ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

Προγραμματισμός στο Διαδίκτυο

Προγραμματισμός στην πλευρά του εξυπηρετητή: PHP

Μανώλης Μαραγκουδάκης

Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

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 Πανεπιστήμιο Αιγαίου**» 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.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

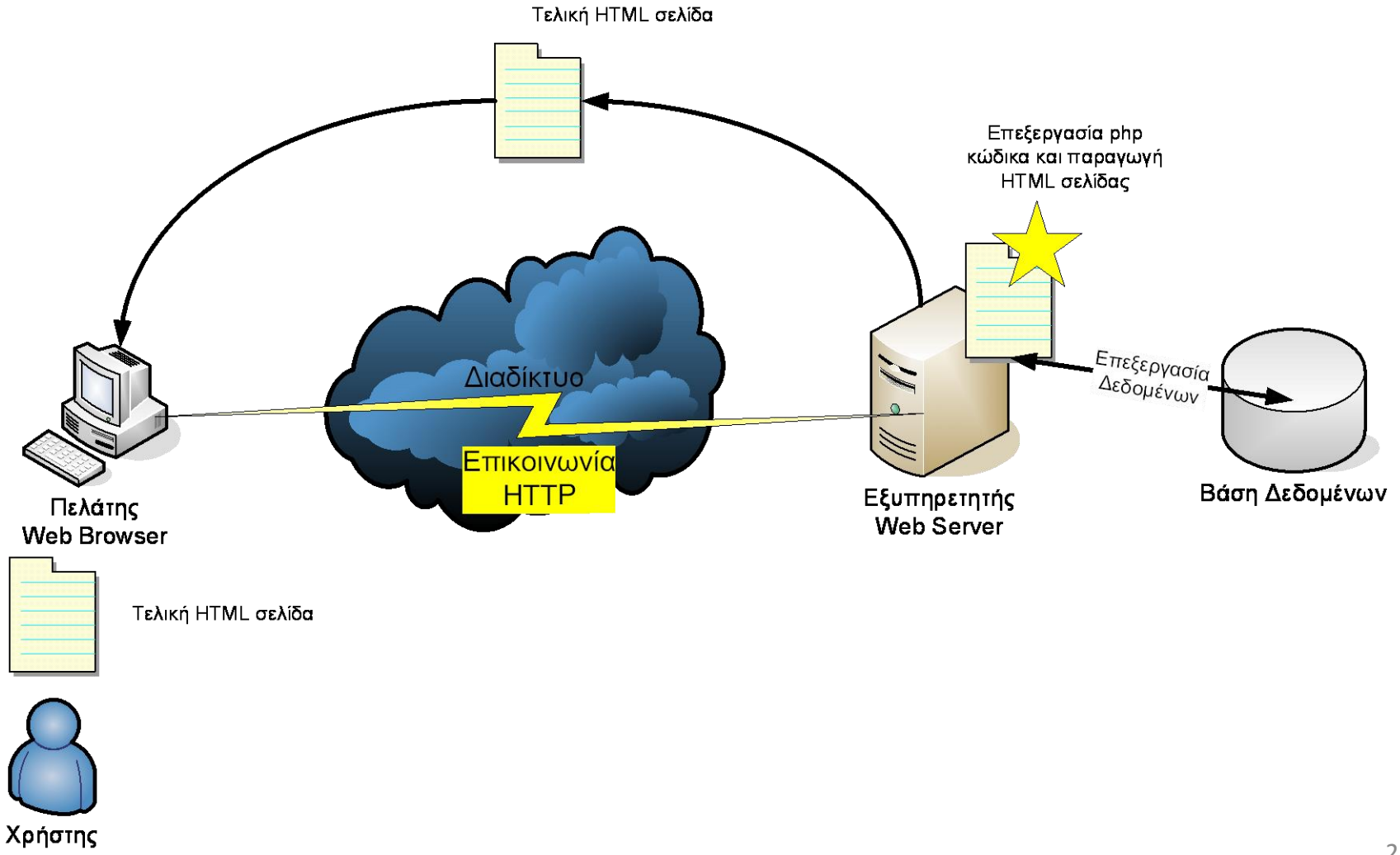
Προγραμματισμός στο Διαδίκτυο

Ενότητα 5α – Προγραμματισμός στην πλευρά του εξυπηρετητή: PHP

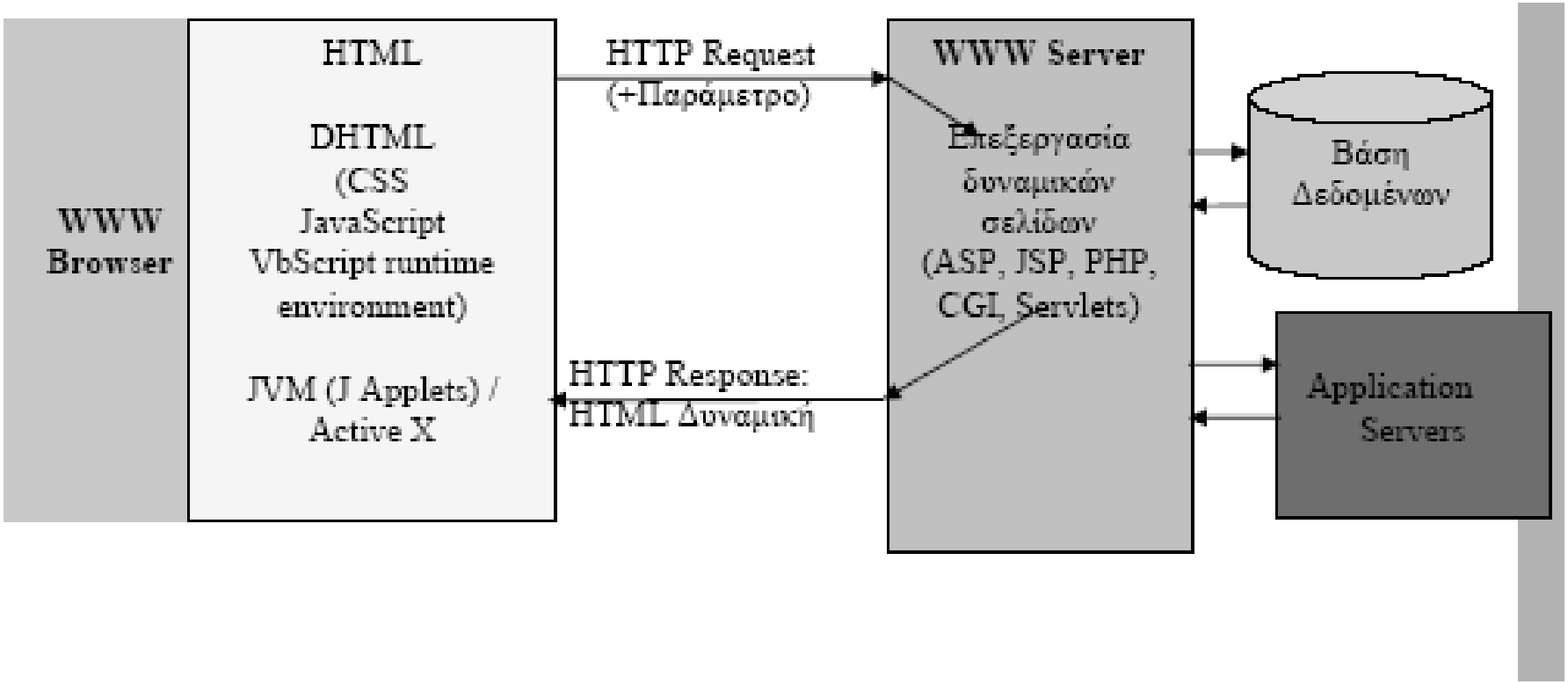
Μανώλης Μαραγκουδάκης

Πανεπιστήμιο Αιγαίου
Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών Συστημάτων

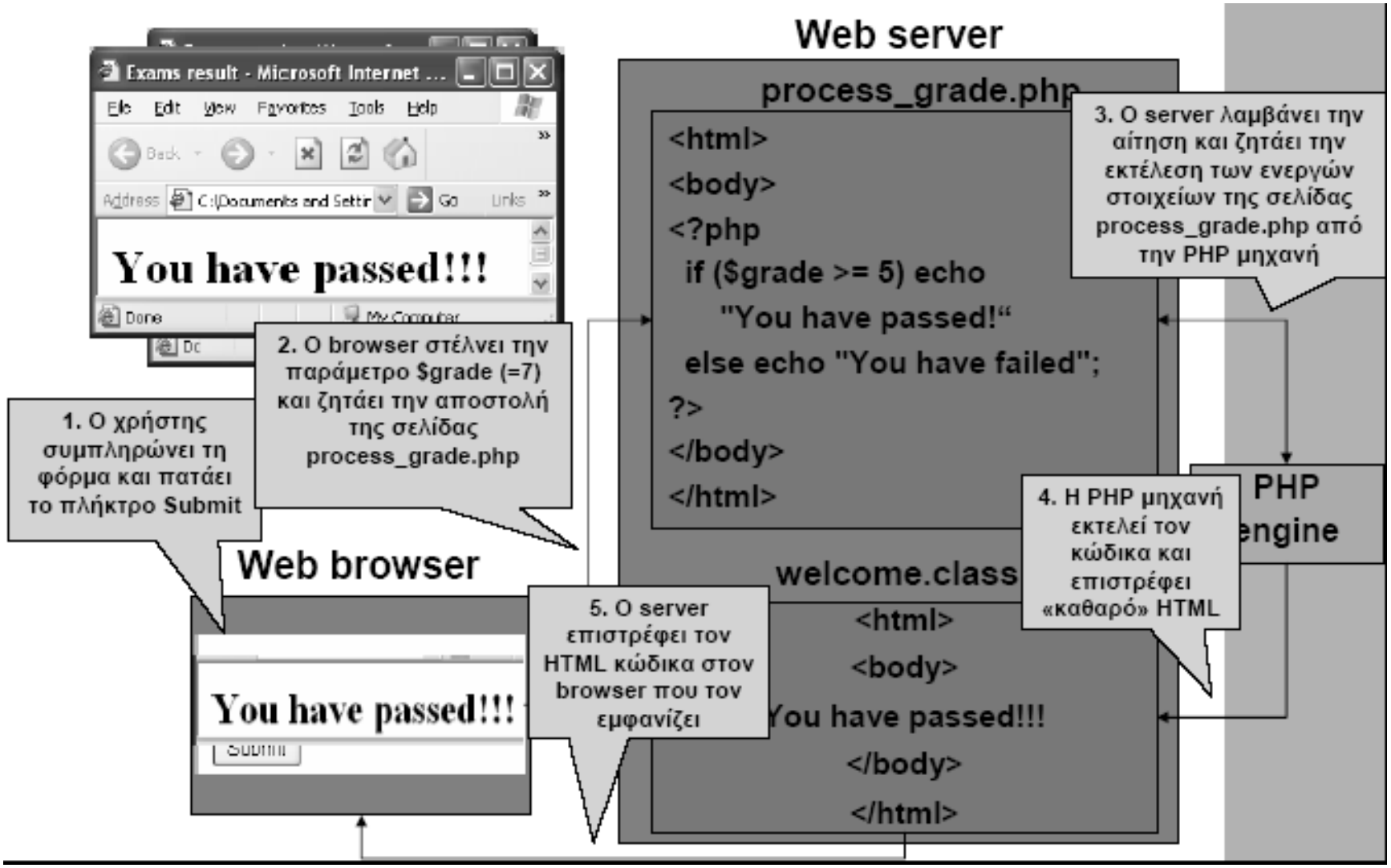
Προγραμματισμός στην πλευρά του εξυπηρετητή



Προγραμματισμός στην πλευρά του εξυπηρετητή



Αλληλεπίδραση browser / web server με χρήση τεχνολογίας server side



Καταλληλότητα, Πλεονεκτήματα, Μειονεκτήματα

- Καταλληλότητα:
  - Δυναμική / Παραμετρική εμφάνιση περιεχομένου
  - Απαραίτητο όταν απαιτείται επικοινωνία (αλληλεπίδραση) με τον Server
  - Δυνατότητα ελέγχου των πελατών, π.χ. μετρητές επισκέψεων (hit counters), ελεγχόμενη πρόσβαση σε κάποιες σελίδες
- Πλεονεκτήματα:
  - Η επεξεργασία μεταφέρεται στο server, χρησιμοποιείται η ισχύς του server
  - Ο κώδικας είναι κρυφός
  - Η εκτέλεση του κώδικα είναι ανεξάρτητη του browser: στέλνεται «καθαρό» HTML που εμφανίζεται πανομοιότυπο σε κάθε browser
  - Η μοναδική λύση για πρόσβαση στο file system του server
- Μειονεκτήματα:
  - Χρησιμοποιεί πολύτιμη επεξεργαστική ισχύ του server.
  - Κλιμάκωση (scalability);

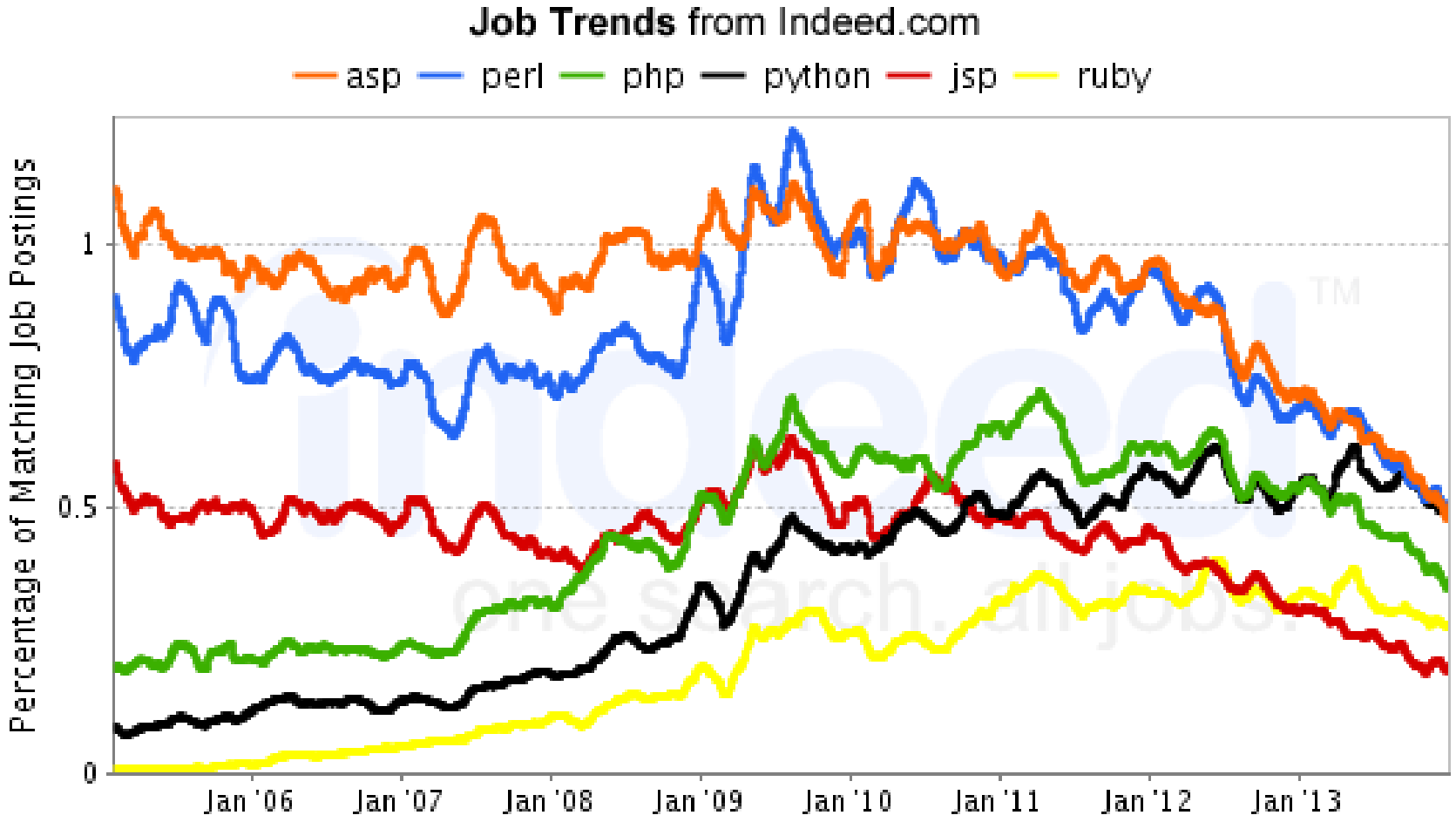
Τεχνολογίες

Τεχνολογία	Server	Πλεονεκτήματα
Active Server Pages (.asp)	MS IIS (Internet Information Server)	Σχετικά εύκολο στη χρήση, καλή ενσωμάτωση/ ολοκλήρωση (integration), καλή υποστήριξη (support) από Microsoft
ASP.NET	IIS	Κλιμάκωση (Scalability), web services.
Java Server Pages (.jsp)		Κλιμάκωση, αξιοπιστία
PHP (.php)	Apache (open source)	Δωρεάν, εύκολη στη χρήση
Java Servlets	Any web server	Προγραμματισμός σε Java, Εύκολη πρόσβαση σε ΒΔ

Τεχνολογίες

Τεχνολογία	Είδος γλώσσας	Παραδείγματα
PHP (.php)	Scriping	Facebook, Wikipedia, Digg, Wordpress
Active Server Pages (.asp)	Scripting (VBScript, JScript)	BN.com, Buy.com, Computerworld, Microsoft, W3Schools
ASP.NET	Compiled (VB.NET, C#, J#, ...)	Costco, Dell, JCPenny, MSN, Microsoft, TechStore, WWU

Τεχνολογίες



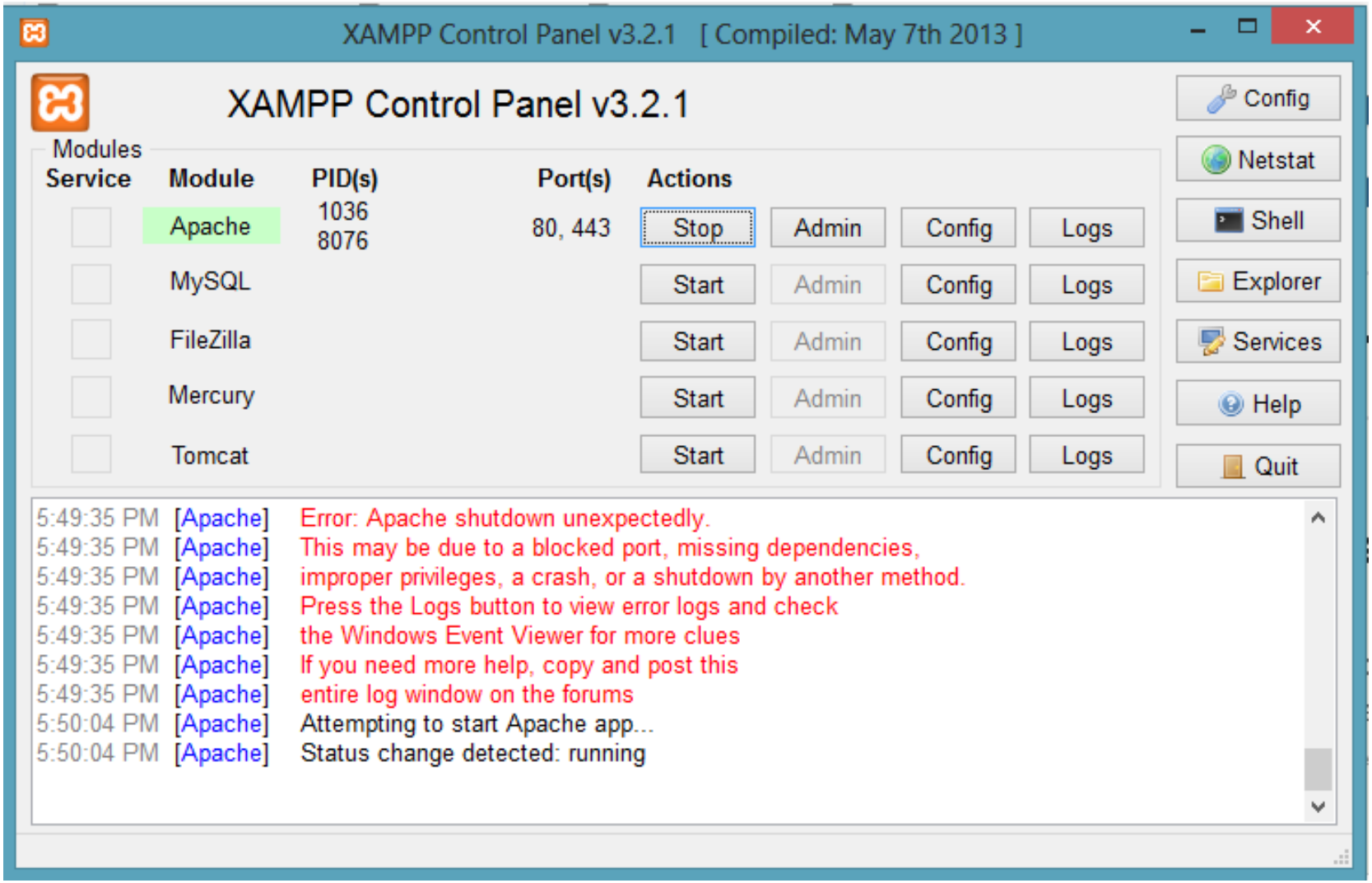
Βασικά για PHP

- PHP = Hypertext Preprocessor
- On-line manual της γλώσσας: http://www.php.net/manual/en/manual.php
- Ο κώδικας πρέπει να περιέχεται σε PHP αρχεία (".php", ".php3", ή".phtml")
- PHP αρχεία μπορούν να περιέχουν text, HTML tags και scripts
- Εκτός από την μηχανή εκτέλεσης της PHP θα πρέπει κανείς να έχει εγκατεστημένο έναν web server (π.χ. Apache) για να πειραματιστεί με την γλώσσα.
  - Εναλλακτικά κάποιο Free hosting Service (π.χ. Freehostia)

Προτεινόμενη πλατφόρμα

- http://www.apachefriends.org/en/xampp.html
- Διαθέτει έκδοση και για Windows και για Linux
- Είναι portable!!!
- Περιέχει:
  - Apache (HTTP web server)
  - MySQL (ΣΔΒΔ)
  - PHP (η γλώσσα…)
  - phpMyAdmin (διαχείριση MySQL στο WWW)
  - FileZilla FTP Server
  - Tomcat (Java HTTP web server, υλοποιεί Java Servlet και JavaServer Pages (JSP) specifications)

XAMPP Control Panel



Τοπική εκτέλεση php αρχείων

- Εκκίνηση του Apache από το XAMPP Control Panel
- Αποθήκευση των αρχείων php κώδικα πηγής στον φάκελο htdocs που βρίσκεται μέσα στον φάκελο εγκατάστασης του xampp
- Φόρτωση της php σελίδας από τον browser ως : http://localhost/example.php
- Αν τα αρχεία είναι σε κάποιον φάκελο π.χ. php1, τότε συμπεριλαμβάνουμε και τον φάκελο: http://localhost/php1/example.php

Γιατί PHP;

- Υψηλή απόδοση: με ένα φτηνό server μπορούν να εξυπηρετηθούν εκατομμύρια επισκέψεων καθημερινά.
- Συνεργάζεται εύκολα με τους περισσότερους database servers (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, κ.α.)
- Σημαντικό για χτίσιμο πληροφοριακών συστημάτων (π.χ. εφαρμογές ηλεκτρονικού εμπορίου)
- Ενσωματωμένες βιβλιοθήκες για συνήθεις web διαδικασίες: δυναμική δημιουργία εικόνων, αποστολή email, χειρισμός cookies

Γιατί PHP;

- Χαμηλό κόστος: δωρεάν!
- Ευκολία μάθησης και χρήσης: η σύνταξή της βασίζεται σε άλλες γλώσσες (μοιάζει με Java, C)
- Υποστηρίζεται από τους περισσότερους web servers σαν module (επιπρόσθετο δομικό στοιχείο)
- Μεταφερσιμότητα (portability): ο ίδιος κώδικας δουλεύει χωρίς αλλαγές και σε άλλο λειτουργικό σύστημα (Windows ⇔ Linux)
- Διαθεσιμότητα του κώδικα προέλευσης (open source): μπορούν να πραγματοποιηθούν αλλαγές στη γλώσσα

PHP - Απλό παράδειγμα

```
<html>
<head><title>PHP Example</title>
</head>
<body>
<?php // ή <? on servers with shorthand support enabled
echo "Hi, I'm a PHP script!";
?>
</body>
</html>
```

PHP παραδείγματα

- print (“ Hello World! ”);
- print (“ Hello World!
Hello Computer! ”);
- print ‘ Hello Google! ’;
- print “A Link”;

- Echo ή Print?
  - Echo: δεν επιστρέφει τιμή, ελάχιστα πιο γρήγορη
  - Print: επιστρέφει τιμή, άρα μπορεί να χρησιμοποιηθεί μέσα σε μια έκφραση

- Επειδή η έξοδος της PHP είναι συνήθως XHTML και η δεύτερη έχει πολλά “” θέλει προσοχή

PHP και XTHML (Α τρόπος)

```
<table style = “border:
1px solid black”>
<tr>
<td>name</td>
<td>John</td>
</tr>
<tr>
<td>address</td>
<td>123 Main
St.</td>
</tr>
</table>
```

```
$name = “John”;
$address = “123 Main St.”;
$output = “”;
$output .= “<table style = \”border: 1px solid
black\”> \n”;
$output .= “ <tr> \n”;
$output .= “ <td>name</td> \n”;
$output .= “ <td>$name</td> \n”;
$output .= “ </tr> \n”;
$output .= “ <tr> \n”;
$output .= “ <td>address</td> \n”;
$output .= “ <td>$address</td> \n”;
$output .= “ </tr> \n”;
$output .= “</table> \n”;
print $output
```

PHP και XTHML (Β τρόπος)

```
<table style = “border:
1px solid black”>
<tr>
<td>name</td>
<td>John</td>
</tr>
<tr>
<td>address</td>
<td>123 Main
St.</td>
</tr>
</table>
```

```
<?
$name = “John”;
$address = “123 Main St.”;
print <<<HERE
<table style = “border: 1px solid black”>
<tr>
<td>name</td>
<td>$name</td>
</tr>
<tr>
<td>address</td>
<td>$address</td>
</tr>
</table>
HERE;
?>
```

HEREDOCS: Διατηρεί τα “”, τα newlines και υποστηρίζει κανονικά ένθεση μεταβλητών

PHP και XTHML (Γ τρόπος)

```
<table style = “border:
1px solid black”>
<tr>
<td>name</td>
<td>John</td>
</tr>
<tr>
<td>address</td>
<td>123 Main
St.</td>
</tr>
</table>
```

```
<?php
$name = “John”;
$address = “123 Main St.”;

?>
<table style = “border: 1px solid black”>
<tr>
<td>name</td>
<td><?php print $name; ?></td>
</tr>
<tr>
<td>address</td>
<td><?php print $address; ?></td>
</tr>
</table>
```

Σχόλια

```
<?php echo “Test"; // one-line comment
/* This is a
multi line comment */
echo “Test"; # This is shell-style style comment
?>
```

Μεταβλητές στην PHP

- Μια μεταβλητή στην PHP αναπαρίσταται με το σύμβολο **$** ακολουθούμενο από το όνομα της μεταβλητής
- Υπάρχουν τριών βασικών τύπων μεταβλητές:
  - Βαθμωτή (scalar) {integer, float, string, boolean}
  - Πίνακας (indexed array)
  - Συσχετιζόμενος πίνακας (associative array)
- Υπάρχει και ο τύπος **Object** (κλάση)
- Οι μεταβλητές είναι ο κύριος μηχανισμός για τη μεταφορά δεδομένων μεταξύ σελίδων ή τμημάτων σελίδων
- Υπάρχουν τρεις βασικές λειτουργίες που μπορούμε να κάνουμε με μία μεταβλητή:
  - να τη θέσουμε,
  - να την επαναθέσουμε
  - να την προσπελάσουμε

Παραδείγματα Μεταβλητών

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

<?php
class Car {
    function Car() {
        $this->model = "VW";
    }
}
// create an object
$herbie = new Car();

// show object properties
echo $herbie->model;
?>

</body>
</html>
```

```
<?php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
?>
```

Τύποι

- Boolean, integer, floating-point number (float), string, array, object, resource, NULL
- Ο τύπος μιας μεταβλητής ΔΕΝ δηλώνεται αλλά προσδιορίζεται από την τιμή που της δίνεται (**Loosely Typed Language**)
- Μετατροπή από ένα τύπο δεδομένων σε άλλο

```
$mydouble = (double)$myint
```

Κανόνες ονομασίας μεταβλητών

- Να αρχίζει με γράμμα ή underscore(_)
- Να αποτελείται από γράμματα, αριθμούς ή underscore (_)
- Να μην είναι δεσμευμένη λέξη (όπως π.χ. print)
- Τα ονόματα των μεταβλητών **είναι case-sensitive**, π.χ. $baby_names και $Baby_names δεν είναι τα ίδια

Τιμές μεταβλητών

- Εκτός από κείμενο, ως τιμές σε μεταβλητές μπορούμε να δώσουμε και αριθμούς καθώς και άλλα αντικείμενα (objects, booleans)
- Για να προβάλλουμε κείμενο χρησιμοποιούμε απλά ή διπλά εισαγωγικά:
  - print (″Αυτό είναι ένα παράδειγμα!″);
- Αν θέλουμε να εκτυπώσουμε το κείμενο μαζί με τα εισαγωγικά, χρησιμοποιούμε το χαρακτήρα διαφυγής \, που ορίζει στην PHP να μη θεωρήσει τον επόμενο χαρακτήρα ως μέρος του κώδικα, αλλά ως απλό κείμενο
  - print (″\″Αυτό είναι ένα παράδειγμα!\″″);

PHP σταθερές

- Σύνταξη
  - define(*name*, *value*, *case-insensitive*)
  - Παράδειγμα
    - <?php
      define("GREETING", "Welcome home!");
      echo GREETING;
      ?>
    - <?php
      define("GREETING", " Welcome home!", true);
      echo greeting;
      ?>
  - Οι σταθερές είναι πάντα Global στην εμβέλεια

Εμβέλεια Μεταβλητών

- Global και Local
  - <?php
    $x = 1; // global εμβέλεια
    function demo() {
    // η χρήση της x μέσα στη συνάρτηση θα δημιουργήσει σφάλμα
    echo " Variable x inside function is: $x ";
    }
    demo();
    echo " Variable x outside function is: $x ";
    ?>

```
<?php
function demo() {
   $x = 1; // local εμβέλεια
   echo "<p>Variable x inside function is: $x</p>";
}
demo();
// // η χρήση της x έξω από τη συνάρτηση θα δημιουργήσει σφάλμα
echo "<p>Variable x outside function is: $x</p>";
?>
```

Εμβέλεια Μεταβλητών

- Η χρήση της λέξης global:
  - ```
    <?php
    $x = 5;
    $y = 10;
    function myTest() {
       global $x, $y;
       $y = $x + $y;
    }
    myTest();
    echo $y; // outputs 15
    ?>
    ```

Η PHP αποθηκεύει τις μεταβλητές και σε ένα πίνακα με όνομα $GLOBALS[*index*]. Όπου Index το όνομα της μεταβλητής

```
<?php
$x = 5;
$y = 10;
function myTest() {
   $GLOBALS['y'] = $GLOBALS['x'] +
$GLOBALS['y'];
}
myTest();
echo $y; // outputs 15
?>
```

Χρήση Static (διατηρεί την τιμή και μετά τη συνάρτηση):

```
<?php
function myTest() {
   static $x = 0;
   echo $x;
   $x++;
}
myTest();
myTest();
myTest();
?>
```
```

Μεταβλητές τύπου String-Συναρτήσεις

- <?php
  echo strlen("Hello world!"); // outputs 12
  ?>
- <?php
  echo str_word_count("Hello world!"); // outputs 2
  ?>
- <?php
  echo strrev("Hello world!"); // outputs !dlrow olleH
  ?>
- <?php
  echo strpos("Hello world!", "world"); // outputs 6
  ?>
- <?php
  echo str_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly!
  ?>

Περισσότερα στο: http://www.w3schools.com/php/php_ref_string.asp

Συναρτήσεις μεταβλητών

- string gettype(mixed var): επιστρέφει μια συμβολοσειρά που περιέχει τον τύπο μιας μεταβλητής ή “unknown type”
  - mixed : η παράμετρος μπορεί να δεχθεί πολλούς τύπους
- bool settype(mixed var, string type): αλλάζει τον τύπο μιας μεταβλητής
- bool is_array(), is_double(), is_int(), is_string(), is_object(): ελέγχουν τύπους
- bool isset(mixed var): ελέγχει αν μια μεταβλητή είναι ορισμένη
- void unset(mixed var): διαγράφει/καταστρέφει μια μεταβλητή
- bool empty(mixed var): ελέγχει αν μια μεταβλητή είναι άδεια
- int intval(mixed var), double doubleval(mixed var), string strval(mixed var): μετατρέπουν την τιμή μιας μεταβλητής στον ονοματιζόμενο τύπο

Μεταβλητές τύπου πίνακα

- Οι μεταβλητές τύπου πίνακα ξεκινούν με $, όπως και οι βαθμωτές. Η συνάρτηση **array()** εκχωρεί μια σειρά τιμών σε έναν πίνακα με τον ακόλουθο τρόπο:

```
$students = array("Μαρία", "Γιάννης", "Λευτέρης");
```

- Η παραπάνω εντολή αυτόματα εκχωρεί ένα **αριθμητικό κλειδί** σε κάθε στοιχείο με τη σειρά δίνοντας στο πρώτο στοιχείο το κλειδί **0**. Μπορούμε τώρα να αναφερόμαστε π.χ. στο στοιχείο “Λευτέρης” ως $students[2].
- Ο ακόλουθος κώδικας θα εκτυπώσει το τρίτο στοιχείο του πίνακα που είναι ο μαθητής Λευτέρης

```
<?php
print "$students[2]";
?>
```

Μεταβλητές τύπου πίνακα

- Υπάρχει και άλλος τρόπος να ορίσουμε έναν πίνακα ή να προσθέσουμε στοιχεία σε έναν ήδη υπάρχοντα πίνακα:

  $students[] = “Μαρία”;

  $students[] = “Γιάννης”;

  $students[] = “Λευτέρης”;

- Για να προσθέσουμε έναν νέο μαθητή γράφουμε (ανεξάρτητα από τον τρόπο που χρησιμοποιήσαμε για τη δημιουργία του πίνακα):

  $students[] = “Βασίλης”;

- Η PHP δίνει αυτόματα στον Βασίλη ένα κλειδί, το αμέσως επόμενο κενό, που στην περίπτωση αυτή είναι το [3].

Μεταβλητές τύπου συσχετιζόμενου πίνακα

- Οι συσχετιζόμενοι πίνακες διαχωρίζουν τα περιεχόμενα στοιχεία όχι με αριθμούς, αλλά **με ονόματα** που εμείς καθορίζουμε. Μέσα στη συνάρτηση array() καθορίζουμε ζεύγη key=>value. Για παράδειγμα:

```
$stud = array( “name”=>”John”,
               “haircolor”=>”black”,
               “eyecolor”=>”green”,
               “age”=>17);
```

- Μπορούμε να πάμε σε οποιοδήποτε στοιχείο του πίνακα μέσω των ονομάτων των κλειδιών που ορίσαμε.
- Για παράδειγμα:

```
print $stud[“eyecolor”];
```

θα δώσει green.

Μεταβλητές τύπου συσχετιζόμενου πίνακα

- Μπορούμε επίσης να θέσουμε κάθε στοιχείο ξεχωριστά:

```
$stud[“name”] = ”John”;
$stud[“haircolor”] = ”black”;
$stud[“eyecolor”] = ”green”;
$stud[“age”] = 17;
```

Πολυδιάστατος πίνακας

Ένας πολυδιάστατος πίνακας είναι ένας πίνακας που περιέχει άλλους πίνακες.

```
$stud = array(
array (“name”=>”John”,”haircolor”=>”black”,”eyecolor”=>”green”,”age”=>17),
array (“name”=>”Mary”,”haircolor”=>”blond”,”eyecolor”=>”blue”,”age”=>16),
array (“name”=>”Kenny”,”haircolor”=>”brown”,”eyecolor”=>” brown”,”age”=>17),
array (“name”=>”Bill”,”haircolor”=>”blond”,”eyecolor”=>”green”,”age”=>16)
);
```

Για να προσπελάσουμε ένα στοιχείο:

```
print $stud[2][“age”];
```

Συναρτήσεις Πινάκων

- Ταξινόμηση απλών πινάκων

  $products=array(“Tires”, ”Oil”); **sort**($products);

  $products=array(13,3,7); **sort**($products);

- Ταξινόμηση συσχετιζόμενων πινάκων

  $prices = array(“Tires”=>100,”Oil”=>10, “Spark Plugs”=>4);

  asort($prices) // (“Spark Plugs”=>4,”Oil”=>10, “Tires”=>100)

  ksort($prices) // (”Oil”=>10, “Spark Plugs”=>4 “Tires”=>100)

- Αντίστροφες Ταξινομήσεις

  $myArray_r = **array_reverse**($myArray);

- Τυχαία σειρά:

  shuffle($myArray);

- Πλήθος στοιχείων πίνακα:

  $num = **count**($products)

asort — Ταξινομεί έναν πίνακα συσχέτισης ως προς την τιμή του στοιχείου διατηρώντας τη συσχέτιση κλειδιού-τιμής.
ksort — Ταξινομεί έναν πίνακα συσχέτισης ως προς το κλειδί του στοιχείου.
array_reverse — επιστρέφει έναν πίνακα με τα στοιχεία σε αντίστροφη σειρά.
shuffle — Ανακάτεμα των στοιχείων ενός πίνακα
count — Καταμέτρηση όλων των στοιχείων ενός πίνακα, ή όλων των ιδιοτήτων ενός αντικειμένου

Τελεστές σύγκρισης

Τελεστής	Περιγραφή
==	Επιστρέφει true εάν η τιμή της μεταβλητής αριστερά είναι ίση με αυτή δεξιά. Αλλιώς, επιστρέφει false.
===	Παρόμοιος με ==, αλλά επιστρέφει true μόνο αν οι μεταβλητές είναι του ίδιου τύπου. (μόνο στην PHP4)
!=	Επιστρέφει true εάν η τιμή της μεταβλητής αριστερά δεν είναι ίση με αυτή δεξιά. Αλλιώς, επιστρέφει false.
<>	Επιστρέφει true εάν η τιμή της μεταβλητής αριστερά δεν είναι ίση με αυτή δεξιά. Αλλιώς, επιστρέφει false.
!==	Παρόμοιος με !=, αλλά επιστρέφει true και αν οι μεταβλητές δεν είναι του ίδιου τύπου. (μόνο στην PHP4)
<	Επιστρέφει true εάν η τιμή της μεταβλητής αριστερά είναι μικρότερη από αυτή δεξιά. Αλλιώς, επιστρέφει false.
>	Επιστρέφει true εάν η τιμή της μεταβλητής αριστερά είναι μεγαλύτερη από αυτή δεξιά. Αλλιώς, επιστρέφει false.
<=	Επιστρέφει true εάν η τιμή της μεταβλητής αριστερά είναι μικρότερη ή ίση από αυτή δεξιά. Αλλιώς, επιστρέφει false.
>=	Επιστρέφει true εάν η τιμή της μεταβλητής αριστερά είναι μεγαλύτερη ή ίση από αυτή δεξιά. Αλλιώς, επιστρέφει false.

Λογικοί και Αριθμητικοί τελεστές

- Λογικοί

  and ή &&, or ή||, xor, !

- Αριθμητικοί

  +, –, *, /, %

- Υπάρχουν ακόμα και οι τελεστές αύξησης/μείωσης και οι σύνθετοι τελεστές (π.χ. $a+=5;)

if-else-elseif

```
if(cond)
        …; // μια εντολή
 if(cond)
 {
     …;
     …;
 }
```

```
if(cond)
 {
     …;
 }
 else
 {
     …;
 }
```

```
if(cond)
 {
     …;
 }
 elseif
 {
     …;
 }
 elseif
 {
     …;
 }
```

Παράδειγμα

```
if ($a > $b) {
      echo "a is bigger than b";
      $b = $a;
   }

if ($a > $b) {
      echo "a is bigger than b";
   } else {
      echo "a is NOT bigger than
   b";
   }

if ($a > $b) {
      echo "a is bigger than b";
   } elseif ($a == $b) {
      echo "a is equal to b";
   } else {
      echo "a is smaller than b";
   }
```

Switch

```
switch($var)
{
    case "a":
        …;
        break;
    case "c":
        …;
        break;
    default :
        …;
        break;
}
```

Η *case* έκφραση μπορεί να είναι οποιαδήποτε έκφραση, αρκεί να έχει τιμή **απλού τύπου**: integer, floating-point, string.

Παράδειγμα

```
switch ($i) {
        case 0:
         print "i equals 0";
            break;
          case 1:
            print "i equals 1";
            break;
          case 2:
            print "i equals 2";
            break;
}
```

```
switch ($name) {
   case 'sylvie': // fall-through
   case 'bruno':
        print('yes');
        break;
   default:
        print('no');
        break;
}
```

while/for/do...while

```
while(cond)
{
      …;
}
```

```
for(exp;cond;exp)
 {
    …;
 }
```

```
do
{
      …;
}
while(cond)
```

foreach

- Δομή ελέγχου για χειρισμό πινάκων
- Δυο εκδόσεις

  foreach (array_expression as $value)

  statement

  foreach (array_expression as $key => $value)

  statement
- Στην πρώτη έκδοση σε κάθε επανάληψη η τιμή του τρέχοντος στοιχείου του πίνακα ανατίθεται στην μεταβλητή **$value** και ο εσωτερικός δείκτης του πίνακα αυξάνεται κατά ένα.
- Η δεύτερη έκδοση λειτουργεί όπως και η πρώτη, με τη διαφορά ότι το κλειδί του τρέχοντος στοιχείου θα ανατεθεί στην μεταβλητή **$key** σε κάθε επανάληψη.

```
<?php
$arr = array("one", "two", "three");

foreach ($arr as $key => $value) {
    echo "Key: $key; Value: $value<br
/>\n";
}
?>
```

Θα τυπώσει:

```
Key: 0; Value: one
Key: 1; Value: two
Key: 2; Value: three
```

Παράδειγμα

```
$a = array ( "one" => 1,  "two" => 2, "three" => 3,
"seventeen" => 17 );
foreach($a as $k => $v) {
    print "\$a[$k] => $v.\n";
}
```

Θα τυπώσει:
$a[one] => 1. $a[two] => 2. $a[three] => 3. $a[seventeen] => 17.

String→Arrays

```
<html>
<head>
<title>explode.php</title>
</head>
<body>
<h1>Using explode</h1>
<?php
$theString = “PARC (Palo Alto Research Center) was one of the single most important hubs of invention for modern computing”;
$theArray = explode(“ “, $theString);
print “<pre> \n”;
print_r($theArray);
print “</pre> \n”;
?>
</body>
</html>
```

Using explode

```
Array
(
    [0] => PARC
    [1] => (Palo
    [2] => Alto
    [3] => Research
    [4] => Center)
    [5] => was
    [6] => one
    [7] => of
    [8] => the
    [9] => single
    [10] => most
    [11] => important
    [12] => hubs
    [13] => of
    [14] => invention
    [15] => for
    [16] => modern
    [17] => computing
)
```

Το κείμενο σε ένα στοιχείο <pre> εμφανίζεται με courier font ενώ διατηρεί τα κενά και τα line breaks

String→Arrays

```
<html>
<head>
<title>split</title>
</head>
<body>
<h1>Using preg_split</h1>
<?php
$theString = ”joe@somebody.net”;
$theArray = preg_split(“/[@\.]/”, $theString);
print ”<pre>\n”;
print_r($theArray);
print ”</pre>\n”;
?>
</body>
</html>
```

Using preg_split

```
Array
(
    [0] => joe
    [1] => somebody
    [2] => net
)
```

Regular Expression!

Σπάσιμο δομών ελέγχου, επαναλήψεων και Script

- Έξοδος από δομή ελέγχου (for, foreach, while, do-while ή switch)

  break

- Μεταπήδηση επόμενη επανάληψη βρόγχου

  continue

- Σταμάτημα εκτέλεσης PHP Script

  exit

Επαναχρησιμοποίηση κώδικα

- Δυνατότητα για επαναχρησιμοποίηση κώδικα από άλλα αρχεία (php, html, οποιοδήποτε άλλο)
- Συνήθως αρχεία .inc αλλά προσοχή! Ο πηγαίος κώδικας ενός **.inc** μπορεί να φανεί αν φορτωθεί απευθείας από browser ⇒ καλύτερα να χρησιμοποιούμε .php ή «κρύψιμο» των .inc (αποθήκευση σε κατάλογο που δεν είναι 'δημοσιευμένος')
- Λειτουργίες **require()** & **include()**
  - Διαφέρουν μόνον ως προς το χειρισμό της μη εύρεσης του αρχείου που εισάγεται (fatal error vs. warning).
  - Require: fatal error (E_COMPILE_ERROR) και σταματάει το script
  - Include: warning (E_WARNING) και συνεχίζει
- Πλεονεκτήματα επαναχρησιμοποίησης κώδικα:
  - Μικρότερο «κόστος» (όχι περιττή επανεγγραφή κώδικα)
  - Αυξημένη αξιοπιστία (αν ο κώδικας δουλεύει κάπου, δουλεύει με τον ίδιο τρόπο παντού)

Λειτουργίες require() & include()

- Γίνεται συμπερίληψη αρχείων βάσει του μονοπατιού που προσδιορίζεται. Το μονοπάτι μπορεί να είναι, είτε σε απόλυτη, είτε σε σχετική μορφή.
- Αν δεν δίνεται μονοπάτι εξετάζεται η μεταβλητή συστήματος include_path (παρόμοια λογική με την μεταβλητή περιβάλλοντος PATH)

  π.χ. include_path=".;c:\php\includes"
- Αν δεν βρεθούν ούτε στο include_path εξετάζεται ο κατάλογος που περιέχει το ίδιο το script καθώς και το τρέχον working directory.
- Όταν γίνεται συμπερίληψη ενός αρχείου, ο κώδικας που περιέχει κληρονομεί την εμβέλεια μεταβλητών της γραμμής του script που εμφανίζεται το require() / include().
- Οποιεσδήποτε μεταβλητές διαθέσιμες σε αυτή την γραμμή στο καλών αρχείο θα είναι διαθέσιμες και στο καλούμενο αρχείο.
- Όμως, όλες οι συναρτήσεις και κλάσεις που ορίζονται στο συμπεριλαμβανόμενο αρχείο έχουν σφαιρική εμβέλεια.

Require

```
// require.php
<?php
echo "Here is a very simple PHP statement.<BR>";
?>

<?php
echo "This is the main file.<BR>";
require( "reusable.php" );
echo "The script will end now.<BR>";
?>
```

Χρήση include μέσα σε συνάρτηση

vars.php

```
<?php
$color = 'green';
$fruit = 'apple';
?>
// include_scoped.php
<?php
function foo()  {
   global $color;
   include 'vars.php';
   echo "A $color $fruit";
}
/* το αρχείο vars.php βρίσκεται στην εμβέλεια της συνάρτησης foo() αρα η
     μεταβλητή $fruit ΔΕΝ είναι διαθέσιμη έξω από την εμβέλεια αυτή. Η  $color ΕΙΝΑΙ
     επειδή δηλώθηκε ως global. */
foo();                  // A green apple
echo "A $color $fruit";   // A green
?>
```

Συναρτήσεις στην PHP

```
function function_name( [[&]parameter [, …] ] )
{
        statements
}
```

Συναρτήσεις στην PHP

- Τα ονόματα των συναρτήσεων ακολουθούν τους ίδιους κανόνες με τα ονόματα των μεταβλητών. Περιορισμοί στην ονομασία συναρτήσεων:
  - Όχι ίδια ονόματα με υπάρχουσες συναρτήσεις
  - Ονόματα μονάχα από γράμματα, ψηφία και χαρακτήρες υπογράμμισης
  - Τα ονόματα δεν μπορούν να ξεκινούν με ψηφίο
  - Έγκυρα ονόματα: name(), name2(), name_three()
  - Άκυρα ονόματα: 5name(), name-six(), explode()
- Παράμετροι
  - 0 ή περισσότερες
- Διαβίβαση παραμέτρων
  - By value
  - By reference
- Επιστροφή τιμής με `return` (Null αν δεν υπάρχει)

Χρησιμοποιώντας συναρτήσεις στην PHP

- Κλήση συνάρτησης ΧΩΡΙΣ πέρασμα παραμέτρων

  function_name();

- Κλήση συνάρτησης ΜΕ πέρασμα παραμέτρων

  function_name(“parameter”);

- Παραδείγματα:

  function_name(2);

  function_name(“string”);

  function_name($variable);

  function_name(&$variable);

Κλήση συναρτήσεων στην PHP

- Η κλήση εξαρτάται από το πρωτότυπο της συνάρτησης

  π.χ. array **explode** (string separator, string str [, int limit])

- Κλήση της συνάρτησης explode:

  $str = “abc def ghi”;

  $str_array = explode(“ ”, str); // επιστρέφει array 3 στοιχείων

  $str_array = explode(“ ”, str, 2); // επιστρέφει μόνο 2 στοιχεία

- Οι κλήσεις σε συναρτήσεις ΔΕΝ είναι ευαίσθητες σε κεφαλαία-πεζά: function_name() = FUNCTION_NAME() = Function_Name()

- ΠΡΟΣΟΧΗ! Τα ονόματα των μεταβλητών ΕΙΝΑΙ ευαίσθητα σε πεζά-κεφαλαία: $name ≠ $Name

Παραδείγματα

```
function gst($amount, $rate=0.12) {
    return $amount*$rate;
}
$tax = gst($purchase, 0.08);
$tax = gst($purchase);
```

```
function doublevalue(&$var) {
    $var = $var * 2;
}
$variable = 5;
doublevalue($variable);
print "\$variable is: $variable"; //$variable is: 10
```

Ορίζοντας τις δικές μας συναρτήσεις

Οι συναρτήσεις δεν απαιτείται να έχουν οριστεί πριν να χρησιμοποιηθούν, εκτός εάν μια συνάρτηση ορίζεται υπό συνθήκη.

```
<?php
    $makefoo = true;
    /* We can't call foo() from here since it doesn't exist yet,
    but we can call bar() */
    bar();
    if ($makefoo) {
      function foo() {
        echo "I don't exist until program execution reaches me.\n";
      }
    }
    /* Now we can safely call foo() since $makefoo evaluated to true */
    if ($makefoo) foo();
    function bar() {
      echo "I exist immediately upon program start.\n";
    }
    ?>
```

Συναρτήσεις και εμβέλεια μεταβλητών

- Τοπικές μεταβλητές
  - Οι μεταβλητές που ορίζονται εντός μιας συνάρτησης είναι τοπικές στη συνάρτηση
- Σφαιρικές μεταβλητές (global)
  - Οι μεταβλητές που ορίζονται έξω από μια συνάρτηση θεωρούνται global
  - Οι global μεταβλητές δεν μπορούν να χρησιμοποιηθούν μέσα σε μια συνάρτηση, εκτός αν δηλωθούν μέσα στη συνάρτηση ως global
- Στατικές μεταβλητές (static)
  - Οι μεταβλητές που ορίζονται ως static διατηρούν την τιμή τους μεταξύ κλήσεων της συνάρτησης

Παραδείγματα

```php
<?php
$a = 1;
include 'b.inc';
?>
```

```php
<?php
$a = 1; /* global scope */

function test()
{
    echo $a; /* reference to local scope variable */
}

test();
?>
```

```php
<?php
$a = 1;
$b = 2;

function Sum()
{
    global $a, $b;

    $b = $a + $b;
}

Sum();
echo $b;
?>
```

```php
<?php
function test()
{
    static $a = 0;
    echo $a;
    $a++;
}
?>
```

Παραδείγματα Χρήσης

```
<html>
<body>
<b>
<?php
echo "Hello, World!";
?>
</b>
<?php
echo " <b> Hello, World! </b>"
?>
</body>
</html>
```

hello_world1.php

Παραδείγματα Χρήσης

```
<html><body>
<?php
$greeting="Hello ";
$num=3+2;
$num=$num+1;
print $greeting.$num." people!"; // use of
    concatenation operator
?>
</body></html>
```

concat.php

Παραδείγματα Χρήσης

```
<html><body>
<?php
$strUser=$_SERVER["HTTP_USER_AGENT"];
print $strUser;
?>
</body></html>
```

globals.php

Παραδείγματα Χρήσης

```
<?php
$h=strftime("%H");
print "<p>".strftime("%m/%d/%Y %H:%M:%S %p")."</p>";
if ($h<12)
    print "Kalhmera";
 else{
 if ($h==12)
    print "Kalo mesimeri";
   else
    print "Kalo apogeyma";
 }
?>
```

strftime.php

Παραδείγματα Χρήσης

```
<?php
for ($i=1; $i<=6; $i=$i+1){
  print "<h".$i.">This is header ".$i."</h".$i.">";
 }
?>
```

for.php

Παραδείγματα Χρήσης

```
<?php
$students[0]="Nikos";
$students[1]="Maria";
for ($i=0; $i<=1; $i=$i+1){
    print $students[$i]."<br>";
}
?>
```

array.php

Παραδείγματα Χρήσης

```
<?php
$students[0]="Nikos";
$students[1]="Maria";
$students[2]="Mpampis";
foreach ($students as $name){
    print $name."<br>";
}
?>
```

foreach.php

Παραδείγματα Χρήσης

```
<h1> Footer Test Page</h1>
<p>
This is the first paragraph.
</p>
<p>2nd p<p>3rd p<p>4th p
<?php include (".\\inc\\footer.htm") ?>
```

include.php

Παράδειγμα Χρήσης

```
<?php
function test()
{
        static $count = 0;

        $count++;
        echo $count.” “;
        if ($count < 10) {
                test();
        }
}
test();
?>
```

recursivefunc.php

Παράδειγμα Χρήσης

```
<?php
    /**
    * Shuffles and displays cards in a deck
    * @author: Eric Anderson
    * @filename: deckofcards.php
    */

    // Create an array of face values
    // and an array of card values
    // then merge them together
    $cards = array_merge(array("J", "Q", "K", "A"), range(2,10));  // 13 cards

    // Shuffle the cards
    shuffle($cards);

    // Create an multidimentional array to hold the 4 suits
    $suits = array(
            'Heart' => array(),
            'Spade' => array(),
            'Diamond' => array(),
            'Club' => array()
    );
```

```
// Add cards to their respective suits
for($i = 0; $i < count($suits); $i++)
{
    for($j = 0; $j < count($cards); $j++)
    {
            $suits['Heart'][$j] = $cards[$j]."<span style=color:#FF0000;>&hearts;</span>";
            $suits['Spade'][$j] = $cards[$j]."&spades;";
            $suits['Diamond'][$j] = $cards[$j]."<span style=color:#FF0000;>&diams;</span>";
            $suits['Club'][$j] = $cards[$j]."&clubs;";
    }
}

// Create a deck
$deck = array();

// Merge the suits into the empty deck array
$deck = array_merge($deck, $suits);
```

```
// Display the deck to the screen
echo "<p><b>Deck of cards:</b></p>";
foreach($deck as $k1 => $v1)
{
    // Display suit name
    echo "<p> $k1's<br /> {<br />  ";
    $acc = 0;

    // Display card value
    foreach($v1 as $k2 => $v2)
    {
        echo "$v2 ";
        $acc++;

        if ($acc == 4)
        {
            echo "<br />  ";
            $acc = 0;
        }
    }
    echo "<br /> }</p>";
}
?>
```