

# Web Programming for Dummies

Γράψαμε αυτό το tutorial, αφού ολοκληρώσαμε την άσκηση Προγραμματισμού στον Παγκόσμιο Ιστό. Προσπαθήσαμε να γράψουμε γενικά πράγματα που βρήκαμε εμείς μετά από αναζητήσεις .Αποφύγαμε να δώσουμε έτοιμες απαντήσεις και λύσεις, αφού τα πρότζεκτ κάθε χρόνο θα είναι διαφορετικά, καθώς και γιατί θα κάνετε το ίδιο πράγμα πολλές φορές (φόρμες, ερωτήσεις στη βάση δεδομένων, php programming), καλύτερο θα ήταν να τα ψάξετε και να τα μάθετε και από μόνοι σας. Επίσης δεν χρειάζεται να υπάρχει ομοιογένεια στα πρότζεκτ. Ο καθένας σκέφτεται με τον δικό του τρόπο, και σύμφωνα με αυτό θα κάνει το πρότζεκτ του. Σε καμία περίπτωση το tutorial αυτό δεν αντικαθιστά τις διαφάνειες ή τις διαλέξεις του μαθήματος, ούτε είναι ένα αυτοτελές walkthrough για το πρότζεκτ, ούτε περιέχει τα πάντα για HTML,PHP κτλ που χρειάζεται να ξέρετε και που μπορείτε να μάθετε.

### Starting Up:

Κατεβάζουμε τα προγράμματα από το site του μαθήματος wamp/xamp server, apache κτλ.

Στο skype στις προχωρημένες ρυθμίσεις, απενεργοποιούμε την επιλογή να χρησιμοποιεί το skype το port 80.

Κατεβάζουμε ένα πρόγραμμα για σχεδιασμό βάσης δεδομένων (εμείς χρησιμοποιήσαμε το Power Designer 16.5).

Όποτε κάνουμε κλικ στο wamp στη μπάρα εργασιών, και επιλέγουμε localhost, θα μας ανοίγει με κάποιον browser το index.php που βρίσκεται στο C:/wamp/www/. Εκεί θα δημιουργούμε ό,τι αρχείο χρειάζεται το πρότζεκτ.

Κλικάροντας στο wamp -> phpmyadmin πηγαίνουμε στην βάση δεδομένων του server. Μόλις φτιάξετε την βάση δεδομένων σας, κάντε την import.

### Δημιουργία Βάσης Δεδομένων:

Ανοίγουμε το Power Designer. New->Conceptual Data. Από δεξιά δημιουργούμε τις οντότητες (entities) που χρειάζονται, πχ χρήστες, αναφορές κτλ.

Διπλό κλικ στο entity, και εκεί γράφουμε τα attributes του κάθε πίνακα. Επιλέγουμε το κατάλληλο datatype. Εάν επιλέξουμε varchar, θα πρέπει να επιλέξουμε κ έναν μέγιστο αριθμό χαρακτήρων στο length.

Κάθε πίνακας θα έχει ένα Auto Increment id, το οποίο θα είναι και το πρωτεύον κλειδί του. Αυτό γίνεται τσεκάροντας το κουτί κάτω από το P (Primary Key). Εάν θέλουμε ένα attribute να είναι υποχρεωτικό (δηλ, ο χρήστης να μην μπορεί να το αφήσει κενό) αλλά παρόλα αυτά να μην είναι Primary Key(δηλαδή να μην είναι διαφορετικό από τα υπόλοιπα), τότε τσεκάρουμε το M (Mandatory=υποχρεωτικό).

Εάν είναι υποχρεωτικό κάθε χρήστης να γράφει το επώνυμο του, τότε αυτό θα ήταν mandatory, καθώς, 2 επώνυμα μπορεί να είναι ίδια (συνεπώς πρέπει να μην είναι Primary Key).

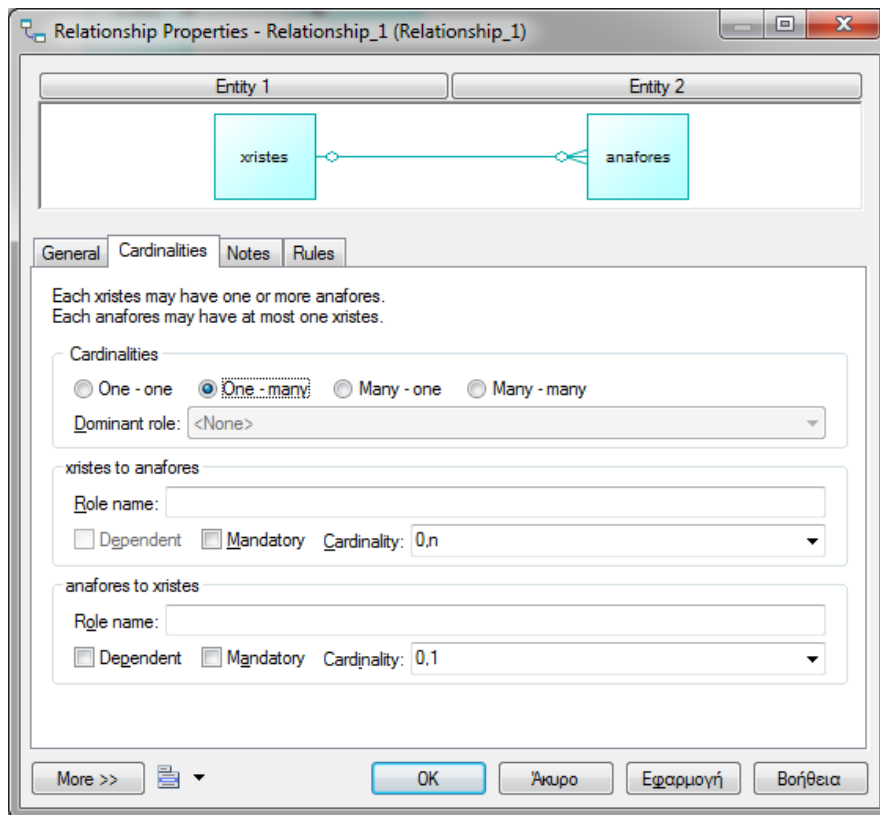
Θέλουμε κάθε κομμάτι γνώσης να αναπαρίσταται μία φορά μόνο στην βάση δεδομένων και να μην επαναλαμβάνεται, και ταυτόχρονα, οι πίνακες να είναι όσο πιο αποκεντρωμένοι μεταξύ τους γίνεται.

Να μην υπάρχει δηλαδή ένας πίνακας που να περιέχει τα στοιχεία των χρηστών, των αναφορών κτλ, αλλά ένας πίνακας χρηστών και ξεχωριστά ένας πίνακας αναφορών. Η σύνδεση θα φτιαχτεί μέσα από το relationship

Αφού τελειώσουμε όλες τις οντότητες, δημιουργούμε τις σχέσεις μεταξύ τους. Κάνουμε drag n drop το συμβολάκι από το conceptual diagram στα δεξιά, κ κλικάρουμε που θέλουμε να συνδεθούν τα 2 άκρα. Δίνουμε όνομα στη σχέση, κ μετά κάνουμε διπλό κλικ->Cardinalities.

Εδώ θα πρέπει να σκεφτούμε τι περιορισμούς συσχέτισης θέλουμε να έχουν οι πίνακες μεταξύ τους. Ένας χρήστης μπορεί δημιουργεί όσες αναφορές θέλει, αλλά μία αναφορά πρέπει να έχει δημιουργηθεί από

αποκλειστικά έναν χρήστη. Αυτή η πληροφορία αναπαρίσται δηλώνοντας ότι οι πίνακες έχουν σχέση One-Many.



Κάτω από την καρτέλα Cardinalities αναγράφονται οι λογικές προτάσεις που έχουμε δημιουργήσει. Σωστά, κάθε χρήστης μπορεί να δημιουργήσει όσες αναφορές θέλει(από 0 μέχρι n).

Οι αναφορές όμως, δηλώσαμε πως πρέπει να έχουν δημιουργηθεί από 1 και μόνο 1 χρήστη. Συνεπώς, πηγαίνουμε στην σχέση που έχουν οι αναφορές ως προς τους χρήστες, και κλικάρουμε το mandatory.

Relationship Properties - Relationship\_1 (Relationship\_1)

Entity 1 Entity 2

General Cardinalities Notes Rules

Each xristes may have one or more anafores.  
Each anafores must have one and only one xristes.

Cardinalities

One - one  One - many  Many - one  Many - many

Dominant role: <None>

xristes to anafores

Role name: \_\_\_\_\_

Dependent  Mandatory Cardinality: 0..n

anafores to xristes

Role name: \_\_\_\_\_

Dependent  Mandatory Cardinality: 1..1

More >>

## HTML: HyperText Markup Language

-**Δεν** είναι γλώσσα προγραμματισμού (δεν έχει δομές ελέγχου, επανάληψης κτλ, δεν δηλώνει μεταβλητές, δεν περιλαμβάνει συναρτήσεις)

- Markup Γλώσσα: Κάθε τι που γράφουμε σε ένα HTML αρχείο περικλείεται μέσα σε tags, δημιουργώντας ένα στοιχείο (element) οπότε το περιεχόμενο είναι markup

-Τρέχει στον browser (client-side)

-Περιγράφει τη δομή της ιστοσελίδας, και όχι την εμφάνιση (αν και έχει μερικώς αυτή την δυνατότητα)

## Server – Client Επικοινωνία

Κάθε φορά που πληκτρολογούμε μια διεύθυνση στο url του browser μας, ο browser στέλνει μια αίτηση για το περιεχόμενο της σελίδας, από την διεύθυνση αυτή. Η σελίδα βρίσκει το κατάλληλο HTML αρχείο, και το στέλνει πίσω στον browser. Ο browser λαμβάνει τον HTML κώδικα, τον τρέχει, και τον μετατρέπει σε αυτό που βλέπουμε. Στην επικοινωνία αυτή, ο browser μας είναι ο client, και η σελίδα είναι ο Server.

Έστω ότι η σελίδα που ζητάμε, περιέχει αυτό τον κώδικα:

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

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

Παρατηρούμε πως ό,τι έχουμε γράψει στην σελίδα μας ('My First Heading' και 'My first paragraph') περικλείεται σε tags που ανοίγουν πριν ξεκινήσει το περιεχόμενο, και κλείνουν αφού τελειώσει.

Αυτό που φαίνεται στην οθόνη μας, είναι αυτό:



Σχηματικά, λοιπόν, μπορούμε να φανταστούμε την δομή κάθε σελίδας έτσι ώστε να προκύπτει μία δομή, με το ένα στοιχείο να περιέχεται σε κάποιο άλλο:

```
<html>
  <body>
    <h1>This a heading</h1>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </body>
</html>
```



PHP: PHP HyperText Preprocessor

-Γλώσσα Προγραμματισμού

-Τρέχει στον Server(Server-Side)

-Χρησιμοποιείται στο Web Design, για δημιουργία δυναμικού περιεχομένου

-Παράγει HTML

Αντίθετα, λοιπόν από την HTML, η PHP τρέχει στον server. Τι σημαίνει αυτό?

Όταν ο browser μας ζητάει μία PHP σελίδα, ο browser δεν κατεβάζει PHP κώδικα. Ο Server, τρέχει πρώτα τον κώδικα αυτό, ο οποίος παράγει τον html κώδικα, που στέλνεται τελικά στον browser.

Το πιο χαρακτηριστικό παράδειγμα για να κατανοηθεί η χρήση της κάθε γλώσσας είναι το παράδειγμα στο οποίο θέλουμε στην σελίδα μας να εμφανίζεται η ώρα:

A screenshot of a terminal window with a black background. The text displayed is the HTML code: `<p>The current server time is 11:00:00 am.</p>`. The opening and closing tags are highlighted in red.

Με τον τρόπο αυτό, όποτε έτρεχε ο κώδικας, θα εμφανιζόταν πως η ώρα είναι 11:00 πμ, πράγμα που **δεν** θέλουμε.

Χρησιμοποιώντας μόνο HTML αυτό θα ήταν αδύνατο να αλλάζουμε την ώρα κάθε δευτερόλεπτο, ώστε να ανταποκρίνεται στην πραγματικότητα σε κάθε refresh της σελίδας.

Αντίθετα, εάν χρησιμοποιούσαμε PHP κώδικα στη σελίδα μας, θα καλούσαμε μία συνάρτηση η οποία θα εμφάνιζε την ώρα όπου θα έτρεχε ο κώδικας κάθε φορά:

```
<p>The current server time is <?php echo date("H:i:s a"); ?>.</p>
```

A screenshot of a code editor with a dark background. The text is in a light color, showing PHP code: <p>The current server time is <?php echo date("H:i:s a"); ?>.</p>. The code is enclosed in a pre tag.

Figure 1 Παρατηρούμε ότι ανοίγουμε ένα tag php και γράφουμε PHP μέσα σε ένα αρχείο HTML

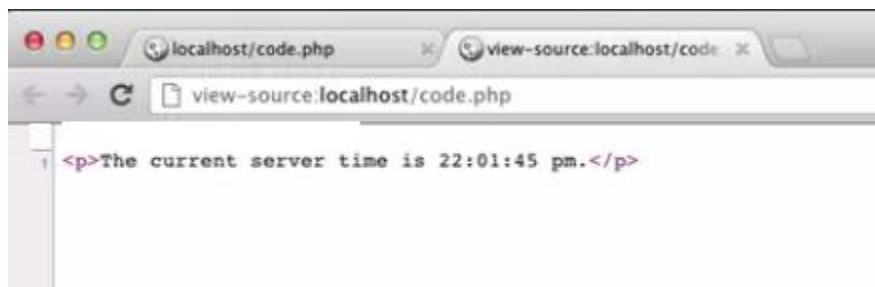
Το αποτέλεσμα θα ήταν το επιθυμητό:

The current server time is 22:01:39 pm.

The current server time is 22:01:45 pm.

**Σημείωση:** Εάν ο Server είναι στην Αμερική, και ο client στην Ελλάδα, τότε η ώρα που θα εμφανίζεται θα είναι η ώρα του server. Εάν θέλουμε να εμφανίζεται η ώρα του client, δηλαδή να προσαρμόζεται το μήνυμα ανάλογα με τη χώρα που προσπελαύνει την σελίδα θα μπορούσαμε να χρησιμοποιήσουμε javascript(μια client side 'γλώσσα')

Σημειώνουμε πως εάν κάναμε View Page Code στη σελίδα αυτή, θα βλέπαμε μόνο τον παραγόμενο HTML κώδικα, και **όχι** τον PHP:



Συνεπώς, θέλουμε έναν συνδυασμό PHP / HTML για να έχουμε το αποτέλεσμα που θέλουμε. Κάθε τι που θέλουμε να γράψουμε στην σελίδα μας, το οποίο δεν είναι προϊόν επεξεργασίας (πχ από δομές επιλογής, επανάληψης κτλ) μπορεί να γραφεί απευθείας στην HTML. Αντίθετα, τα υπόλοιπα πρέπει να τρέχουν μέσα σε PHP.

HTML tags που **πρέπει** να χρησιμοποιούνται σε κάθε αρχείο:

`<!DOCTYPE html>` : Δηλώνει στον browser ότι χρησιμοποιούμε HTML5 (παλιότερες εκδόσεις της HTML έχουν άλλο αναγνωριστικό)

`<html>` και `</html>` : Περιγράφουν την ιστοσελίδα

`<body>` και `</body>` : Περιλαμβάνουν το ορατό περιεχόμενο της σελίδας

`<head >` και `</head>` : Δείχνει τον τίτλο που θα έχει η ιστοσελίδα μας

## Φόρμες

Οι φόρμες χρησιμοποιούνται για να στείλουμε δεδομένα από τον client στον server: πχ θέλουμε να κάνουμε register σε μία σελίδα. Μας παραπέμπει σε μία σχετική φόρμα στην οποία συμπληρώνουμε στα textboxes τα στοιχεία μας. Τα δεδομένα αυτά, στέλνονται στον server, ελέγχονται, και αποθηκεύονται κατάλληλα στην βάση δεδομένων.

### **Form**

- action : ορίζει το url που θα χρησιμοποιηθεί από τον εξυπηρετητή για να επεξεργαστεί τα δεδομένα της φόρμας
- method = {get, post}: ορίζει τη μέθοδο αποστολής των περιεχομένων της φόρμας
  - get : στέλνει τα περιεχόμενα σαν query στο url
  - post : τα στέλνει στο σώμα του http μηνύματος

Η πιο χαρακτηριστική φόρμα GET, είναι αυτή του youtube. Κάνουμε ένα search, το οποίο μετά εμφανίζεται στο Url του browser. Συνεπώς, εάν τα στοιχεία που θα τροφοδοτήσει ο client τον server δεν χρειάζεται να είναι κρυφά (πχ δεν είναι κωδικοί), τότε μπορούμε να χρησιμοποιήσουμε την μέθοδο GET . Η σελίδα αυτή, θα μπορεί να είναι προσβάσιμη ξανά χωρίς να ξανακάνουμε το ίδιο search (πχ μπορεί να γίνει bookmarked)

Αντίθετα, εάν θέλουμε να στείλουμε στον server στοιχεία κρυφά, αυτό το κάνουμε με την μέθοδο POST. Έτσι, δεν υπάρχει όριο χαρακτήρων, στα δεδομένα που θα στείλουμε και τα δεδομένα αυτά θα είναι προσβάσιμα μόνο από τον server και δεν θα φανούν. Ένα τέτοιο παράδειγμα, είναι όταν κάνουμε Login σε μία σελίδα. Όταν πατήσουμε «σύνδεση» δεν θέλουμε να φαίνεται ο κωδικός μας στο url, παρόλα αυτά είναι διαθέσιμος προς επεξεργασία από τον server.

Για να ανακτήσουμε τα δεδομένα που στείλαμε με φόρμα στον server, χρησιμοποιούμε τις Superglobal μεταβλητές `$_POST` και `$_GET*`, στην σελίδα όπου δηλώσαμε ότι θα τα επεξεργαστούμε στο form action. Έτσι, στην σελίδα `px register.php` κάνουμε το register, και βάζοντας στο form action `aeκ.php` σημαίνει ότι τα δεδομένα αυτά θα είναι προσβάσιμα από αυτή την σελίδα.

## Register.php

```
form method="POST" action="aeκ.php">
  Username: <input type="text" name="username"/><br />
  Password: <input type="password" name="password" /><br />
  <p><input type="submit" value="Login" /></p>
</div>
</form>
```

## Aek.php

```
echo $_POST['username'];
echo $_POST['password'];
```

## Sessions

Οι μεταβλητές Session χρησιμοποιούνται για τη διατήρηση πληροφορίας σε περίπτωση διαδοχικών προσβάσεων. Με αυτόν τον τρόπο είναι δυνατή η προσωποποίηση του περιεχομένου. Για παράδειγμα, η αρχική σελίδα του facebook είναι διαφορετική και μοναδική για τον κάθε χρήστη.

Επειδή το HTTP πρωτόκολλο είναι Stateless[ο server δέχεται και επεξεργάζεται κάθε αίτηση ξεχωρίστα και ανεξάρτητα από τις προηγούμενες], πρέπει να ιδρύσουμε εμείς αυτό το αναγνωριστικό μέσα στον κώδικα , μέσα από τον client (cookies/HTML5 Web Storage) ή μέσα από τον server(Sessions). Κάθε μία δυνατότητα έχει συγκεκριμένα πλεονεκτήματα και μειονεκτήματα, εμείς όμως θα αναλύσουμε τα Sessions.

Μέσω Server γίνεται χρησιμοποιώντας την superglobal μεταβλητή \$\_SESSION. Όταν αποθηκεύσουμε μία μεταβλητή σε ένα Session, αυτή θα είναι προσβάσιμη από κάθε σελίδα που θα δημιουργήσουμε, αρκεί να καλέσουμε την συνάρτηση `session_start()`; .Έτσι, εάν θέλουμε πχ να ξέρουμε αν ο χρήστης που έχει συνδεθεί είναι διαχειριστής, τότε

### Login.php

```
Session_start();
If ($admin_stat==1) //ean h metavlhth sthn opoia apothikevoume ta
                    //dikaiwmata tou kathe xrhsth einai 1, tote
$_SESSION['admin']=1; //tha kanoume ton xrhsth diaxeiristh
```

### Yolo.php

```
Session_start();
If (isset($_SESSION['admin']) && ($_SESSION['admin']==1) echo aek;
//ean h metavlhth SESSION admin exeí tethei, kai einai 1, tote emfanise
'aek'
```

Στο login.php ελέγχουμε εάν ο χρήστης που έχει συνδεθεί είναι admin, και εάν είναι, αποθηκεύουμε στην Session μεταβλητή admin την τιμή 1. Σε οποιοδήποτε αρχείο, θέλουμε να μάθουμε τα δικαιώματα του χρήστη, αρκεί να ελέγξουμε την τιμή της μεταβλητής αυτής, όπως κάνουμε στο yolo.php

Δεν ξεχνάμε να καταστρέψουμε το Session, όταν χρειαστεί, χρησιμοποιώντας την unset() ή την session\_destroy() .

## SQL Queries

Κάθε φορά που θέλουμε να προσπελάσουμε την βάση δεδομένων πρέπει να το κάνουμε με sql queries στην php.

Σε **κάθε** σελίδα στην οποία θέλουμε να κάνουμε προσπέλαση την βάση δεδομένων, πρέπει να ορίσουμε την σύνδεση μας με την βάση. Επειδή θα γράψουμε πολλές φορές αυτό τον κώδικα, μπορούμε να το αποθηκεύσουμε σε ένα ξεχωριστό αρχείο, και να το κάνουμε καλούμε χρησιμοποιώντας το include κάθε φορά.

```
<?php
// Create connection
$con=mysqli_connect("localhost","admin","","test");

// Check connection
if (mysqli_connect_errno()) {
    echo "Failed to connect to MySQL: " . mysqli_connect_error();
}
?>
```

Στην μεταβλητή \$con αποθηκεύουμε την σύνδεση μας με την βάση δεδομένων. Κάθε φορά που θα θέλουμε να κάνουμε ένα sql query , θα χρησιμοποιούμε αυτή την μεταβλητή.

Η mysqli\_connect έχει τα εξής ορίσματα:

Host : σε μας localhost, θα μπορούσε να είναι της μορφής www.example.com

Username: το default για χρήστες wamp είναι admin

Password: το default για χρήστες wamp είναι κενό.

Dbname: όνομα της βάσης δεδομένων, στην οποία θέλουμε να συνδεθούμε



Σε περίπτωση που αλλάξουμε τα στοιχεία της βάσης δεδομένων μας, ή του rhrmyadmin, πρέπει να αλλάξουμε κ τα στοιχεία σύνδεσης, έτσι ώστε να γίνεται σωστά η επικοινωνία server με βάση δεδομένων.

Αρχικά, κάνουμε το query που επιθυμούμε. Εδώ θέλουμε να εμφανίσουμε όλους τους χρήστες με όνομα kostas.

```
$query_name = SELECT *  
                FROM pinakas_xristwn  
                WHERE onoma="kostas";
```

Στη συνέχεια, καλούμε την συνάρτηση mysqli\_query για να εκτελεστεί το query που αποθηκεύσαμε στην μεταβλητή \$query\_name.

```
$result=mysqli_query($con,$query_name);
```

Τώρα στην μεταβλητή result έχει αποθηκευτεί ο πίνακας που κάναμε select.

Για να εμφανίσουμε τα στοιχεία του πίνακα , χρησιμοποιούμε την mysqli\_fetch\_array. Κάθε φορά που εκτελείται η mysqli\_fetch\_array, στην μεταβλητή row αποθηκεύεται η επόμενη γραμμή του πίνακα \$result. Όταν ο πίνακας δεν έχει άλλες γραμμές προς προσπέλαση, η while τερματίζει.

```
while($row = mysqli_fetch_array($result))  
{  
    echo $row['username'];  
}
```

Με το echo \$row['username'] δηλώνουμε ποια στήλη του πίνακα της βάσης δεδομένων θέλουμε να χρησιμοποιηθεί. Δηλαδή, εάν ο πίνακας που έχουμε στην \$result, έχει τις στήλες username, onoma, hlikia, με τις μεταβλητές \$row['username'], \$row['onoma'], \$row['hlikia'] θα προσπελάναμε τα κελιά από τις αντίστοιχες στήλες.

Εάν **δεν** είχαμε την `while` και απλά κάναμε

```
$row = mysqli_fetch_array($result);  
echo $row['username'];
```

Θα εμφανιζόταν **μόνο** το πρώτο αποτέλεσμα του πίνακα.

### Σημειώσεις:

1. Θα βρείτε στο ιντερνετ πολλές φορές να χρησιμοποιείται η `mysqli_query` αντί της `mysql_query`. Αυτό γίνεται γιατί η τελευταία, είναι η πιο πρόσφατη έκδοση(**mysqli=mysql Improved**) της `mysql_query`. Στις διαφάνειες του μαθήματος χρησιμοποιείται η **mysqli**.

2.Υπάρχουν πολλοί διαφορετικοί τρόποι να εμφανίσουμε τα δεδομένα μας, χρησιμοποιώντας διαφορετικές συναρτήσεις, όπως `mysqli_fetch_row`, `mysqli_fetch_assoc`.

3.Υπάρχει διαδικαστικός καθώς και αντικειμενοστραφής τρόπος για τα ερωτήματα( όπως επίσης και για κάθε συνάρτηση στην `php`).

4. Με την συνάρτηση `extract($row)` , μέσα στην `while`, δημιουργούνται αυτόματα μεταβλητές με όνομα, το όνομα των στηλών του πίνακα. Άρα, το προηγούμενο παράδειγμα θα γινόταν

```
while($row = mysqli_fetch_array($result))  
{  
    extract($row);  
    echo $username;  
    echo $onoma;  
    echo $hlikia;  
}
```

## JavaScript

- Είναι μία scripting language
- «Ελαφριά» γλώσσα προγραμματισμού
- Μπορεί να προστεθεί σε ένα HTML αρχείο
- «Εκτελείται» από τον Internet browser (Client –Side)
- Δεν σχετίζεται με τη Java, αλλά έχει ομοιότητες με αυτή

### Χρήση της JavaScript:

- Εμφάνιση «δυναμικού» περιεχομένου
- Αντιλαμβάνεται και αντιδρά σε γεγονότα – αλληλεπίδραση με τον χρήστη
- π.χ. ο χρήστης επιλέγει ένα link
- Αλλάζει τα περιεχόμενα σε html elements
- Ελέγχει δεδομένα που δίνονται σε μία φόρμα πριν αυτή γίνει submit

Απλό παράδειγμα χρήσης javascript:

[http://www.w3schools.com/js/tryit.asp?filename=tryjs\\_lightbulb](http://www.w3schools.com/js/tryit.asp?filename=tryjs_lightbulb)

Στο συγκεκριμένο παράδειγμα όταν κλικάρουμε την εικόνα εκτελείται η συνάρτηση `changeImage()` που δημιουργήσαμε. Μέσα στην συνάρτηση χρησιμοποιούμε την εντολή **`document.getElementById`**. Με αυτήν, εντοπίζουμε στον html κώδικα το αντικείμενο με id που ζητάμε ('myImage') και το επεξεργαζόμαστε κατάλληλα. Στο συγκεκριμένο παράδειγμα, ελέγχουμε ποια εικόνα εμφανίζεται. Εάν εμφανίζεται η ανοιχτή λάμπα, εμφανίζουμε την εικόνα με την κλειστή λάμπα και αντίστροφα.

Δίπλα στο id της εικόνας γράφουμε το `onclick="changeImage()",` για να δηλώσουμε πως θα εκτελεστεί αυτή η συνάρτηση κάθε φορά που κλικάρουμε την εικόνα.

Σχόλια/Παρατηρήσεις/Διορθώσεις/Κατακραυγή  
[mavroforos@ceid.upatras.gr](mailto:mavroforos@ceid.upatras.gr)  
[gkofas@ceid.upatras.gr](mailto:gkofas@ceid.upatras.gr)