
php

Τι έχει το μενού σήμερα???

1. Τι είναι η PHP???

Τι είναι η PHP??? Τι είναι ο άνθρωπος???

- Server side, scripting language.

Όπα ώπα…Server-side?

- Server-side: τα προγράμματα τρέχουν στον web server που βρίσκεται εγκατεστημένο το site.

- Client-side: τα προγράμματα τρέχουν στον υπολογιστή που βρίσκεται ο browser του χρήστη.

Scripting Language Vs. Programming Language

- Scripting language: τρέχει σε απάντηση σε κάποιο γεγονός (event). Παραδείγματα τέτοιων γλωσσών είναι οι PHP, Javascript.

- Programming Language: μπορεί να τρέξει ακόμα και αν δεν υπάρχουν γεγονότα ή ακόμα και να δημιουργήσει η ίδια γεγονότα. Παραδείγματα τέτοιων γλωσσών είναι οι Java, C++ και πολλές άλλες.

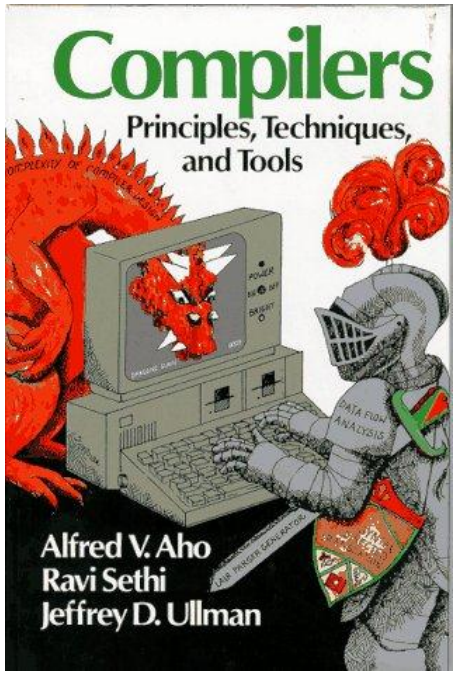
Τι είναι η PHP??? Τι είναι ο άνθρωπος???

- Server side, scripting language
- Χρειάζεται οπωσδήποτε Web Server → Run-As-Is

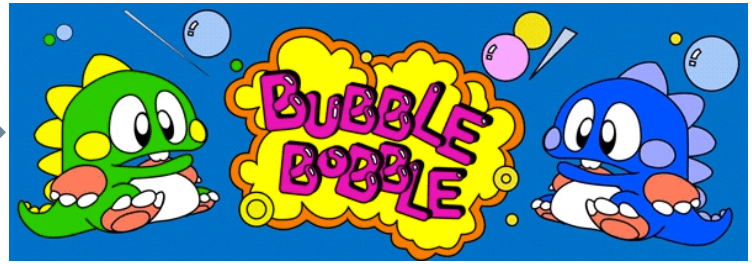
Run-As-Is??? WTF???

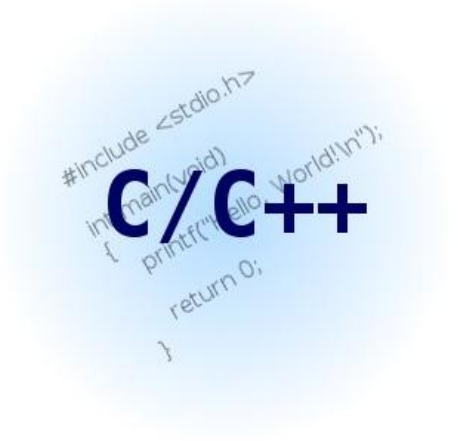
Source Code

Compiler



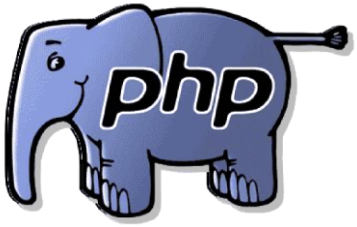
Program



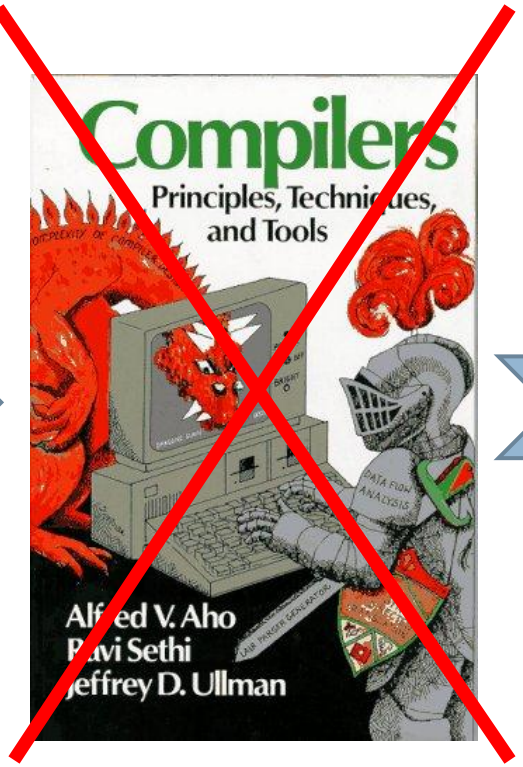


Run-As-Is??? WTF??? (2)

Source Code



Compiler



Program

Τι είναι η PHP??? Τι είναι ο άνθρωπος???

- Server side, scripting language
- Χρειάζεται οπωσδήποτε Web Server → Run-As-Is
- Δουλεύει σε συνεργασία με την HTML

PHP + HTML = L.F.E.

- Τα PHP αρχεία έχουν κατάληξη .php
- Οι εντολές της php βρίσκονται ανάμεσα σε <? php και ?>
- Οι εντολές της php τελειώνουν με semicolon (;) ΠΑΝΤΑ!!!

```
echo "Hello World!";

phpinfo();
```

- Ο υπόλοιπος κώδικάς είναι γραμμένος σε HTML

Παράδειγμα --- Hello World

```
<html>
        <head>
                <title>Hello World!</title>
        </head>
        <body>
                <?php  echo "Hello World!"; ?>
        </body>
</html>
```

Hello World!

Τι είναι η PHP??? Τι είναι ο άνθρωπος???

- Server side, scripting language
- Χρειάζεται οπωσδήποτε Web Server → Run-As-Is
- Δουλεύει σε συνεργασία με την HTML
- Δίνει περισσότερη λειτουργικότητα από την HTML (προφανώς…)

Πρόσθετη Λειτουργικότητα

- HTML Pages
  - Στατικές από την φύση τους
  - Χρησιμοποιούνται για να ενημερώνουν το ευρύ κοινό
- PHP Pages
  - Δυναμικές σελίδες
  - Δίνουν την δυνατότητα interaction με τον χρήστη για να του δώσουν τις πληροφορίες που επιθυμεί

Τι είναι η PHP??? Τι είναι ο άνθρωπος???

- Server side, scripting language
- Χρειάζεται οπωσδήποτε Web Server → Run-As-Is
- Δουλεύει σε συνεργασία με την HTML
- Δίνει περισσότερη λειτουργικότητα από την HTML (προφανώς…)
- Παρόμοια Σύνταξη με Java, C++, Perl και κυρίως ASP.

Τι έχει το μενού σήμερα???

1. Τι είναι η PHP???

2. Ιστορικά Στοιχεία

Ιστορία μου, Αμαρτία μου…

- Version 1 (1994)
  - Rasmus “You’re Da Man” Lerdorf
  - Συντήρηση του Website του με χρήση Perl.
  - Ανάπτυξη εργαλείου για να κάνει την ζωή του πιο εύκολη.

Τι έχει το μενού σήμερα???

1. Τι είναι η PHP???
2. Ιστορικά Στοιχεία
3. Και γιατί όχι...???

Επειδή έτσι μας αρέσει…

A. Open Source / Free Software
B. Cross Platform / Interoperability
C. Powerful, Robust, Scalable
D. Web Development Specific
E. Object Oriented (ειδικά η έκδοση 5)
F. Well Documented
    A. www.php.net/docs.php
G. Large active community
    A. 20 εκατομύρια websites
    B. Wordpress, Joomla, MediaWiki

Τι έχει το μενού σήμερα???

1. Τι είναι η PHP???
2. Ιστορικά Στοιχεία
3. Και γιατί όχι...???
4. Δυνατότητες PHP

Μπαμπά μπαμπά, τι κάνεις με την PHP???

- Δυναμικές σελίδες
- Διαχείριση χρηστών
- Μόνιμη αποθήκευση δεδομένων (server-side)
- Διαχείριση προσωπικού περιεχομένου
  - Προσωπικά μηνύματα
  - Φωτογραφίες που μπορούν να δουν μόνο «οι φίλοι μου»
- Ανέβασμα αρχείων
- Σύνδεση με βάση δεδομένων
- Πολλά, πολλά άλλα

Τι έχει το μενού σήμερα???

1. Τι είναι η PHP???
2. Ιστορικά Στοιχεία
3. Και γιατί όχι...???
4. Δυνατότητες PHP
5. ~~Στήσου μόνος σου... ΜΠΟΡΕΙΣ!!!!~~
   Στήστο μόνος σου...ΜΠΟΡΕΙΣ!!!!

Τι χρειαζόμαστε?

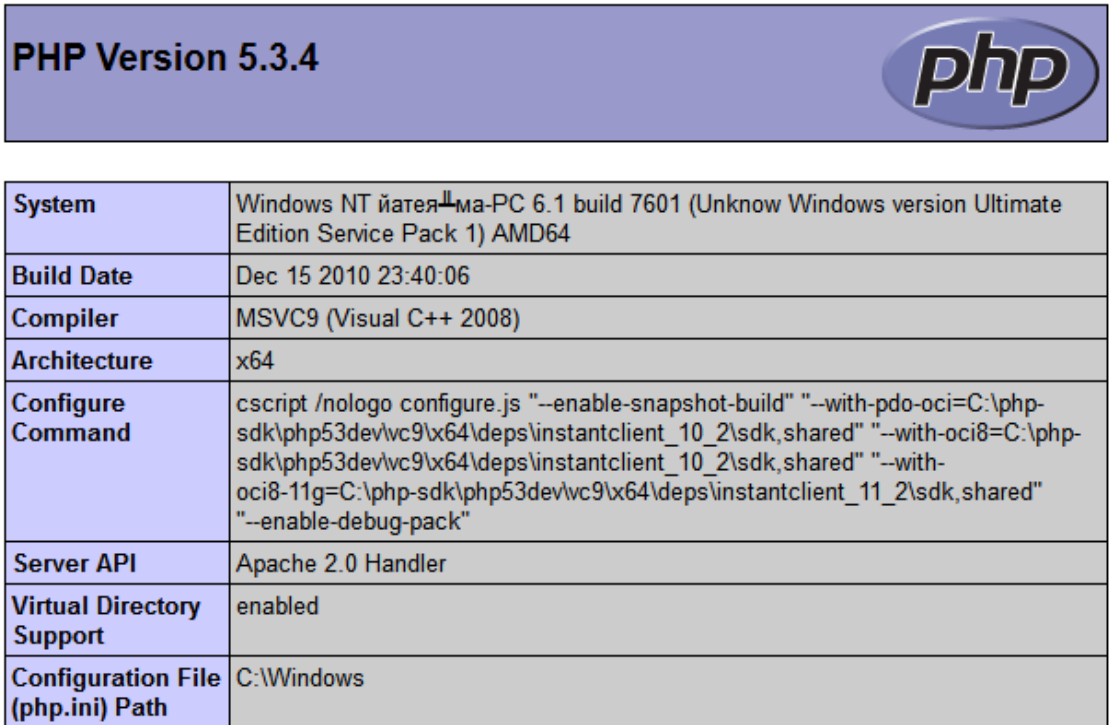
- Web Server
- PHP
- Database
- Text Editor
- Web Browser

‘Ολα αυτά σε ένα???

- WAMP (Windows Apache MySQL PHP)
- LAMP (Linux Apache MySQL PHP)
- MAMP (Macintosh Apache MySQL PHP)
- XAMPP (X Apache MySQL PHP Perl)

My First PHP File

```php
<?php
    phpinfo();
?>
```



PHP Version 5.3.4

System	Windows NT йатея╨ма-PC 6.1 build 7601 (Unknow Windows version Ultimate Edition Service Pack 1) AMD64
Build Date	Dec 15 2010 23:40:06
Compiler	MSVC9 (Visual C++ 2008)
Architecture	x64
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--with-pdo-oci=C:\php-sdk\php53dev\vc9\x64\deps\instantclient_10_2\sdk,shared" "--with-oci8=C:\php-sdk\php53dev\vc9\x64\deps\instantclient_10_2\sdk,shared" "--with-oci8-11g=C:\php-sdk\php53dev\vc9\x64\deps\instantclient_11_2\sdk,shared" "--enable-debug-pack"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\Windows

Παρακαλώ τι θα πάρετε?

1. Βασικές Οδηγίες
2. Μεταβλητές
3. Αλφαριθμητικά - Αριθμοί
4. Τελεστές
5. Συγκρίσεις
6. Βρόχοι επανάληψης
7. Συναρτήσεις
8. Χειρισμός φορμών

Τι έχετε φρέσκο?

1. Βασικές Οδηγίες

PHP Basics

- Τα PHP αρχεία έχουν κατάληξη .php
  - Ο χρήστης **δεν** έχει άμεση πρόσβαση στα αρχεία αυτά καθώς βρίσκονται στον server!

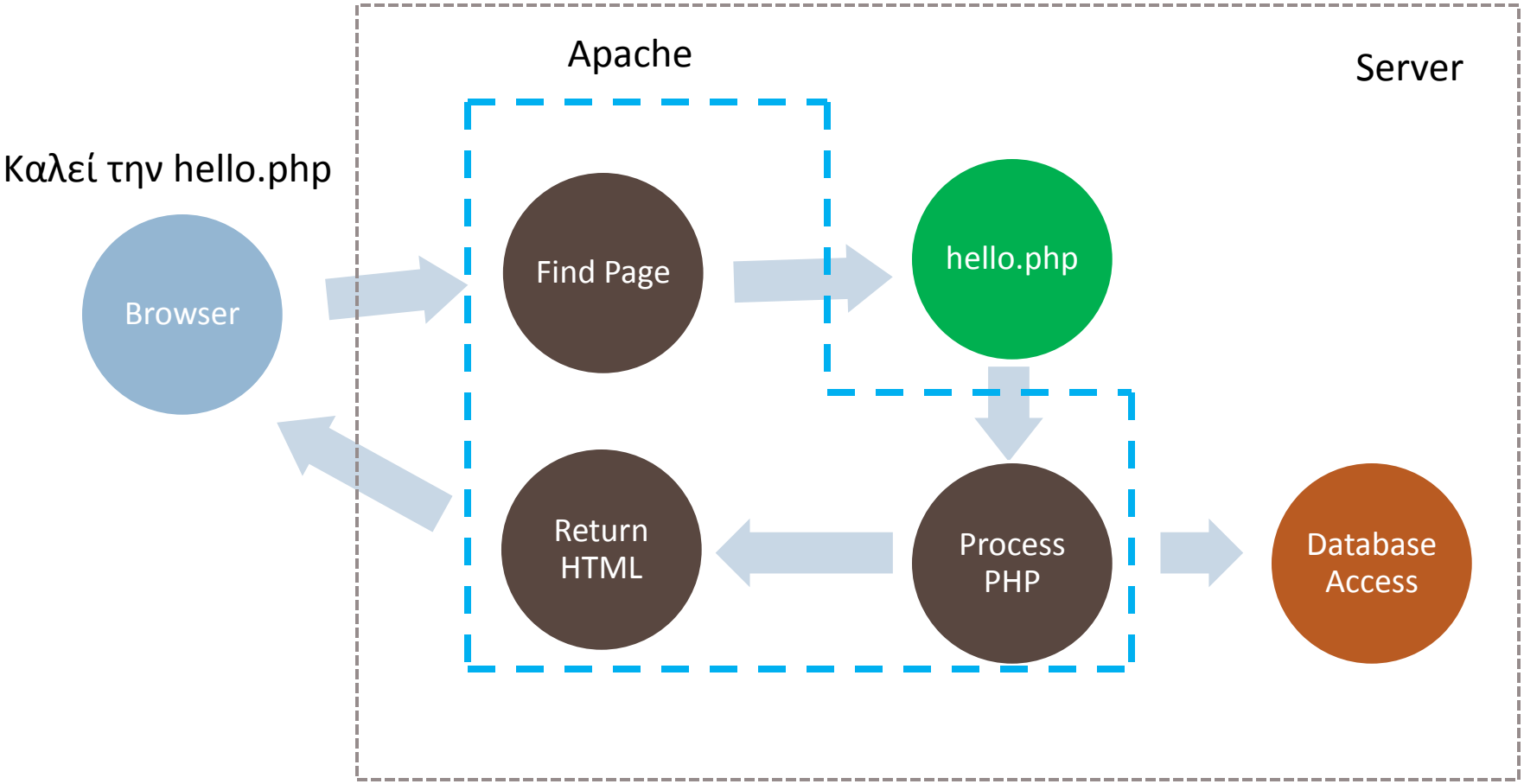
- Ο κώδικας php παρεμβάλλεται στον κώδικα της σελίδας μας.
  - Οι εντολές της php βρίσκονται ανάμεσα σε *<?php* και *?>*
  - Οι εντολές της php τελειώνουν με semicolon (;) ΠΑΝΤΑ!!!
  - Ο κώδικας εκτελείται σειριακά, η μία εντολή μετά την άλλη
  - Space Insensitive

Παράδειγμα --- Hello World

```
<html>
        <head>
                <title>Hello World!</title>
        </head>
        <body>
                <?php  echo "Hello World!"; ?>
        </body>
</html>
```

Hello World!

Διαδρομή Λειτουργιών



Echo - Print

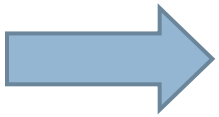
- Η εντολή echo μας δίνει την δυνατότητα να εκτυπώσουμε ένα αλφαριθμητικό στην σελίδα μας. Αυτό μπορεί να είναι από ένα απλό String έως μια μεταβλητή.

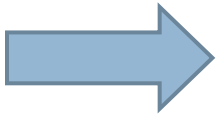
echo "Echo το κινητό σου, θα σι κανονίσω";

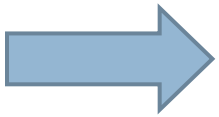
Echo το κινητό σου, θα σι κανονίσω!

Echo – Print (2)

Παραδείγματα:

print “Hello World!”;  Hello World!

echo “Hello” . “ World!”;  Hello World!

echo 5+7;  12

Σχολιασμός: Μανώλης Μαυρομάτης

- Σχόλια για μόνη γραμμή

```
// This is a single line comment
# or this…
```

- Σχόλια για πολλές γραμμές

```
/* This is a multi-line comment
   And is the second line
*/
```

- Τα σχόλια χρησιμοποιούνται για να βοηθήσουν αυτούς που διαβάζουν τον κώδικά μας (και εμάς τους ίδιους).

Ναι, κάτι άλλο?

1. Βασικές Οδηγίες 
2. Μεταβλητές

Μεταβλητές

- Μεταβλητή είναι η συμβολική ονομασία μιας τιμής, η οποία μπορεί να αλλάξει κατά το πέρασμα του χρόνου.

- Μεταβλητές στην PHP
  - Ξεκινούν με $.
  - Ακολουθεί γράμμα ή underscore (_).
  - Μπορεί να περιέχει γράμματα, αριθμούς, underscores (_) ή dashes (\ , /).
  - Δεν μπορεί να περιέχει κενά.
  - Είναι case sensitive (*$AntePAOKARA* <> *$antepaokara*).

Nasty Names

- *$My-name*
- *$___MyBooks*
- *$_Ahahouha*

Μεταβλητές (2)

- Η τιμή αποδίδεται σε μια μεταβλητή με χρήση του τελεστή = (ίσον).
- Παράδειγμα: *$var1 = 10;*
  - Δίνω στην μεταβλητή *$var1* την τιμή 10.

- Στις μεταβλητές δεν ορίζουμε τύπους.
  - Ο τύπος μιας μεταβλητής εξαρτάται από την τιμή της.
  - Εφόσον οι μεταβλητές μπορούν να αλλάξουν τιμή με το πέρας του χρόνου, αυτό σημαίνει ότι μπορούν να αλλάξουν δυναμικά και τύπο.

Βασικοί Τύποι Μεταβλητών

1. *int:* 11 , 256, -32, 0
2. *float:* 3.14, 5.55, 0.22

3. *String:* “Κούλα είναι κωλόπαιδο ο Κυριάκος”, “Σκίστεεεεεε!!!”
4. *boolean:* true, false

Παραδείγματος Χάρη (Κλυνν)

```
<?php
$var1 = “Hello “;
$var2 = “World!”;
echo $var1;
echo $var2;
?>
```

Hello World!

Παραδείγματος Χάριν (2)

```
<?php
$var1 = “Hello “;
echo $var1;
$var1 = 5;
echo $var1;
?>
```

int

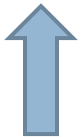
Hello 5

Δυναμικό Σύστημα Τύπων

- Μεταβλητές παίρνουν τύπο τιμής
- Διευκόλυνση στη συγγραφή κώδικα
- Σφάλματα **χρόνου εκτέλεσης** αντί συντακτικά
  - Πιο δύσκολα στον εντοπισμό
- Οι **μετατροπές** τύπων γίνονται **αυτόματα**

```
<?php
$var1 = “3“;
$var2 = 4;
echo $var1 + $var2;
?>
```

7



Μετατροπή σε ακέραιο για να ολοκληρωθεί η πράξη

Αυτό μας τελείωσε, κάτι άλλο?

1. Βασικές Οδηγίες ✓
2. Μεταβλητές ✓
3. Αλφαριθμητικά - Αριθμοί

String Variables (Αλφαριθμητικά)

- Κάθε μεταβλητή είναι **ένα κείμενο**
- Το μήκος μπορεί να αλλάζει
- Δεν υπάρχει περιορισμός μήκους
- Δεν πρόκειται για πίνακες από χαρακτήρες
- Δεν υπάρχει διαφορετικός τύπος χαρακτήρα και αλφαριθμητικού

Συνένωση Αλφαριθμητικών

- Αυτό γίνεται πάρα πολύ εύκολα με την τελεία (.)

```
<?php
    echo “Κάποιος σ’αγαπάει,” . “ είμαι εγώ”
?>
```

Κάποιος σ’αγαπάει, είμαι εγώ

- Μπορεί να γίνει και με κείμενο και μεταβλητή

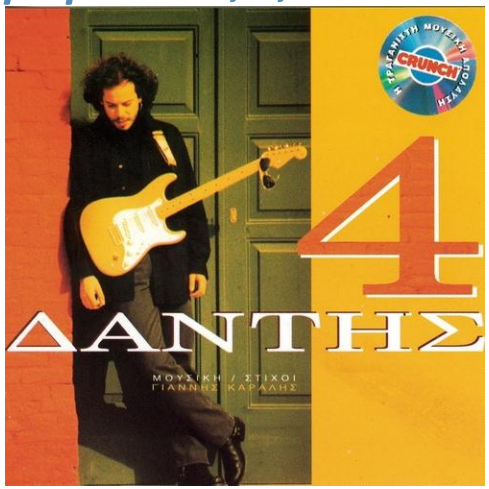
```
<?php
    $var1 = “ είμαι εγώ”
    echo “Κάποιος σε ζητάει,” .  $var1
?>
```

Κάποιος σε ζητάει, είμαι εγώ

Συνένωση Αλφαριθμητικών (2)

- Είτε με δύο μεταβλητές

```
<?php
   $var1 = “καποιος ξενυχτάει και σε περιμένει,”;
   $var2 = “ λεπτό προς λεπτό”;
   echo $var1 .  $var2;
?>
```



Κάποιος ξενυχτάει και σε περιμένει, λεπτό προς λεπτό

Διπλά Αυτάκια Vs. Μονά Αυτάκια

- **Διπλά εισαγωγικά: Προκαλούν αντικαταστάσεις**

```
<?php
        $a = “Pikatsu”;
        echo “Καλώ εσένα, $a!”;  ⟹ Καλώ εσένα, Pikatsu
        // ακόμα καλύτερα “Καλώ εσένα, {$a}!”
?>
```

- **Μονά εισαγωγικά: ΔΕΝ Προκαλούν αντικαταστάσεις**

```
<?php
        $a = “Pikatsu”;
        echo ‘Καλώ εσένα, $a!’;  ⟹ Καλώ εσένα, $a!
?>
```

Χρήσιμες Συναρτήσεις Αλφαριθμητικών

1. ***strlen()***: Υπολογίζει μήκος αλφαριθμητικού
2. ***trim()***: «Κόβει» τα κενά από την αρχή και το τέλος
3. ***strstr()***: Εντοπίζει ένα αλφαριθμητικό μέσα σε άλλο
4. ***strreplace()***: Αντικαθιστά ένα αλφαριθμητικό με ένα άλλο
5. ***strtolower()***: Μετατρέπει από κεφαλαία σε πεζά
6. ***strtoupper()***: Μετατρέπει από πεζά σε κεφαλαία

Μπορείτε να βρείτε και πολλά άλλα από τα docs.

Μερικές Χρήσιμες Συναρτήσεις Δεκαδικών

1. ***round()***: Στρογγυλοποιεί δεκαδικά
2. ***ceil()***: Βρίσκει το ceiling ενός αριθμού
3. ***floor()***: Βρίσκει το floor ενός αριθμού
4. ***abs()***: Βρίσκει την απόλυτη τιμή
5. ***pow()***: Υψώνει σε δύναμη
6. ***sqrt()***: Βρίσκει την τετραγωνική ρίζα
7. ***rand()***: Δημιουργεί τυχαίους αριθμούς
8. ***fmod()***: Βρίσκει το υπόλοιπο διαίρεσης

Ειδικοί Χαρακτήρες

- Δουλεύουν μόνο σε “διπλά εισαγωγικά”

- Παρόμοιοι με C, C++, Java, …
  - ***\n*** → Αλλαγή γραμμής
  - ***\t*** → Tab

Να βάλω και μια ποικιλία για τη μέση?

1. Βασικές Οδηγίες ✓
2. Μεταβλητές ✓
3. Αλφαριθμητικά - Αριθμοί ✓
4. Τελεστές

Τελεστές

Τελεστές	Λειτουργία
+, -, *, /, %	Αριθμητικές πράξεις
||, &&, !	Λογικές πράξεις: ή, και, όχι
==, !=, <, >, <=, >=, ===, !==	Σύγκριση
++, --	Αύξηση, μείωση
.	Ένωση αλφαριθμητικών
=, +=, -=, *=, /=, %=, .=	Ανάθεση τιμής

Τελεστές (2)

Κώδικας	Αποτέλεσμα
3 + 5	**8**
2 - 9	**-7**
1 / 2	**0.5**
*5 * 7*	**35**
102 % 5	**2**
true || false	**true**
!true	**false**
true && true	**true**
3 < 5	**true**
3 != 3	**false**
“Hello “ . ‘, world!’	**Hello, world!**

Παραδείγματα Τελεστών

echo "1" / "2";	0.5
echo 1 . 2;	12
$a = 5; echo "$a";	5
$b = 5; echo "$b" - $b;	0
$c = 5; echo '$c' . $c;	$c5
$a = 3; echo ++$a;	4
$a = 3; echo $a--;	3

$b = "Hello"; $b .= "there!!"; echo $b;

Hello there!!

Τελεστές Ισότητας

- Χρησιμοποιούνται τα ==, !=, ===, !==
- Οι τύποι των μεταβλητών μετατρέπονται για να γίνει σωστά η σύγκριση στην περίπτωση των δύο πρώτων.
  - ***1 == 1*** True
  - ***1 == 2*** False
  - ***0 != "hello"*** True
  - ***"1" == 1*** True
  - ***"1" === 1*** False

Να φέρω μια ή δύο να φτάσουν για όλους?

1. Βασικές Οδηγίες
2. Μεταβλητές
3. Αλφαριθμητικά - Αριθμοί
4. Τελεστές
5. Συγκρίσεις

Εντολή if

```
if ( συνθήκη 1) {
    // εντολές
}
else if (συνθήκη 2) {
   // εντολές
}
…
else {
   // εντολές
}
```

Αν ισχύει η συνθήκη 1

εκτελείται αυτό το σώμα

Αν ισχύει η συνθήκη 2

εκτελείται αυτό το σώμα

…

Αν δεν ισχύει καμία συνθήκη

εκτελείται το τελευταίο σώμα

Παραδείγματα

```
<?php
if ( $a == 5) {
        echo “Five of a kind!”;
}
else {
        echo “Can say if stupid or just trolling!”;
}
?>
```

Παραδείγματα (2)

```
<?php
$a = “Justin Bieber”
if ( $a == “Justin Bieber”) {
        echo “That’s my girl!”;
}
else {
        echo “Bitch please!”;
}
?>
```

Εντολή switch

```
switch ( παράσταση ) {
   case τιμή1:
      σώμα 1;
      break;
   case τιμή2:
      σώμα 2;
      break;
   ...
   default:
     default σώμα
}
```

Αν ισχύει η συνθήκη 1

εκτελείται το σώμα1

Αν ισχύει η συνθήκη 2

εκτελείται το σώμα2

Αν δεν ισχύει καμία συνθήκη

εκτελείται το τελευταίο σώμα

Παράδειγμα

```
<?php
switch ( $day ){
    case “Δευτέρα”:
        echo “Κάτι έχω”;
        break;
    case “Τρίτη”:
        echo “Δεν αντέχω”;
        break;
    case “Τετάρτη”:
        echo “Πως βαριέμαι”;
        break;
…
}
```

Παράδειγμα (2)

```
<?php
$day = “Δευτέρα”;
switch ( $day){
   case “Δευτέρα”:
      echo “Ξέχασες το Break!”;
   case “Τρίτη”:
      echo “Σε σένα μιλάω!”;
   case “Τετάρτη”:
      echo “Είσαι βλακάκος?”;
  default:
      echo “Ό,τι να ’ναι είσαι…”;
}
```

ΠΡΟΣΟΧΗ!!!

Αν δεν βάλουμε το break μετά από κάθε σώμα θα συνεχιστεί η εκτέλεση των εντολών μέχρι το επόμενο break.

Ξέχασες το Break! Σε σένα μιλάω! Είσαι βλακάκος? Ό,τι να ’ναι είσαι…

Από βρόχους τι θα πάρετε?

1. Βασικές Οδηγίες
2. Μεταβλητές
3. Αλφαριθμητικά - Αριθμοί
4. Τελεστές
5. Συγκρίσεις
6. Βρόχοι επανάληψης

Εντολή for

```
for ( αρχική συνθήκη; συνθήκη τερματισμού; Βήμα)
{
        // σώμα εντολών
}
```

1. Αρχικοποιούμε μια μεταβλητή.
2. Εκτελείται το σώμα εντολών του βρόχου.
3. Αλλάζουμε την τιμή της μεταβλητής σύμφωνα με το βήμα που έχουμε.
4. Ελέγχουμε αν ισχύει η συνθήκη τερματισμού
   - Αν ισχύει σταματάμε τις επαναλήψεις.
   - Αν όχι τότε συνεχίζουμε τις επαναλήψεις.

Παράδειγμα

```
<?php
for ($i = 1; $i < 6; $i++){
        echo “High-$i!”;
}
```

High-1!High-2!High-3!High-4!High-5!

Εντολή while

```
while (συνθήκη) {
        // σώμα εντολών
}
```

1. Αρχικά ελέγχεται η συνθήκη.
   - Αν ισχύει τότε εκτελείται το σώμα εντολών του βρόχου.
   - Σε αντίθετη περίπτωση σταματάει η εκτέλεση του βρόχου
2. Ελέγχεται εκ νέου η συνθήκη τερματισμού κτλ

- Υπάρχει περίπτωση το σώμα του βρόχου να μην εκτελεστεί ούτε μια φορά.

Παράδειγμα

```
<?php
$i = 1;
while ($i < 6) {
        echo “High-$i!”;
        $i++;
}
```

High-1!High-2!High-3!High-4!High-5!

Εντολή do…while

```
do {
        // σώμα εντολών
} while (συνθήκη)
```

1. Ακριβώς ίδια με την while.
2. Το σώμα του βρόχου θα εκτελεστεί οπωσδήποτε μια φορά, είτε είναι λανθασμένη η συνθήκη είτε όχι, γιατί ο έλεγχος γίνεται κατόπιν (εορτής).

Παράδειγμα

```
<?php
$i = 5;
do {
      echo “Hi-$i!”;
} while ($i > 6)
```

Hi-5!

Εντολές break - continue

- *break*
  - Εμφανίζεται μέσα σε βρόχους επανάληψης (for, while etc) ή σώμα συγκρίσεων (switch).
  - Διακόπτει την ροή που βρισκόμαστε και συνεχίζει με την επόμενη αμέσως εντολή.
  - Δεν γίνονται άλλες επαναλήψεις.
- *continue*
  - Εμφανίζεται μέσα σε ροή επανάληψης (do…while etc).
  - Διακόπτει την ροή των εντολών πηγαίνοντας στον έλεγχο της συνθήκης τερματισμού.
  - Μπορεί να γίνουν και άλλες επαναλήψεις μετά από το continue.

Αυτά, κάτι άλλο?

1. Βασικές Οδηγίες
2. Μεταβλητές
3. Αλφαριθμητικά - Αριθμοί
4. Τελεστές
5. Συγκρίσεις
6. Βρόχοι επανάληψης
7. Συναρτήσεις

Συναρτήσεις

- Οι συναρτήσεις είναι υπό-ρουτίνες που φέρουν σε πέρας μια συγκεκριμένη εργασία.
- Ορίζονται με την λέξη-κλειδί function στην αρχή της γραμμής.
- Στην συνέχεια ακολουθεί το όνομα της συνάρτησης και μέσα σε παρένθεση τα ορίσματα που παίρνει αυτή, χωρισμένα με κόμμα (,).

```
function όνομα-συνάρτησης ( ορίσματα ){
        // σώμα εντολών
}
```

Ορίσματα Συναρτήσεων

- **Δίνουν** πληροφορίες σε μία συνάρτηση
- Ακολουθούν ίδια ονοματολογία με μεταβλητές
- Αρχίζουν με ***$***, ακολουθεί το **όνομα**
- Το όνομα...
  - Αρχίζει με γράμμα ή _
  - Περιέχει γράμματα, αριθμούς, _
  - Έχει ευαισθησία σε πεζά-κεφαλαία

Εντολή επιστροφής (return)

- Η εντολή *return* επιστρέφει πίσω στο κυρίως πρόγραμμα μια τιμή που έχει προκύψει κατά την ροή της συνάρτησης.
- Η τιμή αυτή λαμβάνεται στο σημείο που έγινε η κλήση της συνάρτησης.
- Η *return* πρέπει να είναι η τελευταία εντολή στο σώμα εντολών γιατί μετά σταματάει η εκτέλεση της συνάρτησης.
- Δεν ορίζεται τύπος επιστροφής, όπως στις άλλες γλώσσες προγραμματισμού.
- Δεν είναι υποχρεωτική.

Κλήση Συνάρτησης

- Για να καλέσουμε μια συνάρτηση χρησιμοποιούμε το όνομα της.
- Στην συνέχεια, ορίζουμε μέσα σε παρένθεση τις τιμές που θα πάρουν τα ορίσματα της.
- Έχει σημασία η σειρά των ορισμάτων
  - Πρώτο όρισμα → πρώτη τιμή
  - Δεύτερο όρισμα → δεύτερη τιμή…

Κλήση χωρίς επιστροφή

όνομα-συνάρτησης (ορίσματα)

Κλήση με επιστροφή

$a = όνομα-συνάρτησης (ορίσματα)

Ορισμός Συναρτήσεων

```
function multiply( $a , $b ){
        $c = $a * $b;
        return $c;
}
```

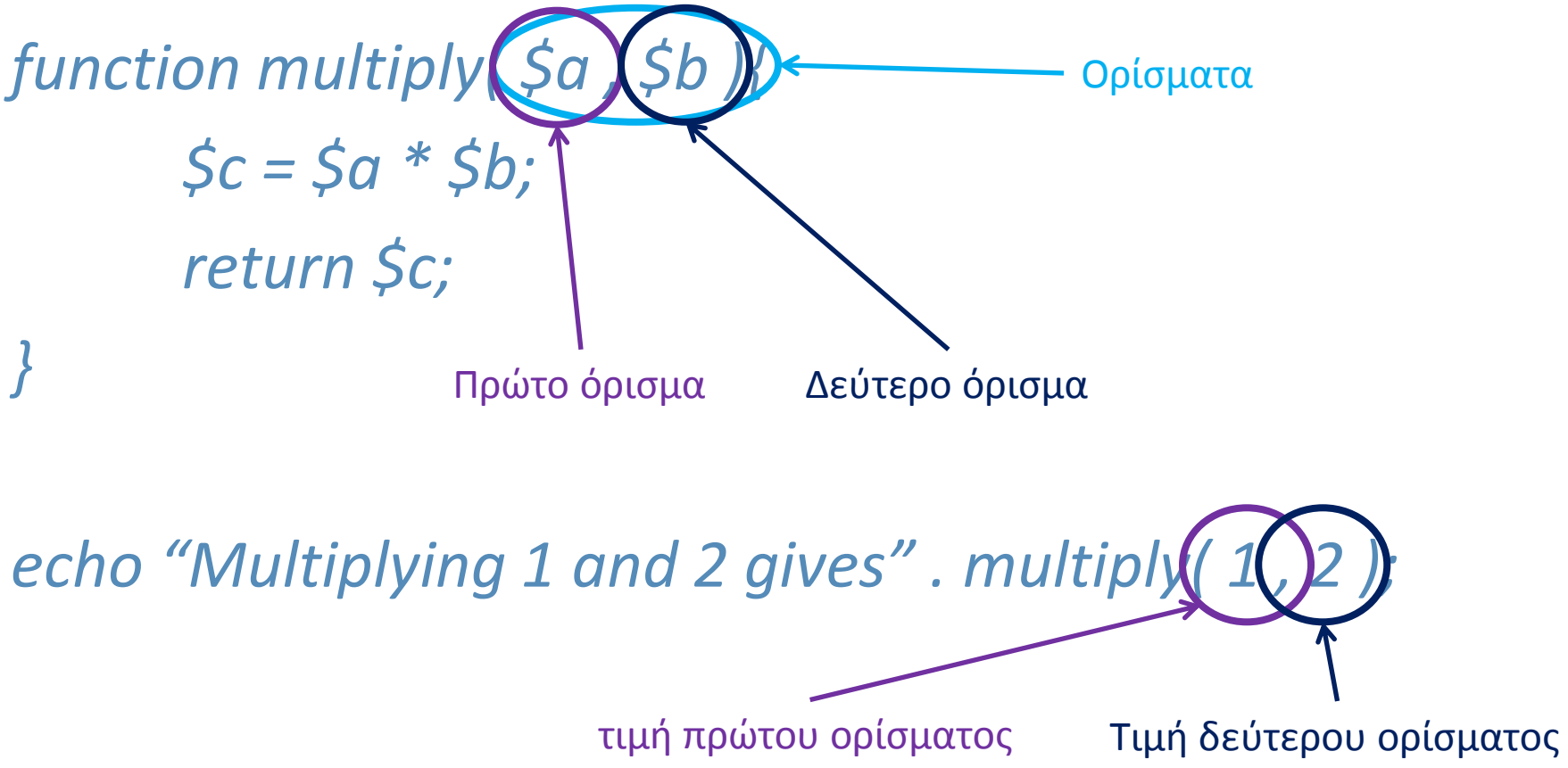
Όνομα συνάρτησης

ορισμός συνάρτησης

```
echo “Multiplying 1 and 2 gives” . multiply( 1 , 2 );
```

Κλήση συνάρτησης

Ορισμός Συναρτήσεων (2)



Ορισμός Συναρτήσεων (3)

```
function multiply( $a , $b ){
        $c = $a * $b;
        return $c;
}
```

Επιστροφή

```
echo "Multiplying 1 and 2 gives" . multiply( 1 , 2 );
```

μπαίνει στην θέση του

Multiplying 1 and 2 gives 2

Παράδειγμα

```
function average( $a, $b ) {
    $c = $a + $b;
    return $c / 2;
}

echo 'The average of 3, 5: ' . average( 3, 5 );
echo "\n";
echo 'The average of 1, 9: ' . average( 1, 9 );
```

Παραδείγμα (2)

```
function choose($a){
        if ($a == “1”){
                return “Pikachu”;
        }
        else {
                return “Καμπαμαρού”;
        }
}

echo choose(3);
```

Προαιρετικά Ορίσματα

- Μπορούμε αν θέλουμε κάποια από τα ορίσματα μιας συνάρτησης να τα θέσουμε κατά τον ορισμό της.
  - Τα προαιρετικά ορίσματα πρέπει να είναι τα τελευταία σε σειρά.
  - Μπορούν να είναι όσα θέλουμε σε αριθμό.
  - Πρέπει να ορίσουμε με ίσον (=) την τιμή που θα έχουν ως προεπιλεγμένη.

Παράδειγμα

```
function play( $title, $band = “Mazoo and the Zoo”){
        $str = “I am playing “;
        $str .= $title . “ by ” . $band;
        return $str;
}

echo play(“Η Αρκούδα”);
```

Προαιρετικό όρισμα

Αν είχαμε όμως

```
function play( $title = “Η Αρκούδα”  , $band )
```

στον ορισμό θα ήταν λάθος.

Παράδειγμα

```
function play
        ( $title  = “Careless Whisper”, $band = “George Michael”){
            $str = “I am playing “;
            $str .= $title . “ by ” . $band;
            return $str;
}

echo play();
```

Όλα είναι προαιρετικά ορίσματα

Θα πιείτε κάτι?

1. Βασικές Οδηγίες
2. Μεταβλητές
3. Αλφαριθμητικά - Αριθμοί
4. Τελεστές
5. Συγκρίσεις
6. Βρόχοι επανάληψης
7. Συναρτήσεις
8. Χειρισμός Φορμών

Χειρισμός Δεδομένων Φόρμας

- Για να πάρουμε δεδομένα χρησιμοποιούμε δύο μεθόδους:
  - HTTP GET

    Μεταβλητή ***$_GET***

    $_GET[όνομα παραμέτρου]

  - HTTP POST

    Μεταβλητή ***$_POST***

    $_POST[όνομα παραμέτρου]

- Οι μεταβλητές αυτές ορίζονται αυτόματα από την PHP

Παράδειγμα

test.html

```
<form action="test.php" method="post">
        <input type="text" name="foo" />
        <input type="submit" value="Πάρε" />
</form>
```

Παράδειγμα

test.php

```
<p>
        Δώσε
        <?php
                echo $_POST[ 'foo' ];
        ?>!
</p>
```

Εφημερίδες! Έκτακτο Παράρτημα!

1. Booleans – Null (κενό)
2. Πίνακες
3. Εντολή foreach
4. Διαχωρισμός σε αρχεία
5. Εμβέλεια Μεταβλητών
6. Συναρτήσεις Αρχείων
7. Διαχείριση Αρχείων
8. Headers (Κεφαλίδες)

Πρώτη Είδηση!

1. Booleans – Null (κενό)

Booleans

- Οι μεταβλητές τύπου boolean παίρνουν δύο τιμές:
  1. *true (1)*
  2. *false (0)*

- Στην εκτύπωση των τιμών αυτών από την PHP αυτό που βλέπουμε στην οθόνη ειναι τα αριθμητικά τους ισοδύναμα

  ...ή μήπως όχι???

Παράδειγμα

```
<?php
$var1 = true;
$var2 = false;
echo “var1: “ . $var1;
echo “var2: “ . $var2;
?>
```

var1: 1

var2: (???????)

Τι μαγικό ήταν αυτό???

- Στην περίπτωση που εχουμε *false*, δεν εμφανίζεται τίποτα στην οθόνη, ούτε *false* ούτε *0*.

- Οπότε δεν τρομάζουμε όταν συμβαίνει αυτό, είναι φυσιολογικό.

NULL (Κενό)

- Η λέξη *NULL* υποδηλώνει μια μεταβλητή που δεν περιέχει μέσα της καμία τιμή ούτε είναι κάποιου συγκεκριμένου τύπου, είναι απλά ορισμένη ως όνομα.

- Οταν εκτυπώνουμε μεταβλητές που έχουν τιμή *NULL* αυτές επιστρέφουν κένο, όπως ακριβώς και οι μεταβλητές boolean με τιμή *false*.

Χρήσιμες Συναρτήσεις Μεταβλητών Boolean

- *isset(όνομα μεταβλητής)*
  - Συνάρτηση που δείχνει αν μια μεταβλητή υπάρχει.
  - Επιστρέφει *true/false*.

```
<?php
$lyke = 1;
echo “Λύκε, Λύκε είσαι εδώ?  ” . isset($lyke);
?>
```

Λύκε, Λύκε είσαι εδώ? 1

Χρήσιμες Συναρτήσεις Μεταβλητών Boolean (2)

- *unset(όνομα μεταβλητής)*
  - Συνάρτηση που διαγράφει τα περιεχόμενα μιας μεταβλητής.
  - Κάνει την τιμή της μεταβλητής ίση με *NULL*.

```
<?php
$me = “ΕΓΩ!!!!!”;
unset ($me);
echo “Ποιος είναι ο καλύτερος στην PHP??” . $me;
?>
```

Ποιος είναι ο καλύτερος στην PHP??

Χρήσιμες Συναρτήσεις Μεταβλητών Boolean (3)

- *empty(όνομα μεταβλητής)*
  - Ελέγχει αν μια μεταβλητή είναι άδεια.
  - Επιστρέφει *true/false*.

- Ως άδεια νοείται μια μεταβλητή
  - με τιμή *NULL* ή *false*.
  - με τιμή *0*.
  - με τιμή “” ή “*0*”.
  - πίνακας χωρίς περιεχόμενα.
  - μεταβλητή ορισμένη χωρίς τιμή (*$var1*).

Και μετά, και μετά??

1. Booleans – Null (κενό) 
2. Πίνακες

Πίνακες

- Ο πίνακας είναι τύπος δεδομένων που μπορεί να αποθηκεύσει περισσότερες από μια τιμές.

- Οι τιμές που παίρνει κάθε θέση του πίνακα μπορεί να είναι οτιδήποτε εμείς θέλουμε.
  - Αριθμός.
  - Αλφαριθμητικό.
  - Boolean.
  - ακόμα και άλλος πίνακας.

Κλειδιά - Αγκύλες

- Οι θέσεις ενός πίνακα ορίζονται με κλειδιά, τα οποία μπορεί να είναι
  1. Ακέραιοι αριθμοί όπως στους πίνακες του αντικειμενοστρεφούς προγραμματισμού.
  2. *Labels* (ετικέτες) που ορίζουμε εμείς.

- Με τις αγκύλες ορίζουμε την θέση του πίνακα που θέλουμε να πάρουμε ή να ορίσουμε την τιμή του
  - *$array1[0] = 10; $array2[“Coyote”] = “HELP!!!”*
  - *echo $array1[0] . $array2 [“Coyote”]*

Ορισμός Πίνακα

- Ένας πίνακας ορίζεται με την εντολή *array()* και τις τιμές χωρισμένες με κόμμα (,)
  - Σε αυτή τη περίπτωση έχουμε δείκτες-κλειδιά ακέραιους (0, 1, 2 κτλ).

```
<?php
$array1 = array(2, " good to be ", true);
echo $array1[0] . $array1[1] . $array1[2];
echo $array1;
?>
```

2 good to be 1
array

Ορισμός Πίνακα (2)

- Στην περίπτωση των *labels* έχουμε

```
<?php
$array1 = array(“Babis” => “2 be” ,
                “Soulara” =>“ or not 2 be”);
echo $array1[“Babis”] . $array1[“Soulara”];
?>
```

2 be or not 2 be

Ορισμός Πολυδιάστατου Πίνακα

- Στην PHP αυτό επιτυγχάνεται βάζοντας πίνακα μέσα σε πίνακα

```
<?php
$array1 = array(array(9,10,11), array(4,5,6));
echo $array1[0][0] . " to " . $array1[1][1];
?>
```

9 to 5

Διαγραφή από Πίνακα

- Η διαγραφή ενός στοιχείου πίνακα γίνεται με την εντολή unset.
  - *unset $array1[0];*

- ΠΡΟΣΟΧΗ!!!! Μετά την διαγραφή του στοιχείου, τα υπόλοιπα στοιχεία μένουν στις θέσεις που είχαν, δεν μεταφέρονται δηλαδή προς τα αριστερά.

Παράδειγμα

```
<?php
$array1 = array(“I”, “am”, “bad”, “ass”)
echo “$array1[0] $array1[1] $array1[2] $array1[3] \n”;
unset $array1[2];
echo “$array1[0] $array1[1] $array1[2] $array1[3]”;
?>
```

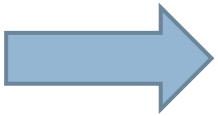
I am bad ass

I am ass

Συναρτήσεις Πινάκων

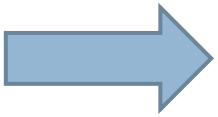
- *print_r()*: εμφανίζει τα περιεχόμενα ενός πίνακα.
  - Παίρνει σαν όρισμα την μεταβλητή – πίνακα.
  - Συνδυάζεται με πολύ καλά με την *<pre></pre>*.

```
<?php
$array1 = array(1,2,3);
print_r($array1)
?>
```



```
Array([0] => 1 , [1]=>2 , [2]=>3)
```

```
<pre>
    <?php print_r($array1)
    ?>
</pre>
```

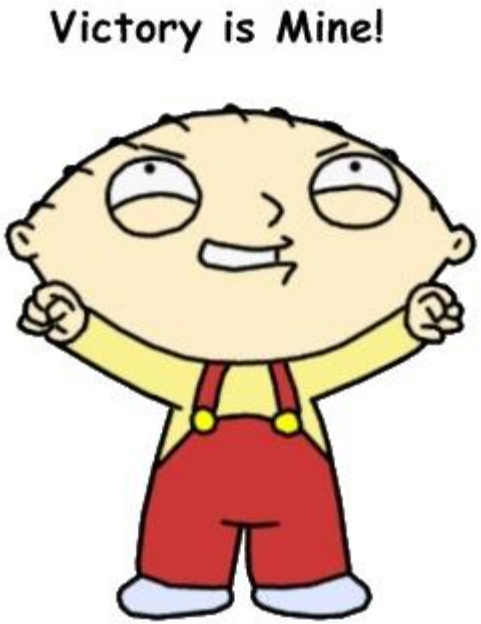


```
Array
(
    [0] => 1 ,
    [1] => 2 ,
    [2] => 3
)
```

Συναρτήσεις Πινάκων (2)

- *count():* επιστρέφει τον αριθμό των στοιχείων
- *max():* επιστρέφει το μέγιστο στοιχείο
- *min():* επιστρέφει το ελάχιστο στοιχείο
- *sort():* ταξινομεί τα στοιχεία του πίνακα κατά αύξουσα σειρά
- *r_sort():* ταξινομεί τα στοιχεία του πίνακα κατά φθίνουσα σειρά
- *in_array():* βρίσκει αν μια τιμή υπάρχει στον πίνακα. Επιστρέφει *true/false*.

Συναρτήσεις Πινάκων (3)



- *implode():* μετατρέπει έναν πίνακα σε string

```
$array1 = array(“Damn” , “you” , “vile” , “woman!”);
echo $string = implode(“ ” , $array1);
```

Damn you vile woman!

- *explode():* μετατρέπει ένα string σε πίνακα

```
$string = “Victory is mine”;
$array1 = explode(“ ” , $string);
<pre><?php print_r($array1)?>
</pre>
```

Και πολλές άλλες!

```
Array(
    [0] => “Victory” ,
    [1] => “is” ,
    [2] => ”mine”
)
```

Συνέχεια στην σελίδα 3…

1. Booleans – Null (κενό) ✓
2. Πίνακες ✓
3. Εντολή foreach

Εντολή foreach

- Η εντολή *foreach* χρησιμοποιείται πάντα με κάποιον πίνακα.

- Με χρήση αυτής, μπορούμε να προσπελάσουμε όλες τις θέσεις ενός πίνακα και να εκτελέσουμε τον κώδικα που περιλαμβάνεται στο σώμα της συνάρτησης για κάθε μια από τις τιμές του.

- Σύνταξη

```
foreach ( $array as $value) {
    // σώμα εντολών
}
```

όπου η μεταβλητή *$value* περιέχει τις τιμές του πίνακα.

Εντολή foreach (2)

- Εναλλακτική σύνταξη

```
foreach ( $array as $key => $value) {
    // σώμα εντολών
}
```

  - *$key* έχει τα κλειδί του τρέχοντος στοιχείου.
  - *$value* περιέχει τις τιμή του.

- Αν αλλάξει η τιμή του *$value* μέσα στο σώμα, δεν αλλάζουν οι τιμές που έχει ο πίνακας.

Παράδειγμα

```
$heroes = array(
        “Batman” => “AWESOME” ,
        “Robin” => “gay”
);
foreach ( $heroes as $hero => $power ) {
        echo “ $hero is $power!! \n”;
}
```

Batman is AWESOME!!

Robin is gay!!

foreach Πολυδιάστατων Πινάκων

```
$matrix = array(
        array( 1, 2, 3 ),
        array( 4, 5, 6 ),
        array( 7, 8, 9 ),
);
foreach ( $matrix as $row ) {
        foreach ( $row as $element ) {
                echo "$element ";
        }
        echo "\n";
}
```

Πολιτική

1. Booleans – Null (κενό) ✓
2. Πίνακες ✓
3. Εντολή foreach ✓
4. Διαχωρισμός σε Αρχεία

Διαχωρισμός σε Αρχεία

- Η PHP μας δίνει την πολύ χρήσιμη δυνατότητα να φορτώσουμε τον κώδικά ενός αρχείου σε ένα άλλο αρχείο.

- Με αυτόν τον τρόπο καταφέρνουμε να
  - Έχουμε καλύτερα οργανωμένο τον κώδικά μας
  - Επαναχρησιμοποιούμε κώδικα κατά βούληση

- Συνήθως χρησιμοποιείται για να
  - φορτώσουμε βιβλιοθήκες συναρτήσεων
  - φορτώσουμε header (κεφαλίδα) και footer (υποσέλιδο) στην σελίδα μας

Εντολή include

- Με την εντολή *include* φορτώνουμε εξωτερικά αρχεία στον κώδικά μας.
  - *include 'library.php'*

- Με αυτόν τον τρόπο τρέχουμε τον κώδικα του library.php
  - Ειναι το ίδιο με το να κάναμε copy/paste τον κώδικα που περιλαμβάνει το αρχείο που κάναμε *include*.

- Αν το αρχείο δεν βρεθεί τότε παράγει ένα warning και συνεχίζει.

Εντολή require

- Η εντολή require είναι παρόμοια σε χρησιμότητα με την include. Συντάσσεται ακρίβως το ίδιο.
  - *require 'library.php'*

- Η διαφορά των δύο εντολών έγκειται στο ότι με το require σε περίπτωση που δεν βρεθεί το αρχείο έχουμε error και όχι warning, καθώς και διακοπή της φόρτωσης της σελίδας.

include_once – require_once

- Σύνταξη
  - *include_once library.php;*
  - *require_once library.php;*

- Οι εντόλές αυτές κάνουν το ίδιο πράγμα με τις αντίστοιχες που αναφέραμε προηγουμένως με μόνη διαφορά ότι τρέχουν τον κώδικα μόνο την πρώτη φορά που φορτώνει ένα αρχείο.

- Πολύ χρήσιμο σε περίπτωση που έχουμε πολλά αρχεία που μπορεί το ένα να κάνει *include* το άλλο.

Μικρές Αγγελίες

1. Booleans – Null (κενό) ✓
2. Πίνακες ✓
3. Εντολή foreach ✓
4. Διαχωρισμός σε Αρχεία ✓
5. Συναρτήσεις Αρχείων

Συνάρτηση *opendir()*

- Συνάρτηση *opendir(όνομα φακέλου)*
  - Σαν όρισμα βάζουμε απλά το όνομα του φακέλου αν είναι μέσα στο ίδιο φάκελο με το αρχείο μας
  - Αλλίως βάζουμε το path (σχετικό ή απόλυτο)

- Το αποτέλεσμα
  - Επιστρέφει ένα αντικείμενο τύπου φάκελος αρχείων
  - Επιστρέφει *false* αν δεν υπάρχει ο φάκελος
  - Χρησιμοποιείται από άλλες συναρτήσεις, που θα δούμε παρακάτω

Συνάρτηση *readdir()*

- Συνάρτηση *readdir()*
  - Δέχεται σαν όρισμα το αντικείμενο που επιστρέφει η *opendir()*.
- Το αποτέλεσμα
  - Επιστρέφει το επόμενο αρχείο μέσα στον φάκελο που επιλέξαμε.
  - Επιστρέφει *false* αν αποτύχει να επιστρέψει αρχείο.

Συνάρτηση *closedir()*

- Συνάρτηση *closedir()*
  - Δέχεται σαν όρισμα το αντικείμενο που επιστρέφει η *opendir()*.
- Το αποτέλεσμα
  - Κλείνει τον φάκελο που επιλέξαμε με την *opendir()* όταν τελειώσουμε τις εργασίες μας με αυτόν.
  - Επιστρέφει *false* αν αποτύχει να κλείσει τον φάκελο.

Συναρτήσεις Αρχείων

- Συνάρτηση *file_get_contents()*
  - Δέχεται σαν όρισμα ένα αντικείμενο τύπου αρχείο που επιστρέφει η *readdir()* ή ένα link σε αρχείο (απόλυτο ή σχετικό).
  - Επιστρέφει τα περιεχόμενα του αρχείου
  - Επιστρέφει *false* αν αποτύχει να διαβάσει το αρχείο.

- Συνάρτηση *file_put_contents()*
  - Δέχεται σαν όρισμα ένα αντικείμενο τύπου αρχείο που επιστρέφει η *readdir()* και τα περιεχόμενα που θέλουμε να του δώσουμε.
  - Αποθηκεύει τα περιεχόμενα στο αρχείο και σβήνει τα προηγούμενα.
  - Επιστρέφει *false* αν αποτύχει να διαβάσει το αρχείο.

Συναρτήσεις Αρχείων (2)

- Άλλες χρήσιμες συναρτήσεις
  - *fopen()*
  - *fclose()*
  - *fwrite()*
  - *fget()*

  και άλλες πολλές.

  http://gr.php.net/manual/en/ref.filesystem.php

Αναγγελίες Γάμων

1. Booleans – Null (κενό) ✓
2. Πίνακες ✓
3. Εντολή foreach ✓
4. Διαχωρισμός σε Αρχεία ✓
5. Συναρτήσεις Αρχείων ✓
6. Διαχείρηση Αρχείων

Ανέβασμα Αρχείων

- Τα αρχεία στέλνονται στον server με αίτημα *POST*.
- Αποθηκεύονται προσωρινά στην μνήμη όσο τρέχει το πρόγραμμά μας.
- Πρέπει να τα αποθηκεύσουμε αλλιώς χάνονται με το πέρας της εκτέλεσης.

```
<form enctype="multipart/form-data" method=“post”>
        Send this file:
        <input name="userfile" type="file" />
        <input type="submit" value="Send File" />
</form>
```

Μεταβλητή $_FILES

- Η μεταβλητή *$_FILES* περιέχει όλα τα αρχεία που ανέβασε ο χρήστης.

- *$_FILES['userfile']['name']*
  - Περιέχει το πραγματικό όνομα κάθε αρχείου.
- *$_FILES['userfile']['tmp_name']*
  - Περιέχει το προσωρινό όνομα κάθε αρχείου.
- *$_FILES['userfile']['size']*
  - Περιέχει το μέγεθος κάθε αρχείου σε bytes.

Αποθήκευση Αρχείων

- Συνάρτηση αποθήκευσης uploaded αρχείων
  - *move_uploaded_files($filename, $destination)*
  - γίνεται έλεγχος πριν την αποθήκευση ότι το αρχείο προέρχεται από ανέβασμα.
  - η μεταβλητή *$filename* είναι το αρχείο που ανέβηκε.
  - η μεταβλητή *$destination* είναι εκεί που θα αποθηκευτεί.

```
<?php
  $destination = 'C:\Uploads\'
  if( !empty( $_FILES ) ) {
    $destination .= $_FILES[ 'foo' ][ 'name' ];
    $filename = $_FILES[ 'foo' ][ 'tmp_name' ];
    move_uploaded_file( $filename, $destination );
  }
?>
```

Επικήδειοι

1. Booleans – Null (κενό) ✓
2. Πίνακες ✓
3. Εντολή foreach ✓
4. Διαχωρισμός σε Αρχεία ✓
5. Συναρτήσεις Αρχείων ✓
6. Διαχείρηση Αρχείων ✓
7. Headers (Κεφαλίδες)

Συνάρτηση *header()*

- Η συνάρτηση *header()* παίρνει ως όρισμα ένα String και το στέλνει στην κεφαλίδα της απάντησης.
- Πρέπει να τρέξει πριν από οποιοδήποτε output
  - Είτε είναι html.
  - Είτε space, tab ή enter.

```
<html>
<?php
        header( 'Location: http://www.apisti.com/‘ );
?>
```

Χρήσιμα Headers

- ***header('Location: http://www.apisti.com/‘)***
  - Κάνει redirect σε μια διεύθυνση.
  - Η διεύθυνση πρέπει να είναι απόλυτη.

- ***header('Content-type: text/html; charset=utf-8‘)***
  - Είναι εντολή ισοδύναμη με τον κώδικα

```
<meta
        http-equiv=“Content-type”
        content=“text/html; charset=utf-8”
/>
```

THE END!!!