Τ.Ε.Ι ΗΠΕΙΡΟΥ

Τ.Ε.Ι OF EPIRYS

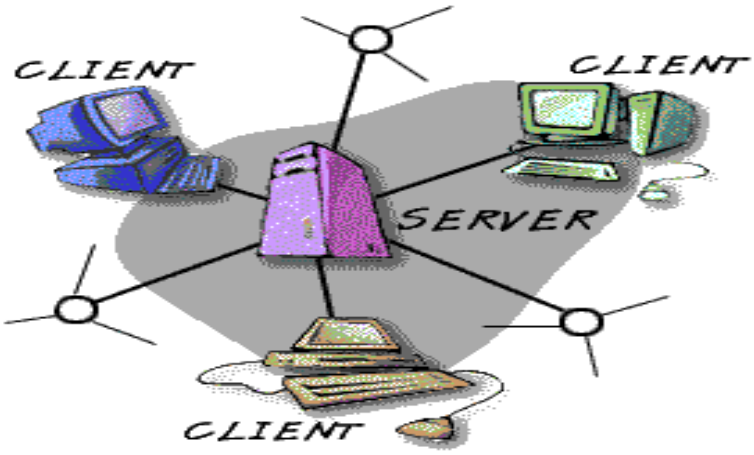
Σχολή Διοίκησης & Οικονομίας (Σ.Δ.Ο)
Τμήμα Τηλεπληροφορικής & Διοίκησης

School Of Management And Economics
Department Of Communications, Informatics And Management

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΕΞΑΜΗΝΟ Η'

ΘΕΜΑ: **ΤΕΧΝΟΛΟΓΙΕΣ INTERNET**
PHP– APACHE– JAVASCRIPT– CSS
ΚΑΙ ΧΡΗΣΗ ΤΟΥΣ ΓΙΑ ΤΗΝ ΑΝΑΠΤΥΞΗ ΜΙΑΣ On-Line ΕΦΑΡΜΟΓΗΣ



ΟΝΟΜΑ ΦΟΙΤΗΤΗ:
ΔΗΜΗΤΡΙΟΣ Χ. ΒΟΥΖΙΑΝΑΣ

ΥΠΕΥΘΥΝΟΣ ΚΑΘΗΓΗΤΗΣ:
ΒΑΣΙΛΕΙΟΣ ΧΑΡΙΛΟΓΗΣ

ΑΡΤΑ ΔΕΚΕΜΒΡΙΟΣ 2004

ΠΕΡΙΕΧΟΜΕΝΑ:

ΚΕΦΑΛΑΙΟ IV – Javascript

ΚΕΦΑΛΑΙΟ V – Apache HTTP WEB Server

ΚΕΦΑΛΑΙΟ VI – MySQL

ΚΕΦΑΛΑΙΟ VII – PHP

Εισαγωγή

Δεν αναρωτηθήκατε ποτέ πως δουλεύει μια εφαρμογή του internet ;

Έχετε ακούσει ποτέ τις λέξεις HTML-PHP-MySQL-APACHE;

Σ'αυτή την εργασία θα προσπαθήσω να αναπτύξω ένα μέρος της επιστήμης, αλλά και γιατί όχι της τέχνης που απαιτείται ώστε να καταφέρει κάποιος να αναπτύξει μια αλληλεπιδραστική (interactive) Web εφαρμογή;

Τα εργαλεία και οι τεχνολογίες που θα χρησιμοποιήσω για την ανάπτυξη της εφαρμογής που μέσα από αυτή την εργασία θα παρουσιάσω είναι :

- Η γλώσσα προγραμματισμού διαδικτιακών εφαρμογών PHP
- Το σύστημα διαχείρισης βάσεων δεδομένων MySQL
- Η γλώσσα προγραμματισμού - υποστήριξης Web εφαρμογών Javascript
- Η τεχνολογία CSS
- Και φυσικά η κατεξοχήν γλώσσα του Internet η HTML.

Κοινό γνώρισμα των παραπάνω τεχνολογιών, είναι ότι γίνονται σχετικά εύκολα κατανοητές χωρίς να χρειάζεται κάποιος να έχει προηγμένες γνώσεις προγραμματισμού, αλλά είναι και ιδιαίτερα ευχάριστες αφού τα αποτελέσματα των εργασιών , εμφανίζονται μπροστά μας απλός μέσο ενός φυλλομετρητή – Browser όπως είναι ο Internet Explorer ή ο Netscape Navigator.

Με το πέρας της εργασίας αυτής, ελπίζω πως ο άγνωστης θα έχει αποκτήσει μια αρκετά σαφή εικόνα για το πώς δημιουργούνται η πραγματικά πολλές φορές αξιοθαύμαστες σύγχρονες Web εφαρμογές.

Καλή ανάγνωση λοιπόν!

ΔΗΜΗΤΡΙΟΣ Χ. ΒΟΥΖΙΑΝΑΣ

Πολλά είναι αυτά που ακούμε καθημερινά για το Internet.

Ποσά όμως από αυτά αγγίζουν την πραγματικότητα, σε ποιον ανήκει και ποιοι είναι αυτοί που ορίζουν τις εξελίξεις σ'αυτό των τεραστίων δυνατοτήτων μέσο;

Παρακάτω γίνετε μια προσπάθεια να δοθούν απαντήσεις σε ερωτήσεις όπως οι παραπάνω, οι οποίες αν και είναι αρκετά γενικές , ωστόσο επιβάλλετε να γνωρίζει την απάντηση τους κάποιος που θέλει να ασχοληθεί με την ανάπτυξη web εφαρμογών.

Οι Οργανισμοί του Internet

Το Internet έχει επιφέρει μια επανάσταση στον κόσμο των υπολογιστών και των επικοινωνιών όσο τίποτα άλλο μέχρι σήμερα.

Η εφεύρεση του τηλεγράφου του τηλεφώνου, του ραδιοφώνου και του υπολογιστή ήταν το βήμα γι' αυτήν την άνευ προηγουμένου ολοκλήρωση των επιτευγμάτων. Το Internet είναι μια τεχνολογία για παγκόσμια επικοινωνία (broadcasting), ένας μηχανισμός για διασπορά πληροφοριών και ένα μέσο για συνεργασία και αλληλεπίδραση ανάμεσα σε ιδιώτες και τους υπολογιστές τους χωρίς να αποτελεί εμπόδιο η γεωγραφική τοποθεσία.

Το Internet αντιπροσωπεύει ένα από τα πιο επιτυχημένα παραδείγματα από τα οφέλη που προέκυψαν από τη συνεχή επένδυση και αφοσίωση στην έρευνα και την ανάπτυξη της υποδομής της πληροφορικής.

Ξεκινώντας με τις πρώτες έρευνες στη μεταγωγή πακέτων (packet switching), η κυβέρνηση, η βιομηχανία και η ακαδημαϊκή κοινότητα ήταν συνεργάτες στην εξέλιξη και την ανάπτυξη αυτής της εκπληκτικής νέας τεχνολογίας.

Στην πραγματικότητα κανένας δεν είναι κάτοχος (κύριος) του Internet και κανένα άτομο ή οργανισμός δεν ελέγχει πλήρως το Internet. Περισσότερο σαν έννοια παρά σαν μια πραγματική χειροπιαστή οντότητα, το Internet βασίζεται σε μια φυσική υποδομή (physical infrastructure) η οποία διασυνδέει δίκτυα (networks) μ' άλλα δίκτυα. Υπάρχουν πολλοί οργανισμοί, εταιρείες, κυβερνήσεις, σχολεία, ιδιώτες και παροχείς υπηρεσιών (service providers) που κατέχουν ένα κομμάτι αυτής της υποδομής, αλλά δεν υπάρχει κανένας που να το κατέχει ολόκληρο.

Υπάρχουν, όμως, οργανισμοί που επιτηρούν και ορίζουν τα πρότυπα (standards) γι' ό,τι συμβαίνει στο Internet και εκχωρούν IP διευθύνσεις (IP addresses) και ονόματα περιοχής (domain names).

Τέτοιοι οργανισμοί είναι το National Science Foundation (NSF), το Internet Engineering Task Force (IETF), το ICANN, το InterNIC και το Internet Architecture Board (IAB).

Το ***IETF (Internet Engineering Task Force)*** είναι ο κύριος οργανισμός προτύπων για το Internet και πρόκειται για μια μεγάλη ανοικτή διεθνή κοινότητα που αποτελείται από σχεδιαστές δικτύων, χειριστές, πωλητές και ερευνητές που ασχολούνται (ενδιαφέρονται) για την εξέλιξη της αρχιτεκτονικής του Internet και την ομαλή λειτουργία του. Είναι ανοικτό σ' οποιονδήποτε ενδιαφερόμενο ιδιώτη.

Το ***InterNIC*** είναι προς τον παρόν ένας πληροφοριακός δικτυακός τόπος (Web site) που δημιουργήθηκε για να παρέχει στο κοινό πληροφορίες σχετικά με την καταχώρηση ενός ονόματος περιοχής (domain name registration).

Το ***ICANN (Internet Corporation for Assigned Names and Numbers)*** είναι ένας μη κερδοσκοπικός οργανισμός που έχει αναλάβει την ευθύνη για την καταχώρηση χώρου (space allocation) των IP διευθύνσεων, την εκχώρηση των παραμέτρων του πρωτοκόλλου, τη διαχείριση του συστήματος ονοματοδοσίας περιοχής (domain name system management) και τη διαχείριση του συστήματος του διακομιστή (root server system management). Οι εργασίες αυτές γινόντουσαν παλαιότερα με συμβόλαιο από την κυβέρνηση των ΗΠΑ.

Το ***IAB (Internet Architecture Board)*** είναι μια τεχνική συμβουλευτική ομάδα του Internet Society, οι αρμοδιότητες της οποίας περιλαμβάνουν :

- Επίβλεψη του Internet Engineering Task Force (IETF).
- Επίβλεψη της διαδικασίας δημοσίευσης των προτύπων του Internet.
- Διαχείριση του Request for Comments (RFCs).

Το InterNIC είναι αρμόδιο για την εκχώρηση τάξεων (classes) σε διαφορετικούς οργανισμούς σύμφωνα με τον αριθμό των hosts που έχουν.

Μια Σύντομη Ιστορία του Internet

Το Υπουργείο Άμυνας των ΗΠΑ έθεσε τα θεμέλια του Internet τριάντα περίπου χρόνια νωρίτερα (αρχές της δεκαετίας του '70) μ' ένα δίκτυο υπολογιστών που αποκαλείτο ARPANET. Αλλά το ευρύ κοινό δεν χρησιμοποιούσε πολύ το Internet μέχρι την ανάπτυξη του Παγκόσμιου Ιστού (World Wide Web) στις αρχές της δεκαετίας του '90. Μέχρι τον Ιούνιο του 1993 υπήρχαν μόλις 130 Web sites, ενώ σήμερα υπάρχουν περισσότερα από 2 δισεκατομμύρια Web sites. Το ARPANET ήταν ένα μεγάλο δίκτυο ευρείας περιοχής (WAN, Wide-Area Network) που δημιουργήθηκε από την Υπηρεσία Προηγμένων Ερευνητικών Εργασιών του Υπουργείου Άμυνας των ΗΠΑ, που αποκαλείτο ARPA (Advanced Research Project Agency).

Με έτος ίδρυσης το 1969, το ARPANET χρησίμευσε σαν ένα πλαίσιο δοκιμών για τις νέες τεχνολογίες δικτύωσης, διασυνδέοντας πολλά πανεπιστήμια και ερευνητικά κέντρα. Οι δύο πρώτοι κόμβοι (nodes) που σχημάτισαν το ARPANET ήταν το UCLA (University of California, Los Angeles) και το Stanford Research Institute, ακολουθούμενα σύντομα από το Πανεπιστήμιο της Utah.

Από το ARPANET στο Internet

Τον Οκτώβριο του 1972, το ARPANET έγινε «δημόσιο». Στην πρώτη Διεθνή Διάσκεψη για Θέματα Υπολογιστών και Επικοινωνιών (First International Conference on Computers and Communication), που έλαβε χώρα στην Washington DC (ΗΠΑ), οι επιστήμονες της ARPA έκαναν μια επίδειξη της λειτουργίας του συστήματος, συνδέοντας υπολογιστές μεταξύ τους από 40 διαφορετικές τοποθεσίες.

Το γεγονός αυτό ήταν το ερέθισμα για επιπλέον έρευνες στην επιστημονική κοινότητα του δυτικού κόσμου. Σύντομα εμφανίσθηκαν κι άλλα δίκτυα. Η Διάσκεψη της Washington ίδρυσε επίσης μια Διεθνή Ομάδα Εργασίας (Internetworking Working Group – IWG) με εργασία τον συντονισμό των σχετικών ερευνών. Εντωμεταξύ, οι επιστήμονες της ARPA εργάζονταν για τη βελτίωση του συστήματος και την επέκταση των δυνατοτήτων του :

Το 1972, εργάστηκαν με επιτυχία σ' ένα νέο πρόγραμμα που θα επέτρεπε την αποστολή μηνυμάτων μέσω του δικτύου, δίνοντας τη δυνατότητα για απευθείας επικοινωνία πρόσωπο με πρόσωπο, κάτι που το γνωρίζουμε σήμερα με τον όρο e-mail (ηλεκτρονική αλληλογραφία). Επίσης, στις αρχές της δεκαετίας του '70, οι επιστήμονες ανέπτυξαν τα πρωτόκολλα host-to-host. Πριν απ' αυτά, το σύστημα επέτρεπε μόνο σ' ένα απομακρυσμένο τερματικό (remote terminal) να έχει πρόσβαση στα αρχεία του κάθε ξεχωριστού host.

Το 1974, οι επιστήμονες της ARPA, εργαζόμενοι από κοινού με ειδικούς στο Stanford, ανέπτυξαν μια κοινή γλώσσα που θα επέτρεπε σε διαφορετικά δίκτυα να επικοινωνούν μεταξύ τους. Αυτό έγινε γνωστό με τον όρο transmission control protocol/internet protocol, το πασίγνωστο σήμερα CP/IP. Η ανάπτυξη του TCP/IP ήταν ένα σημαντικό στάδιο στην ανάπτυξη της δικτύωσης και είναι σημαντικό να επικεντρωθούμε στις επιπτώσεις που υπήρξαν στις έννοιες της σχεδίασης (design concepts). Αν και το 1974 σημάδεψε το ξεκίνημα του TCP/IP, θα χρειάζονταν αρκετά χρόνια τροποποιήσεων και επανασχεδίασης πριν ολοκληρωθεί και γίνει παγκοσμίως αποδεκτό.

Εντωμεταξύ, η δικτύωση των υπολογιστών (computer networking) αναπτυσσόταν ταχύτατα.

Το 1974 το Stanford δημιούργησε το Telnet, την πρώτη δημόσια ευρέως αποδεκτή υπηρεσία πακέτου δεδομένων (packet data service), μια εμπορική παραλλαγή του ARPANET. Τη δεκαετία του '70, το Υπουργείο Ενέργειας των ΗΠΑ καθιέρωσε το MFENet για τους ερευνητές που εργάζονταν στην Ενέργεια της Μαγνητικής Σύντηξης (Magnetic Fusion Energy), από το οποίο δημιουργήθηκε το HEPNet, που είναι αφοσιωμένο στη Φυσική Υψηλής Ενέργειας (High Energy Physics).

Αυτό ενέπνευσε τους φυσικούς της NASA στο να καθιερώσουν το SPAN για τους διαστημικούς φυσικούς (space physicists). Το 1976 αναπτύχθηκε ένα πρωτόκολλο Unix-to-Unix από τα εργαστήρια της AT&T Bell και διανεμήθηκε δωρεάν σ' όλους τους χρήστες υπολογιστών UNIX και εφόσον το UNIX ήταν τότε το βασικό λειτουργικό σύστημα που χρησιμοποιούσαν στα πανεπιστήμια, αυτό το γεγονός έκανε διαθέσιμη τη δικτύωση στην ευρύτερη ακαδημαϊκή κοινότητα.

Από το Internet στο World Wide Web

Μια σημαντική ανάπτυξη ήταν η εισαγωγή το 1984 των DNS (Domain Name Servers). Μέχρι τότε ο κάθε host υπολογιστής είχε εκχωρημένο ένα όνομα και υπήρχε μια μοναδική λίστα ονομάτων και διευθύνσεων την οποία μπορούσε εύκολα να συμβουλευθεί ο καθένας. Το νέο σύστημα εισήγαγε μερικά επιθέματα στις διευθύνσεις internet των ΗΠΑ, όπως edu (educational), com. (commercial), gov (governmental) εκτός από την org. (international organization) και μια σειρά από κωδικούς κρατών. Αυτό συνετέλεσε ώστε τα ονόματα των host υπολογιστών να είναι πιο ευκολομνημόνευτα.

Μια δεύτερη ανάπτυξη ήταν η απόφαση εθνικών κυβερνήσεων να ενθαρρύνουν τη χρήση του internet στην ανώτατη εκπαίδευση, ανεξάρτητα σχολής. Το 1984 η βρετανική κυβέρνηση ανακοίνωσε τη δημιουργία του JANET (Joint Academic Network) για την εξυπηρέτηση των βρετανικών πανεπιστημίων αλλά ακόμα πιο σημαντική ήταν η απόφαση, το επόμενο έτος, του US National Science Foundation να καθιερώσει το NSFNet για τον ίδιο σκοπό που η χρήση των πρωτοκόλλων TCP/IP ήταν υποχρεωτική για όλους τους συμμετέχοντες στο πρόγραμμα.

Τελικά, το NSFNet συμφώνησε να παρέχει τη ραχοκοκαλιά (backbone) για την υπηρεσία του Internet στις ΗΠΑ και παρείχε πέντε υπερυπολογιστές για την εξυπηρέτηση της αναμενόμενης κυκλοφορίας (traffic). Οι πρώτοι υπολογιστές παρείχαν μια χωρητικότητα δικτύου των 56.000 bytes ανά δευτερόλεπτο αλλά η χωρητικότητα αναβαθμίσθηκε το 1988 στα 1.544.000.000 bytes ανά δευτερόλεπτο.

Η επίδραση της δημιουργίας του NSFNet ήταν δραματική. Αρχικά έσπασε τη συμφόρηση της χωρητικότητας του συστήματος και κατά δεύτερο ενθάρρυνε την εκτόξευση στη χρήση του Internet. Χρειάσθηκε μια δεκαετία ώστε ο αριθμός των hosts υπολογιστών που είναι συνδεδεμένοι στο «Δίκτυο» να φθάσουν στους χίλιους. Μέχρι το 1986 ο αριθμός των hosts είχε φθάσει τις 5.000 και έναν χρόνο αργότερα ο αριθμός είχε σκαρφαλώσει στους 28.000 hosts.

Αν και είχε ξεκινήσει η εμπορική εκμετάλλευση του δικτύου, η επέκταση του Internet συνεχίσθηκε καθοδηγούμενη από την κυβέρνηση και τις ακαδημαϊκές κοινότητες. Γινόταν επίσης όλο και πιο διεθνές. Μέχρι το 1989 ο αριθμός των hosts ξεπέρασε τις 100.000 για πρώτη φορά και έφθασε τις 300.000 έναν χρόνο αργότερα.

Το τέλος της δεκαετίας του 1980 και το ξεκίνημα της δεκαετίας του 1990 αποδείχθηκε βολικό για πολλούς λόγους :

- Το 1990 το ARPANET (το οποίο είχε απεμπλακεί από τις στρατιωτικές έρευνές του το 1983) υπήρξε ένα θύμα της ίδιας του της επιτυχίας. Το δίκτυο είχε συρρικνωθεί σε μια αχνή σκιά του προηγούμενου εαυτού του.
- Το 1990, η πρώτη μηχανή αναζήτησης (search-engine) στο Internet για την εύρεση και την ανάκτηση αρχείων υπολογιστών, το Archie, αναπτύχθηκε στο Πανεπιστήμιο McGill, στο Montreal.
- Το 1991, το NSF απομάκρυνε την απαγόρευσή του για ιδιωτική πρόσβαση στους backbone υπολογιστές του.

Η Διαφορά Ανάμεσα στα Δύο

Πολλοί χρησιμοποιούν τους όρους Internet και World Wide Web εναλλάξ, αλλά στην πραγματικότητα οι δύο όροι δεν είναι συνώνυμοι. Το Internet και το Web είναι ξεχωριστά αλλά συσχετισμένα πράγματα. Το Internet είναι ένα τεράστιο δίκτυο από δίκτυα, μια δικτυακή υποδομή (networking infrastructure). Συνδέει εκατομμύρια υπολογιστών μαζί σ' όλον τον κόσμο, δημιουργώντας ένα δίκτυο στο οποίο ο κάθε υπολογιστής μπορεί να επικοινωνήσει μ' έναν οποιοδήποτε άλλον υπολογιστή για όσο χρόνο είναι και οι δύο συνδεδεμένοι στο Internet. Οι πληροφορίες μεταδίδονται μέσω του Internet μέσω μιας ποικιλίας από γλώσσες που είναι γνωστές σαν πρωτόκολλα (protocols).

Το World Wide Web, ή απλά Web, είναι ένας τρόπος πρόσβασης σε πληροφορίες μέσω του Internet. Είναι ένα μοντέλο διαμοιρασμού πληροφοριών που είναι κτισμένο στην κορυφή του Internet. Το Web χρησιμοποιεί το πρωτόκολλο HTTP, που είναι μια μόνο από τις γλώσσες που ομιλούνται στο Internet, για τη μεταφορά δεδομένων. Οι υπηρεσίες του Web, οι οποίες χρησιμοποιούν το HTTP για να επιτρέψουν στις εφαρμογές να επικοινωνούν ώστε να ανταλλάσσουν επιχειρηματικές πληροφορίες, χρησιμοποιούν το Web για να μοιράζονται πληροφορίες. Το Web χρησιμοποιεί επίσης και τους φυλλομετρητές (browsers), όπως είναι ο Internet Explorer ή Netscape, για να έχει πρόσβαση σε έγγραφα του Web που αποκαλούνται ιστοσελίδες (Web pages), οι οποίες συνδέονται μεταξύ τους με υπερσυνδέσμους (hyperlinks). Τα έγγραφα του Web (Web documents) περιέχουν επίσης γραφικά, ήχους, κείμενο και video.

Το Web αποτελεί έναν μόνο από τους τρόπους που οι πληροφορίες μπορούν να διασπαρούν στο Internet. Το Internet, όχι το Web, χρησιμοποιείται επίσης για την ηλεκτρονική αλληλογραφία (e-mail), η οποία βασίζεται στο πρωτόκολλο SMTP, στις ομάδες ειδήσεων (news groups) του Usenet, στο instant messaging και στο πρωτόκολλο FTP. Έτσι το Web αποτελεί απλά ένα κομμάτι του Internet, αν και ένα μεγάλο κομμάτι, αλλά οι δύο όροι δεν είναι συνώνυμοι και δεν θα πρέπει να συγχέονται.

Η Ιστορία του World Wide Web

Το World Wide Wed (Παγκόσμιος Ιστός) προτάθηκε από τον Tim Berners-Lee στο Ευρωπαϊκό Εργαστήριο για Πρακτική Φυσική (European Laboratory for Practical Physics – CERN) στη Γενεύη της Ελβετίας το 1989. Το 1993 ήταν το έτος του Mosaic, του πρώτου γραφικού φυλλομετρητή (Web browser). Ο Mosaic αναπτύχθηκε στο National Center for Supercomputing Applications (NCSA) και στο Πανεπιστήμιο του Illinois. Η πρώτη διανομή του Mosaic στο Internet προκάλεσε το τεράστιο ενδιαφέρον του κόσμου για το World Wide Web.

Το 1994, ο Marc Andreessen, ένας από τους δημιουργούς του Mosaic, άφησε το NCSA, ίδρυσε μαζί με άλλους την Netscape Communications Corp. και παρουσίασε στο κοινό τον Netscape Navigator, έναν γραφικό Web browser, τον Οκτώβριο του

1994. Ο δωρεάν διανεμόμενος Netscape Navigator για τα λειτουργικά συστήματα UNIX, Windows και Macintosh OS προκάλεσε το παγκόσμιο ενδιαφέρον του κοινού για το Internet και το Web. Σημάδεψε το ξεκίνημα της επιχειρησιακής εποχής του Internet. Το 1995, η Microsoft εισήλθε στην αγορά των Web browsers, παρουσιάζοντας την παραλλαγή (version) 1.0 του Internet Explorer. Από τότε και στο εξής, ξεκίνησε ο πόλεμος των Web browsers, ο πυρετός του χρυσού για το Internet, η εποχή του ηλεκτρονικού εμπορίου (e-commerce) …

Tim Berners-Lee : Ο Πατέρας του Web

Το World Wide Web δημιουργήθηκε το 1991, χάρις στον ερευνητή Tim Berners-Lee και άλλους στο Ευρωπαϊκό Εργαστήριο Σωματιδιακής Φυσικής (European Laboratory for Particle Physics), που είναι επίσης γνωστό σαν Conseil Europenne pour la Recherche Nuclaire (CERN). Η ομάδα του CERN δημιούργησε το πρωτόκολλο που βασίζεται στο υπερκείμενο (hypertext) το οποίο κάνει δυνατή τη σύνδεση του περιεχομένου στο Web με τους υπερσυνδέσμους (hyperlinks). Ο Berners-Lee διευθύνει τώρα το World Wide Web Consortium (W3C), που είναι μια ομάδα από εκπροσώπους της βιομηχανίας και των πανεπιστημίων και η οποία εποπτεύει τα standards της τεχνολογίας του Web.

Νωρίτερα, το Internet ήταν περιορισμένο σε μη εμπορικές χρήσεις επειδή το δίκτυο στήριξης (backbone) παρεχόταν κυρίως από το National Science Foundation, τη NASA (National Aeronautics and Space Administration) και το Υπουργείο Ενέργειας των ΗΠΑ, με χρηματοδότηση από την κυβέρνηση.

Αλλά καθώς τα ανεξάρτητα δίκτυα άρχισαν να φυτρώνουν, οι χρήστες μπορούσαν να έχουν πρόσβαση στα εμπορικά Web sites χωρίς να χρησιμοποιούν το χρηματοδοτούμενο από την κυβέρνηση δίκτυο. Μέχρι το τέλος του 1992, ο πρώτος εμπορικός παροχέας υπηρεσιών online (commercial online service provider), ο Delphi, προσέφερε πλήρη πρόσβαση στο Internet στους συνδρομητές του και ακολούθησαν και πολλοί άλλοι παροχείς (providers). Τον Ιούνιο του 1993, το Web είχε μόλις 130 sites, αλλά έναν χρόνο αργότερα, ο αριθμός είχε φθάσει σχεδόν στις 3.000 και τον Απρίλιο του 1998, υπήρχαν περισσότερα από 2,2 εκατομμύρια sites στο Web

Από το 1991 που δημιουργήθηκε το Web μέχρι το 1995 που ουσιαστικά ξεκίνησε ο πόλεμος ανάμεσα στις Netscape και Microsoft οι άνθρωποι που ασχολούνταν με το Internet, ήξεραν πως αν και ήταν σπουδαία αυτά που είχαν πετύχει σε πιλοτικό και ακαδημαϊκό επίπεδο, το Internet έπρεπε να στραφεί στην αγορά αν ήθελε να επιβιώσει και να συνεχίσει να αναπτύσσετε.

Από εκείνο το σημείο και μετά άρχισαν να αναπτύσσονται ραγδαία και οι εφαρμογές client-server οι οποίες σαφώς εξυπηρετούν κατά πολύ μεγάλο βαθμό τις ανάγκες τις αγοράς την οποία το μόνο που την ενδιαφέρει είναι η χρήση των νέων τεχνολογιών για τη μείωση του κόστους και την αύξηση των κερδών .

Ας δούμε όμως τι είναι τελικά αυτό που λέμε client-server computing,ποια η χρησιμότητα του, και ποια τα οφέλη που προκύπτουν από την ανάπτυξή του.

Αυτό που κυρίως όμως μας ενδιαφέρει να δούμε είναι οι λύσεις που έδωσε σε καθημερινά προβλήματα που αντιμετώπιζαν χιλιάδες χρηστές .

Όταν μια εταιρεία θέλει να δημιουργήσει ένα ηλεκτρονικό κατάστημα ή να προσφέρει κάποια άλλη υπηρεσία στους συνδρομητές του δικτύου (π.χ. πλειστηριασμοί, portal, job site κ.λπ.) έχει συνήθως να επιλύσει κάποια προβλήματα:
Τα σημαντικότερα από αυτά είναι :
-Χαμηλότερο κόστος (συνήθως)
-Εγγυημένη απόδοση (επειδή το χρησιμοποιούν και άλλοι είναι ευκολότερο να πληροφορηθούμε από αυτούς τα προτερήματα και τα ελαττώματά του)
-Καλή υποστήριξη (αν ο κατασκευαστής είναι μια μεγάλη και επώνυμη εταιρεία)
-Αυξημένες δυνατότητες (ενσωματώνει όχι μόνο τις ιδέες του κατασκευαστή, αλλά και τις γνώσεις, εμπειρίες και απαιτήσεις των πελατών του)

Τα πλεονεκτήματα της χρήσης του client server computing σε τέτοιες καταστάσεις είναι:
-Διαφοροποίηση (μπορείς να δημιουργήσεις κάτι που δεν υπάρχει πουθενά αλλού και να ξεχωρίσεις από τον ανταγωνισμό)
-Ταχύτητα προσαρμογής (μπορείς να αλλάξεις κάτι πολύ γρήγορα χωρίς να περιμένεις μέχρι να αναβαθμίσει το λογισμικό του ο παραγωγός του έτοιμου προϊόντος, ή ο πελάτης.)
-Customization (το λογισμικό κατασκευάζεται έτσι ώστε να καλύπτει απόλυτα τις ανάγκες του κάθε πελάτη)
-Επεκτασιμότητα. Συνήθως, για λόγους εξοικονόμησης πόρων, κάθε υπηρεσία Internet αρχίζει τη λειτουργία της με μια φθηνή πλατφόρμα υλοποίησης με πρόβλεψη αναβάθμισής της, αν αυτή αποδειχθεί δημοφιλής. Δυστυχώς όμως, υπάρχουν πολλές περιπτώσεις όπου οι αρχικές επιλογές δεσμεύουν την υπηρεσία και περιορίζουν τις δυνατότητες επέκτασής της. Γι' αυτό και κάθε αξιολόγηση πλατφόρμας εγκατάστασης υπηρεσιών Internet θα πρέπει να λαμβάνει υπ' όψιν της όχι μόνο τις τρέχουσες δυνατότητες της προτεινόμενης αρχιτεκτονικής, αλλά και την ικανότητά της να προσαρμοστεί στις μελλοντικές απαιτήσεις μας.
-Πραγματικό κόστος. Το πρόβλημα αυτό εμφανίζεται συνήθως στον τομέα του λογισμικού όπου πολλές εταιρείες προσφέρουν χαμηλές τιμές πρώτης αγοράς, αλλά απαιτούν εξοντωτικά license fees για επεκτάσεις τους σε περισσότερα μηχανήματα ή σε μεγαλύτερο αριθμό χρηστών. Η κατάσταση γίνεται ακόμη δυσκολότερη, καθώς κάθε εταιρεία έχει την τάση να χρησιμοποιεί ιδιόμορφους τρόπους τιμολόγησης με αποτέλεσμα να δυσχεραίνεται πάρα πολύ η σύγκριση μεταξύ διαφορετικών προϊόντων.
Στο client server computing αν και αυτό το πρόβλημα είναι μια πραγματικότητα, ωστόσο υπάρχει μια παγκόσμια-ελεύθερη κοινωνία προγραμματιστών που η κοινωνική ευαισθησία της δίνει τη δυνατότητα σε χιλιάδες επιχειρήσεις ανα την υφήλιο να δουλεύουν με αρκετά δύσχρηστα , αλλά free εργαλεία.

ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΕΝΝΟΙΑ ΤΟΥ CLIENT-SERVER COMPUTING

Τι είναι το client-server computing;

Γενικά, το client-server computing αναφέρεται σε μια βασική αλλαγή στο στυλ των υπολογιστών, την αλλαγή από τα συστήματα που βασίζονται στα μηχανήματα στα συστήματα που βασίζονται στον χρήστη.

Ειδικότερα, ένα σύστημα client-server είναι ένα σύστημα στο οποίο το δίκτυο ενώνει διάφορους υπολογιστικούς πόρους, ώστε οι clients (ή αλλιώς front end) να μπορούν να ζητούν υπηρεσίες από έναν server (ή αλλιώς back end), ο οποίος προσφέρει πληροφορίες ή επιπρόσθετη υπολογιστική ισχύ.

Με άλλα λόγια, στο client-server μοντέλο, ο client θέτει μια αίτηση και ο server επιστρέφει μια ανταπόκριση ή κάνει μια σειρά από ενέργειες. Ο server μπορεί να ενεργοποιείται άμεσα για την αίτηση αυτή ή να προσθέτει την αίτηση σε μια ουρά. Η άμεση ενεργοποίηση για την αίτηση μπορεί ,για παράδειγμα, να σημαίνει ότι ο server υπολογίζει έναν αριθμό και τον επιστρέφει αμέσως στον client. Η τοποθέτηση της αίτησης σε μια ουρά μπορεί να σημαίνει ότι η αίτηση πρέπει να τεθεί σε αναμονή για να εξυπηρετηθεί. Ένα καλό παράδειγμα για αυτό είναι όταν εκτυπώνουμε ένα κείμενο σε ένα εκτυπωτή δικτύου. Ο server τοποθετεί την αίτηση σε μια ουρά μαζί με αιτήσεις εκτυπώσεων και από άλλους clients. Μετά επεξεργάζεται την αίτηση με βάση την σειρά προτεραιότητας, η οποία , σε αυτή την περίπτωση, καθορίζεται από τη σειρά με την οποία ο server παρέλαβε την απαίτηση.

Το client-server computing είναι πολύ σημαντικό, διότι επιτυγχάνει τα εξής:

- Αποτελεσματική χρήση της υπολογιστικής ισχύος.
- Μείωση του κόστους συντήρησης, δημιουργώντας συστήματα client-server που απαιτούν λιγότερη συντήρηση και κοστίζουν λιγότερο στην αναβάθμιση.
- Αύξηση της παραγωγικότητας, προσφέροντας στους χρήστες ξεκάθαρη πρόσβαση στις αναγκαίες πληροφορίες μέσω σταθερών και εύκολων στην χρήση διασυνδέσεων.
- Αύξηση της ευελιξίας και της δυνατότητας δημιουργίας συστημάτων που υποστηρίζουν πολλά περιβάλλοντα.

Με βάση αυτούς τους σκοπούς, οι οργανισμοί που κινούνται προς την κατεύθυνση της client-server τεχνολογίας αυξάνουν κατά πολύ την ανταγωνιστική τους θέση.

Το βασικό client-server μοντέλο

Η πλευρά του client πρώτα στέλνει ένα μήνυμα για να καλέσει σε ετοιμότητα τον server. Από τη στιγμή που ο client και ο server έχουν επικοινωνία μεταξύ τους, ο client μπορεί να υποβάλλει την αίτησή του.

Client

Ο client είναι ο αιτών των υπηρεσιών. Ο client δεν μπορεί παρά να είναι ένας υπολογιστής. Οι υπηρεσίες που ζητούνται από τον client μπορεί να υπάρχουν στους ίδιους σταθμούς εργασίας ή σε απομακρυσμένους σταθμούς εργασίας που συνδέονται μεταξύ τους μέσω ενός δικτύου. Ο client ξεκινάει πάντα την επικοινωνία.

Τα συστατικά του client είναι πολύ απλά. Ένας client υπολογιστής πρέπει να μπορεί να κάνει τα ακόλουθα:

- Να τρέχει το λογισμικό των γραφικών διεπαφών χρηστών (GUIs).
- Να δημιουργεί τις αιτήσεις για πληροφορίες και να τις στέλνει στον server.
- Να αποθηκεύει τις επιστρεφόμενες πληροφορίες.

Αυτές οι αιτήσεις καθορίζουν πόση μνήμη χρειάζεται, ποια ταχύτητα επεξεργασίας θα μπορούσε να βελτιώσει τον χρόνο ανταπόκρισης, και πόση χωρητικότητα αποθήκευσης απαιτείται.

Server

Ο server απαντάει στις αιτήσεις που γίνονται από τους clients. Ένας client μπορεί να ενεργεί ως server εάν λαμβάνει και επεξεργάζεται αιτήσεις όπως ακριβώς και τις στέλνει (για παράδειγμα, ένας σταθμός εργασίας που χρησιμοποιείται και ως server εκτυπώσεων από άλλους). Οι server δεν ξεκινάνε τις επικοινωνίες -περιμένουν τις αιτήσεις των clients.

Επιστρέφοντας στο παράδειγμα του server εκτυπώσεων ενός δικτύου, ο client ζητάει από τον server να εκτυπώσει ένα κείμενο σε έναν συγκεκριμένο εκτυπωτή και ο server προσθέτει την εκτύπωση σε μια ουρά και ενημερώνει τον client όταν το κείμενο εκτυπωθεί επιτυχημένα. Η διαδικασία του client μπορεί να ανήκει φυσικά στον ίδιο σταθμό εργασίας με την διαδικασία του server. Στο παράδειγμα εδώ, μια εντολή εκτύπωσης μπορεί να εκδίδεται στον server του σταθμού εργασίας του δικτύου, χρησιμοποιώντας την διαδικασία του server εκτυπώσεων σε αυτόν τον σταθμό εργασίας.

Τα συστατικά του server είναι πολύ απλά. Μια server μηχανή πρέπει να μπορεί να κάνει τα ακόλουθα :

- Να αποθηκεύει, να ανακτά και να προστατεύει πληροφορίες.
- Να επιθεωρεί τις αιτήσεις των clients.
- Να δημιουργεί εφαρμογές διαχείρισης πληροφοριών, όπως δημιουργία αντιγράφων, ασφάλεια κτλ.
- Να διαχειρίζεται πληροφορίες.

Δίκτυα

Τα δίκτυα είναι τα πιο άγνωστα συστατικά στην εξίσωση των client-server. Γενικά οι άνθρωποι δεν ξέρουν πολλά για το πώς λειτουργούν τα δίκτυα στα συστήματα client-server, διότι τα συστήματα αυτά είναι σχεδιασμένα για να κάνουν τα δίκτυα διάφανα στον χρήστη. Επιπλέον, τα δίκτυα πρέπει να είναι αξιόπιστα. Πρέπει να μπορούν να υποστηρίζουν την επικοινωνία, να ελέγχουν σφάλματα και να ξεπερνούν αμέσως τις αποτυχίες.

Τα δίκτυα ελέγχονται από το λογισμικό λειτουργικών συστημάτων και διαχείρισης για να ελέγχουν τις υπηρεσίες επικοινωνίας του server και να προστατεύουν τα προγράμματα του client και του server από το να έχουν άμεση σύνδεση μεταξύ τους. Το λογισμικό διαχείρισης εστιάζεται στη παροχή αξιόπιστων υπηρεσιών, στην ελαχιστοποίηση των προβλημάτων στο δίκτυο και στην ελαχιστοποίηση των χρόνων «πτώσης» του δικτύου.

Πώς αναπτύχθηκε η client-server τεχνολογία;

Η τεχνολογία των υπολογιστών αναπτύχθηκε βαθμιαία, με τέτοιο τρόπο που κάθε καινούργια αρχιτεκτονική έπαιρνε τα πλεονεκτήματα από τις τεχνικές που ήδη υπήρχαν, ώστε να εκμεταλλεύεται όλες τις δυνατότητες των υπολογιστών. Σήμερα οι υπολογιστές είναι μικρότεροι, γρηγορότεροι και φθηνότεροι από ότι παλιότερα. Σαν αποτέλεσμα, η γενική κατεύθυνση είναι η διανομή της επεξεργασίας της πληροφορίας αλλά και της ίδιας της πληροφορίας σε ένα πλήθος αυτών των νέων υπολογιστών.

Ο όρος αρχιτεκτονική συνήθως χρησιμοποιείται για να περιγράφει συστήματα διαχείρισης βάσεων δεδομένων, λειτουργικά συστήματα και άλλους υπολογιστικούς μηχανισμούς λογισμικού και υλικού. Οι αρχιτεκτονικές περιγράφουν πως οι συσκευές και τα λογισμικά πακέτα ταιριάζουν για να φτιάξουν είναι εύκολο στην χρήση και στην διαχείριση σύνολο.

Η κλασσική αρχιτεκτονική αποτελείται από έναν υπολογιστή μεγάλης ισχύος, (που παίζει το ρόλο του οικοδεσπότη) με ένα ή περισσότερα απλά τερματικά. Οι εφαρμογές ελέγχονται και διανέμονται από τον υπολογιστή-«οικοδεσπότη». Σε αυτόν πραγματοποιούνται όλες οι διαχειρίσεις πληροφοριών, η λογική των εφαρμογών και η μορφοποίηση της εμφάνισης τους. Οι χρήστες αλληλεπιδρούν με το κεντρικό σύστημα μέσω των εφαρμογών που έχουν φτιαχτεί και είναι υπεύθυνες για την ομαλή επικοινωνία clients και server. Αυτή είναι η πιο συνηθισμένη αρχιτεκτονική σήμερα.

Ένα καλά οργανωμένο σύστημα που χρησιμοποιεί αυτήν την κλασσική αρχιτεκτονική προσφέρει τις ακόλουθες δυνατότητες:

- Ένα υψηλό επίπεδο αξιοπιστίας .
- Κεντρικό έλεγχο και κεντρική διαχείριση των πληροφοριών.
- Ισχυρή διαχείριση των πληροφοριών και δυνατότητα αποθηκεύσεων .

Παλαιότερα , οι εφαρμογές περιορίζαν την ευελιξία των τελικών χρηστών. Η διασύνδεση των χρηστών δεν ήταν γραφική, κάτι που έκανε το σύστημα

δυσκολότερο στη χρήση και σήμαινε ότι ο χρήστης έπρεπε να μάθει πως να χρησιμοποιήσει την εκάστοτε γλώσσα προγραμματισμού του server. Επίσης, οι εφαρμογές εξαρτώνταν από μια πλατφόρμα, που σημαίνει ότι εάν κάτι συνέβαινε στον υπολογιστή-«οικοδεσπότη», ο χρήστης δεν μπορούσε να χρησιμοποιήσει το σύστημα, έως ότου το σύστημα άρχιζε να επαναλειτουργεί.

Στην client-server αρχιτεκτονική, η client εφαρμογή τρέχει σε έναν πλήρη σταθμό εργασίας. Αυτός ο σταθμός μπορεί να είναι ένας προσωπικός υπολογιστής(PC), ένας UNIX σταθμός εργασίας ή ένας Mac. Η client εφαρμογή βασίζεται στις υπηρεσίες που προσφέρει ο server και επικοινωνούν μέσω πρωτοκόλλων, όπως το πρωτόκολλο του Internet (TCP/IP) ή του Novell (IPX/SPX).

Το περιβάλλον του client-server έχει πολλά πλεονεκτήματα σε σχέση με τις κλασσικές αρχιτεκτονικές. Η διαχείριση της διασύνδεσης των χρηστών και άλλες επεξεργασίες είναι αποφορτισμένα από τον «οικοδεσπότη», ενώ ο server ακόμη προσφέρει συγκεντρωμένο έλεγχο των κοινών πόρων. Επειδή ο client επικοινωνεί με τον server μέσω ενός καθορισμένου συστήματος διασύνδεσης, δεν χρειάζεται να γνωρίζει που ανήκει ο server ή πως ενεργεί. Ο σταθμός εργασίας τρέχει την εφαρμογή και εμφανίζει τις πληροφορίες στον χρήστη. Μόνο όταν ο client προσπελάζει πληροφορίες, τότε εγκαθίσταται επικοινωνία με τον server. Ο φόρτος εργασίας μειώνεται δραματικά στον υπολογιστή-«οικοδεσπότη» όσο αυξάνεται η ισχύς κάθε σταθμού εργασίας.

Οι οργανισμοί έχουν να κάνουν με συνεχώς περισσότερα δεδομένα, τα οποία πρέπει να τα διαχειρίζονται και να τα εκμεταλλεύονται στις εργασίες τους. Η αύξηση του όγκου των δεδομένων, σε συνδυασμό με την προσπάθεια των οργανισμών να μειώσουν το κόστος, να αυξήσουν την παραγωγικότητα και να βελτιώσουν τις υπηρεσίες των πελατών (με καλύτερη χρήση πληροφοριών και ταχύτερο χρόνο ανταπόκρισης στους πελάτες ταυτόχρονα), έχουν συμβάλει σε μια ώθηση για δημιουργία και χρήση client-server εφαρμογών.

Σε συνδυασμό με τις μεταβαλλόμενες απαιτήσεις των επιχειρήσεων, η ανάπτυξη της τεχνολογίας των client-server έχουν οδηγήσει στα ακόλουθα:

- Πρόοδο στο υλικό.
- Πρόοδο στο λογισμικό.
- Πρόοδο στο δίκτυο.

Πρόοδο στο λογισμικό

Η πρόοδος στο λογισμικό που χρησιμοποιείται από τα client-server συστήματα αύξησε πολύ την ευκολία και την αποτελεσματικότητα με την οποία μπορούν οι χρήστες να προσπελάσουν πληροφορίες. Στα πρώτα συστήματα οι χρήστές δεν μπορούσαν να ζητήσουν συγκεκριμένες εγγραφές. Οι χρήστες έπρεπε να φορτώσουν ένα ολόκληρο σετ δεδομένων από τον server και να προσπελάσουν τις πληροφορίες τοπικά.

Στο μεταξύ, άλλοι χρήστες έμεναν εκτός δικτύου, ή δεν μπορούσαν να προσπελάσουν το ίδιο σετ πληροφοριών. Πάντως, σήμερα οι clients μπορούν να

στείλουν αιτήσεις για συγκεκριμένες εγγραφές. Η χρήση της SQL επιτρέπει στους χρήστες να ζητούν συγκεκριμένες πληροφορίες χωρίς να χρειάζεται να μαθαίνουν πως να χρησιμοποιούν το λειτουργικό σύστημα του server και τα DBMS (Database Management System). Τα DBMS μπορούν όχι μόνο να προσπελάσουν πληροφορίες που ανήκουν σε μια δομή συγγενικών βάσεων δεδομένων, αλλά μπορούν να προσπελάσουν πληροφορίες που ανήκουν σε διαφορετικού τύπου μηχανές.

Αυτό που κάνει τις πληροφορίες εύκολα προσπελάσιμες στον χρήστη είναι η ανάπτυξη σταθερών, εύκολων στη χρήση γραφικών διασυνδέσεων για τους χρήστες. Η γραφική διεπαφή των χρηστών (GUI) είναι το μέσο με το οποίο ο τελικός χρήστης επικοινωνεί με την εφαρμογή.

Οι σταθμοί εργασίας είναι υπεύθυνοι για την εμφάνιση των πληροφοριών και χρησιμοποιούν πιο σύνθετες γραφικές απεικονίσεις, που περιλαμβάνουν πιο πολύπλοκα γραφικά ή ακόμα και κίνηση. Η αρχή των γραφικών διεπαφών χρηστών (GUIs) βασίζεται στην αντίληψη ότι οι άνθρωποι ανταποκρίνονται καλύτερα σε εικόνες παρά σε λέξεις. Η πραγματικότητα έχει δείξει ότι τα πιο δημοφιλή συστήματα υπολογιστών χρησιμοποιούν γραφική διεπαφή χρήστη (GUI).

Τα γραφικά παράθυρα επιπλέον επιτρέπουν την πραγματοποίηση πολλών διαφορετικών εργασιών μέσα στην ίδια εφαρμογή. Τα πολλαπλά παράθυρα ή ακόμα και οι πολλαπλές εκδοχές του ίδιου του παραθύρου, είναι τώρα δυνατό να υπάρχουν στην ίδια οθόνη μέσα στην ίδια την εφαρμογή.

Η πολυνηματική (Multi-threaded) επεξεργασία είναι επίσης μια από τις πιο σημαντικές εξελίξεις στην πληροφορική. Τα treads είναι η δυνατότητα του λειτουργικού συστήματος, που επιτρέπει στις εφαρμογές να τρέχουν πολλές διεργασίες ταυτόχρονα. Τα αρχικά λειτουργικά συστήματα ήταν single-threaded, που σήμαινε ότι μπορούσαν να εκτελέσουν μόνο μια διεργασία την φορά. Πάντως, τα συστήματα client-server μπορούν να προσπελάσουν πολλά threads την φορά, που σημαίνει ότι οι clients-servers μπορούν να κάνουν αποτελεσματική χρήση των λογισμικών και του υλικού των συστημάτων.

ΣΥΣΤΑΤΙΚΑ ΤΟΥ CLIENT-SERVER COMPUTING: Ποιος είναι ο ρόλος του client;

Η διαδικασία client-server μπορεί να απλοποιηθεί στα ακόλουθα βήματα:

1. Ο χρήστης δημιουργεί μια αίτηση ή ένα ερώτημα.
2. Ο client μορφοποιεί το ερώτημα και το στέλνει στο server.
3. Ο server ελέγχει την δυνατότητα πρόσβασης του χρήστη.
4. Ο server επεξεργάζεται το ερώτημα και επιστρέφει τα αποτελέσματα.
5. Ο client λαμβάνει την ανταπόκριση και τη μορφοποιεί για τον χρήστη.
6. Ο χρήστης βλέπει και χειρίζεται την πληροφορία.

Πέρα από τα έξι αυτά βήματα, ο client παίζει τέσσερις βασικούς ρόλους. Ο client είναι στην πραγματικότητα το κέντρο της client-server εφαρμογής. Ο χρήστης αλληλεπιδρά με τον client, ο client ξεκινάει το μεγαλύτερο μέρος της ανάπτυξης της εφαρμογής, και ο server υπάρχει για να απαντάει στις ανάγκες του client.

Ο client εκτελεί τις ακόλουθες λειτουργίες :

- Προσφέρει μια εύκολη στη χρήση διασύνδεση χρηστών.
- Στέλνει απαιτήσεις.
- Δέχεται ανταποκρίσεις.
- Επιτρέπει στον χρήστη να βλέπει και να χειρίζεται τις πληροφορίες.

Για κάθε έναν από τους τέσσερις ρόλους, δηλαδή της παροχής μιας εύκολης στη χρήση διασύνδεσης, της αποστολής απαιτήσεων, της αποδοχής ανταποκρίσεων και της δυνατότητας στο χρήστη να παίρνει και να χειρίζεται πληροφορίες, ο client έχει συγκεκριμένες ευθύνες.

Παροχή μια εύκολης στη χρήση διασύνδεσης

Μια εύκολη στη χρήση διασύνδεση αποτελείται από δυο σημαντικές εργασίες: αποδοχή των εισερχόμενων και εμφάνιση των εξερχόμενων. Για παράδειγμα, ο client δέχεται τα εισερχόμενα, επιτρέποντας σε κάποιον που τροφοδοτεί κάποια πράγματα να διαθέσει μια ειδική παραγγελία σε έναν πελάτη. Ο client μπορεί επίσης να εμφανίσει τις πληροφορίες του πελάτη στον τροφοδότη.

Η διασύνδεση των χρηστών είναι ένα από τα πιο σημαντικά κομμάτια της client εφαρμογής. Ελέγχει την όψη (τα στοιχεία της οθόνης) και την αίσθηση (τον τρόπο που ο χρήστης κάνει αιτήσεις και παίρνει απαντήσεις) του προγράμματος.

Η ανάπτυξη του client βασίζεται σε αρχές σχεδίασης εστιασμένες στον χρήστη. Αυτές οι αρχές είναι οι ακόλουθες:

- Διατηρεί τη διασύνδεση συνεπή, ώστε οι χρήστες να πάρουν μια οικεία όψη και αίσθηση από τις εφαρμογές και τις πλατφόρμες.
- Δεν ξεχνά ότι ο υπολογιστής εξυπηρετεί τον χρήστη. Ο χρήστης θα πρέπει να ελέγχει την σειρά των εργασιών. Ο υπολογιστής δεν θα πρέπει ποτέ να αγνοεί τον χρήστη, αλλά να διατηρεί τον χρήστη ενήμερο και να προσφέρει άμεσες απαντήσεις.
- Χρησιμοποιεί μεταφορές, τόσο φραστικές, όσο και οπτικές για να βοηθήσει τους χρήστες να αναπτύξουν θεμελιώδεις απεικονίσεις. Για παράδειγμα, η αποθήκευση αρχείων σε φακέλους στον υπολογιστή, ώστε ο χρήστης να μπορεί να συγκεντρωθεί στη δουλειά παρά να αποκαλύπτει πως λειτουργεί ο υπολογιστής.
- Δεν ζητάει από τον χρήστη να θυμάται εντολές. Οι εντολές μπορούν να είναι διαθέσιμες στον χρήστη για να τις επιλέγει, ώστε ο χρήστης να μπορεί να βασίζεται στην αναγνώριση, παρά στην απομνημόνευση.
- Επιτρέπει στην διασύνδεση να συγχωρεί τα λάθη του χρήστη. Οι καταστροφικές ενέργειες απαιτούν επιβεβαίωση, και οι χρήστες μπορούν να ανατρέψουν ή να ακυρώσουν την τελευταία ενέργεια.

Πρότυπα: Μια πρότυπη εγκαταστημένη διασύνδεση παρέχει εγγύηση ότι οι κατευθυντήριες γραμμές έχουν δοκιμαστεί για συνέπεια και εύκολη αποτελεσματική χρήση. Η IBM, η Microsoft και η Macintosh ανήκουν στα διαθέσιμα πρότυπα.

Στον υπολογιστή, οι εφαρμογές και τα αντικείμενα αναπαρίστανται ως εικονίδια (μικρές γραφικές εικόνες). Το εικονίδιο είναι προκαθορισμένο να ξεκινάει το πρόγραμμα, το οποίο εμφανίζει το ίδιο το παράθυρο, στο οποίο η εφαρμογή ή η συγκεκριμένη εργασία μπορεί να εκτελεστεί. Για παράδειγμα, ένα πρόγραμμα εκτύπωση μπορεί να αναπαρασταθεί από μια μικρή εικόνα ενός εκτυπωτή. Χρησιμοποιώντας ένα δείκτη για να κάνουμε διπλό κλικ στο εικονίδιο, η εφαρμογή μπορεί τότε να ξεκινήσει. Η εφαρμογή εκτύπωσης ανοίγει ένα παράθυρο στο οποίο η ουρά εκτύπωσης μπορεί να εμφανιστεί. Τα εικονίδια, οι δείκτες και τα παράθυρα είναι μερικά από τα στοιχεία του GUIs. Άλλα στοιχεία είναι: οι μπάρες ολίσθησης , οι κέρσορες, τα controls και η βοήθεια.

Τα εικονίδια μπορούν να αναπαραστήσουν μια εφαρμογή στον υπολογιστή. Ο χρήστης μπορεί να χειρίζεται τα εικονίδια κατευθείαν για να ξεκινάει εφαρμογές, να μετακινεί και να αποθηκεύει αρχεία ή να ανοίγει αρχεία. Τα Windows εμφανίζουν όψεις των αρχείων και των αντικειμένων που είναι δυνατόν να διαχειρίζονται. Τα Windows επίσης εμφανίζουν μηνύματα και οδηγίες βοήθειας, και παρουσιάζουν επιλογές που μπορούν να πραγματοποιηθούν. Περισσότερα από ένα παράθυρα μπορούν να είναι ανοιχτά την ίδια στιγμή .

Διάφοροι τύποι παραθύρων επιτρέπουν τον καλύτερο έλεγχο των εφαρμογών. Το ενεργό παράθυρο είναι ένα παράθυρο στο οποίο επιτελείται εργασία. Ορισμένες φορές το σύστημα χρειάζεται περισσότερες πληροφορίες για να ολοκληρώσει μια απαιτούμενη εργασία, οπότε το σύστημα εμφανίζει ένα δευτερεύον παράθυρο. Για παράδειγμα, αν δουλεύουμε σε ένα αρχείο επεξεργασίας κειμένου και επιλέξουμε την επιλογή save, ένα άλλο παράθυρο εμφανίζεται, ζητώντας το όνομα του αρχείου και την τοποθεσία αποθήκευσης του. Άλλα παράθυρα δίνουν μηνύματα που επιτρέπουν να μαθαίνουμε ότι κάτι έχει συμβεί (για παράδειγμα, κάποιο σφάλμα) ή ότι κάποια ενέργεια βρίσκεται σε εκτέλεση.

Μερικές φορές οι πληροφορίες που εμφανίζονται δεν χωράνε στο παράθυρο. Σε αυτές τις περιπτώσεις, μια μπάρα ολίσθησης χρησιμοποιείται για να ανεβοκατεβαίνουμε μέσα στο παράθυρο. Για παράδειγμα, αν έχουμε μια λίστα από 100 ονόματα, το πιθανότερο είναι να μην γίνουν όλα ορατά αμέσως μέσα στο παράθυρο. Πάντως, μπορούμε να χρησιμοποιήσουμε τις μπάρες ολίσθησης για να δούμε όλα τα ονόματα που βρίσκονται στα σημεία του παραθύρου που δεν είναι προς στιγμήν ορατά.

Οι δείκτες και οι δρομείς δείχνουν στον χρήστη το που θα πραγματοποιηθεί η ενέργεια. Τα αντικείμενα επιλέγονται αφού τα προσδιορίσουμε, συνήθως φωτίζοντας τα με ένα κέρσορα ή επιλέγοντας τα με ένα δείκτη (όπως το ποντίκι). Από την στιγμή που το αντικείμενο επιλέγεται, μπορούμε να αποφασίσουμε εάν θέλουμε ή όχι να εφαρμόσουμε την ενέργεια. Για παράδειγμα, μπορούμε να επιλέξουμε ένα εικονίδιο μιας εφαρμογής, κάνοντας κλικ μια φορά με τον δείκτη. Τότε μπορούμε να αποφασίσουμε εάν θέλουμε ή όχι να ξεκινήσουμε την εφαρμογή.

Τα controls επιτρέπουν την επιλογή των ενεργειών. Για παράδειγμα, ένα μενού είναι ένα control. Μπορούμε να εμφανίσουμε το μενού και να επιλέξουμε μια από τις

επιλογές. Ορισμένα controls επιτρέπουν την δυνατότητα να κάνουμε επιλογές είτε χρησιμοποιώντας έναν δείκτη, είτε τυπώνοντας οδηγίες, που επιτρέπουν τη χρησιμοποίηση της πιο αποτελεσματικής μεθόδου εργασίας.

Η on-line βοήθεια είναι ένα σταθερό χαρακτηριστικό των περισσοτέρων εφαρμογών τώρα. Η βοήθεια δίνει στον χρήστη περισσότερες πληροφορίες για μια επιλογή, για ένα πεδίο ή για το πώς εκτελείται μια εργασία. Η βοήθεια συνήθως προσπελάζεται από την μπάρα των μενού, αν και μπορεί επίσης να προσπελαστεί με το πάτημα ενός κουμπιού ή ενός κλειδιού.

Η δυνατότητα αλληλεπίδρασης που παρουσιάστηκαν παραπάνω είναι μόνο ένα τμήμα των δυνατοτήτων που είναι διαθέσιμες. Χρησιμοποιώντας έναν καλό οδηγό προτύπων, μπορούμε να βεβαιωθούμε ότι προσφέρεται στον χρήστη ένας πιο εύκολος, πιο αποτελεσματικός τρόπος για να ολοκληρώσει τις εργασίες του και ότι η διασύνδεση είναι συνεπής.

Αποστολή Αιτήσεων

Το να σταλεί μια αίτηση σημαίνει την μορφοποίηση της αίτησης και την αποστολή της στον server, με τρόπο τέτοιο που να μπορεί ο τελευταίος να καταλάβει. Εάν ο τροφοδότης θέλει να εμφανίσει όλες τις απλήρωτες ειδικές παραγγελίες, ο client μορφοποιεί την αίτηση σε SQL που χρησιμοποιείται από τον server DBMS (Database Management System) και την στέλνει μέσω του δικτύου στον server.

Το βασικό πλεονέκτημα των συστημάτων client- server είναι ότι η λογική της εφαρμογής και η βάση δεδομένων είναι χωρισμένα. Ο διαχωρισμός αυτός προσφέρει τα παρακάτω πέντε ευδιάκριτα πλεονεκτήματα:

- Από την στιγμή που η βασική επεξεργασία γίνεται στον server, ο client δεν χρειάζεται τόση πολύ ισχύ, και οι πόροι δεν δεσμεύονται μετά την αποστολή των απαιτήσεων.
- Διαχωρίζοντας τη λογική, μειώνεται ο φόρτος στο δίκτυο, επειδή το δίκτυο δεν χρειάζεται να διαβιβάζει ολόκληρα αρχεία πίσω και μπροστά. Χρησιμοποιώντας SQL, απλοποιούνται τα προβλήματα στο δίκτυο με ερωτήματα και απαντήσεις. Αυτή η απλοποίηση μπορεί να είναι τεράστια, ειδικά σε μεγάλα δίκτυα με πολλούς clients.
- Οι χρήστες δεν περιορίζονται σε μια client πλατφόρμα. Η SQL τυποποιεί τα ερωτήματα ώστε οι πληροφορίες να μπορούν να μεταφέρονται από μια πλατφόρμα σε άλλη με ξεκάθαρο τρόπο.
- Έχοντας τη βάση δεδομένων στον server, διαφυλάσσεται η ακεραιότητα των πληροφοριών. Είναι δυνατό να προσφέρονται υπηρεσίες, όπως ασφάλεια και προστασία πληροφοριών, δημιουργία αντιγράφων ασφαλείας, κ.τ.λ.. Αυτό το επίπεδο προστασίας των πληροφοριών γίνεται πιο δύσκολο, όσο ο έλεγχος γίνεται πιο αποκεντρωμένος (όπως με ένα πιο κατανεμημένο σύστημα)
- Η εκτέλεση της επεξεργασίας ενημερώνει όλες τις αλλαγές που έγιναν στη βάση δεδομένων σε μια χρονική περίοδο για να βεβαιωθεί ότι οι τροποποιήσεις έχουν καταγράφει κανονικά. Η ενημέρωση μπορεί επίσης να χρησιμοποιηθεί για να ανακτηθεί η βάση δεδομένων σε περίπτωση που «πέσει» το σύστημα.

Το πώς ο χρήστης κάνει την αίτηση εξαρτάται από τον client. Γενικά, ο client προτρέπει τον χρήστη να μπει στα πεδία για να ψάξει και μετά δημιουργεί την

πραγματική απαίτηση σε SQL. Για παράδειγμα, ένας τροφοδότης θέλει να δημιουργήσει μια λίστα από συγκεκριμένους πελάτες που περιμένουν να παραλάβουν ειδικές παραγγελίες που πραγματοποιήθηκαν πάνω από 30 ημέρες πριν. Τα κριτήρια του ερωτήματος θα περιλαμβάνουν συγκεκριμένους πελάτες που περιμένουν να παραλάβουν ειδικές παραγγελίες που πραγματοποιήθηκαν πάνω από 30 ημέρες πριν.

Το πιο πιθανό είναι ότι η εφαρμογή θα προτρέψει τον τροφοδότη να εισάγει στο πεδίο των ερωτημάτων το εξής:
customers = current, status = expecting, date > 30 days.

Τα RPCs (Remote Procedure Calls) ελέγχουν την πραγματική αποστολή της αίτησης του τροφοδότη. Τα RPCs κάνουν την σχεδίαση των client-server εφαρμογών ευκολότερη, διότι κάνουν πολλές από τις λεπτομέρειες του δικτύου ξεκάθαρες όχι μόνο στον χρήστη, αλλά επίσης και στον χειριστή του client. Το πώς οι ίδιες πληροφορίες είναι κατανεμημένες επίσης επιδρά στο πώς η αλληλεπίδραση διαβάσματος και εγγραφής μπορεί να καθορίσει το πόσο καλά θα τοποθετήσουμε τις πληροφορίες για να ελαχιστοποιήσουμε τα προβλήματα στο δίκτυο.

Οι πληροφορίες μπορούν να τοποθετηθούν με τρεις βασικούς τρόπους:
- Επικεντρωμένες στον server
- Τοποθετημένες σε πολλαπλά πανομοιότυπα αντίγραφα σε διαφορετικές τοποθεσίες
- Χωρισμένες σε διάφορες τοποθεσίες

Η συγκέντρωση των πληροφοριών στον server προσφέρει περισσότερο έλεγχο, διότι υπάρχει μόνο ένα αντίγραφο για να προσπελάσουμε και να διατηρήσουμε, και περιέχει όλες τις απαραίτητες πληροφορίες. Ωστόσο, εάν ο server ‘πέσει’, οι πληροφορίες δεν είναι διαθέσιμες. Επίσης, συγκεντρώνοντας τις πληροφορίες, η δραστηριότητα στο δίκτυο αυξάνεται, διότι όλοι πρέπει να προσπελάσουν τις ίδιες πληροφορίες στην ίδια περιοχή.

Τα βασικά πλεονεκτήματα του είναι να έχουμε πολλαπλά πανομοιότυπα αντίγραφα, είναι η βελτίωση στην επίδοση και η ασφάλεια της ύπαρξης περισσοτέρων από ένα αντιγράφων σε περίπτωση που ο server ‘πέσει’. Όταν δυο ομάδες χρηστών σε διαφορετικούς κόμβους σε ένα δίκτυο, προσπελάσουν την ίδια πληροφορία, μπορεί να βελτιωθεί σημαντικά η επίδοση όταν υπάρχουν δυο αντίγραφα της πληροφορίας, έναν σε κάθε κόμβο.

Υπάρχουν δυο βασικές επιλογές όταν υπάρχουν πολλαπλά αντίγραφα:
- Οι χρήστες μπορούν να αντιγράφουν τις απαιτούμενες πληροφορίες στις δικές τους μηχανές και να δουλέψουν πάνω σε αυτές εκεί.
- Ένα σύστημα διαχείρισης βάσεων δεδομένων μπορεί να διανείμει αντίγραφα της πληροφορίας σε τακτικά διαστήματα.

Χρησιμοποιώντας DBMS είναι πιο αξιόπιστο από να αφήνεις τους χρήστες να δουλεύουν με πολλαπλά αντίγραφα της πληροφορίας, διότι το DBMS χειρίζεται τον συγχρονισμό των πολλαπλών αντιγράφων της πληροφορίας. Αφήνοντας τους χρήστες να αντιγράφουν την πληροφορία που χρειάζονται δεν είναι αρκετό, όταν χρειάζεται να είμαστε σίγουροι ότι οι χρήστες δουλεύουν σε ενημερωμένα, συγχρονισμένα αντίγραφα της πληροφορίας.

Ο χωρισμός της πληροφορίας σε διάφορες περιοχές είναι πιο πολύπλοκος και λιγότερο αξιόπιστος. Ο χωρισμός δημιουργεί διάφορα δυναμικά προβλήματα που πρέπει να διαχειριστούμε. Οι τεμαχισμένες πληροφορίες πρέπει να είναι ολοφάνερες στον χρήστη. Ως αποτέλεσμα, ο χρόνος επεξεργασίας αυξάνεται, διότι κάθε τμήμα πρέπει να ανακτηθεί και να συναρμολογηθεί.

Πρότυπα: Τα πρότυπα βοηθούν να σχεδιαστεί ο τρόπος, με τον οποίο ο client στέλνει τις αιτήσεις στον server. Η ανάπτυξη της SQL προσφέρει μια σταθερή μέθοδο για προσπέλαση πληροφοριών. Τα RPCs προσφέρουν ξεκάθαρη επικοινωνία μεταξύ των προγραμμάτων. Τα IPCs (Interprocess Communications protocols) προσφέρουν τα πρότυπα για να κάνουν τις client-server αλληλεπιδράσεις διάφανες στον χρήστη, καθώς επίσης και στους ίδιους τους clients και servers.

Η SQL επιτρέπει στους χρήστες να δημιουργούν ερωτήματα, χωρίς να χρειάζεται να μάθουν κάτι για το λειτουργικό σύστημα του server ή το DBMS.

Η SQL επιτρέπει τα ακόλουθα:
- Μια απλή μέθοδο προσπέλασης πληροφοριών σε πολλαπλές πλατφόρμες.
- Μια απλή μέθοδο προσπέλασης πληροφοριών σε πολλαπλές εφαρμογές.
- Απλοποίηση των προβλημάτων του δικτύου, διότι, αντί για μεταφορά ολόκληρων αρχείων, μόνο οι απαιτούμενες πληροφορίες επιστρέφονται.

Τα RPCs είναι σχεδιασμένα να μοιάζουν σαν κλήσεις τοπικών διαδικασιών στην εφαρμογή. Από την στιγμή που γίνεται η κλήση, η τρέχουσα βιβλιοθήκη είναι υπεύθυνη για να βρει την απόμακρη διαδικασία και να χειριστεί την επικοινωνία.

Τα IPCs επιτρέπουν δυο εφαρμογές να τρέχουν στο ίδιο ή σε διαφορετικό περιβάλλον για να στείλουν και να λάβουν απαιτήσεις και ανταποκρίσεις .

Τα IPCs είναι πολύ σημαντικά για το client, γιατί εκτελούν τα ακόλουθα:
- Συντονίζουν το πως οι clients στέλνουν απαιτήσεις στον server λαμβάνουν ανταποκρίσεις από τον server.
- Ελέγχουν την ταχύτητα μεταφοράς πληροφοριών μεταξύ του client και του server.
- Καθιστούν την τοποθεσία του δικτύου του server διάφανη στον client.

Ένα απλό παράδειγμα ενός IPC που χρησιμοποιείται σήμερα είναι τα named pipes. Τα named pipes προσφέρουν διάφανη επικοινωνία μεταξύ των προγραμμάτων. Τα named pipes είναι σαν αρχεία που πολλοί επεξεργαστές μπορούν να χρησιμοποιήσουν ταυτόχρονα. Αυτά μπορεί να είναι write-only, read-only ή write-and-read.

Λήψη Απόκρισης και Διαχείριση της Πληροφορίας

Ο server ελέγχει την δυνατότητα πρόσβασης του χρήστη, επεξεργάζεται το ερώτημα, και επιστρέφει την αιτούμενη πληροφορία. Σε αυτό το σημείο, ο client δέχεται τα αποτελέσματα και τα μετατρέπει σε μια μορφή που μπορεί να χρησιμοποιήσει ο client.

Η διαδικασία της λήψης της απόκρισης είναι παρόμοια με την διαδικασία που χρησιμοποιεί ο client για να στείλει αιτήσεις, αλλά αντιστροφή.

Μετά, το API (Application Programming Interface) εξάγει την απαιτούμενη πληροφορία και την περνάει στην εφαρμογή. Με άλλα λόγια, αντί να μορφοποιεί την αίτηση και να την στέλνει στον client, ο client δέχεται την απόκριση και την μορφοποιεί για τον χρήστη. Το τελευταίο βήμα είναι η δυνατότητα στον χρήστη να βλέπει και να χειρίζεται την πληροφορία.

Από την στιγμή που η πληροφορία επιστρέφει, το πλεονέκτημα είναι ότι ο client θα θέλει να πάρει κάποια πρωτοβουλία. Σε αυτό το σημείο η εφαρμογή υιοθετεί ξανά τον πρώτο της ρόλο: προσφέρει μια εύκολη στη χρήση αλληλεπίδραση για να δεχθεί εισερχόμενα δεδομένα και να εμφανίσει εξερχόμενα.

ΣΥΣΤΑΤΙΚΑ ΤΟΥ CLIENT-SERVER COMPUTING: Ο SERVER

Αν και ο client κατέχει ένα μεγάλο μέρος της προσοχής, αφού με τον client αλληλεπιδρούν οι χρήστες, ο server είναι η καρδιά του client-server συστήματος. Οι servers είναι τα σημεία όπου αποθηκεύονται οι πληροφορίες και εκτελούνται οι εργασίες. Σήμερα, ο server μπορεί να είναι οποιαδήποτε μορφή υπολογιστή. Ωστόσο η αύξηση της ισχύως και η μείωση του κόστους των προσωπικών υπολογιστών τους κάνει γενικά την πιο συμφέρουσα οικονομικά επιλογή. Ακόμα και αν ο server είναι ένας σταθερός προσωπικός υπολογιστής αυτό που κάνει τη διαφορά από ένα σταθερό προσωπικό σύστημα είναι ότι είναι εξειδικευμένος και έχει συγκεκριμένες πρωτοβουλίες.

Τύποι των Servers

Οι servers μπορούν να διαιρεθούν σε έξι τύπους:

- Server Εφαρμογών (Application servers).
- Server Πληροφοριών (Data servers).
- Server Υπολογισμών (Compute servers).
- Server Βάσεων Δεδομένων (Database servers).
- Server Πόρων ή Επικοινωνιών (Resource or Communications servers).

Ο τύπος του server που χρησιμοποιείται εξαρτάται από την απαιτούμενη εργασία. Επίσης, αυτοί οι έξι ρόλοι μπορούν να συνδυαστούν σε ένα σύστημα ή να διαιρεθούν σε περισσότερα. Για παράδειγμα, η ίδια μηχανή μπορεί να εξυπηρετήσει σαν ένας server εφαρμογών και ένας server βάσεων δεδομένων.

Οι περισσότεροι servers που χρησιμοποιούνται σήμερα στις επιχειρήσεις είναι servers αρχείων (file servers). Οι servers αρχείων επιτρέπουν στους clients να

προσπελάσουν αρχεία και να μοιραστούν πληροφορίες και λογισμικό. Αυτοί οι servers είναι συνήθως ένας προσωπικός υπολογιστής ή ένα UNIX σύστημα με έναν επεξεργαστή. Πολλοί άνθρωποι μπορούν να προσπελάσουν τον server αρχείων την ίδια στιγμή, που σημαίνει ότι ο server έχει πολλαπλές μονάδες δίσκων και κάρτες προσαρμογής δικτύου, αλλά μόνο ένα άτομο μπορεί να προσπελάσει ένα συγκεκριμένο αρχείο εκείνη τη στιγμή.

Server Εφαρμογών (Application servers)

Οι servers εφαρμογών (application servers) τρέχουν λογισμικό εφαρμογών, που είναι πολύ σημαντικό όταν διανέμονται λογικές εφαρμογών μεταξύ του client και του server. Η τοποθέτηση εφαρμογών στον server σημαίνει ότι αυτές οι εφαρμογές είναι διαθέσιμες σε πολλούς clients. Πολλοί clients μπορούν να χρησιμοποιήσουν τα RPCs(Remote Procedure Calls) για να θέσουν σε λειτουργία μια επεξεργασία στον server. Πολλοί servers εφαρμογών μπορούν ακόμα και να εργαστούν μαζί για να απαντήσουν στην απαίτηση του client. Κάθε server μπορεί να τρέξει ένα διαφορετικό λειτουργικό σύστημα σε μια διαφορετική πλατφόρμα υλικού, αλλά αυτές οι λεπτομέρειες είναι ξεκάθαρες στον client -ο client μπορεί να κάνει αιτήσεις χωρίς να υπολογίζει τον τύπο της μηχανής που θα ανταποκριθεί.

Server Πληροφοριών (Data servers)

Οι servers πληροφοριών (data severs) χρησιμοποιούνται μόνο για αποθήκευση και διαχείριση πληροφοριών και χρησιμοποιούνται σε συνδυασμό με servers υπολογισμών (compute servers). Αυτοί οι servers ερευνούν και ελέγχουν την αξιοπιστία των πληροφοριών, αλλά γενικά δεν μεταβιβάζουν μεγάλη ποσότητα πληροφοριών στο δίκτυο.

Server Υπολογισμών (Compute servers)

Οι servers υπολογισμών (compute servers) παίρνουν τις αιτήσεις των clients για πληροφορίες στον server πληροφοριών και μετά προωθούν τα αποτελέσματα των αιτήσεων πίσω στον client.

Server Βάσεων Δεδομένων (Database servers)

Οι servers βάσεων δεδομένων (database servers) είναι τυπικά client-server συστήματα, και έχουν να κάνουν την ίδια εργασία με αυτή που κάνουν οι servers πληροφοριών και υπολογισμών μαζί. Οι servers βάσεων δεδομένων τρέχουν DBMS (Database Management System) λογισμικό και πολύ πιθανό και κάποια λογική client-server εφαρμογή, που σημαίνει ότι αυτός ο τύπος του server χρειάζεται περισσότερη ισχύ. Τα DBMS προσφέρουν εξειδικευμένες υπηρεσίες: την δυνατότητα να ανακτά πληροφορίες και να διαχειρίζεται πληροφορίες. Οι servers που συνδυάζουν τις λειτουργίες του server βάσεων δεδομένων και του server εφαρμογών είναι επίσης γνωστοί ως server συναλλαγών (transaction servers).

Server Πόρων ή Επικοινωνιών (Resource or Communication servers)

Οι servers πόρων (resource servers), που περικλείουν τους servers επικοινωνιών (communications servers) επιτρέπουν σε πολλούς clients την προσπέλαση συγκεκριμένων πόρων, που είναι ουσιαστικά πολύ ακριβοί για να βρίσκονται σε έναν client. Για παράδειγμα, οι servers εκτυπώσεις (print servers) συνδέουν πολλούς clients με πολλούς εκτυπωτές. Οι servers επικοινωνιών συνδέουν απομακρυσμένα συστήματα. Άλλοι servers πόρων μπορούν να συνδέσουν clients με άλλες συσκευές, όπως πολυμέσα. Συνήθως από τη στιγμή που οι servers πόρων είναι συνδεδεμένοι σε μια συγκεκριμένη συσκευή, δεν απαιτείται τόσο πολύ ισχύ, όση αυτή των servers που προσφέρουν περισσότερο περιπλοκές υπηρεσίες.

Ένας εύκολος τρόπος για να ξεχωρίσουμε τους servers εφαρμογών, βάσεων δεδομένων και συναλλαγών είναι πώς ο client κάνει αιτήσεις στον server. Οι servers δέχονται τους παρακάτω τύπους απαιτήσεων από τους clients:

- Οι servers εφαρμογών ενεργούν κάπως πιο αποκεντρωμένα από τη βάση δεδομένων σε απάντηση του client.
- Οι servers βάσεων δεδομένων επιστρέφουν πληροφορία σαν απάντηση σε μια αίτηση του client , που γίνεται σε SQL.
- Οι servers συναλλαγών επιστρέφουν πληροφορία σαν απάντηση σε ένα μήνυμα που αποτελείται από ένα σύνολο εντολών SQL. Αυτό το σύνολο επιτυγχάνει ή αποτυγχάνει σαν μια μονάδα.

Από τους έξι τύπους, οι client-server εφαρμογές συνήθως χρησιμοποιούν πιο πολύ servers εφαρμογών, βάσεων δεδομένων και συναλλαγών ή κάποιο συνδυασμό αυτών των τριών.

Ποιος είναι ο ρόλος του server;

Η client-server διαδικασία μπορεί να απλοποιηθεί στα παρακάτω βήματα:

1. Ο χρήστης στέλνει μια αίτηση ή ένα ερώτημα, μέσω του client, στον server.
2. Ο server ακούει την αίτηση του client.
3. Από τη στιγμή που ο server ακούει την αίτηση, ελέγχει την δυνατότητα πρόσβασης του χρήστη.
4. Ο server επεξεργάζεται το ερώτημα.
5. Ο server επιστρέφει τα αποτελέσματα στον client.
6. Ο client δέχεται τα αποτελέσματα και τα παρουσιάζει στον χρήστη.

Από αυτά τα έξι βήματα, ο server παίζει τέσσερις σημαντικούς ρόλους. Όπως είδαμε, ο server είναι η καρδιά της client -server εφαρμογής. Ο server υπάρχει για να απαντήσει στις ανάγκες του client, και ο client εξαρτάται από την αξιοπιστία και την έγκαιρη απάντηση του server.

Ο server πρέπει να εκτελέσει τις ακόλουθες λειτουργίες:

- Να ακούσει την αίτηση του client .
- Να ελέγξει την δυναμικότητα πρόσβασης του χρήστη.
- Να επεξεργαστεί την αίτηση.
- Να επιστρέψει τα αποτελέσματα.

Ο server δεν εγκαινιάζει καμιά ενέργεια. Αντίθετα, ο server περιμένει παθητικά να φτάσουν οι αιτήσεις του client μέσω του δικτύου. Ο server πρέπει πάντα να απαντάει στους clients, ακόμα και όταν πολλοί clients κάνουν ταυτόχρονες αιτήσεις.

Από την στιγμή που ο server δέχεται από τον client την απαίτηση, ο server πρέπει να βεβαιωθεί ότι ο client είναι εξουσιοδοτημένος να λάβει την πληροφορία ή την απάντηση. Αν ο client δεν είναι εξουσιοδοτημένος, ο server απορρίπτει την αίτηση και στέλνει μήνυμα στον client. Εάν ο client είναι εξουσιοδοτημένος, ο server συνεχίζει και επεξεργάζεται την αίτηση.

Η επεξεργασία της αίτησης περιλαμβάνει την παραλαβή της αίτησης του client, την μετατροπή του σε μια μορφή που μπορεί ο server να χρησιμοποιήσει και την επεξεργασία της ίδιας της αίτησης.

Όταν η επεξεργασία ολοκληρώνεται, ο server στέλνει τα αποτελέσματα πίσω στον client. Μετά, ο client μπορεί να μεταφράσει και να χρησιμοποιήσει τις πληροφορίες.

Δεν υπάρχει προκαθορισμένος διαχωρισμός στις ευθύνες για τις client-server εφαρμογές. Ανάλογα με τις ανάγκες μας, μπορούμε και να διαχωρίσουμε την εφαρμογή. Το ισχυρό client μοντέλο δίνει περισσότερες λειτουργίες στον client, ενώ το ισχυρό server μοντέλο δίνει περισσότερες λειτουργίες στον server. Οι servers εφαρμογών και συναλλαγών τείνουν να είναι ισχυροί servers, ενώ οι servers βάσεων δεδομένων και αρχείων τείνουν να έχουν ισχυρούς clients.

Ανεξάρτητα του πώς διαχωρίζουμε την εφαρμογή, η βασική ευθύνη του server παραμένει η ίδια: να εξυπηρετεί τους clients που κάνουν αιτήσεις.

Επεξεργάζοντας την αίτηση

Ο server πρέπει να είναι ικανός να ανταποκριθεί στην αίτηση του client αμέσως. Εάν πολλοί clients κάνουν αιτήσεις ταυτόχρονα, ο server πρέπει να είναι ικανός να βάζει σε προτεραιότητα τις αιτήσεις των clients, και να επεξεργάζεται πολλές αιτήσεις την στιγμή.

Από την στιγμή, που ο server επιβεβαιώνει ότι ο χρήστης είναι εξουσιοδοτημένος να κάνει αιτήσεις στον server, ο server μπορεί να αποκαλύψει την απαίτηση και να την επεξεργαστεί.

Η αίτηση μπορεί να έχει μια από τις ακόλουθες τέσσερις μορφές:
- Μια απόμακρη αίτηση είναι μια απλή αίτηση για πληροφορίες από έναν απλό client.
- Μια απόμακρη συναλλαγή περιλαμβάνει πολλαπλές αιτήσεις για πληροφορίες από έναν απλό client.

- Μια κατανεμημένη συναλλαγή περιλαμβάνει πολλαπλές αιτήσεις για πληροφορίες από έναν απλό client, οι οποίες πληροφορίες ανήκουν σε πολλούς server.
- Μια κατανεμημένη αίτηση είναι μια συναλλαγή που σχηματίζεται από πολλαπλές αιτήσεις για πληροφορίες από πολλαπλούς clients, οι οποίες πληροφορίες ανήκουν σε πολλαπλούς servers.

Αυτές οι αιτήσεις πρέπει να περάσουν από το λεγόμενο ACID τεστ: Ατομικότητα (Atomicity), Συνέπεια (Consistency), Απομόνωση(Isolation) και Αντοχή (Durability). Η ατομικότητα σημαίνει ότι ολόκληρη η συναλλαγή πρέπει να πετύχει ή να αποτύχει, δεν μπορεί να ολοκληρωθεί ως προς ένα κομμάτι της. Η συνέπεια σημαίνει ότι το σύστημα πάει από ένα σταθερό σημείο σε ένα άλλο σταθερό σημείο. Η απομόνωση σημαίνει ότι, από τη στιγμή που μια συναλλαγή ολοκληρώνεται με επιτυχία, τα αποτελέσματα της δεν είναι ορατά σε άλλες συναλλαγές. Η αντοχή σημαίνει ότι από τη στιγμή που η συναλλαγή ολοκληρώνεται με επιτυχία, δεσμεύεται μόνιμα από το σύστημα και επακόλουθες αποτυχίες δεν θα το επηρεάσουν. Εάν η συναλλαγή αποτύχει, το σύστημα οπισθοχωρεί στο σημείο που ήταν πριν προσπαθήσει να επεξεργαστεί την συναλλαγή.

Η διαχείριση της συναλλαγής ελέγχεται είτε από το DBMS είτε από το TPM (Transaction Processing Manager). Οι διαχειριστές της συναλλαγής προστατεύουν την ακεραιότητα των πληροφοριών που είναι μια απόλυτη αξίωση. Ο server είναι υπεύθυνος για προστασία και την διατήρηση της ακρίβειας των πληροφοριών.

Ας επιστρέψουμε στην αίτηση του client. Η αίτηση μπορεί απλά να ζητάει για πληροφορία ή μπορεί επίσης να επικαλείται μια αποθηκευμένη διαδικασία. Οι αποθηκευμένες διαδικασίες βελτιώνουν την επίδοση διότι είναι ήδη μεταγλωττισμένα στον υπολογιστή-«οικοδεσπότη» και είναι έτοιμα να εκτελεστούν. Επίσης μειώνουν τα προβλήματα στο δίκτυο. Ένα παράδειγμα μιας αποθηκευμένης διαδικασίας είναι το trigger: εάν μια πώληση εισέρχεται στην βάση δεδομένων ενός καταστήματος, αυτή η ενέργεια μπορεί να προκαλέσει τον server να προσθέσει την ποσότητα της πώλησης σε μια τιμή συνολικής πωλούμενης ποσότητας.

Η SQL είναι η σταθερή μέθοδος προσπέλασης πληροφοριών στις client-server εφαρμογές. Η SQL περιλαμβάνει τα ακόλουθα συστατικά:
- Γλώσσα καθορισμού των πληροφοριών.
- Γλώσσα διαχείρισης των πληροφοριών.
- Γλώσσα ελέγχου των πληροφοριών.

Η γλώσσα καθορισμού των πληροφοριών καθορίζει τις δομές των πληροφοριών, η γλώσσα διαχείρισης των πληροφοριών μετακινεί και ενημερώνει τις πληροφορίες και η γλώσσα ελέγχου των πληροφοριών καθορίζει τους περιορισμούς πρόσβασης και ασφάλειας. Η SQL επεξεργάζεται πληροφορίες σε ομάδες. Έτσι, ο server μπορεί να στείλει πολλαπλές εγγραφές για να ικανοποιήσει την αίτηση του client. Η SQL μπορεί επίσης να φλιτάρει, να μετασχηματίσει, ή να συνδυάσει πληροφορίες πριν τις στείλει στον client.

Δυο πρότυπα κατευθύνουν τις συγγενικές βάσεις δεδομένων και την απομακρυσμένη προσπέλαση πληροφοριών: DRDA (Distributed Relational Database Architecture) της IBM και το RDA (Remote Data Access) της ISO. Το DRDA προσφέρει ένα συνηθισμένο πρωτόκολλο και χρησιμοποιεί SQL ως συνηθισμένη

γλώσσα πρόσβασης για σύνδεση εφαρμογών και DBMS. Το RDA πρωτόκολλο χρησιμοποιεί ένα υποσύνολο της SQL ως ένα συνηθισμένο σύστημα μεταφοράς.

Επιστρέφοντας τα αποτελέσματα

Όταν ο server τελειώνει την επεξεργασία των αποτελεσμάτων και είναι έτοιμος να επιστρέψει τα αποτελέσματα στον client, πρέπει να μορφοποιήσει τα αποτελέσματα και να τα στείλει με ένα τρόπο που μπορεί ο client να καταλάβει.

Ο server παραδίδει τις πληροφορίες στο πρωτόκολλο, που διευθύνει ένα πακέτο, μορφοποιεί τις πληροφορίες για να τις τοποθετήσει στο πακέτο και περνάει το πακέτο στο δίκτυο. Το δίκτυο μετά βεβαιώνεται ότι το πακέτο πηγαίνει στον client.

Η ΣΗΜΑΣΙΑ ΤΟΥ CLIENT-SERVER COMPURTING

Ποια είναι η σημασία του client-server στην ανάπτυξη των εφαρμογών;

Η client-server αρχιτεκτονική επιτρέπει την εκμετάλλευση των δυνατοτήτων που παρέχουν οι clients, οι servers και το δίκτυο, όταν αναπτύσσεται μια εφαρμογή. Για να επωφεληθούμε από αυτές τις δυνατότητες, πρώτα πρέπει να γνωρίζουμε ότι η πιο συνηθισμένη λειτουργία μιας client-servers εφαρμογής είναι η παροχή δυνατότητας πρόσβασης του χρήστη στις πληροφορίες, αποτελεσματικά και εύκολα. Είναι αναγκαίο να ενοποιούνται ομαλά τα GUIs, οι κατανεμημένες εφαρμογές, οι συγγενικές βάσεις δεδομένων και τα δίκτυα.

Οι πληροφορίες δεν αποθηκεύονται ή ελέγχονται από κεντρικούς μεγάλους υπολογιστές (mainframes). Αντίθετα, είναι εύκολα προσπελάσιμες στους servers του δικτύου.

Από την στιγμή που το client-server computing είναι διαφορετικό από τα κλασσικά μοντέλα, αυτή ενισχύει τις αδυναμίες των παραδοσιακών μεθοδολογιών ανάπτυξης συστημάτων. Οι πληροφορίες που ελέγχονται από αυτό το σύστημα είναι περισσότερες από τις προηγούμενες αρχιτεκτονικές. Η ασφάλεια ρυθμίζεται σε διάφορα επίπεδα συμπεριλαμβανομένου των σταθμών εργασιών, των πληροφοριών και του χρόνου. Οι clients και οι servers προσδιορίζονται από το λογισμικό και όχι από το υλικό. Τα RPCs, που επιτρέπουν στον client να απαιτήσουν μια υπηρεσία από τον server, είναι πολύ σημαντικά στο client-server computing.

Αναπτύσσοντας Εφαρμογές

Η ανάπτυξη client-server εφαρμογών διαφέρει από τον παραδοσιακό προγραμματισμό. Για να αναπτύξουμε μια RPC εφαρμογή, ακολουθούνται τα παρακάτω γενικά βήματα:

1. Προσδιορίζεται το πρωτόκολλο επικοινωνίας του client και server.
2. Αναπτύσσονται τα clients και servers προγράμματα.
3. Μεταγλωττίζονται τα προγράμματα.
4. Διασυνδέονται οι βιβλιοθήκες.
5. Εξετάζονται οι εφαρμογές τοποθετώντας τον server σε μια απομακρυσμένη μηχανή και τρέχοντας τον client τοπικά.

Εξαιτίας της πολυπλοκότητας της client-server αρχιτεκτονικής, η ανάπτυξη client-server εφαρμογών απαιτεί πιο λεπτομερή σχεδιασμό-ειδικότερα, πως να διαχωριστεί η εφαρμογή ανάμεσα στον client και τον server και πως να κατανεμηθούν οι πληροφορίες μεταξύ client και server. Οι εφαρμογές που χρησιμοποιούν RPCs είναι κατανεμημένες διότι τα RPCs είναι ουσιαστικά clients και απομακρυσμένοι server επεξεργαστές. Οι ίδιες οι εφαρμογές συνήθως διαιρούνται σε δυο τμήματα -το τμήμα του client και το τμήμα του server.

Οι clients κάνουν κλήσεις και οι servers εξυπηρετούν τις κλήσεις.

Το σύστημα διασύνδεσης των χρηστών ανήκει στον client. Εργασίες που δουλεύουν καλά στον client είναι η μορφοποίηση ερωτημάτων για τον server,η δημιουργία αναφορών και ο έλεγχος των σφαλμάτων. Η λογική της εφαρμογής στον client ονομάζεται front end.

Ο server, από την άλλη, βασικά ευθύνεται για την ανάκτηση, τον χειρισμό και την ασφάλεια των πληροφοριών. Φυσικά, αυτό εξαρτάται από το πώς οι πληροφορίες κατανέμονται. Η λογική της εφαρμογής στον server ονομάζεται back end.

Κατανομή πληροφοριών

Αφού αποφασιστεί ο διαχωρισμός της εφαρμογής μεταξύ του client και του server πρέπει να αποφασιστεί το πώς θα κατανέμονται οι πληροφορίες. Ένας από τους κύριους λόγους που χρειάζεται να κατανέμονται οι πληροφορίες είναι η ελαχιστοποίηση και συνεπώς ο περιορισμός των προβλημάτων στο δίκτυο.

Υπάρχουν τέσσερις βασικές επιλογές κατανομής των πληροφοριών, όπως φαίνονται παρακάτω:

- Η ύπαρξη πολλαπλών πανομοιότυπων αντιγράφων.
- Η διαίρεση των πληροφοριών σε διάφορες περιοχές.
- Η αντίληψη πληροφοριών από άλλες περιοχές.
- Η αντιγραφή των πληροφοριών σε περιοχές προσπέλασης, σε υψηλή ταχύτητα.

Η αρχή που παίζει μεγάλο ρόλο στον καθορισμό του πως κατανέμονται οι πληροφορίες είναι η κομβική χωροθέτηση (nodal residency). Αυτό σημαίνει, ότι οι πληροφορίες είναι αποθηκευμένες πλησιέστερα στους χρήστες που τις χρειάζονται. Έτσι, εάν ένα αρχείο αναφοράς που δεν πρόκειται να αλλάζει συχνά χρησιμοποιείται από πολλαπλούς χρήστες, είναι λογικό να υπάρχουν πολλαπλά αντίγραφα αυτού του

αρχείου στους clients. Εάν κάποιες πληροφορίες χρησιμοποιούνται από όλους τους clients και άλλες πληροφορίες είναι ειδικές για συγκεκριμένους clients, είναι πιο πρακτικό να αποθηκεύονται οι πρώτες πληροφορίες στον server και οι δεύτερες πληροφορίες στον client. Εάν κάποιες πληροφορίες που αλλάζουν συχνά, μπορούν να υπολογιστούν γρήγορα από υπάρχουσες πληροφορίες στον server, δεν χρειάζεται να σπαταλείται χρόνος ή χώρος για να αποθηκευτούν. Ή, εάν οι clients χρησιμοποιούν συνήθως το ίδιο τμήμα των αρχείων, είναι βολικό να αντιγράφονται τα τμήματα που χρειάζονται οι χρήστες κάθε φορά που απαιτούν τις πληροφορίες. Από τη στιγμή που αποφασιστεί το πώς θα διαιρεθούν οι εργασίες και οι πληροφορίες μεταξύ του client και του server, τότε αρχίζει η δημιουργία των προγραμμάτων.

Οι clients-server εφαρμογές μετακινούν το επίκεντρο του προγραμματισμού από τις μηχανές προς τους χρήστες. Οι τελικοί χρήστες προσδοκούν τα ακόλουθα:

- Πρόσβαση σε πολλαπλές πληροφορίες, δηλαδή, οι πληροφορίες να είναι διαθέσιμες σε όλους τους εξουσιοδοτημένους χρήστες.
- Ολοκληρωμένες υπηρεσίες.
- Πρόσβαση σε πόρους μέσω διαφόρων πλατφόρμων.
- Ανταλλαγή και εκμετάλλευση πληροφοριών.
- Πρόσβαση σε οποιεσδήποτε πληροφορίες, η ακόμα και σε φαινομενικά απροσπέλαστες πληροφορίες.
- Ευκολότερη διατήρηση και συντήρηση των πληροφοριών.

Γίνεται εύκολα αντιληπτό από τα παραπάνω πως τα οφέλη από την επένδυση ενός κεφαλαίου από την πλευρά μιας επιχ/σης στις νέες τεχνολογίες του Web και ιδιαίτερα στις client –server τεχνολογίες είναι σχεδόν σίγουρο, αν τα επιχειρησιακά στελέχη αναλύσουν την σχεδόν νηπιακή σε σχέση με άλλες αγορές αγορά του Διαδίκτυου, και διαπιστώσουν εμπορικά κίνητρα.

Ένα βήμα μετά την ανάλυση της αγοράς και των αναγκών της είναι να μπορέσει μια επιχ/ση να απευθυνθεί στα κατάλληλα άτομα για να μπορέσει να αναπτύξει τα σχέδια της στην πράξη πλέον .

Εδώ έρχονται οι διάφορες εταιρείες ανάπτυξης διαδικτυακών εφαρμογών οι οποίες αν και όλες παρέχουν σχεδόν παρόμοιες υπηρεσίες , οι τεχνολογίες με τις οποίες αναπτύσσουν από την πλευρά του Software τα προγράμματα τους είναι διαφορετικές.

Μερικές από τις τελειότερες τεχνολογίες που χρησιμοποιούνται σήμερα στην ανάπτυξη Web εφαρμογών είναι το σχεδόν πια κλασσικό δίδυμο των τεχνολογιών PHP και MySQL καθώς και οι βοηθητικές θα λέγαμε τεχνολογίες της JAVASCRIPT και των CASCADING STYLE SHEETS(CSS), οι οποίες σε συνδυασμό με την πρωταρχική γλώσσα ανάπτυξης ιστοσελίδων την HTML έχουν δώσει στο Web το σύγχρονο «Δυναμικό» πρόσωπο του.

Αυτές είναι και οι τεχνολογίες τις οποίες θα παρουσιάσω παρακάτω σε μια προσπάθεια να ξεκαθαρίσω το πώς πραγματικά δημιουργείται ένα σύγχρονο δυναμικό πλέον Web Site.

Κοινό γνώρισμα όλων των παρακάτω τεχνολογιών είναι πως όλες γνώρισαν τρομερή ανάπτυξη κατά την πενταετία 1995-2000 γεγονός το οποίο δεν ξενίζει κανέναν καθώς αν κάποιος αποφασίσει να τις γνωρίσει, θα διαπιστώσει πως εκτός του ότι δεν απαιτούνται προχωρημένες προγραμματιστικές γνώσεις για να δουλέψει κανείς με αυτές, το κυριότερο είναι πως είναι όλες free και τα αποτελέσματα τους φαίνονται απλά και μόνο μέσω ενός Browser.

Αυτός ο τελευταίος παράγοντας είναι που ώθησε χιλιάδες ανθρώπους να ασχοληθούν με τις τεχνολογίες του web, γεγονός που σαφέστατα επηρέασε στην παγκόσμια ανάπτυξη και υπεροχή του έναντι οποιουδήποτε αλλού μέσου.

Αρκετά όμως με την παρουσίαση αυτών των τεχνολογιών , ας γνωρίσουμε την κάθε μία ξεχωριστά.

CSS(Cascading Style Sheets) – Εισαγωγή

Δεν θα ήταν υπέροχο αν θα μπορούσατε να λέγατε σε όλους τους φυλλομετρητές που εμφανίζουν την Ιστοσελίδα σας - "Έτσι θέλω να χειριστείς το κείμενο μου";

Μπορείτε να κάνετε όλες τις H3 εντολές σε γραμματοσειρά Arial;

Μπορείτε να κάνετε τις εντολές των παραγράφων ώστε να απέχουν μισή ίντσα;

Μπορείτε να ορίσετε το κενό διάστημα ανάμεσα στης σειρές σας;

Ο έλεγχος της μορφής (layout) των σελίδων Web υπήρξε παραδοσιακά μια δύσκολη διαδικασία.

Στην αρχή, οι δημιουργοί των σελίδων Web είχαν ελάχιστο ή καθόλου έλεγχο πάνω στην εμφάνιση των σελίδων τους. Με τον καιρό, διάφορες μέθοδοι αναπτύχθηκαν με σκοπό τον καλύτερο έλεγχο των αντικειμένων της σελίδας (κείμενο και γραφικά) και τη σωστότερη εμφάνιση των σελίδων σύμφωνα με την αρχική σχεδίασή τους.

Μετά την εισαγωγή των Tables, οι δημιουργοί υιοθέτησαν αυτή τη μέθοδο για το «στήσιμο» των αντικειμένων στις σελίδες. Μάλιστα, τόση ήταν η ανάγκη για σωστό layout, ώστε ορισμένοι δημιουργοί να έχουν αποκτήσει πολύ μεγάλη πείρα στη χρήση των Tables από τη συχνή χρήση τους.

Η επιθυμία για ένα τέλειο layout είναι αποτέλεσμα της μεταφοράς του έντυπου υλικού στον Web. Η τοποθέτηση του κειμένου και της εικόνας σε ένα έντυπο είναι απλή διαδικασία με τη χρήση προγραμμάτων ηλεκτρονικής στοιχειοθέτησης, όπως των Adobe PageMaker και Quark Xpress. Ωστόσο, η μεταφορά αυτών τον πληροφοριών από το έντυπο στο Διαδίκτυο απαιτεί απόλυτο έλεγχο πάνω στα αντικείμενα της σελίδας .

Με την παρουσίαση των Cascading Style Sheets (CSS) οι δημιουργοί απέκτησαν τον απαιτούμενο έλεγχο στα αντικείμενα των Web σελίδων τους.

Τα CSS μάς έδωσαν τη δυνατότητα να δημιουργήσουμε **layers.**

Τα layers είναι «επιφάνειες» της σελίδας.

Κάθε «επιφάνεια» περιλαμβάνει κάποια αντικείμενα και έχει ορισμένα ιδιότητες. Θεωρητικά, όλες οι «επιφάνειες» είναι «διάφανεις» (transparent) και εμείς μπορούμε να επεμβαίνουμε σε όλες τις ιδιότητές τους: διάσταση, σειρά, διαφάνεια, θέση κ.ά.

Τα layers σε συνδυασμό με την JavaScript και το ευρύτερο σύνολο της DHTML δημιουργούν πραγματικά αριστουργήματα. Εκτός από τα layers, τα CSS δίνουν επίσης στο δημιουργό έλεγχο σε όλους τους τομείς παρουσίασης των κειμένων και των γραμματοσειρών, περιλαμβάνοντας, τον τύπο της γραμματοσειράς, την απόσταση των γραμμάτων (spacing), το μέγεθος και το είδος (bold, italic).

Τα CSS είναι πραγματικό δώρο για όσους συνήθιζαν να παλεύουν προσπαθώντας να κάνουν μια σελίδα να μοιάζει το ίδιο στον Internet Explorer και τον Netscape Navigator, τόσο για Macintosh όσο και για PC.

Ο Internet Explorer 3 και ανώτεροι και ο Netscape 4.0 (και οι ανώτεροι) προσφέρουν περισσότερο έλεγχο σε αυτά που ονομάζονται "Style Sheets". Αντί να γράφετε συνέχεια το μέγεθος της γραμματοσειράς, τα περιθώρια, τα διαστήματα κτλ, κτλ... τα γράφετε μια καλή σε μια σειρά εντολών που θα σας λύσουν τα χέρια.

Ορισμός

Ο όρος "Cascading" Style Sheets χρησιμοποιείτε επειδή παραπάνω από ένα στυλ μπορεί να επηρεάσει μια σελίδα.

Για παράδειγμα, αν χρησιμοποιείτε Style Sheet στην HTML εντολή (ονομάζεται in-line) σε ένα έγγραφο και ένα εξωτερικό style sheet που καλείται από διάφορες σελίδες (ονομάζεται span), και τα δυο μπορούν να επηρεάσουν τα αντικείμενα της σελίδας. Αν και τα δυο, το in-line και το span Style Sheet προσπαθούν να επηρεάσουν το ίδιο αντικείμενο, λόγου χάρη την εντολή <H1>, το κοντινότερο στην εντολή <H1> εφαρμόζεται. Αυτό είναι το in-line στην συγκεκριμένη περίπτωση.

Σε περίπτωση που αναρωτιέστε, αν υπάρχουν δύο span Style Sheets που επηρεάζουν το ίδιο αντικείμενο, εφαρμόζεται το δεύτερο. Είναι το κοντινότερο, σε σχέση με το πρώτο.

Τα Στυλ Λύνουν ένα Κοινό Πρόβλημα

Τα tags της HTML σχεδιάσθηκαν αρχικά για να ορίσουν το περιεχόμενο ενός εγγράφου. Η δουλειά τους ήταν να έλεγαν "Αυτή είναι μια επικεφαλίδα", "Αυτή είναι μια παράγραφος", "Αυτός είναι ένας πίνακας", χρησιμοποιώντας tags όπως <h1>, <p>, <table> κοκ. Η διάταξη (layout) του εγγράφου ήταν υπόθεση του φυλλομετρητή, χωρίς τη χρήση tags μορφοποίησης (formatting tags).

Καθώς οι δύο κύριοι φυλλομετρητές, ο Netscape και ο Internet Explorer, συνέχισαν να προσθέτουν νέα HTML tags και χαρακτηριστικά (attributes), όπως το tag <font> και το attribute color, στις αρχικές προδιαγραφές της HTML, γινόταν ολοένα και δυσκολότερη η δημιουργία Web sites όπου το περιεχόμενο των εγγράφων HTML να μπορεί να ξεχωρίζει καθαρά από τη διάταξη παρουσίασης του εγγράφου.

Για να μπορέσει να λύσει αυτό το πρόβλημα, το *World Wide Web Consortium (W3C)* – το μη κερδοσκοπικό consortium ορισμού στάνταρτς που είναι υπεύθυνο για την καθιέρωση της HTML – δημιούργησε τα ΣΤΥΛ (STYLES) σαν προσθήκη στην HTML 4.0.

Τα Φύλλα Στυλ μάς Γλιτώνουν από Πολλή Δουλειά

Τα στυλ στην HTML 4.0 ορίζουν το πώς εμφανίζονται τα HTML στοιχεία, όπως ακριβώς το tag font και το attribute color στην HTML 3.2. Τα στυλ αποθηκεύονται συνήθως σε αρχεία που είναι εξωτερικά (external) στα HTML έγγραφά μας.

Τα εξωτερικά φύλλα στυλ (external style sheets) μάς δίνουν τη δυνατότητα να αλλάξουμε την εμφάνιση και τη διάταξη όλων των σελίδων στο δικό μας Web site, με απλή επεξεργασία ενός μόνου CSS εγγράφου. Αν έχουμε ποτέ προσπαθήσει να αλλάξουμε τη γραμματοσειρά (font) ή το χρώμα (color) όλων των επικεφαλίδων (headings) σ' όλες τις ιστοσελίδες μας, θα καταλάβουμε γιατί τα CSS μπορούν να μας γλιτώσουν από πολλή δουλειά.

Τα CSS αποτελούν μια μεγάλη επιτυχία στον σχεδιασμό του Web (Web design) επειδή δίνουν τη δυνατότητα στους developers να ελέγξουν το στυλ και τη διάταξη πολλών ιστοσελίδων μονομιάς. Σαν Web developer μπορούμε να ορίσουμε ένα στυλ για κάθε HTML στοιχείο και να το εφαρμόσουμε σ' όσες ιστοσελίδες θέλουμε. Για να κάνουμε μια καθολική (global) αλλαγή, απλά αλλάζουμε το στυλ μία φορά και όλα τα στοιχεία του Web ενημερώνονται αυτόματα.

Πολλαπλά Στυλ Καταλήγουν σ' ένα

Τα Φύλλα Στυλ επιτρέπουν τον ορισμό των πληροφοριών στυλ με πολλούς τρόπους. Τα στυλ μπορούν να ορισθούν μέσα σ' ένα μόνο HTML στοιχείο, μέσα στο στοιχείο <head> μιας HTML σελίδας ή σ' ένα εξωτερικό αρχείο CSS. Πολλά εξωτερικά Φύλλα Στυλ μπορούν να χρησιμοποιούνται μέσα απ' ένα μόνο HTML έγγραφο.

Ποιο στυλ θα χρησιμοποιηθεί όταν υπάρχουν περισσότερα από ένα καθορισμένα στυλ για ένα HTML στοιχείο ;

Σε γενικές γραμμές μπορούμε να πούμε ότι όλα τα στυλ θα καταλήξουν (cascade) σ' ένα νέο εικονικό (virtual) Φύλλο Στυλ σύμφωνα με τους παρακάτω κανόνες, όπου ο μεγαλύτερος αριθμός έχει και την υψηλότερη προτεραιότητα :

1. Προεπιλογή του φυλλομετρητή.
2. Εξωτερικό Φύλλο Στυλ (External Style Sheet).
3. Εσωτερικό Φύλλο Στυλ (Internal Style Sheet), μέσα στο τμήμα header του εγγράφου.
4. Inline Style, μέσα στο HTML στοιχείο.

Η Σύνταξη των CSS

Η σύνταξη των CSS αποτελείται από τρία μέρη : έναν επιλογέα (selector), μια ιδιότητα (property) και μια τιμή (value) :

επιλογέας {ιδιότητα: τιμή}

selector {property: value}

Ο *επιλογέας* είναι συνήθως το στοιχείο/tag που θέλουμε να ορίσουμε, η *ιδιότητα* είναι το χαρακτηριστικό που θέλουμε να αλλάξουμε και η κάθε ιδιότητα μπορεί να πάρει μια *τιμή*. Η ιδιότητα και η τιμή ξεχωρίζουν από τον χαρακτήρα : και περικλείονται από τους χαρακτήρες { }, ως εξής :

body {color: black}

Αν η τιμή αποτελείται από πολλές λέξεις, πρέπει να τοποθετήσουμε εισαγωγικά :

p {font-family: "sans serif"}

Αν θέλουμε να ορίσουμε περισσότερες από μία ιδιότητες, πρέπει να ξεχωρίσουμε την κάθε ιδιότητα με τον χαρακτήρα **;** .Το παρακάτω παράδειγμα δείχνει πώς μπορούμε να ορίσουμε μια κεντραρισμένη παράγραφο με χρώμα κειμένου κόκκινο :

p {text-align: center; color: red}

Για να κάνουμε τους ορισμούς των στυλ πιο ευανάγνωστους, μπορούμε να γράψουμε από μία ιδιότητα σε κάθε γραμμή, ως εξής :

```
p
{
    text-align: center;
    color: black;
    font-family: arial
    }
```

Ομαδοποίηση (Grouping)

Μπορούμε να ομαδοποιήσουμε τους επιλογείς. Ξεχωρίζουμε τον κάθε επιλογέα με κόμμα. Στο παρακάτω παράδειγμα έχουμε ομαδοποιήσει όλα τα στοιχεία επικεφαλίδας (header elements). Το κάθε στοιχείο επικεφαλίδας θα είναι πράσινο :

```
h1, h2, h3, h4, h5, h6

{

   color: green

}
```

Το Χαρακτηριστικό (Attribute) Class

Με το χαρακτηριστικό class μπορούμε να ορίσουμε διαφορετικά στυλ για το ίδιο στοιχείο (element). Ας υποθέσουμε ότι θέλουμε να έχουμε δύο είδη παραγράφων στο έγγραφό μας : μια δεξιά στοιχισμένη παράγραφο και μια κεντραρισμένη παράγραφο.

Να πώς μπορούμε να το κάνουμε αυτό με τα στυλ :

```
p.right {text-align: right}

p.center {text-align: center}
```

Πρέπει να χρησιμοποιήσουμε το χαρακτηριστικό class στο HTML έγγραφο, ως εξής :

```
<p class="right"> Αυτή είναι μια παράγραφος.

Το κείμενο αυτής της παραγράφου θα είναι δεξιά στοιχισμένο. </p>

<p class="center"> Αυτή είναι μια άλλη παράγραφος.

Το κείμενο αυτής της παραγράφου θα είναι κεντραρισμένο. </p>
```

Το Χαρακτηριστικό (Attribute) Id

Με το χαρακτηριστικό id μπορούμε να ορίσουμε ένα μοναδικό στυλ που μπορούμε να χρησιμοποιήσουμε σε πολλά στοιχεία. Να πώς μπορούμε να το κάνουμε αυτό με τα στυλ :

#right {text-align: right}

Στο HTML έγγραφο πρέπει να γράψουμε τα εξής :

* Αυτή είναι μια παράγραφος.*

*Το κείμενο αυτής της παραγράφου θα είναι δεξιά στοιχισμένο. *

<h3 id="right"> Αυτή είναι μια επικεφαλίδα.

Αυτή η επικεφαλίδα θα είναι επίσης δεξιά στοιχισμένη. </h3>

Το χαρακτηριστικό id πρέπει να έχει μια μοναδική τιμή στο έγγραφο.

Τα Σχόλια (Comments) στα CSS

Μπορούμε να εισάγουμε σχόλια (comments) στα CSS για να εξηγούμε τον κώδικά μας και τα οποία μπορούν να μας βοηθήσουν όταν θα χρειαστεί κάποια στιγμή να τροποποιήσουμε τον πηγαίο κώδικα (source code). Τα σχόλια αγνοούνται από τον φυλλομετρητή. Ένα CSS σχόλιο αρχίζει με τους χαρακτήρες **/*** και τελειώνει με τους χαρακτήρες ***/**, ως εξής :

```
/* Αυτό είναι ένα σχόλιο (comment) */

p

{

   text-align: center;

   /* Αυτό είναι ένα ακόμη σχόλιο */

   color: black;

   font-family: arial

}
```

Εισάγοντας ένα Style Sheet

Υπάρχουν δύο τρόποι:

-Ο εξωτερικός
-Ο εσωτερικός.

ΕΣΩΤΕΡΙΚΟΣ

Αυτός είναι ο πιο συνήθης τρόπος χρήσης των Style Sheets. Για να λέμε τα πράγματα με ακρίβεια , όταν χρησιμοποιούμε τον εσωτερικό τρόπο, τεχνικά δεν είναι style sheet, αλλά ένα in-line style block. Αλλά η χρήση της δεύτερης ονομασίας δεν χρησιμοποιείτε, γιατί μπερδεύει πολλούς και τα ξεχωρίζουν σαν δυο διαφορετικά αντικείμενα.

Ακολουθήστε τους παρακάτω απλούς κανόνες για την τοποθέτηση ενός Style Sheet (Block) στην σελίδα σας:

Πρέπει να γραφτεί ανάμεσα στα <HEAD> και </HEAD> tags. Το κείμενο-εντολές πρέπει να εισαχθεί ανάμεσα στο <STYLE TYPE="text/css"> και </STYLE> Επειδή το style sheet είναι κείμενο, αν το τοποθετήσετε στην Ιστοσελίδα, θα εμφανιστεί σαν κείμενο, οπότε πρέπει να το εισάγετε και ανάμεσα σε tags σχολίων <!-- και -- > Αν τα κάνατε όλα σωστά πρέπει να εμφανιστεί έτσι:

```
<HEAD>
<STYLE TYPE="text/css">
<!--

  Οι εντολές για το Style Sheet γράφονται εδώ...

-->
</STYLE>
</HEAD>
```

"classes" σε ένα tag

Αν θέλετε διαφορετικά "classes" σε ένα tag είναι πολύ απλό:

```
H3.first {font-size: 20pt; color: #FF00FF}
H3.scnd {font size: 18pt; color #DD00FF}
```

Ονομάζουμε τις εντολές χρησιμοποιώντας μια τελεία "." και ένα όνομα που θέλουμε. Στο παράδειγμά μας, χρησιμοποιήσαμε το "first" για τον πρώτο τύπο και το "scnd" για τον δεύτερο. (Μπορείτε να χρησιμοποιήσετε όποιο όνομα θέλετε αλλά προσέξτε να μην εμπλέκεται με τις εντολές και τον κώδικα). Μετά στην σελίδα προσθέτετε με τον εξής τρόπο :

```
<H3 CLASS="first">Θα επηρεαστεί από το "H3.first"</H3>
<H3 CLASS="scnd">Θα επηρεαστεί από το "H3.scnd"</H3>
```

ΕΞΩΤΕΡΙΚΟΣ

Ανοίξτε το notepad ή τον css editor σας. Εδώ η μορφή που πρέπει να ακολουθηθεί είναι η εξής :

<STYLE TYPE="text/css"> <!-- Οι εντολές για το Style Sheet γράφονται εδώ... --> </STYLE> Μετά αποθηκεύετε το αρχείο σας με κατάληξη **.css**. Η κατάληξη χρησιμοποιείται από τον φυλλομετρητή ώστε να καταλάβει ότι πρόκειται για CSS και όχι απλά γράμματα.

Σαν επόμενο βήμα, πρέπει να τοποθετήσετε την παρακάτω εντολή του Style Sheet σε όσες σελίδες θέλετε να επηρεάζονται από αυτό: <LINK REL="stylesheet" HREF="http://www.Η Σελίδα σας / Όνομα Αρχείου.css" TYPE="text/css">

Αν δεν θέλετε μια λέξη ή ένα αντικείμενο να επηρεαστεί από το CSS, απλά στο tag του αντικειμένου δηλώνετε το style και γράφετε τι στυλ θέλετε να έχει... πχ.

<STYLE TYPE="text/css"> <!-- H2 {font-family: arial} --> </STYLE> <h2 style="font-family: verdana">Επικεφαλίδα 2</h2>

Εισαγωγή στις εντολές ...

Βασική Μορφή

```
TAG {
ορισμός;
ορισμός;
ορισμός
}
```

Δείτε για παράδειγμα παρακάτω:

```
H2 {
font-size: 16pt;
font-style: italic;
font-family: arial
}
```

Προσοχή στα εξής:

Πριν τον ορισμό υπάρχει άγκιστρο "{" και όχι παρένθεση ή αγκύλη - "(" και "[" αντίστοιχα - Σας συμβουλεύω να αφήνετε κενές σειρές γιατί σας διευκολύνουν στην ανάγνωση του CSS κώδικα και στην επεξεργασία του. Απλά μην ξεχνάτε το ":" πριν δώσετε την τιμή και το ";" μόλις τελειώνει ο ορισμός.

Προσοχή :Ο τελευταίος ορισμός δεν θέλει ερωτηματικό!!!!

ΥΠΟΣΤΗΡΙΞΗ ΤΩΝ CSS ΑΠΟ ΤΟΥΣ BROWSERS

Τα CSS υποστηρίζονται μόνο από τις τελευταίες εκδόσεις των browsers. Τόσο ο Internet Explorer 4 (υποστήριξε πρώτος, από την έκδοση 3, τα CSS) όσο και ο Netscape Navigator 4 υποστηρίζουν πλήρως τα CSS.

Όσο περίεργο και αν ακούγεται, και οι δύο browsers είναι –σχεδόν- πλήρως συμβατοί. Είναι γνωστό σε όλους ότι οι δύο browsers σχεδόν ποτέ δεν συμβαδίζουν, με αποτέλεσμα να χρειάζεται άλλος τρόπος συγγραφής για τον πρώτο και άλλος για τον δεύτερο. Στα CSS όμως δεν ισχύει αυτός ο κανόνας. Ως αποτέλεσμα, τα CSS συχνά καλούνται ως η «νεκρή ζώνη» για τους browsers τέταρτης γενιάς.

Τι συμβαίνει όμως με τα CSS σε παλιότερους browsers; Ορισμένοι, όπως ο Netscape 2.0 και 3.0, είναι αρκετά «έξυπνοι» ώστε να αγνοήσουν Style Sheets που βρίσκονται μεταξύ του tag <STYLE>.

Ορισμένοι ακόμα πιο παλιοί, όπως ο Internet Explorer 2 και ο Netscape 1, θα αγνοήσουν μεν το <STYLE> tag, αλλά θα εμφανίσουν τα περιεχόμενα μεταξύ των tags. Τα περιεχόμενα αυτά θα τυπωθούν στη κορυφή της σελίδας σε μία γραμμή – και αυτό δεν είναι καθόλου καλό. Για να λυθεί και αυτό το πρόβλημα, η λύση είναι να κάνουμε σχόλιο τα περιεχόμενα του tag με τη χρήση των <!-- --> , όπως στο ακόλουθο παράδειγμα:

```
<HTML>
<TITLE>Κρύβοντας τα CSS σε παλιότερους browsers</TITLE>
 <STYLE type=”text/css”>
 <!—
 #stoixeio {
   position: absolute;
   z-index: 1;
   left: 30px;
   top: 30px; }
 -->
 </STYLE>
<BODY>
<DIV id=stoixeio>
<IMG SRC=”images/stoixeio.gif” width=120 height=90 border=0>
</DIV>
 </BODY>
</HTML>
```

Με αυτή την τεχνική, μπορούμε να είμαστε σίγουροι ότι δεν θα συμβεί κάποιο «ατύχημα» σε παλαιότερο browser.

ΔΙΑΦΟΡΕΣ ΜΕΤΑΞΥ ΤΩΝ INTERNET EXPLORER 4 & NETSCAPE 4

Μια μεγάλη διαφορά υπάρχει μεταξύ των Internet Explorer 4 και Netscape Navigator 4: το στοιχείο visibility (ορατότητα).

Αυτό το στοιχείο ελέγχει εάν ένα αντικείμενο είναι αρχικά ορατό στην οθόνη ή παραμένει κρυφό. Αποτελεί ένα αρκετά δυνατό εργαλείο, όταν συνδυάζεται με το DOM (Document Object Modeling) και την JavaScript. Χρησιμοποιώντας JavaScript ή VBScript, μπορούμε να καθιστούμε ένα αντικείμενο φανερό ή κρυφό ανάλογα με την επίδραση του χρηστή.

Ο Internet Explorer 4 ακολουθεί τις τελευταίες προτάσεις του W3C (World Wide Web Consortium) για τα CSS. Ένα ορατό αντικείμενο έχει την τιμή Visible (=ορατό) και ένα κρυφό αντικείμενο την τιμή Hidden (=κρυφό). Στο ακόλουθο παράδειγμα, το γραφικό stoixeio.gif θα είναι ορατό όταν φορτώσει η σελίδα. Εάν η παράμετρος Visibility ήταν Hidden, τότε θα φορτωνότανε μεν στη μνήμη, αλλά θα παρέμενε κρυφό. Στο κατωτέρω παράδειγμα η παράμετρος Visibility διατηρεί το αντικείμενο ορατό.

```
<HTML>
<TITLE>Κρύβοντας τα CSS σε παλαιότερους browsers</TITLE>
 <STYLE type=”text/css”>
 <!—
 #stoixeio {
   position: absolute;
   z-index: 1;
   left: 30px;
   top: 30px;
VISIBILITY: visible; }
 -->
 </STYLE>
<BODY>
<DIV id=stoixeio>
<IMG SRC=”images/stoixeio.gif” width=120 height=90 border=0>
</DIV>
 </BODY>
</HTML>
```

Η Netscape παρουσίασε τον browser της όταν οι τελικές προτάσεις για τα CSS ήταν ακόμα υπό συζήτηση. Έτσι, προσπάθησε να μαντέψει ποιες θα ήταν οι παράμετροι για το Visibility. Μάντεψε όμως λάθος. Στο Netscape 4 για να κάνουμε ένα αντικείμενο ορατό χρησιμοποιούμε το Show, ενώ για να το κάνουμε κρυφό το Hide. Στη συνέχεια βλέπουμε τη διαφορά αυτή:

```
<HTML>
<TITLE>Κρύβοντας τα CSS σε παλαιότερους browsers</TITLE>
 <STYLE type=”text/css”>
 <!—
```

```
#stoixeio {
  position: absolute;
  z-index: 1;
  left: 30px;
  top: 30px;
VISIBILITY: show; }
-->
</STYLE>
<BODY>
<DIV id=stoixeio>
<IMG SRC=”images/stoixeio.gif” width=120 height=90 border=0>
</DIV>
</BODY>
</HTML>
```

Η Netscape έχει δηλώσει ότι θα συμβιβαστεί με τις τελικές προτάσεις του W3C στο Netscape 5. Θα συνεχίσει βεβαίως να υποστηρίζει τα Show και Hide, ώστε να είναι συμβατό με παλαιότερο κώδικα.

ΡΥΘΜΙΣΕΙΣ DEFAULT

Εάν η παράμετρος Visibility δεν συμπεριληφθεί στο style sheet, τότε το αντικείμενο θα είναι ορατό.

Ο Netscape browser θα θεωρήσει την τιμή Show και ο Internet Explorer την τιμή Visible. Παραλείποντας την παράμετρο, ξεπερνάμε το πρόβλημα της ασυμβατότητας. Μόνο όταν εμφανίζουμε και κρύβουμε αντικείμενα με τη χρήση script προκύπτει ασυμβατότητα.

ΤΑ TAGS DIV & SPAN

Τα tags DIV (division) και SPAN είναι νέα στοιχεία στην HTML και εισήχθησαν για να υποστηρίξουν τα Style Sheets.

Το DIV είναι το πιο ισχυρό tag και επιτρέπει να δημιουργούμε περιοχές στη σελίδα. Μια περιοχή είναι ένα κομμάτι της σελίδας με συγκεκριμένο ύψος και πλάτος, κάτι σαν ένα floating layer. Αυτή η περιοχή μπορεί να γίνει αντικείμενο επεξεργασίας ανεξάρτητα από την υπόλοιπη σελίδα. Για παράδειγμα, μπορεί να είναι animated ή κρυφή, επίσης δύναται να καταστεί «διάφανη» ή μπορεί τα συμπεριλαμβανόμενα γραφικά να εναλλάσσονται μεταξύ ορατού και κρυφού. Οι περιοχές DIV μπορούν να έχουν και το δικό τους style sheet. Το style sheet δύναται να ορίζει τα στοιχεία που περιλαμβάνονται στο DIV και να τα τοποθετεί στη σελίδα σε συντεταγμένες x, y, z. Το DIV χρησιμοποιείται συχνά για να δημιουργήσουμε στη σελίδα ένα στοιχείο το οποίο θα επεξεργαστούμε ανεξάρτητα από την υπόλοιπη σελίδα.

Τέτοια tags μπορούν να περιλαμβάνουν διάφορα γραφικά ή οποιονδήποτε συνδυασμό στοιχείων. Μπορούν επίσης να περιλαμβάνουν και Java applets και plug-in όπως το Shock-Wave.

Το tag SPAN χρησιμοποιείται για να εφαρμόσουμε έναν συγκεκριμένο τύπο style sheet σε μια περιοχή κειμένου ή γραφικών μιας γραμμής. Τα DIV tags, από την άλλη, μπορούν να περιλαμβάνουν περισσότερες από μία γραμμές. Φανταστείτε ότι έχετε ένα κομμάτι κειμένου όπου κάθε τρίτη λέξη διαθέτει τις εξής ιδιότητες: Bold, γραμματοσειρά Times, μέγεθος 35 points, κείμενο υπογραμμισμένο και Italic. Μοιάζει μεγάλος μπελάς – αλλά δεν είναι, με τη χρήση του SPAN tag. Στο ακόλουθο παράδειγμα φαίνεται πώς μπορούμε να το πετύχουμε:

```
<HMTL>
<TITLE>Ξεμπερδέματα</TITLE>
<STYLE>
<!--
 span.trith {
  Font-family: Times;
  Font-style: Italic;
  Font-weight: Bold;
  Line-height: 35pt;
  Text-Decoration: underline; }
-->
</STYLE>
<BODY>
 Κάθε τρίτη <SPAN class=trith>λέξη</SPAN> σε αυτή <SPAN
class=trith>την</SPAN> πρόταση πρέπει <SPAN class=trith>να</SPAN>
ξεχωρίζει.
</BODY>
</HTML>
```

ΔΙΑΦΟΡΕΣ ΜΕΤΑΞΥ ΤΩΝ LAYERS & ΤΩΝ STYLE SHEETS

Δεν είμαι πεπεισμένος ότι υπάρχει οποιοσδήποτε λόγος να μάθουμε το tag <LAYER> για το Netscape 4.

Το <LAYER> tag εισήχθη από τη Netscape προτού τεθούν οι προϋποθέσεις για τα style sheets. Μάλιστα, προτάθηκε στο W3C ως μέρος των προϋποθέσεων για το CSS1,αλλα απορρίφθηκε.

Στο Netscape 4, τόσο τα layers όσο και το DIV tag υποστηρίζονται ως τρόποι δήλωσης περιοχών της σελίδας. Μπορούμε να ορίσουμε μια περιοχή είτε με το DIV είτε με το LAYER και μετά να την επεξεργαστούμε. Ωστόσο, το LAYER tag χρησιμοποιείται μόνο από το Netscape 4, ενώ το DIV tag χρησιμοποιείται και από το Netscape 4 και από τον Internet Explorer 4. Προφανώς το LAYER tag θα συνεχίσει να υποστηρίζεται από μελλοντικές εκδόσεις του Netscape, αλλά δεν θα γίνει ποτέ cross-platform.

ΔΗΛΩΣΗ ΤΟΥ STYLE SHEET

Το style sheet το δηλώνουμε εκτός του περιεχομένου της σελίδας, όπως στο πρώτο παράδειγμα. Δηλώνουμε τις ιδιότητες του αντικειμένου στην αρχή και τους δίνουμε ένα ξεχωριστό όνομα. Ο δεύτερος τρόπος είναι να συμπεριλάβουμε τις ιδιότητες κατευθείαν στο αντικείμενο, όπως κατωτέρω**:**

<IMG SRC=”images/stoixeio.gif” WIDTH=120 HEIGHT=90 BORDER=0 style=”position:absolute; left:30; top:30; z-index:1;”>

ΤΟΠΟΘΕΤΗΣΗ ΣΤΟΙΧΕΙΩΝ ΣΤΗΝ ΟΘΟΝΗ

Η πιο σημαντική δουλειά των style sheets είναι η τοποθέτηση των στοιχείων της σελίδας στην οθόνη. Η πρώτη παράμετρος, Position, ορίζει σε σχέση με τι είναι τοποθετημένο το αντικείμενο.

Όταν είναι Absolute, το στοιχείο τοποθετείται σε σχέση με την άνω αριστερή γωνία της οθόνης, που έχει συντεταγμένες 0,0.

Όταν είναι Relative, το στοιχείο τοποθετείται σε σχέση με τη θέση κάποιου άλλου αντικειμένου.

Αυτή η μέθοδος όμως δεν χρησιμοποιείται συχνά.

Οι παράμετροι Left και Top είναι οι συντεταγμένες x και y αντίστοιχα της άνω αριστερής γωνίας του αντικειμένου. Με άλλες λέξεις, εάν θέλουμε να τοποθετήσουμε ένα γραφικό 30 pixels από την άνω αριστερή γωνία και 30 pixels από το αριστερό κενό του browser, τότε πρέπει να θέσουμε τις εξής τιμές: Left: 30px; Top: 30px;

Η παράμετρος Z-Index επιτρέπει στα στοιχεία να τοποθετηθούν πάνω από άλλα στοιχεία. Με αυτή την παράμετρο μπορούμε να δημιουργούμε επικαλυπτόμενα layers, όπως στο Adobe Photoshop. Οσο πιο μεγάλος είναι ο αριθμός τόσο πλησιέστερα στο θεατή βρίσκεται το αντικείμενο. Εάν το layer 1 περιέχει ένα γραφικό και το layer 2 περιέχει κείμενο και χρησιμοποιήσουμε το Z-INDEX, τότε το κείμενο θα εμφανιστεί πάνω στο γραφικό.

ΚΑΘΟΡΙΣΜΟΙ ΓΡΑΜΜΑΤΟΣΕΙΡΩΝ

Τα CSS παρέχουν ολικό έλεγχο στην παρουσίαση των γραμματοσειρών. Στο ακόλουθο πίνακα βλέπουμε μερικά από τα πιο σημαντικά tags για τον έλεγχο των γραμματοσειρών. Όλες αυτές οι ρυθμίσεις μπορούν να τοποθετηθούν μέσα στο style sheet, όπως δείξαμε ανωτέρω, είτε με τη χρήση του <STYLE> είτε συνημμένα στο αντικείμενο. Η σειρά δεν παίζει κανένα ρόλο και δεν είναι υποχρεωτική η χρήση όλων.

ΠΛΑΤΟΣ ΚΑΙ ΥΨΟΣ

Το μήκος και το πλάτος ορίζουν την περιοχή του DIV. Εάν δεν οριστούν αυτές οι μεταβλητές, τότε μια αυτόματη διαδικασία θα υπολογίσει το χώρο που χρειάζονται τα περιεχόμενα του DIV για να εμφανιστούν. Δυστυχώς, όμως, αυτή η αυτόματη διαδικασία δεν λειτουργεί πάντα σωστά.

ΣΤΟΙΧΙΣΗ ΚΕΙΜΕΝΟΥ

Η Στοίχιση κειμένου (alignment) είναι μια εύκολη διαδικασία. Με τη χρήση του DIV μπορούμε να στοιχίσουμε οποιοδήποτε στοιχείο, γραφικό, κείμενο ή ακόμα Java applet και plug-in.

ΧΡΩΜΑ ΚΑΙ ΕΙΚΟΝΑ ΦΟΝΤΟΥ

Μπορούμε, εάν θέλουμε, να ρυθμίσουμε το χρώμα ή και την εικόνα του φόντου της περιοχής DIV. Το χρώμα αυτό και η εικόνα θα εμφανίζονται τότε πίσω από όλα τα περιεχόμενα της περιοχής DIV. Εάν η περιοχή DIV περιέχει κείμενο, τότε το κείμενο αυτό θα έχει έγχρωμο φόντο, ενώ αν περιέχει ένα αρχείο transparent GIF, τότε γύρω του θα υπάρχει το έγχρωμο φόντο.

ΟΛΟΚΛΗΡΩΝΟΝΤΑΣ

Βέβαια, υπάρχουν πολύ περισσότερα tags από αυτά που αναφέραμε, τα οποία θα δούμε στα υπόλοιπα άρθρα. Τα Cascading Style Sheets είναι μια επανάσταση στο χώρο του Web authoring και μας προσφέρουν άπειρες δυνατότητες για να δημιουργήσουμε ελεύθερα. Το μόνο όριο πλέον είναι η φαντασία μας

Εξωτερικά Φύλλα Στυλ (External Style Sheets)

Ένα εξωτερικό φύλλο στυλ (external style sheet) είναι ιδανικό όταν το στυλ εφαρμόζεται σε πολλές σελίδες. Μ’ ένα εξωτερικό φύλλο στυλ μπορούμε να αλλάξουμε την εμφάνιση ενός ολόκληρου Web site αλλάζοντας ένα μόνο αρχείο. Η κάθε σελίδα πρέπει να έχει έναν δεσμό (link) προς το φύλλο στυλ που χρησιμοποιεί το tag <link>, ο οποίος βρίσκεται μέσα στο τμήμα head, ως εξής :

```
<head>
    <link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

Ο φυλλομετρητής θα διαβάζει τους ορισμούς στυλ από το αρχείο mystyle.css και θα μορφοποιήσει το έγγραφο σύμφωνα μ’ αυτό το αρχείο.

Ένα εξωτερικό φύλλο στυλ μπορεί να γραφεί σ' έναν οποιονδήποτε text editor. Το αρχείο δεν πρέπει να περιέχει καθόλου html tags και πρέπει να αποθηκευθεί με την επέκταση ***.css***. Ένα παράδειγμα ενός αρχείου φύλλου στυλ είναι το εξής :

```
hr {color: sienna}

p {margin-left: 20px}

body {background-image: url("images/back40.gif")}
```

Εσωτερικά Φύλλα Στυλ (Internal Style Sheets)

Ένα εσωτερικό φύλλο στυλ (internal style sheet) πρέπει να χρησιμοποιηθεί όταν ένα έγγραφο έχει ένα μοναδικό στυλ. Ορίζουμε τα εσωτερικά στυλ στο τμήμα head χρησιμοποιώντας το tag <*style*>, ως εξής :

```
<head>
    <style type="text/css">
        hr {color: sienna}
        p {margin-left: 20px}
        body {background-image: url("images/back40.gif")}
    </style>
</head>
```

Ο φυλλομετρητής θα διαβάσει τους ορισμούς των στυλ και θα μορφοποιήσει ανάλογα το έγγραφο.

Ένας φυλλομετρητής κανονικά αγνοεί τα άγνωστα tags. Αυτό σημαίνει ότι ένας παλιός φυλλομετρητής που δεν υποστηρίζει στυλ, θα αγνοήσει το tag <style>, αλλά το περιεχόμενο του tag <style> θα εμφανισθεί στη σελίδα.

Μπορούμε να εμποδίσουμε έναν παλιό φυλλομετρητή από το να εμφανίσει αυτό το περιεχόμενο κρύβοντάς το με το στοιχείο σχολίου της HTML, ως εξής :

```
<head>
    <style type="text/css">
        <!--
        hr {color: sienna}
```

```
            p {margin-left: 20px}

            body {background-image: url("images/back40.gif")}

            -->

        </style>

    </head>
```

Τα Inline Styles

Ένα inline style χάνει πολλά από τα πλεονεκτήματα των φύλλων στυλ αναμειγνύοντας το περιεχόμενο με την παρουσίαση. Πρέπει να χρησιμοποιούμε αυτή τη μέθοδο με φειδώ, όπως όταν ένα στυλ πρόκειται να εφαρμοσθεί σε μία μοναδική εμφάνιση ενός στοιχείου.

Για να χρησιμοποιήσουμε τα inline styles χρησιμοποιούμε το χαρακτηριστικό (attribute) style στο σχετικό tag. Το χαρακτηριστικό style μπορεί να περιέχει οποιαδήποτε ιδιότητα CSS. Το παρακάτω παράδειγμα δείχνει πώς μπορούμε να αλλάξουμε το χρώμα και το αριστερό περιθώριο μιας παραγράφου :

```
    <p style="color: sienna; margin-left: 20px">

        Αυτή είναι μια παράγραφος

    </p>
```

Τα Πολλαπλά Φύλλα Στυλ

Αν μερικές ιδιότητες έχουν ορισθεί για τον ίδιο επιλογέα (selector) σε διαφορετικά φύλλα στυλ, οι τιμές θα κληρονομηθούν από το γενικότερο φύλλο στυλ. Για παράδειγμα, ένα εξωτερικό φύλλο στυλ έχει αυτές τις ιδιότητες για τον επιλογεά h3 :

```
    h3

    {

        color: red;

        text-align: left;

        font-size: 8pt
```

```
}
```

Και ένα εσωτερικό φύλλο στυλ έχει αυτές τις ιδιότητες για τον επιλογέα h3 :

```
h3
{
        text-align: right;
        font-size: 20pt
}
```

Αν η σελίδα με το εσωτερικό φύλλο στυλ συνδέεται επίσης στο εξωτερικό φύλλο στυλ, οι ιδιότητες του h3 θα είναι οι εξής :

```
color: red;
text-align: right;
font-size: 20pt
```

Το χρώμα κληρονομείται από το εξωτερικό φύλλο στυλ και η στοίχιση κειμένου και το μέγεθος γραμματοσειράς αντικαθίστανται από το εσωτερικό φύλλο στυλ.

Παραδείγματα Κώδικα CSS

1ο Παράδειγμα

Ένα HTML αρχείο χρησιμοποιεί το tag <link> για να συνδεθεί μ' ένα εξωτερικό φύλλο στυλ (external style sheet) :

```
<html>
    <head>
        <link rel="stylesheet" type="text/css" href="ex1.css">
    </head>
    <body>
```

```
        <h1> This header is 36 pt </h1>

        <h2> This header is blue </h2>

        <p> This paragraph has a left margin of 50 pixels </p>

    </body>

</html>
```

Το αρχείο του εξωτερικού φύλλου στυλ το οποίο προφανώς θα βρίσκετε στον ίδιο φάκελο με τις σελίδες στις οποίες θα αναφέρετε αλλιώς θα πρέπει να δηλωθεί το path του, είναι το εξής (ex1.css) :

```
    body {background-color: yellow}

    h1 {font-size: 36pt}

    h2 {color: blue}

    p {margin-left: 50px}
```

2ο Παράδειγμα

Ένα HTML αρχείο χρησιμοποιεί το tag <link> για να συνδεθεί μ' ένα εξωτερικό φύλλο στυλ (external style sheet) :

```
<html>

    <head>

        <link rel=stylesheet type="text/css" href="ex2.css">

    </head>

    <body>

        <h1> This is a header 1 </h1>

        <hr>

        <p> You can see that the style sheet formats the text </p>

        <p><a href="http://www.microsoft.com" target="_blank">

            This is a link </a></p>

    </body>
```

</html>

Το αρχείο του εξωτερικού φύλλου στυλ είναι το εξής (ex2.css) :

body {background-color: tan}

h1 {color:maroon; font-size:20pt}

hr {color:navy}

p {font-size:11pt; margin-left: 15px}

a:link {color:green}

a:visited {color:yellow}

a:active {color:blue}

a:hover {color:black}

3ο Παράδειγμα

Καθορισμός της Απόστασης των Χαρακτήρων

<html>

<head>

<style>

h1 {letter-spacing: -3px}

h4 {letter-spacing: 0.5cm}

</style>

</head>

<body>

* *

* Note: Netscape 4 does not support the "letter-spacing" property.*

* *

<h1> This is header 1 </h1>

<h4> This is header 4 </h4>

```
        </body>

</html>
```

4ο Παράδειγμα

Ορισμός της Γραμματοσειράς Κειμένου

```
<html>
 <head>
   <style>
         h3 {font-family: times}
        p {font-family: courier}
       p.sansserif {font-family: "sans-serif"}
   </style>
 </head>

<body>
        <h3> This is header 3 </h3>
        <p> This is a paragraph </p>
       <p class="sansserif"> This is a paragraph </p>
</body>

</html>
```

5ο Παράδειγμα

Σημάδια σε μη Αριθμημένες Λίστες

```
<html>

        <head>

                <style>

                        ul.disc

                        {

                                list-style-type: disc

                        }

                        ul.circle

                        {

                                list-style-type: circle

                        }

                        ul.square
```

```
                {
                        list-style-type: square
                }
        </style>
    </head>
    <body>
        <ul class="disc">
            <li> Coffee </li>
            <li> Tea </li>
            <li> Coca Cola </li>
        </ul>
        <ul class="circle">
            <li> Coffee </li>
            <li> Tea </li>
            <li> Coca Cola </li>
        </ul>
        <ul class="square">
            <li> Coffee </li>
            <li> Tea </li>
            <li> Coca Cola </li>
        </ul>
    </body>
</html>
```

6ο Παράδειγμα

Ορισμός μιας Εικόνας σαν Φόντο

```
<html>
    <head>
        <style>
            body
            {
                background-image: url("../images/bgdesert.jpg")
            }
        </style>
    </head>
    <body>
    </body>
</html>
```

Η Γλώσσα Προγραμματισμού JavaScript

Εισαγωγή στην JavaScript

*Η **JavaScript*** είναι μια γλώσσα συγγραφής σεναρίων (scripting language) που χρησιμοποιείται για να προσθέσει εφέ και διαλογικότητα (αλληλεπίδραση-διαδραστικότητα, interactivity) στις ιστοσελίδες μας και είναι ανταγωνιστική της γλώσσας προγραμματισμού VBScript.

Δημιουργήθηκε από την εταιρεία Netscape και το αρχικό της όνομα ήταν *LiveScript*.

Ο κώδικας της JavaScript γράφεται σε καθαρό κείμενο (ASCII μορφή) και ενσωματώνεται μέσα στον κώδικα της HTML, μπορεί δε να εκτελεσθεί αμέσως ή όταν λαμβάνει χώρα ένα συμβάν (event). Δεν γίνεται μεταγλώττιση (compilation) του κώδικα της JavaScript, αρκεί μόνο ο φυλλομετρητής (browser) να υποστηρίζει την JavaScript.

Αν και ακούγουνται ίδιες, η Java και η JavaScript δεν έχουν καμία απολύτως σχέση μεταξύ τους, ούτε στη σύνταξή τους σαν γλώσσες προγραμματισμού ούτε και στις εφαρμογές που χρησιμοποιούνται.

Η JavaScript αποτελεί το πρώτο βήμα στην απλοποίηση της Java. Παρόλο που η Netscape ξεκίνησε την ανάπτυξη της JavaScript αυτόνομα, γρήγορα συνεργάστηκαν με τη Sun, συμφωνώντας να κάνουν τη JavaScript ένα υποσύνολο της Java. Η JavaScript μοιάζει κατά κάποιο τρόπο με τη Java αλλά είναι πολύ πιο απλή στη χρήση. Δεν υπάρχει ανάγκη βοηθητικών εργαλείων, ούτε χρειάζεται μεταγλώττιση εφαρμογών της Java. Το μόνο που χρειάζεται είναι να εισάγετε ένα σενάριο στις ηλεκτρονικές σελίδες σας και όταν ένας browser που υποστηρίζει την JavaScript φτάνει στη σελίδα, διαβάζει το σενάριο και ακολουθεί τις οδηγίες.

Που μπορείτε να βρείτε τη JavaScript

Αν θέλετε να δημιουργήσετε προγράμματα σε C ++ ή Visual Basic χρειάζεστε ένα περιβάλλον ανάπτυξης λογισμικού και έναν μεταγλωττιστή.

Δεν υπάρχει όμως περιβάλλον ανάπτυξης λογισμικού για την JavaScript και δεν θα χρειαστεί ποτέ να μεταγλωττίσετε τα σενάριά σας. Αυτό που θα χρειαστείτε είναι ένας συμβατός με την JavaScript browser. Δηλαδή ένας World Wide Web browser που μπορεί να διαβάσει, να μεταφράσει και να τρέξει τα σενάρια που δημιουργείτε. Αυτό ισχύει για Netscape Navigator 4.0 ή Internet Explorer 4.0 ή νεώτερες εκδόσεις.

Σε τι ωφελεί η JavaScript

Με τη JavaScript μπορείτε να φτιάξετε σενάρια που να εκτελούν αυτόματες εργασίες, π.χ όταν μια σελίδα του Web ανοίγει ή κλείνει. Επίσης μπορείτε να κάνετε την JavaScript να εκτελεί ενέργειες ανταποκρινόμενη σε ένα συγκεκριμένο γεγονός. Για παράδειγμα όταν ο χρήστης επιλέγει ένα κουμπί ή ένα σύνδεσμο, όταν εστιάζει από ένα στοιχείο μιας φόρμας σε ένα άλλο στοιχείο της κ.ο.κ. Οι ενέργειες αυτές μπορεί να είναι απλές. Τα σενάρια μπορεί να ανοίγουν νέα παράθυρα στον browser και να εμφανίζουν συγκεκριμένα HTML έγγραφα ή να παρουσιάζουν μια σελίδα επιλεγμένη από τον κατάλογο ιστορικού του browser. Μπορεί επίσης να είναι και περίπλοκες δηλαδή ένα σενάριο μπορεί να ελέγχει τα περιεχόμενα μιας φόρμας που θέλει να υποβάλει ένας χρήστης και στη συνέχεια να προειδοποιεί τον χρήστη αν τα δεδομένα είναι λάθος. Το σενάριο μπορεί να ψάξει για πληροφορίες σε μια μικρή βάση δεδομένων ή να κάνει πολύπλοκους υπολογισμούς οικονομικών στοιχείων.

Τι μπορείτε να κάνετε με την JavaScript

- Πολυμερή έγγραφα με πλαίσια
- Επαναφόρτωση μέρους του παραθύρου
- Δημιουργώντας έγγραφα με αλληλεπίδραση
- Περισσότερος έλεγχος στην αλληλεπίδραση με το χρήστη
- Έγγραφα με μνήμη
- Ζωντανά έγγραφα
- Μηνύματα που ολισθαίνουν
- Ρολόγια
- Χρονικός μηχανισμός αντίστροφης μέτρησης
- Έγγραφα με αυτόματη ενημέρωση

Η Εντολή document.write()

Ένα πρώτο απλό παράδειγμα σε JavaScript είναι το εξής :

```
<html>

  <body>

    <script type="text/javascript">

      document.write("Hello from JavaScript")

    </script>

  </body>

</html>
```

Ο κώδικας της JavaScript περιέχεται ανάμεσα στα tags **<*script*>** και **</*script*>** και σαν χαρακτηριστικό (attribute) μπορούμε να χρησιμοποιήσουμε το

type="text/javascript" ή το ***language="JavaScript"***. Μέσα σ' ένα αρχείο HTML μπορούμε να έχουμε όσα σύνολα tags <script> και </script> χρειαστούμε, είτε στο τμήμα head ή στο τμήμα body του εγγράφου.

Η εντολή ***document.write()*** χρησιμοποιείται για να μπορούμε να εμφανίσουμε κείμενο στην οθόνη του φυλλομετρητή μέσα από εντολές της JavaScript. Το κείμενο ***Hello from JavaScript*** θα εμφανισθεί μαζί με το υπόλοιπο κείμενο του HTML κώδικα. Αν και εδώ δεν φαίνεται καθαρά η αξία της JavaScript, θα φανεί όταν χρησιμοποιήσουμε εντολές ελέγχου, όπως είναι η if, για να μπορούμε να εμφανίσουμε διαφορετικό κείμενο ανάλογα με την τιμή κάποιων μεταβλητών ή ανάλογα με τις επιλογές που έχει κάνει ο χρήστης.

Events - Γεγονότα

Τα γεγονότα (Events) και οι χειριστές γεγονότων (event handlers) είναι ένα πολύ σημαντικό μέρος στον JavaScript προγραμματισμό. Τα Events προκαλούνται από τις πράξεις του χρήστη. Αν ο χρήστης πατήσει ένα κουμπί, τότε συμβαίνει ένα *Click*-event. Αν ο δείκτης του mouse κινηθεί πάνω από μια διεύθυνση (link), τότε συμβαίνει ένα *MouseOver*-event.

Υπάρχουν πολλά events ,αλλά εμείς Θέλουμε το javaScript πρόγραμμά μας να αντιδρά σε συγκεκριμένα events. Αυτό μπορεί να γίνει με την βοήθεια των event-handlers (χειριστές γεγονότων). Ένα κουμπί μπορεί να εμφανίζει ένα pop-up παράθυρο όταν πατιέται. Αυτό σημαίνει ότι το pop-up παράθηρο πρέπει να εμφανιστεί σαν απάντηση στο *Click*-event. Ο event-handler που χρειαζόμαστε λέγεται *onClick*. Αυτός λέει στον υπολογιστή τι να γίνει όταν συμβεί το ανάλογο event. Ο επόμενος κώδικας δείχνει ένα απλό παράδειγμα του event-handler *onClick*:

```
<form>
<input type="button" value="Πάτησε με" onClick="alert('Καλημέρα')">
</form>
```

Υπάρχουν πολλά νέα πράγματα σε αυτόν τον κώδικα - γι'αυτό ας τα δούμε βήμα-βήμα. Μπορείτε να δείτε ότι σχεδιάζουμε μία φόρμα με ένα κουμπί (αυτό γίνεται με HTML και γι'αυτό δεν μπαίνω σε λεπτομέρειες). Το νέο μέρος είναι onClick="alert('Καλημέρα')" μέσα στο <input>. Όπως είπαμε ήδη, αυτό καθορίζει τι θα συμβεί όταν το κουμπί πατηθεί. Έτσι αν ένα *Click*-event συμβεί, ο υπολογιστής θα εκτελέσει το alert('Καλημέρα'). Αυτό είναι JavaScript κώδικας (Σημειώστε ότι δεν χρησιμοποιούμε το <script language= javascript> σε αυτή την περίπτωση.

Η εντολή *alert()* μας επιτρέπει να δημιουργούμε pop-up παράθυρα. Μέσα στη παρένθεση πρέπει να βάλετε το κείμενο του pop-up παραθύρου. Σε αυτή τη περίπτωση είναι *'Καλημέρα'*. Έτσι το script δημιουργεί ένα παράθυρο με το κείμενο *'Καλημέρα'* όταν ο χρηστής πατήσει το κουμπί.

Ένα πράγμα που μπερδεύει λίγο: Στην εντολή document.write() χρησιμοποιούμε δύο άνω τελείες " και στο συνδυασμό alert() χρησιμοποιήσαμε μία άνω τελεία ' - γιατί; Βασικά μπορούμε να χρησιμοποιήσουμε και τα δύο. Όμως στο τελευταίο παράδειγμα γράψαμε onClick="alert('Καλημέρα')" - βλέπετε ότι χρησιμοποιούνται και τα δύο σύμβολα. Αν γράφαμε onClick="alert("Καλημέρα")" ο υπολογιστής θα μπερδευόταν μιας και δεν είναι εμφανές ποιο μέρος ανήκει στο onClick event-handler και ποιο όχι. Γι'αυτό πρέπει να εναλλαχθούν οι δύο με μία άνω τελεία σε αυτή τη περίπτωση. Δεν

έχει σημασία με ποια σειρά θα χρησιμοποιηθούν τα δύο σύμβολα. Αυτό σημαίνει ότι θα μπορούσα επίσης να γράψω onClick='alert("Καλημέρα")' και να έχω το ίδιο αποτέλεσμα.

Υπάρχουν πολλοί διαφορετικοί event-handlers που μπορούν να χρησιμοποιηθούν.

Αν χρησιμοποιείτε το Netscape Navigator το popup παράθυρο θα περιέχει το κείμενο *JavaScript alert* στο caption Bar (στο πάνω αριστερά μέρος του παραθύρου). Αυτός είναι περιορισμός ασφαλείας. Μπορείτε να δημιουργήσετε ένα παρόμοιο popup παράθυρο με την εντολή prompt(). Αυτό το παράθυρο δέχεται και τιμή που καθορίζει ο χρήστης για να χρησιμοποιηθεί αργότερα απ'το script. Π.χ. μπορεί αν χρησιμοποιηθεί σε ένα script που μιμείται ένα σύστημα προστασίας και να ρωτά για ένα συγκεκριμένο password. Το κείμενο στο popup παράθυρο δείχνει ότι το παράθυρο έρχεται από τον browser και όχι από το Operating system. Μιας και το Caption Bar είναι περιορισμός ασφαλείας, δεν μπορεί να αλλάξει.

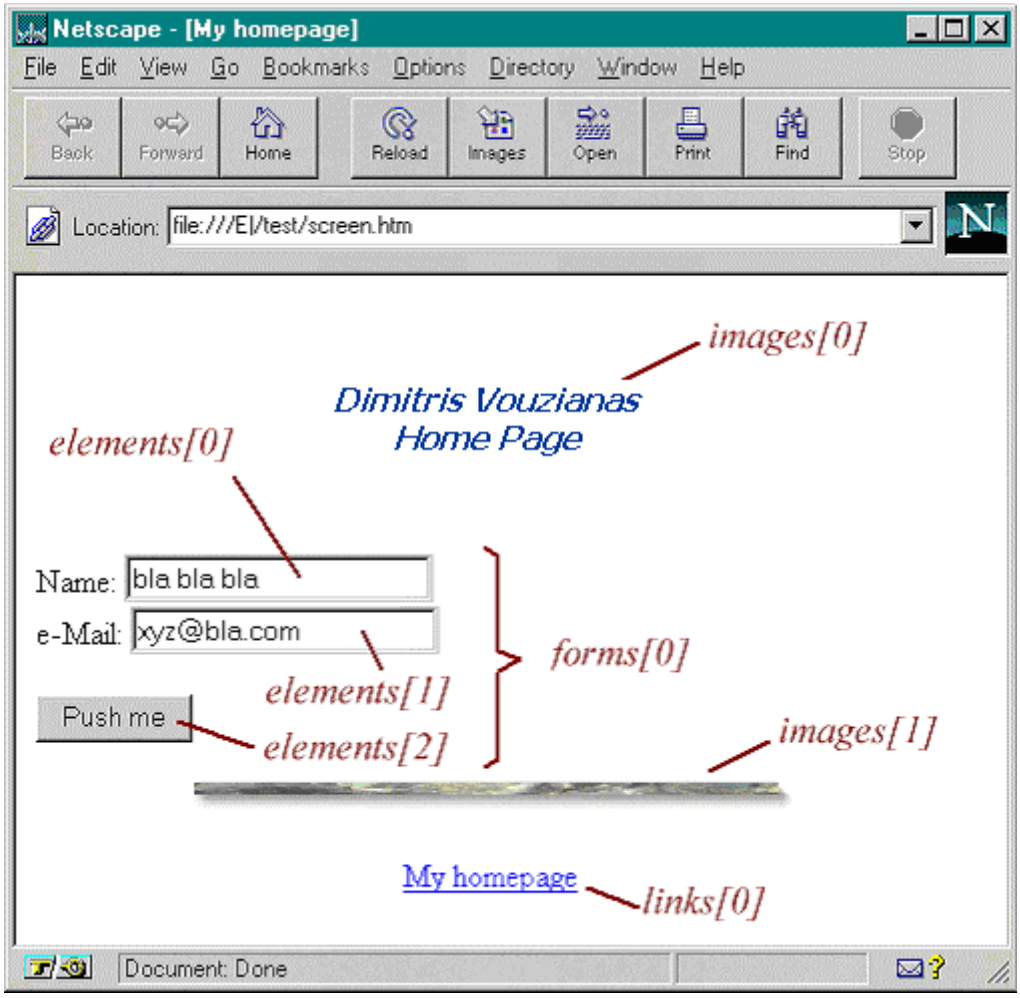
Ιεραρχία JavaScript

Το JavaScript οργανώνει όλα τα στοιχεία μιας web σελίδας σε μια ιεραρχία. Κάθε στοιχείο της σελίδας βλέπεται σαν αντικείμενο. Κάθε αντικείμενο έχει τις δικές του ιδιότητες (properties) και μεθόδους (methods). Με τη βοήθεια του JavaScript μπορείτε εύκολα να διαχειριστείτε τα αντικείμενα. Γι'αυτό είναι σημαντικό να καταλάβετε την ιεραρχία μιας HTML σελίδας. θα καταλάβετε εύκολα πως γίνεται αυτή η ιεραρχία με ένα παράδειγμα. Ο ακόλουθος κώδικας είναι μια απλή HTML σελίδα.

```
<html>
<head>
<title>My homepage
</head>
<body bgcolor=#ffffff>

<center>
<img src="home.gif" name="pic1" width=200 height=100>
</center>

<p>

<form name="myForm">
Name:
<input type="text" name="name" value=""><br>
e-Mail:
<input type="text" name="email" value=""><br><br>
<input type="button" value="Push me" name="myButton" onClick="alert('Hello')">
</form>

<p>
<center>
<img src="ruler.gif" name="pic4" width=300 height=15>
```

```
<p>

<a href="http://www.teiep.gr ">My homepage</a>
</center>

</body>
</html>
```

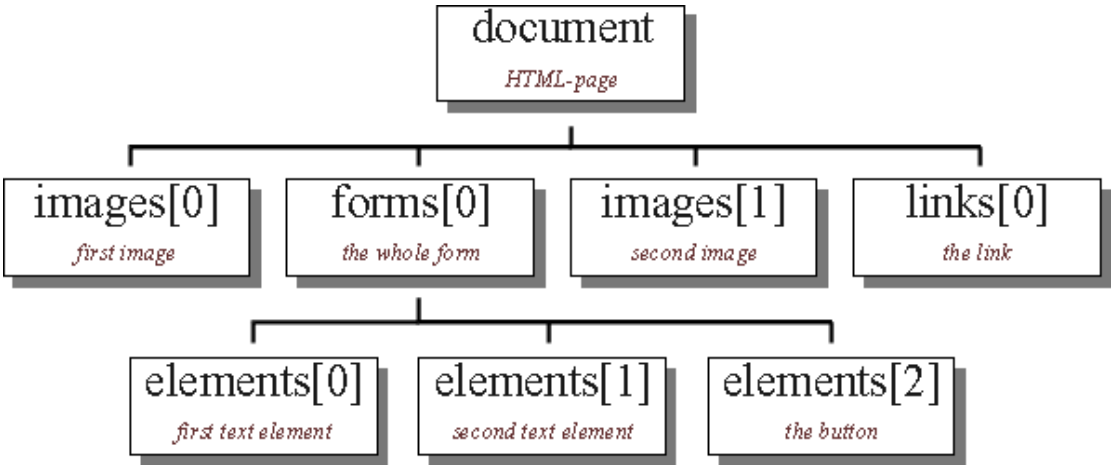
Εδώ βλέπουμε την σελίδα (έχω προσθέσει κάποια πράγματα με κόκκινο χρώμα. Επίσης, χρησιμοποιούνται οι αγγλικοί όροι, μιάς και έτσι τους καταλαβαίνει το JavaScript):



Έχουμε δύο εικόνες, ένα link και μία φόρμα με δύο πεδία κειμένου και ένα κουμπί. Από πλευράς JavaScript το παράθυρο του browser βλέπεται σαν ένα αντικείμενο, εν ονόματι *window*. Το αντικείμενο windows περιέχει συγκεκριμένα στοιχεία, όπως η StatusBar. Μέσα στο παράθυρο, μπορούμε να φορτώσουμε ένα HTML έγγραφο (ή άλλου τύπου αρχείο - προς το παρόν, θα περιοριστούμε σε HTML αρχεία).

Αυτή η σελίδα είναι το αντικείμενο *document*. Αυτό σημαίνει ότι το αντικείμενο document αντιπαριστά το HTML έγγραφο το οποίο βλέπει ο χρήστης εκείνη τη στιγμή. Το document είναι ένα πολύ σημαντικό αντικείμενο στο JavaScript - θα χρησιμοποιείται συχνά. Οι ιδιότητες (properties) του document είναι για παράδειγμα το χρώμα του φόντου. Αλλά το πιο σημαντικό είναι ότι όλα τα αντικείμενα HTML είναι ιδιότητες του document. Αντικείμενα είναι για παράδειγμα τα links, ή οι

φόρμες.
Η παρακάτω εικόνα παρουσιάζει την ιεραρχία του HTML εγγράφου που δείξαμε παραπάνω:



Θέλουμε να μπορούμε να πάρουμε πληροφορίες για τα διάφορα αντικείμενα και να τα διαχειριστούμε. Γι'αυτό πρέπει να ξέρουμε πώς να έχουμε πρόσβαση στα διάφορα αντικείμενα. Μπορείτε να δείτε τα ονόματα των αντικειμένων στην ιεραρχία. Αν θέλετε να ξέρετε πώς να απευθυνθείτε στην πρώτη εικόνα (image). Πρέπει να ξεκινήσετε απ'την αρχή. Το πρώτο αντικείμενο λέγεται *document*. Η πρώτη εικόνα της σελίδας αντιπροσωπεύεται από το *images[0]*. Αυτό σημαίνει ότι μπορούμε να έχουμε πρόσβαση στην εικόνα τυπώνοντας *document.images[0]*.
Αν για παράδειγμα θέλετε να ξέρετε τι τιμή έχει το πρώτο πεδίο κειμένου, πρέπει να σκεφτείτε ποιά διαδρομή μας δίνει πρόσβαση σε αυτό το πεδίο κειμένου. Ξανά αρχίζουμε από το πάνω μέρος της ιεραρχίας. Ακολουθήστε το "μονοπάτι" που φτάνει στο *elements[0]* - βάλτε όλα τα ονόματα που συναντάτε στη διάρκεια του μονοπατιού μαζί. Αυτό σημαίνει ότι έχετε πρόσβαση στο πρώτο πεδίο κειμένου με: *document.forms[0].elements[0]*
Αλλά τώρα πώς θα ξέρουμε το εισαχθέν κείμενο; Για να μάθουμε τις ιδιότητες και τις μεθόδους ενός αντικειμένου, πρέπει να διαβάσουμε την ανάλογη JavaScript reference Εδώ βλέπετε ότι ένα πεδίο κειμένου έχει και την ιδιότητα *value*. Η ιδιότητα αυτή αντιπροσωπεύει το κείμενο (την τιμή του πεδίου). Τώρα μπορείτε να διαβάσετε το κείμενο που εισήγαγε ο χρήστης:

```
name= document.forms[0].elements[0].value;
```

Η τιμή του κειμένου αποθηκεύεται στη μεταβλητή *name*. Μπορούμε τώρα να δουλέψουμε με αυτή τη μεταβλητή. Για παράδειγμα, μπορούμε να δημιουργήσουμε ένα popup παράθυρο alert("Hi " + name). Αν η τιμή είναι *'Arta'* η εντολή alert("Hello " + name) θα ανοίξει ένα popup παράθυρο με το κείμενο *'Hi Arta'*
Αν έχετε ιδιαίτερα μεγάλες σελίδες το να απευθύνεστε στα αντικείμενα με αριθμούς μπορεί να σας μπερδέψει αρκετά - π.χ. *document.forms[3].elements[17]* ή μήπως ήταν *document.forms[2].elements[18]*; Για να αποφύγετε αυτό το πρόβλημα μπορείτε να ονομάσετε τα διάφορα αντικείμενα. Ακολουθεί ένα τέτοιο παράδειγμα:

```
<form name="myForm">
```

```
Name:
<input type="text" name="name" value=""><br>
...
```

Αυτό σημαίνει ότι το *forms[0]* λέγεται επίσης *myForm*. Αντί για *elements[0]* μπορούμε να χρησιμοποιήσουμε το *name* (όπως προσδιορίζει η ιδιότητα NAME="" στο <input>). Έτσι αντί να γράφουμε

```
name= document.forms[0].elements[0].value;
```

μπορούμε να γράψουμε

```
name= document.myForm.name.value;
```

Αυτό το κάνει πολύ πιο εύκολο - ειδικά όταν έχουμε μεγάλες σελίδες με πολλά αντικείμενα. (Σημειώστε ότι το αν τα γράμματα είναι μικρά ή κεφαλαία έχει σημασία - αυτό σημαίνει ότι δεν μπορείτε να γράψετε *myform* αντί για *myForm*) Πολλές ιδιότητες των αντικειμένων δεν περιορίζονται στην δυνατότητα να τις "διαβάζεις". Μπορείς κάλλιστα να ρυθμίσεις τις τιμές τους. Για παράδειγμα μπορείτε να γράψετε ένα νέο κείμενο το πεδίο κειμένου.

Κοιτάξτε το παρακάτω παράδειγμα:

Μπλα, μπλα,

Ιδού ο κώδικας του πιο πάνω παραδείγματος - το ενδιαφέρον κομμάτι είναι μέσα στην ιδιότητα onClick-property στο δεύτερο <input>:

```
<form name="myForm">
<input type="text" name="input" value="Μπλα, μπλα, ">
<input type="button" value="Write"
 onClick="document.myForm.input.value= 'Καλημέρα!'; ">
```

Έχω γράψει εδώ ένα μικρό παράδειγμα. Εδώ μπορείτε να δείτε τη χρήση διάφορων αντικειμένων.

Πρώτα κοιτάξτε τον κώδικα να δείτε τι κάνει:

Ο πηγιαίος κώδικας:

```
<html>
<head>
<title>Objects</title>

<script language="JavaScript">
<!-- hide

function first() {

  // δημιουργεί ένα popup παράθυρο με το
  // κείμενο που έχει εισαχθεί στο πεδίο κειμένου

  alert("Η τιμή του πεδίου κειμένου είναι: " +
    document.myForm.myText.value);
}

function second() {

  // Αυτό το function κοιτάζει την κατάσταση του Check Box

  var myString= "The checkbox is ";

  // Είναι το CheckBox τικαρισμένο ή όχι?
```

```
  if (document.myForm.myCheckbox.checked) myString+= "τικαρισμένο"
    else myString+= "μη τικαρισμένο";

  // output string
  alert(myString);
}

// -->
</script>
</head>
<body bgcolor=lightblue>

<form name="myForm">
<input type="text" name="myText" value="Μπλα, μπλα, μπλα">
<input type="button" name="button1" value="Κουμπί 1"
  onClick="first()">
<br>
<input type="checkbox" name="myCheckbox" CHECKED>
<input type="button" name="button2" value="Κουμπί 2"
  onClick="second()">
</form>

<p><br><br>

<script language="JavaScript">
<!-- hide

document.write("Το χρώμα που έχει το φόντο είναι: ");
document.write(document.bgColor + "<br>");

document.write("Το κείμενο στο δεύτερο κουμπί είναι: ");
document.write(document.myForm.button2.value);

// -->
</script>

</body>
</html>
```

Το αντικείμενο location

Εκτός από τα αντικείμενα window και document υπάρχει άλλο ένα σημαντικό αντικείμενο: το location. Αυτό το αντικείμενο αντιπροσωπεύει την διεύθυνση του HTML εγγράφου που απεικονίζεται. Έτσι αν φορτώσατε την σελίδα *http://www.xyz.com/page.html* τότε η τιμή του *location.href* είναι ίδια με την πάνω διεύθυνση.

Ακόμα πιο σημαντικό είναι ότι μπορείτε να βάλετε νέες τιμές στο *location.href*. Αυτό το κουμπί, παραδείγματος χάριν, ανοίγει ένα άλλο έγγραφο σε αυτό το παράθυρο:

```
<form>
```

```
<input type=button value="Yahoo"
 onClick="location.href='http://www.yahoo.com'; ">
</form>
```

Frames

Δημιουργώντας frames

Μια ερώτηση που εμφανίζεται αρκετά συχνά είναι πώς μπορούν frames και JavaScript να συνεργαστούν. Πρώτα, θέλω να εξηγήσω τι είναι τα frames και σε τι χρησιμεύουν. Μετά απ'αυτό, θα δούμε πώς μπορούμε να συνδυάσουμε frames και JavaScript.
Το παράθυρο του browser μπορεί να χωριστεί σε πολλά frames. Αυτό σημαίνει ότι frame είναι μια τετράγωνη περιοχή μέσα σε ένα παράθυρο του browser. Κάθε frame δείχνει το δικό του έγγραφο (τις περισσότερες φορές HTML έγγραφα). Έτσι μπορείτε για παράδειγμα, να δημιουργήσετε δυο frames. Στο πρώτο να φορτώσετε τη σελίδα της Netscape και στο δεύτερο τη σελίδα της Microsoft.

Παρόλο που η δημιουργία frames είναι υπόθεση της HTML, θέλω να περιγράψω μερικά βασικά πράγματα. Για να δημιουργήσουμε frames, χρειαζόμαστε δύο εντολές: <frameset> και <frame>. Ένα έγγραφο με δύο frames μπορεί να δείχνει ως εξής:

```
<html>
<frameset rows="50%,50%">
  <frame src="page1.htm" name="frame1">
  <frame src="page2.htm" name="frame2">
</frameset>
</html>
```

Αυτό θα παράγει δύο frames. Μπορείτε να δείτε ότι χρησιμοποιούνται οι ιδιότητες *rows* στην εντολή <frameset>. Αυτό σημαίνει ότι τα δύο frames είναι το ένα πάνω στο άλλο. Το πάνω frame απεικονίζει την σελίδα *page1.htm* και το κάτω τη σελίδα *page2.htm*. Αν πατήσετε το επόμενο κουμπί, μπορείτε να δείτε πώς δείχνει:

Αν θέλετε να έχετε στήλες (columns) αντί για σειρές (rows), πρέπει να γράψετε *cols* αντί για *rows* στην εντολή <frameset>. Το *"50%,50%"* δείχνει πόσο μεγάλα θα είναι τα δύο έγγραφα. Μπορείτε επίσης να γράψετε *"50%,*"* αν δεν θέλετε να υπολογίσετε πόσο μεγάλο πρέπει να είναι το δεύτερο παράθυρο για να φτάσει το 100%. Μπορείτε επίσης να γράψετε το μέγεθος σε pixels, αν παραλείψετε το σύμβολο %. Κάθε frame παίρνει τη δικιά του *name* ιδιότητα στην εντολή <frame>. Αυτό θα μας βοηθήσει να έχουμε πρόσβαση στα frames μέσω JavaScript.

Μπορείτε να έχετε πολλά <frameset> το ένα να φωλιάζεται μέσα στο άλλο. Βρήκα αυτό το παράδειγμα στο έγγραφο της Netscape (το άλλαξα λίγο):

```
<frameset cols="50%,50%">
  <frameset rows="50%,50%">
    <frame src="cell.htm">
    <frame src="cell.htm">
  </frameset>
```

```
  <frameset rows="33%,33%,33%">
    <frame src="cell.htm">
    <frame src="cell.htm">
    <frame src="cell.htm">
  </frameset>
</frameset>
```

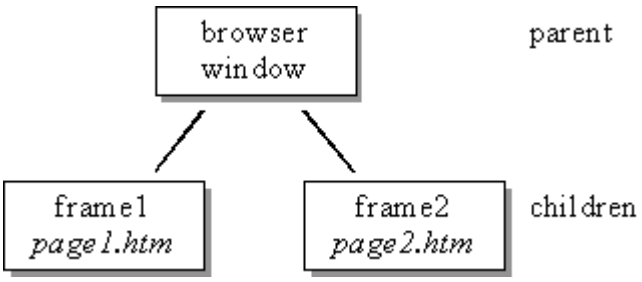
Κοιτάξτε αυτό το παράδειγμα:

Μπορείτε να ρυθμίσετε το μέγεθος της διαχωριστικής γραμμής, αλλάζοντας την τιμή της ιδιότητας *border σ*το <frameset>. border=0 σημαίνει ότι δεν θέλετε να υπάρχει διαχωριστική γραμμή (δεν δουλέυει στο Netscape 2.x).

Frames και JavaScript

Τώρα θέλουμε να ρίξουμε μια ματιά στο πώς το JavaScript 'βλέπει' τα frames σε ένα παράθυρο του browser. Γι'αυτό θα δημιουργήσουμε δύο frames, όπως στο πρώτο παράδειγμα, του πρώτου μέρους.

Έχουμε δει πως το JavaScript ταξινομεί όλα τα στοιχεία του παραθύρου σε μια ιεραρχική σειρά. Το ίδιο γίνεται και με τα frames. Η επόμενη εικόνα δείχνει την ιεραρχία του πρώτου παραδείγματος:



Στο πάνω μέρος της ιεραρχίας έχουμε το window. Το παράθυρο αυτό είναι χωρισμένο σε δύο frames. Το παράθυρο είναι ο γονέας (στην αγγλική γλώσσα parent) σε αυτή την ιεραρχία και τα δύο frames είναι τα παιδιά (children). Δώσαμε στα δύο frames τα ονόματα *frame1* και *frame2*. με τη βοήθεια των δύο ονομάτων μπορούμε να ανταλλάσσουμε πληροφορίες μεταξύ των δυο frames.

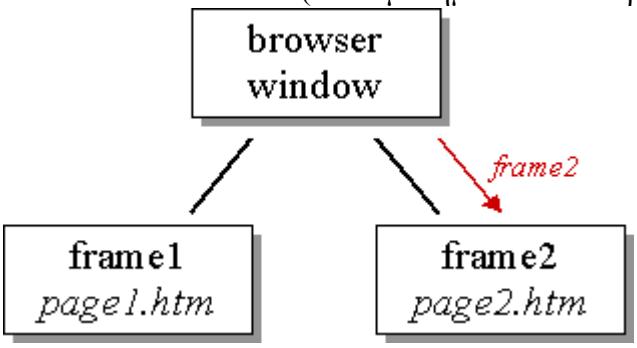
Ένα script πιθανώς θα πρέπει να λύσει το εξής πρόβλημα: ο χρήστης διαλέγει ένα link στο ένα frame - αλλά ο συγγραφέας θέλει η σελίδα να εμφανιστεί στο δεύτερο frame, κι όχι στο πρώτο. Αυτό μπορεί να χρησιμοποιηθεί για σελίδες Menu (εν ονόματι MenuBars ή NavigationBars) όπου το ένα frame μένει συνέχεια το ίδιο και προσφέρει πολλά διαφορετικά links, για την περιπλάνηση μέσα στην σελίδα του συγγραφέα.
Πρέπει να κοιτάξουμε τρεις περιπτώσεις:

- Το <I>parent window/frame αποκτά πρόσβαση στο *child frame*
- Το <I>child frame αποκτά πρόσβαση στο parent *window/frame*
- Το <I>child frame αποκτά πρόσβαση στο άλλο *child frame*

Από την πλευρά του παραθύρου τα δύο frames λέγονται *frame1* κει *frame2*. Μπορείτε να δείτε στην πάνω εικόνα ότι υπάρχει απ'ευθείας σχέση μεταξύ parent

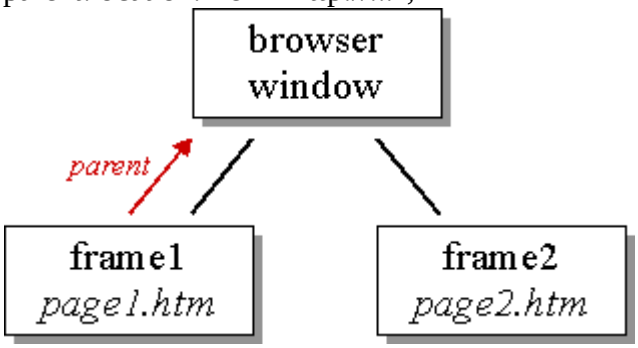
window και κάθε frame. Έτσι αν έχουμε ένα script στο parent window - αυτό σημαίνει στην σελίδα που δημιουργεί τα frames - και θέλουμε να έχουμε πρόσβαση στα frames, μπορούμε να χρησιμοποιήσουμε απλώς το όνομα του frame. Για παράδειγμα, μπορείτε να γράψετε:

frame2.document.write("Ενα μύνημα απ'το πατέρα παράθυρο.");



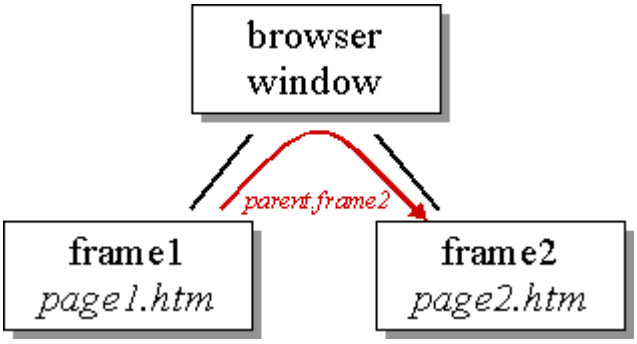
Μερικές φορές χρειάζεστε να έχετε πρόσβαση στο parent window από ένα frame. Αυτό χρειάζεται π.χ. αν θέλουμε να σβήσουμε τα frames. Σβήνοντας τα frames, σημαίνει να φορτωθεί μια νέα σελίδα στη θέση της σελίδας που δημιούργησε τα frames. Αυτό είναι στην περίπτωσή μας το parent window. Μπορούμε να έχουμε πρόσβαση στο parent window (ή parent frame) από τα child frames με το αντικείμενο *parent*. Για να φορτώσουμε ένα νέο document, πρέπει να δώσουμε νέα τιμή στο *location.href*. Εφόσον θέλουμε να διώξουμε τα frames, πρέπει να χρησιμοποιήσουμε το αντικείμενο location του parent window. Εφόσον το κάθε frame μπορεί να φορτώσει την δικιά του σελίδα έχουμε διαφορετικό αντικείμενο location για κάθε frame. Μπορούμε να φορτώσουμε νέα σελίδα στο parent window με την εντολή:

parent.location.href= "http://...";



Πολύ συχνά θα χρειαστείτε να έχετε πρόσβαση σε ένα child frame από ένα άλλο child frame. Πώς μπορείτε να γράψετε κάτι από το πρώτο frame στο δεύτερο frame - αυτό σημαίνει ποια εντολή πρέπει να χρησιμοποιήσετε στο κείμενο *page1.htm*; Στην εικόνα, βλέπετε ότι δεν υπάρχει απ'ευθείας σύνδεση μεταξύ των δύο frames. Αυτό σημαίνει ότι δεν μπορούμε απλώς να καλέσουμε *frame2* από το *frame1* μιας και αυτό το frame δεν ξέρει τίποτα για την ύπαρξη του άλλου frame. Για το parent window το δεύτερο frame λέγεται *frame2* και το parent window λέγεται *parent* για το πρώτο frame. Γι'αυτό πρέπει να γράψουμε τα εξής για να αποκτήσουμε πρόσβαση στο αντικείμενο document του δεύτερου frame:

parent.frame2.document.write("Γεια, το frame1 επιφέρει αυτές τις αλλαγές.");



Navigationbars

Ας κοιτάξουμε ένα Navigationbar. Θα έχουμε αρκετά links σε ένα frame. Αν ο χρήστης διαλέξει ένα απ'αυτά τα links, η σελίδα δεν θα εμφανιστεί στο ίδιο frame - θα εμφανιστεί στο άλλο frame.

Παράδειγμα:

Αρχίζοντας, χρειαζόμαστε ένα script το οποίο δημιουργεί τα frames. Αυτό το έγγραφο μοιάζει με το πρώτο παράδειγμα:

frames3.htm

```
<html>
<frameset rows="80%,20%">
  <frame src="start.htm" name="main">
  <frame src="menu.htm" name="menu">
</frameset>
</html>
```

Η σελίδα *start.htm* είναι η σελίδα που θα δείχνει το *main* frame στην αρχή.
Δεν υπάρχουν ιδιαίτερες απαιτήσεις γι'αυτή τη σελίδα.
Η επόμενη σελίδα είναι το frame *menu*:

menu.htm

```
<html>
<head>
<script language="JavaScript">
<!-- hide

function load(url) {
  parent.main.location.href= url;
}

// -->
</script>
</head>
<body>
```

```
<a href="javascript:load('first.htm')">Πρώτο link</a>
<a href="second.htm" target="main">Δεύτερο link</a>
<a href="third.htm" target="_top">Τρίτο link</a>

</body>
</html>
```

Πάνω μπορείτε να δείτε τρεις διαφορετικούς τρόπους για να φορτώσετε μια σελίδα στο frame *main*. Το πρώτο link χρησιμοποιεί τη function *load()*. Κοιτάξτε πώς καλείται αυτή η function:

```
<a href="javascript:load('first.htm')">first</a>
```

Μπορείτε να δείτε ότι μπορούμε να αφήσουμε τον browser να εκτελέσει το κώδικα JavaScript αντί για να φορτώσει άλλη σελίδα - απλώς πρέπει να χρησιμοποιήσουμε την παράμετρο *javascript:* στην ιδιότητα *href*. Μπορείετε να δείτε πως γράφω *'first.htm'* μέσα στη παρένθεση. Η τιμή αυτή περνά στην function *load()*. Η function *load()* έχει ρυθμιστεί ως:

```
function load(url) {
  parent.main.location.href= url;
}
```

Εδώ βλέπετε ότι γράφω *url* μέσα στη παρένθεση. Αυτό σημαίνει ότι η τιμή 'first1.htm' αποθηκεύεται στη μεταβλητή *url*. Μέσα στη function *load()* μπορούμε τώρα να χρησιμοποιήσουμε την μεταβλητή.

Το δεύτερο link χρησιμοποιεί την ιδιότητα *target*. Αυτό, στην πραγματικότητα δεν είναι JavaScript. Είναι ενέργεια της HTML. Βλέπετε ότι πρέπει να καθορίσουμε το όνομα του frame. Σημειώστε ότι δεν πρέπει να βάλουμε *parent* πριν απ'το όνομα του frame. Αυτό μπορεί να σας μπερδέψει λίγο. Ο λόγος γι'αυτό είναι ότι το *target* είναι HTML και όχι javascript.

Το τρίτο link δείχνει πώς να βγάλουμε τα frames χρησιμοποιώντας την ιδιότητα *target*.
Αν θέλετε να βγάλετε τα frames με την function *load()* πρέπει απλώς να γράψετε parent.location.href= url μέσα στο function.

Ποιο τρόπο πρέπει να διαλέξετε; Αυτό εξαρτάται απ'το script και τι θέλετε να κάνετε. Η ιδιότητα *target* είναι πολύ απλή στη χρήση. Μπορείτε να τη χρησιμοποιήσετε αν θέλετε να φορτώσετε άλλη σελίδα σε κάποιο frame. Η λύση JavaScript (όπως το πρώτο link) χρησιμοποιείται συνήθως όταν θέλουμε να κάνουμε πολλά πράγματα σαν απάντηση στο πάτημα του link. Ένα συνηθισμένο πρόβλημα είναι να φορτώσετε δύο σελίδες σε δύο διαφορετικά frames. Παρόλο που μπορείτε να το λύσετε αυτό χρησιμοποιώντας την ιδιότητα *target*, χρησιμοποιώντας JavaScript functions είναι πιο άμεσο. Ας πούμε ότι έχετε τρία frames, με ονόματα *frame1*,*frame2* και *frame3*. Ο χρήστης πατά ένα link στο *frame1*. Τότε θέλετε να φορτώσετε δύο διαφορετικές σελίδες στα δύο άλλα frames. Μπορείτε, παραδείγματος χάριν, να χρησιμοποιήσετε την επόμενη function:

```
function loadtwo() {
  parent.frame1.location.href= "first.htm";
  parent.frame2.location.href= "second.htm";
}
```

Αν θέλετε να διατηρήσετε την function πιο εύκαμπτη, μπορείτε να χρησιμοποιήσετε variable passing. Η function θα δείχνει ως εξής:

```
function loadtwo(url1, url2) {
  parent.frame1.location.href= url1;
  parent.frame2.location.href= url2;
}
```

Μπορείτε να καλείτε αυτή τη function γράφοντας loadtwo("first.htm", "second.htm") ή loadtwo("third.htm", "forth.htm"). Χρησιμοποιώντας Variable passing κάνει τη function πιο εύκαμπτη. Μπορείτε να τη χρησιμοποιήσετε ξανά και ξανά με διαφορετικό περιεχόμενο.

Φόρμες

Εξετάζοντας την φόρμα

Οι φόρμες χρησιμοποιούνται συχνά στο Internet. Η φόρμα στέλνεται συνήθως μέσω του server ή μέσω ταχυδρομείου σε ένα συγκεκριμένο e-mail λογαριασμό. Μα πώς μπορούμε να είμαστε σίγουροι ότι ο χρήστης συμπλήρωσε σωστά τα πεδία της φόρμας? Με τη βοήθεια του JavaScript, η φόρμα μπορεί εύκολα να ελεγχθεί πριν σταλθεί. Πρώτα, θέλω να σας δείξω πώς μπορούμε να εξετάσουμε μια φόρμα. Μετά, θα ρίξουμε μια ματιά στο να στέλνουμε πληροφορίες μέσω Internet.

Πρώτα απ'όλα, θέλουμε να δημιουργήσουμε ένα script. Η HTML θα περιέχει δύο πεδία κειμένου. Ο χρήστης θα γράφει το όνομά του στο πρώτο και την e-mail διεύθυνση του στο δεύτερο. Μπορείτε να εισάγετε οτιδήποτε στα πεδία κειμένου και να πατήσετε το κουμπί. Επίσης, προσπαθήστε να μη βάλετε τίποτα και να πατήσετε το κουμπί.

Το όνομά σας:

Η e-mail διεύθυνση σας:

Όσον αφορά το πρώτο πεδίο κειμένου, θα λάβετε ένα μύνημα λάθους, αν δεν γράψετε τίποτα. Κάθε άλλη τιμή του πεδίου θεωρείται σωστή. Φυσικά, αυτό δεν εμποδίζει τον χρήστη απ'το να γράψει ένα ψεύτικο όνομα. Ο browser δέχεται ακόμα και αριθμούς. Αν θέλετε λοιπόν να βάλετε την τιμή '17' θα πάρετε αποτέλεσμα 'Γεια 17!'. Γι'αυτό λοιπόν, αυτός δεν είναι ο καλύτερος έλεγχος.
Η δεύτερη φόρμα είναι πιο έξυπνα φτιαγμένη. Προσπαθήστε να βάλετε ένα απλό κείμενο - το όνομά σας για παράδειγμα. Δεν θα δουλέψει (εκτός αν έχετε ένα @ στο όνομά σας...). Αυτό το κριτήριο για να δεχθεί την τιμή σαν κανονική e-mail διεύθυνση είναι το @. Ένα απλό @ θα κάνει την δουλειά - αλλά αυτό δεν έχει και πολύ νόημα. Κάθε Internet e-mail διεύθυνση περιέχει @, έτσι φαίνεται σωστό να ελέγχουμε για @ εδώ.

Πώς δείχνει το script γι'αυτά τα δύο πεδία κειμένου και την εξέτασή τους; Ιδού ο κώδικας:

```
<html>
<head>
<script language="JavaScript">
<!-- Hide

function test1(form) {
  if (form.text1.value == "")
    alert("Παρακαλώ, εισάγετε μια τιμή!")
  else {
   alert("Γεια "+form.text1.value+"! Η φόρμα είναι ok!");
  }
}

function test2(form) {
  if (form.text2.value == "" ||
      form.text2.value.indexOf('@', 0) == -1)
       alert("Η e-mail διεύθυνση δεν είναι σωστά γραμμένη!");
  else alert("OK!");
}
// -->
</script>
</head>

<body>
<form name="first">
Enter your name:<br>
<input type="text" name="text1">
<input type="button" name="button1" value="Δοκιμή πεδίου"
onClick="test1(this.form)">
<P>
Enter your e-mail address:<br>
<input type="text" name="text2">
<input type="button" name="button2" value="Δοκιμή πεδίου"
onClick="test2(this.form)">
</body>
</html>
```

Πρώτα πρέπει να ρίξουμε μια ματιά στον HTML κώδικα στο κύριο μέρος. Δημιουργούμε δύο πεδία κειμένου και δύο κουμπιά. Τα κουμπιά καλούν τις functions test1(...) και test2(...), ανάλογα με ποιό κουμπί πατιέται. Περνάμε την τιμή *this.form* στις functions για να μπορούμε να καθορίσουμε τα σωστά πεδία στις functions αργότερα.

Η function test1(form) δοκιμάζει αν το πεδίο είναι κενό. Αυτό γίνεται γράφοντας *if (form.text1.value == "")...* . Το 'form' είναι η μεταβλητή που λαμβάνει την τιμή 'this.form' όταν καλείται η function. Μπορούμε να πάρουμε την τιμή του πεδίου κειμένου χρησιμοποιόντας το 'value' σε συνδυασμό με το *form.text1*. Για να κοιτάξουμε αν η τιμή είναι κενή, την συγκρίνουμε με "". Αν η τιμή είναι ίση με "", τότε δεν υπάρχει τιμή. Ο χρήστης θα πάρει το μύνημα λάθους. Αν η τιμή δεν ισούται με "", τότε ο χρήστης θα λάβει ok.

Το πρόβλημα εδώ είναι ότι ο χρήστης μπορεί να εισάγει μόνο κενά. Αυτό θα βλέπεται σαν σωστή τιμή! Αν θέλετε, μπορείτε να ελέγχετε γι'αυτές τις πιθανότητες

και να τις αποκλείετε. Πιστεύω ότι θα'ναι εύκολο με τις πληροφορίες που δίνονται εδώ.

Τώρα, πρέπει να κοιτάξουμε στη function *test2(form)*. Αυτή η function συγκρίνει ξανά την τιμή με το "", για να δει αν έχει εισαχθεί τιμή ή όχι. Αλλά έχουμε προσθέσει κάτι στο if. Το || λέγεται OR operator. Το έχετε μάθει αυτό στο μέρος 6 αυτής της εισαγωγής.

Το if ελέγχει αν το πρώτο ή το δεύτερο κριτήριο σύκγρισης είναι σωστό. Αν τουλάχιστον το ένα απ'τα δύο είναι σωστό, η όλη if εντολή είναι σωστή και το ακόλουθο κείμενο θα εκτελεστεί. Αυτό σημαίνει ότι θα πάρετε ένα μύνημα λάθους αν η τιμή είναι κενή ή δεν υπάρχει @ στο κείμενο. Το δεύτερο operation στην εντολή if κοιτάζει αν υπάρχει @ στη τιμή.

Ελέγχοντας για συγκεκριμένους χαρακτήρες

Μερικές φορές θέλετε να περιορίσετε τη φόρμα σε συγκεκριμένους χαρακτήρες ή αριθμούς. Σκεφτείτε ένα τηλεφωνικό αριθμό - η τιμή θα έπρεπε να έχει μόνο αριθμούς (υπολογίζουμε ότι ο αριθμός δεν περιέχει άλλους χαρακτήρες).Θα μπορούσαμε να ελέγξουμε αν το κείμενο είναι μόνο αριθμοί. Αλλά πολλοί άνθρωποι χρησιμοποιούν άλλους χαρακτήρες στους αριθμούς - π.χ.: 01234-56789, 01234/56789 ή 01234 56789 (με κενό, δηλαδή, ενδιάμεσα). Ο χρήστης δεν πρέπει να αναγκαστεί να μην χρησιμοποιήσει αυτά τα σύμβολα. Έτσι πρέπει να επεκτείνουμε το script να ελέγχει για ψηφία και σύμβολα.

Telephone:

Ιδού ο πηγιαίος κώδικας:

```
<html>
<head>
<script language="JavaScript">
<!-- hide

// ********************************************************
// ΕΛΕΓΧΟΣ ΓΙΑ ΣΥΓΚΕΚΡΙΜΕΝΟΥΣ ΧΑΡΑΚΤΗΡΕΣ
// ********************************************************


function check(input) {
  var ok = true;

  for (var i = 0; i < input.length; i++) {
    var chr = input.charAt(i);
    var found = false;
    for (var j = 1; j < check.length; j++) {
      if (chr == check[j]) found = true;
    }
    if (!found) ok = false;
  }
```

```
  return ok;
}

function test(input) {

  if (!check(input, "1", "2", "3", "4",
       "5", "6", "7", "8", "9", "0", "/", "-", " ")) {

    alert("Input not ok.");
  }
  else {
    alert("Input ok!");
  }
}

// -->
</script>
</head>

<body>
<form>
Telephone:
<input type="text" name="telephone" value="">
<input type="button" value="Έλεγχος"
  onClick="test(this.form.telephone.value)">
</form>
</body>
</html>
```

Η function test() προσδιορίζει ποιοι χαρακτήρες επιτρέπονται.

Στέλνοντας μια φόρμα

Ποίοι διαφορετικοί τρόποι υπάρχουν για να στείλεις μία φόρμα; Ο απλός τρόπος είναι η αποστολή της φόρμας μέσω e-mail. Αυτή τη μέθοδο θα την αναλύσουμε σ'αυτό το σημείο.

Αν θέλετε τη φόρμα να τη χειριστεί ο server, χρειάζεστε να χρησιμοποιήσετε CGI (Common Gateway Interface). Αυτό επιτρέπει την διαχείριση της φόρμας αυτόματα. Ο server μπορεί να δημιουργήσει για παράδειγμα, μια database από τις φόρμες που παραλήφθηκαν απ'τους πελάτες. Άλλο παράδειγμα είναι οι index-pages όπως το Yahoo. Έχουν μια φόρμα για να ψάχνουν στην database. Ο χρήστης παίρνει απάντηση γρήγορα, όταν πατηθεί το κουμπί submit. Δεν είναι ανάγκη να περιμένει μέχρι οι άνθρωποι που συντηρούν τον server να διαβάσουν την φόρμα και να κοιτάξουν για την πληροφορία. Αυτή η δουλειά γίνεται αυτόματα απ'το server. Το JavaScript δεν μπορεί να κάνει τέτοια πράγματα.

Δεν μπορείτε να δημιουργήσετε guestbooks με την JavaScript, γιατί το JavaScript δεν είναι ικανό να γράψει ένα αρχείο στον server. Αυτό μπορεί να γίνει μόνο μέσω CGI. Φυσικά, μπορείτε να δημιουργήσετε

ένα guestbook μέσω e-mail. Αλλά θα πρέπει να βάζετε τις τιμές μόνοι σας. Αυτό είναι εντάξει αν δεν περιμένετε 1000 γράμματα την ημέρα. Το script είναι μόνο HTML. Έτσι, το JavaScript δεν χρειάζεται εδώ! Μόνο, αν χρειαστείτε να ελέγξετε τις τιμές των πεδίων θα χρειαστεί το JavaScript. Θα πρέπει, επίσης να προσθέσω, ότι η εντολή mailto δεν πιάνει παντού - π.χ. ο Microsoft Internet Explorer δεν το υποστηρίζει με αυτή τη μορφή (μόνο ως anchor).

```
<form method=post action="mailto:your.address@goes.here" enctype="text/plain">
Σας αρέσει αυτή η σελίδα?
  <input name="choice" type="radio" value="1">καθόλου.<br>
  <input name="choice" type="radio" value="2" CHECKED>Σπατάλημα χρόνου
είναι.<br>
  <input name="choice" type="radio" value="3">Η χειρότερη σελίδα που έχω
δει.<br>
  <input name="submit" type="submit" value="Αποστολή">
</form>
```

Η ιδιότητα *enctype="text/plain"* χρησιμοποιείται για να στείλει απλό κείμενο, κι όχι μέρη κώδικα. Αυτό κάνει την ανάγνωση του γράμματος πιο εύκολη.

Αν θέλετε να ελέγξετε την φόρμα πριν σταλθεί, μπορείτε να χρησιμοποιήσετε το onSubmit event-handler. Μπορείτε να βάλετε αυτόν τον event-handler στην εντολή <form>. Ο κώδικας θα δείχνει ως εξής:

```
function validate() {
  // Έλεγχος της φόρμας
  // ...

  if (inputOK) return true
    else return false;
}

...

<form ... onSubmit="return validate()">

...
```

Με αυτό τον κώδικα, η φόρμα δεν στέλνεται, αν έχει γίνει κάποιο λάθος.

Πώς να κάνετε focus σε ένα συγκεκριμένο αντικείμενο

Με τη βοήθεια της μεθόδου focus(), μπορείτε να κάνετε την φόρμα σας πιο φιλική ως προς τον χρήστη. Μπορείτε να καθορίσετε ποιο στοιχείο της φόρμας είναι εστιασμένη στην αρχή. Ή μπορείτε να πείτε στον browser να εστιάσει στο πεδίο της φόρμας που είχε λάθος τιμή. Αυτό σημαίνει ότι ο browser θα τοποθετήσει τον κέρσορα στο στοιχείο της φόρμας, έτσι ο χρήστης δεν είναι ανάγκη να κάνει click στο πεδίο πριν εισάγει τιμή Το ακόλουθο script θα κάνει την δουλειά αυτή:

```
function setfocus() {
  document.first.text1.focus();
}
```

bla bla bla

Αυτό το script θα εστίαζε το πρώτο πεδίο κειμένου χάρις στο script που έδειξα πάνω. Πρέπει να καθορίσετε το όνομα της φόρμας - που εδώ λέγεται *first* - και το όνομα του κάθε στοιχείου - εδώ *text1*. Αν θέλετε να εστιάσετε αυτό το στοιχείο της φόρμας όταν φορτώνεται η σελίδα, μπορείτε νε θέσετε την ιδιότητα onLoad στην εντολή <body>. Αυτό θα έδειχνε ως εξής:

<body onLoad="setfocus()">
Μπορούμε επίσης να μεγαλώσουμε το κώδικα ως εξής:

```
function setfocus() {
  document.first.text1.focus();
  document.first.text1.select();
}
```

Δοκιμάστε αυτό το κώδικα:

bla bla bla

Βλέπετε ότι το πεδίο κειμένου εστιάζεται και επιλέγεται η τιμή του.

Αναφορά σε Φόρμες και σε Στοιχεία Φόρμας

Μπορούμε να αναφερόμαστε σε μια φόρμα χρησιμοποιώντας τον πίνακα ***forms[ ]***, ο οποίος περιέχει μια λίστα απ' όλα τα αντικείμενα φόρμας που υπάρχουν στο έγγραφο. Παρόμοια, μπορούμε να χρησιμοποιήσουμε την ιδιότητα (πίνακα) ***elements[ ]*** μιας φόρμας για να αναφερόμαστε στα στοιχεία μιας φόρμας, η οποία ιδιότητα είναι μια λίστα απ' όλα τα αντικείμενα-στοιχεία που περιέχει η φόρμα. Η αρίθμηση ξεκινά από το 0.

Υπάρχουν δύο τρόποι για να αναφερθούμε σε μια φόρμα. Αν η φόρμα που μας ενδιαφέρει λέγεται secondForm και είναι η δεύτερη φόρμα σε μια ιστοσελίδα και θέλουμε να βρούμε πόσα στοιχεία (elements) περιέχει, μπορούμε να χρησιμοποιήσουμε την ιδιότητα length της φόρμας μ' έναν από τους εξής δύο τρόπους :

document.forms[1].length

document.secondForm.length

Παρόμοια, μπορούμε να αναφερθούμε στα στοιχεία μιας φόρμας με το όνομα ή τη θέση τους μέσα στον πίνακα στοιχείων. Για παράδειγμα, για να πάρουμε την τιμή ενός πλαισίου κειμένου με όνομα Lname που βρίσκεται στη φόρμα secondForm και είναι το τέταρτο κατά σειρά στοιχείο της φόρμας, μπορούμε να χρησιμοποιήσουμε έναν από τους εξής τέσσερις τρόπους :

document.secondForm.Lname.value

document.secondForm.elements[3].value

document.forms[1].Lname.value

document.forms[1].elements[3].value

Layers

Τι είναι layers?

Τα Layers είναι μια νέα δυνατότητα του Netscape Navigator 4.0. Σας επιτρέπει την ακριβή τοποθέτηση αντικειμένων, όπως εικόνες. Επιπλέον, μπορείτε να μετακινήσετε αντικείμενα μέσα στη σελίδα. Μπορείτε επίσης, να κρύψετε αντικείμενα. Τα Layers μπορούν να χειριστούν εύκολα μέσω JavaScript. Ελπίζω να σας αρέσουν τα layers όσο σε εμένα.

Προς το παρόν μπορείτε να χρησιμοποιήσετε layers μόνο με τον Microsoft Internet Explorer 4.0 και πάνω και Netscape Navigator 4.0 και πάνω!

Τι ακριβώς είναι τα layers; Αυτό μπορεί να εξηγηθεί εύκολα: παίρνετε πολλά κόλλες χαρτιών. Σε ένα απ'αυτά ζωγραφίζετε μια εικόνα. Σε ένα άλλο γράφετε κείμενο. Στο τρίτο μια εικόνα, στο τέταρτο κείμενο, κ.ο.κ. Τώρα, απλώστε τις κόλλες σε ένα τραπέζι. Ας πούμε ότι κάθε χαρτί είναι ένα layer. Απ'αυτή την οπτική γωνία, ένα layer είναι ένα είδος container. Μπορεί να περιλαμβάνει μέσα του αντικείμενα - σε αυτή τη περίπτωση εικόνες και κείμενο.

Τώρα πάρτε ένα χαρτί που έχει εικόνα. Μετακινήστε το πάνω στο τραπέζι. Παρακολουθήστε την εικόνα καθώς μετακινείτε το χαρτί. Αν μετακινήσετε το χαρτί στα δεξιά, η εικόνα θα ακολουθήσει! Τι μαθαίνουμε απ'αυτό; Τα Layers τα οποία μπορούν να περιλαμβάνουν διάφορα αντικείμενα - όπως εικόνες, φόρμες, κείμενο, κ.λ.π. - μπορούν να τοποθετηθούν στη σελίδα και να μετακινηθούν. Αν μετακινήσετε ένα layer, όλα τα αντικείμενα που ανήκουν σε αυτό, θα μετακινηθούν ανάλογα. Τα Layers μπορούν να τοποθετηθούν το ένα πάνω απ'το άλλο, όπως και τα χαρτιά στο τραπέζι. Κάθε layer μπορεί να έχει διάφανα μέρη. Δημιουργήστε μια τρύπα στο χαρτί. Τώρα βάλτε αυτό το χαρτί πάνω από ένα άλλο. Η τρύπα είναι το διαφανές μέρος του πρώτου χαρτιού - το περιεχόμενο του δεύτερου χαρτιού φαίνεται μέσω της τρύπας.

Δημιουργώντας layers

Για να δημιουργήσουμε ένα layer χρειαζόμαστε είτε το <layer> ή το <ilayer>. Μπορείτε να δείτε τις διάφορες ιδιότητές τους:

name="*layerName*"	Το όνομα του layer
left=*xPosition*	Η οριζόντια απόσταση από την αριστερή πλευρά του παραθύρου
top=*yPosition*	Η κάθετη απόσταση από την πάνω πλευρά του παραθύρου
z-index=*layerIndex*	Αριθμός αναφοράς του layer

width=*layerWidth*	Το πλάτος του layer σε pixel
clip="*x1_offset, y1_offset, x2_offset, y2_offset*"	Καθορίζει την επιφάνεια του layer η οποία θα φαίνεται
above="*layerName*"	Καθορίζει πάνω από ποιο layer θα εμφανιστεί
below="*layerName*"	Καθορίζει κάτω από ποιο layer θα εμφανιστεί
Visibility=show|hide|inherit	Καθορίζει αν το layer θα είναι ορατό ή όχι00
bgcolor="*rgbColor*"	Το χρώμα του φόντου - είτε το όνομα του χρώματος είτε rgb τιμή
background="*imageURL*"	Εικόνα για φόντο

Το <layer> χρησιμοποιείται για τα layers τα οποία μπορούν να τοποθετηθούν σε οποιαδήποτε θέση. Αν δεν θέσετε θέση (με τις ιδιότητες *left* και *top*), το layer θα τοποθετηθεί στο πιο αριστερό πάνω μέρος του παραθύρου.
Το <ilayer> δημιουργεί ένα layer του οποίου η θέση εξαρτάται από τη ροή του κειμένου.

Ας αρχίσουμε με ένα εύκολο παράδειγμα. Θέλουμε να δημιουργήσουμε δύο layers. Στο πρώτο layer βάζουμε μια εικόνα, και στο δεύτερο layer βάζουμε κείμενο. Αυτό που θέλουμε να πετύχουμε είναι να βάλουμε το κείμενο πάνω απ'την εικόνα.

Το κείμενο φαίνεται πάνω απ'την εικόνα
Ο κώδικας είναι ο εξής:

```
<html>

<layer name=pic z-index=0 left=200 top=100>
<img src="img.gif" width=160 height=120>
</layer>

<layer name=txt z-index=1 left=200 top=100>
<font size=+4> <i> Layers-Demo </i> </font>
</layer>

</html>
```

Βλέπετε ότι δημιουργούμε δύο layers με την εντολή <layer>. Και τα δύο layers έχουν τοποθετηθεί στη θέση 200/100 (που καθορίστηκε απ'τις ιδιότητες *left* και *top*).

Οτιδήποτε μεταξύ του <layer> και του </layer> (ή <ilayer> και </ilayer>) ανήκουν σ'αυτό το Layer.

Βλέπετε ότι χρησιμοποιούμε την ιδιότητα *z-index*. Αυτό καθορίζει με ποια σειρά θα εμφανιστούν τα layers - στην περίπτωσή μας, καθορίζουμε ότι το κείμενο θα είναι πάνω απ΄την εικόνα. Το layer με το μεγαλύτερο z-index εμφανίζεται από πάνω. Δεν είναι ανάγκη να χρησιμοποιήσετε τις τιμές 0 και 1 για το z-index - κάθε θετική τιμή είναι εντάξει.

Αν γράψετε *z-index=100* στο πρώτο <layer>, το κείμενο θα εμφανιστεί κάτω απ'την εικόνα - μιας και το layer με το κείμενο έχει μικρότερη τιμή z-index (z-index=1). Εδώ βλέπετε το κείμενο κάτω απ'την εικόνα, μιας και χρησιμοποίησα διάφανο φόντο (gif89a format).

Το κείμενο εμφανίζεται κάτω απ'την εικόνα

Layers και JavaScript

Τώρα θα χειριστούμε τα layers μέσω JavaScript. Θέλουμε για αρχή να δημιουργήσουμε ένα κουμπί που όταν πατηθεί το layer εξαφανίζεται, και αν ξαναπατηθεί, εμφανίζεται.

Πρώτα απ'όλα πρέπει να δούμε πώς λέγονται τα layers στο JavaScript. Όπως συνήθως, υπάρχουν πολλοί τρόποι. Ο καλύτερος είναι να εισάγουμε όνομα στο layer. Αν καθορίσουμε ένα layer ως εξής

<layer ... name=myLayer>
...
</layer>

θα έχουμε πρόσβαση σε αυτό το layer γράφοντας *document.layers["myLayer"]*. Σύμφωνα με το έγγραφο της Netscape μπορούμε επίσης να γράψουμε *document.myLayer* - αλλά για κάποιο λόγο αυτό δεν δουλεύει στον browser μου. Αυτό όμως πρέπει να είναι πρόβλημα μόνο της preview έκδοσης και θα έχει λυθεί στην τελική έκδοση (χρησιμοποιώ Netscape Navigator 4.0 PR3 σε WinNT αυτή τη στιγμή). Μέχρι τώρα όμως δεν είχα προβλήματα με το *document.layers["myLayer"]* - έτσι θα χρησιμοποιήσουμε αυτόν τον τρόπο.

Μπορείτε επίσης να έχετε πρόσβαση στα layers μέσω ενός αριθμού που αντιπροσωπεύει την σειρά του layer στην κατάταξη όλων των layers. Για να έχετε πρόσβαση στο πρώτο layer, πρέπει να γράψετε *document.layers[0]*. Σημειώστε ότι αυτός ο αριθμός (index) δεν έχει σχέση με την ιδιότητα z-index. Αν έχετε για παράδειγμα δύο layers με ονόματα *layer1* και *layer2* με z-index αριθμούς 17 κει 100 έχετε πρόσβαση σε αυτά τα layers μέσω *document.layers[0]* και *document.layers[1]* και **ΟΧΙ** γράφοντας *document.layers[17]* και *document.layers[100]*.

Υπάρχουν πολλές ιδιότητες του κάθε layer οι οποίες μπορούν να αλλάξουνε χρησιμοποιώντας JavaScript. Το ακόλουθο παράδειγμα δείχνει ένα κουμπί που σε

αφήνει να εξαφανίσεις ή να εμφανίσεις το layer (απαιτείται Netscape Navigator 4.0 - ή ανώτερη έκδοση!).

This text is inside a layer

Ο κώδικας είναι ο εξής:

```
<html>
<head>
<script language="JavaScript">
<!-- hide

function showHide() {
  if (document.layers["myLayer"].visibility == "show")
    document.layers["myLayer"].visibility= "hide"
  else document.layers["myLayer"].visibility= "show";
}

// -->
</script>
</head>
<body>

<ilayer name=myLayer visibility=show>
<font size=+1 color="#0000ff"><i>This text is inside a layer</i></font>
</ilayer>

<form>
<input type="button" value="Show/Hide layer" onClick="showHide()">
</form>

</body>
</html>
```

Το κουμπί καλεί την function *showHide()*. Μπορείτε να δείτε ότι η function αυτή αποκτά πρόσβαση στην ιδιότητα *visibility* του layer *myLayer*. Με το να ρυθμίζει την τιμή σε *"show"* ή *"hide"* την ιδιότητα *document.layers["myLayer"].visibility* μπορείτε εσείς να διαλέξετε εάν θα εμφανίζεται ή όχι. Σημειώστε ότι το *"show"* και *"hide"* είναι τιμές κειμένου - όχι keywords, αυτό σημαίνει ότι **δεν μπορείτε να γράψετε** *document.layers["myLayer"].visibility= show.*

Έχω χρησιμοποιήσει το <ilayer> αντί για το <layer> επειδή ήθελα να πάρει θέση σύμφωνα με τη ροή του κειμένου (δηλ. λες και είναι ένα οποιοδήποτε άλλο αντικείμενο).

Μετακινώντας layers

Οι ιδιότητες *left* και *top* καθορίζουν τη θέση του layer. Μπορείτε να ρυθμίσετε νέες τιμές στις δύο αυτές ιδιότητες με σκοπό να αλλάξετε τη θέση του layer. Η ακόλουθη γραμμή κώδικα θέτει την θέση του οριζόντια στα 200 (σε pixel):

```
document.layers["myLayer2"].left= 200;
```

Τώρα θα φτιάξουμε ένα κινούμενο layer - αυτό δείχνει κάπως σαν scroller.

This text is inside a layer

Ο κώδικας:

```
<html>
<head>
<script language="JavaScript">
<!-- hide

function move() {
  if (pos < 0) direction= true;
  if (pos > 200) direction= false;

  if (direction) pos++
    else pos--;

  document.layers["myLayer2"].left= pos;
}

// -->
</script>
</head>
<body onLoad="setInterval('move()', 20)">

<ilayer name=myLayer2 left=0>
<font size=+1 color="#0000ff"><i>This text is inside a layer</i></font>
</ilayer>

</body>
</html>
```

Δημιουργούμε ένα layer με το όνομα *myLayer2*. Μπορείτε να δείτε ότι χρησιμοποιούμε το *onLoad* μέσα στο <body>. Αυτό το κάνουμε επειδή Θέλουμε να αρχίσουμε την κίνηση του layer μόλις φορτώσει η σελίδα. Χρησιμοποιούμε το *setInterval()* μέσα στον *onLoad* event-handler. Αυτή είναι νέα μέθοδος του JavaScript 1.2 (δηλαδή την έκδοση JavaScript η οποία βρίσκεται στο Netscape Navigator 4.0). Αυτή χρησιμοποιείται για να καλούμε μια function ξανά και ξανά ανά κάποιο χρονικό διάστημα. Χρησιμοποιήσαμε το *setTimeout()* για αυτό στο τελευταίο μάθημα. Το *setInterval()* δουλεύει σχεδόν ίδια - μόνο που πρέπει να το καλέσετε μόνο μια φορά. Χάρη στο *setInterval()* καλούμε την *move()* κάθε 20 milliseconds. Η function *move()* θέτει το layer σε μια συγκεκριμένη θέση. Εφόσον καλούμε την function αυτή ξανά και ξανά έχουμε σαν αποτέλεσμα το κυλιόμενο κείμενο. Το μόνο που πρέπει να κάνουμε στη function *move()* είναι να υπολογίσουμε τη θέση του layer και να ρυθμίσουμε αυτή τη τιμή στο *document.layers["myLayer2"].left= pos*.

Ο ακόλουθος κώδικας θα εκτελεστεί μόνο από browsers που καταλαβαίνουν JavaScript 1.2:

```
<script language="JavaScript1.2">
<!-- hide
document.write("Χρησιμοποιήστε JavaScript 1.2 browser.");
```

// -->
</script>
Αυτό είναι το ίδιο πρόβλημα που είχαμε στο αντικείμενο Image. Μπορούμε να ξαναγράψουμε τον κώδικα σε ένα παρόμοιο τρόπο. Θέτοντας την μεταβλητή *browserOK* λύνεται το πρόβλημα.

Το ακόλουθο παράδειγμα δείχνει ότι οι layers μπορούν να επικαλύψουν ο ένας τον άλλο:

Η Εντολή *document.lastModified*

Για να εμφανίσουμε την ημερομηνία τελευταίας τροποποίησης μιας ιστοσελίδας, γράφουμε τον εξής κώδικα της JavaScript μέσα στον HTML κώδικα :

```
<html>

  <body>

    <script type="text/javascript">

      document.write("Ημερομηνία τελευταίας τροποποίησης της σελίδας : ")

      document.write(document.lastModified)

    </script>

  </body>

</html>
```

Άνοιγμα Δευτερογενών Παραθύρων

Για να εμφανίσουμε ένα άλλο παράθυρο (δευτερογενές ή εστιασμένο) εκτός από το βασικό παράθυρο του φυλλομετρητή και να τοποθετήσουμε μέσα σ' αυτό τα περιεχόμενα ενός άλλου HTML εγγράφου, γράφουμε τον εξής κώδικα :

```
<html>

  <head>

    <script type="text/javascript">

      function WinOpen() {
```

```
            window.open("page01.html", "Window1", "toolbar=yes");

        }

    </script>

  </head>

</html>
```

Δημιουργήσαμε μια συνάρτηση (function) με όνομα ***WinOpen()*** η οποία περιέχει την εντολή ***window.open()***, με την οποία μπορούμε να εμφανίσουμε ένα δευτερογενές παράθυρο, και η οποία έχει τις εξής παραμέτρους (ορίσματα) :

- *page01.html*, που είναι το όνομα της ιστοσελίδας (εγγράφου) που θα εμφανισθεί μέσα στο δευτερογενές παράθυρο,
- *Window1*, που είναι το όνομα του νέου παραθύρου και
- *toolbar=yes*, δηλ. το νέο παράθυρο θα έχει γραμμή εργαλείων (toolbar).

Για να κληθεί τώρα η παραπάνω συνάρτηση και να εμφανισθεί το δευτερογενές παράθυρο, πρέπει μέσα στο ίδιο HTML έγγραφο και στο τμήμα <body> να γράψουμε τον εξής κώδικα για τη δημιουργία μιας φόρμας (form) :

```
<form>

    <input type="button" name="WindowButton"

        value="Κάντε κλικ εδώ για να εμφανισθεί ένα νέο παράθυρο"

        onclick="WinOpen()">

</form>
```

Δημιουργήσαμε μια φόρμα και τοποθετήσαμε μέσα της ένα πλήκτρο εντολής (button) με όνομα *WindowButton* και με μια ετικέτα (τίτλο) που προτρέπει τον χρήστη να κάνει κλικ. Η ιδιότητα ***onclick*** της φόρμας είναι ένας από τους *χειριστές συμβάντων (event handlers)* της JavaScript και εδώ καθορίζει ότι αν κάνουμε κλικ στο πλήκτρο εντολής θα κληθεί η συνάρτηση *WinOpen()* οπότε και θα εμφανισθεί το παράθυρο με όνομα *Window1* και μέσα του το έγγραφο (ιστοσελίδα) *page01.html*.

Αυτόματη προώθηση με επιβεβαίωση

Μπορείτε να αντιμετωπίσετε αυτό το παραπάνω πρόβλημα τοποθετώντας ένα πλαίσιο επιβεβαίωσης. Ο χρήστης θα δει το μήνυμα, αλλά θα έχει την επιλογή ανάμεσα στα κουμπιά OK και Cancel. Όταν ο χρήστης δει για πρώτη φορά το μήνυμα θα πατήσει OK και θα μεταφερθεί στη σελίδα σας. Όταν όμως γυρίσει πίσω με το κουμπί back μπορεί να πατήσει το Cancel και μετά πάλι το back. Αυτή τη φορά δεν θα πάει μπρος-πίσω. Να πως γίνεται αυτό :

Υπόδειξη :

```
<SCRIPT LANGUAGE = ΄΄JAVASCRIPT">
<!--
function redirect ( ) {
if ( confirm ( ΄΄Η σελίδα έχει μεταφερθεί. Παρακαλούμε σημειώστε την καινούργια
της θέση και ζητήστε από τον ιδιοκτήτη της σελίδας από την οποία ήρθατε να
αλλάξει τον σύνδεσμο. Πατήστε OK για να πάτε στη νέα σελίδα.")) {
location='ααα.htm'
}
}
<!--Τέλος-->
</SCRIPT>
</HEAD>
<BODY onload = "redirect ( )">
```

Αυτό είναι πιο περίπλοκο. Δημιουργήσαμε μια λειτουργία που καλέσαμε redirect, η οποία περιέχει μια πρόταση if. Η if πρόταση χρησιμοποιεί μια ενσωματωμένη λειτουργία που λέγεται confirm που μοιάζει πολύ με τη λειτουργία alert με μόνη διαφορά ότι δημιουργεί ένα πλαίσιο επιβεβαίωσης με τα κουμπιά OK και Cancel. Στο τέλος της πρότασης if βρίσκεται η εντολή location = 'ααα.htm'. Που σημαίνει "αν ο χρήστης πατήσει το OK, εμφάνισε το έγγραφο ααα.htm".

Προσθήκη Πλήκτρων Πλοήγησης

Στην JavaScript μπορούμε να δημιουργήσουμε πλήκτρα πλοήγησης (navigation buttons), τα οποία εκτελούν τις ίδιες λειτουργίες με τα γνωστά πλήκτρα *Back (Πίσω)* και *Εμπρός (Forward)* που υπάρχουν στη γραμμή εργαλείων ενός φυλλομετρητή.

```
<form>

    <input type="button" value="Πίσω δύο σελίδες" onclick="history.go(-2)">

    <input type="button" value="Προηγούμενη σελίδα" onclick="history.go(-1)">

    <input type="button" value="Επόμενη σελίδα" onclick="history.go(1)">

    <input type="button" value="Μπροστά δύο σελίδες" onclick="history.go(2)">

</form>
```

Ο χειριστής συμβάντων ***onclick()*** καλεί τη συνάρτηση ***history.go()*** με όρισμα τη σχετική σελίδα στην οποία θέλουμε να πάμε.

Δημιουργία Μηνυμάτων σε Πλαίσια Διαλόγου

Για να εμφανίσουμε μηνύματα σε πλαίσια διαλόγου σε μια ιστοσελίδα, χρησιμοποιούμε τη συνάρτηση ***alert()***, η οποία εμφανίζει το κείμενο που της περνάμε σαν όρισμα σ' ένα πλαίσιο (παράθυρο) διαλόγου. Για να κάνουμε αλλαγή γραμμής στο κείμενο αυτό, πρέπει να συμπεριλάβουμε τον ειδικό χαρακτήρα ***n.**

Το παρακάτω παράδειγμα εμφανίζει ένα μήνυμα μέσα σε μια ιστοσελίδα :

```
<script type="text/javascript">

    alert("Καλώς ήρθατε στη σελίδα της πτυχιακής μου!!")

</script>
```

Μπορούμε να χρησιμοποιήσουμε και τους χειριστές συμβάντων ***onload()*** και ***onunload()***, οι οποίοι καλούνται όταν έχει ολοκληρωθεί η φόρτωση μιας ιστοσελίδας ή όταν έχουμε φύγει από μια ιστοσελίδα, αντίστοιχα. Οι χειριστές αυτοί τοποθετούνται στο tag <body> ενός εγγράφου.

```
<body onload = alert("Καλώς ήρθατε στη σελίδα της πτυχιακής μου!!")>

<body onunload = alert("Καλώς ήρθατε στη σελίδα της πτυχιακής μου!!")>
```

Μηνύματα στη Μπάρα Κατάστασης

Για να εμφανίσουμε ένα μήνυμα στη γραμμή κατάστασης (status bar) του παραθύρου του φυλλομετρητή, καταχωρούμε μια τιμή στη μεταβλητή *status* του αντικειμένου *window*, ως εξής :

```
<a href="teiep.html" onmouseover="window.status='Αν κάνετε κλικ θα πάτε στη σελίδα του ΤΕΙ ΗΠΕΙΡΟΥ';return true"> Άρτα </a>
```

Ο χειριστής συμβάντων ***onmouseover()*** εκτελεί μια ενέργεια όταν ο δείκτης του ποντικιού βρεθεί πάνω από τον σύνδεσμο και στη συγκεκριμένη περίπτωση καταχωρεί μια τιμή στη μεταβλητή *window.status*, που έχει σαν αποτέλεσμα να εμφανισθεί ένα μήνυμα στη γραμμή κατάστασης.

Προσθήκη Μηνυμάτων σε Φόρμες

Μπορούμε να προσθέσουμε μηνύματα στις φόρμες μιας HTML σελίδας τα οποία θα εμφανίζονται όταν κάνουμε κλικ σ' ένα στοιχείο της φόρμας ή όταν εστιάζουμε σ' ένα άλλο στοιχείο της φόρμας. Για να δημιουργήσουμε ένα τέτοιο μήνυμα, γράφουμε πρώτα το παρακάτω script στο τμήμα head :

```
<script type="text/javascript">
```

```
function AlertBox() {

    alert("Καλημέρα από την Άρτα")

}

</script>
```

Στο τμήμα head δηλώνουμε τις συναρτήσεις που θα χρησιμοποιήσουμε και μετά στον κώδικα για τη δημιουργία της φόρμας, γράφουμε τα εξής :

```
<form>

    <input type="checkbox" name="check1" onclick="AlertBox()">

    Υπογράψτε

</form>
```

Όταν κάνουμε κλικ στο πλαίσιο ελέγχου, καλείται η συνάρτηση *AlertBox()* η οποία με τη σειρά της καλεί την ενσωματωμένη συνάρτηση *alert()* και εμφανίζεται το μήνυμα.

Αυτόματη Προώθηση σε Άλλη Σελίδα

Για να προωθήσουμε αυτόματα τους επισκέπτες μιας ιστοσελίδας σε μια άλλη ιστοσελίδα, γράφουμε τα εξής :

```
<head>

  <script type="text/javascript">

    alert("Η διεύθυνση της σελίδας έχει αλλάξει")

  </script>

</head>

<body onload = "location= 'new.html' ">
```

Μόλις ο φυλλομετρητής αρχίζει να διαβάζει το HTML έγγραφο, εμφανίζεται το πλαίσιο διαλόγου alert με το προειδοποιητικό μήνυμα και μόλις κάνουμε κλικ στο OK για να το κλείσουμε, φορτώνεται η ιστοσελίδα, εκτελείται ο χειριστής συμβάντων *onload* και φορτώνεται η ιστοσελίδα new.html.

Η Συνάρτηση *Confirm()*

Η συνάρτηση ***confirm()*** εμφανίζει ένα πλαίσιο επιβεβαίωσης σ' ένα πλαίσιο διαλόγου με τα πλήκτρα OK και Cancel, όπου αν κάνουμε κλικ στο OK θα επιστραφεί η τιμή true, ενώ αν κάνουμε κλικ στο Cancel θα επιστραφεί η τιμή false.

Ακολουθεί ένα παράδειγμα :

```
<head>
   <script type="text/javascript">
   function redirect() {
      if (confirm("Η διεύθυνση της σελίδας έχει αλλάξει")) {
            location = 'new.html'
         }
   }
   </script>
</head>
<body onload = "redirect()">
```

Αφού ολοκληρωθεί το φόρτωμα της σελίδας θα κληθεί η συνάρτηση *redirect()* και θα εμφανισθεί ένα πλαίσιο διαλόγου επιβεβαίωσης με το παραπάνω επεξηγηματικό κείμενο και τα πλήκτρα OK και Cancel. Αν κάνουμε κλικ στο πλήκτρο OK, θα εκτελεσθεί η εντολή *if* και θα πάμε στην καινούργια ιστοσελίδα, σύμφωνα με την τιμή της μεταβλητής *location.*

Τα Αρχεία Σεναρίων

Μπορούμε να γράψουμε σενάρια (scripts) της JavaScript σ' ένα ξεχωριστό αρχείο κειμένου, διαφορετικό από το HTML έγγραφο, και μετά να τα συνδέσουμε χρησιμοποιώντας την ιδιότητα ***SRC*** της ετικέτας (tag) <script>. Όταν ο φυλλομετρητής διαβάσει τη σελίδα, θα δει την ιδιότητα SRC και θα διαβάσει το αρχείο σεναρίου σαν αυτό να βρισκόταν μέσα στο HTML έγγραφο.

Αυτή η ιδιότητα είναι ιδιαίτερα χρήσιμη όταν δημιουργούμε πολύπλοκα σενάρια και τα χρησιμοποιούμε σε πολλές ιστοσελίδες. Για παράδειγμα, μπορούμε να δημιουργήσουμε ένα απλό αρχείο κειμένου με το εξής κείμενο :

```
<!--
```

document.write("Ημερομηνία τελευταίας τροποποίησης της σελίδας : ")

document.write(document.lastModified)

//-->

Αποθηκεύουμε το αρχείο αυτό μ' ένα όνομα με επέκταση ***.js***, π.χ. *doc01.js*, και μέσα στον κώδικα HTML της ιστοσελίδας γράφουμε τα εξής :

<script type="text/javascript" SRC="doc01.js">

</script>

Τα Σχόλια στην JavaScript

Για να εισάγουμε ***σχόλια (comments)*** στην JavaScript, μπορούμε να γράψουμε τα εξής :

// αυτό είναι ένα σχόλιο

ή

/ αυτή είναι η αρχή ενός μεγαλύτερου σχολίου,*

αυτή είναι η δεύτερη γραμμή του

*και αυτή είναι η τελευταία γραμμή του */*

Για να βάλουμε ένα απλό σχόλιο σ' ένα script, χρησιμοποιούμε τους χαρακτήρες //, οπότε ο φυλλομετρητής αγνοεί ο,τιδήποτε βρίσκεται μετά και πηγαίνει στην επόμενη γραμμή. Τους χαρακτήρες // μπορούμε να τους τοποθετήσουμε και στη μέση μιας γραμμής εντολής ενός script.

Αν θέλουμε να γράψουμε ένα μεγαλύτερο σχόλιο που να επεκτείνεται σε πολλές γραμμές, μπορούμε να ξεκινήσουμε την πρώτη γραμμή με τους χαρακτήρες /* και να κλείσουμε το σχόλιο τελειώνοντας την τελευταία γραμμή με τους χαρακτήρες */.

Η Συνάρτηση *Prompt()*

Η συνάρτηση ***prompt()*** εμφανίζει ένα μήνυμα σ' ένα πλαίσιο διαλόγου στην οθόνη και ό,τι γράψουμε το εκχωρεί σε μια μεταβλητή. Ακολουθεί ένα παράδειγμα :

function function1() {

```
   var sWord=" "

   while (sWord != "Arta") {

        sWord = prompt("Γράψτε Arta για έξοδο", " ")

   }

   alert("Ο βρόχος ολοκληρώθηκε")

}
```

Οι Συναρτήσεις (Functions)

Ο ορισμός μιας συνάρτησης (function) είναι μια διαδικασία κατά την οποία δηλώνουμε το όνομά της και εξηγούμε τι κάνει. Η σύνταξη μιας συνάρτησης είναι η εξής :

```
   function ΌνομαΣυνάρτησης() {

        ... εντολές ...

   }
```

Μπορούμε να ορίσουμε συναρτήσεις οπουδήποτε σε μια ιστοσελίδα, αλλά πρέπει να προσέχουμε να τις ορίζουμε πριν τις καλέσουμε και είναι καλύτερα να τις ορίζουμε στο τμήμα <head> ενός εγγράφου. Οι συναρτήσεις επιστρέφουν τιμές που μπορούμε να εκχωρήσουμε σε μεταβλητές, αν η κλήση τους γίνει ως εξής :

```
   μεταβλητή = ΌνομαΣυνάρτησης()
```

Ακολουθεί ένα παράδειγμα :

```
function function1() {

   var sTypedText

   sTypedText = prompt("Γράψτε κάτι και πατήστε OK", " ")

   return sTypedText

}
```

Έχουμε ορίσει μια συνάρτηση με όνομα *function1()*, η οποία περιέχει μια τοπική μεταβλητή στοιχειοσειράς με όνομα *sTypedText* και χρησιμοποιούμε την ενσωματωμένη συνάρτηση *prompt()* για να γράψουμε ένα κείμενο σ' ένα πλαίσιο

κειμένου, το οποίο κείμενο καταχωρείται στη μεταβλητή *sTypedText* και επιστρέφεται εκεί απ' όπου κλήθηκε η συνάρτηση με την εντολή *return*.

Στην ίδια ιστοσελίδα, μπορούμε να γράψουμε τα εξής :

```
var sText

sText = function1()

document.write("Γράψατε : <p><h3> " + sText + "</h3></p>")
```

Μπορούμε να διαβιβάσουμε τιμές σε μια συνάρτηση, διαχωρίζοντάς τες με κόμμα, όπως στο παρακάτω παράδειγμα :

```
function Percentage(nNum, nPerc) {

    return nNum * (nPerc/100)

}
```

Για να καλέσουμε αυτή τη συνάρτηση, μπορούμε να γράψουμε ένα σενάριο σαν το εξής :

```
var nValue1 = prompt("Γράψτε έναν αριθμό", " ")

var nValue2 = prompt("Γράψτε το ποσοστό %", " ")

var nPercent = Percentage(nValue1, nValue2)

document.write("<h3>" + nValue2 + " % " + nValue1 + " = " + nPercent + "<h3>")
```

Οι Πίνακες (Arrays)

Για να δημιουργήσουμε έναν πίνακα στην JavaScript, απαιτούνται τα εξής τρία στάδια :

- Δημιουργία μιας νέας τάξης αντικειμένων με μια ειδική συνάρτηση.
- Κλήση της συνάρτησης.
- Εισαγωγή των δεδομένων στον πίνακα.

Για να δημιουργήσουμε μια τάξη αντικειμένων που να περιγράφει έναν πίνακα πρέπει να ορίσουμε έναν ειδικό τύπο συνάρτησης, ως εξής :

```
function makeArray(n) {
```

```
    this.length=n

    for (var i=1; i<=n; i++) {

        this[i]=null;

    }

    return this

}
```

Για να δημιουργήσουμε τώρα έναν πίνακα με όνομα area και με 10 θέσεις, καλούμε την προηγούμενη συνάρτηση, ως εξής :

```
var area=new makeArray(10)
```

και καταχωρούμε τιμές στα στοιχεία του πίνακα, ως εξής :

```
area[1]="Γιάννενα"
area[2]="Καστοριά"

...

area[10]="Άρτα"
```

Στις νεότερες εκδόσεις της JavaScript μπορούμε να δημιουργήσουμε έναν πίνακα με την εξής απλή εντολή :

```
var area=new Array()
```

Η αρίθμηση του παραπάνω πίνακα ξεκινά από το 0 και όχι από το 1.

Αντικείμενα, Ιδιότητες και Μέθοδοι

Ένα αντικείμενο (object) διαθέτει ένα σύνολο από ιδιότητες (properties) και μεθόδους (methods) που συνδέονται μ' αυτό. Η JavaScript περιέχει έναν αριθμό αντικειμένων, όπως είναι το αντικείμενο *document*, που είναι η ηλεκτρονική σελίδα που δημιουργεί η HTML, το αντικείμενο *window* και το αντικείμενο *form.*

Οι ***ιδιότητες*** ενός αντικειμένου είναι μεταβλητές που ανήκουν στο αντικείμενο και καθορίζονται με τη χρήση μιας τελείας (dot notation), ως εξής :

```
όνομαΑντικειμένου.όνομαΙδιότητας
```

Μια ***μέθοδος*** είναι απλά μια συνάρτηση που συνδέεται μ' ένα συγκεκριμένο αντικείμενο και μπορεί να επιστρέψει μια τιμή. Ο γενικός τρόπος χρήσης μιας μεθόδου είναι ο εξής :

όνομαΑντικειμένου.όνομαΜεθόδου(παράμετροι)

Ακολουθούν παραδείγματα :

document.write(navigator.appName)

Η ιδιότητα *appName* ανήκει στο αντικείμενο *navigator* και περιέχει το όνομα αναγνώρισης του φυλλομετρητή. Το αντικείμενο *navigator* περιγράφει τον φυλλομετρητή που διαβάζει την ιστοσελίδα που περιέχει το σενάριο. Ακόμη, στο παραπάνω παράδειγμα, η μέθοδος *write()* συνδέεται με το αντικείμενο εγγράφου *document* και εμφανίζει κάποιο κείμενο στην ιστοσελίδα.

document.bgColor="red"

Το παραπάνω παράδειγμα ορίζει την ιδιότητα του χρώματος φόντου της σελίδας σε κόκκινο (red).

var sText

var sUpper

sText = prompt("Γράψτε κάτι με μικρά γράμματα", " ")

sUpper = sText.toUpperCase()

document.write("Το κείμενο με κεφαλαία γράμματα είναι : <p><h3>" + sUpper + "</h3>")

Εδώ δηλώσαμε τις μεταβλητές sText και sUpper και με τη μέθοδο prompt δώσαμε τιμή στη μεταβλητή sText. Η prompt είναι μια μέθοδος του αντικειμένου window, όπως και οι μέθοδοι alert και confirm, αλλά δεν είναι ανάγκη να γράψουμε ολόκληρα τα window.prompt, window.alert και window.confirm. Μετά, με τη μέθοδο toUpperCase() των αντικειμένων στοιχείο σειράς, μετατρέπουμε σε κεφαλαία γράμματα το κείμενο της μεταβλητής sText και το καταχωρούμε στη μεταβλητή sUpper.

document.Customer.Lastname.value = "Arta"

Η παραπάνω εντολή καταχωρεί το κείμενο Arta στην ιδιότητα value του πλαισίου κειμένου Lastname της φόρμας Customer του τρέχοντος εγγράφου.

Το Αντικείμενο Date

Το αντικείμενο Date μάς δίνει τη δυνατότητα να δουλεύουμε με ημερομηνίες και ώρες. Για να γίνει αυτό, πρέπει να δημιουργήσουμε ένα αντίγραφο αντικειμένου με τη λέξη κλειδί ***new***, ως εξής :

Today = new Date()

Όταν δημιουργούμε ένα αντικείμενο Date, μπορούμε να πάρουμε την τρέχουσα ημερομηνία και ώρα του συστήματος, όπως κάναμε πριν, ή να ορίσουμε μια δική μας ημερομηνία και ώρα, μ' έναν από τους εξής τρόπους :

Date1 = new Date("8, 30, 2001, 10:50:14")

Date2 = new Date(2001, 8, 30)

Date3 = new Date(2001, 8, 30, 10, 50, 14)

Για το αντικείμενο Date υπάρχουν οι εξής μέθοδοι :

- *getDate()*, επιστρέφει την ημερομηνία.
- *getDay()*, επιστρέφει την ημέρα της εβδομάδας.
- *getHours()*, επιστρέφει την ώρα.
- *getMinutes()*, επιστρέφει τα λεπτά.
- *getMonth()*, επιστρέφει τον μήνα.
- *getSeconds()*, επιστρέφει τα δευτερόλεπτα.
- *getTime()*, επιστρέφει την πλήρη ώρα.
- *getTimeZoneoffset()*, επιστρέφει τη διαφορά από την ώρα Greenwich.
- *getYear()*, επιστρέφει το έτος,
- *parse()*, επιστρέφει τα χιλιοστά του δευτερολέπτου από 1/1/1970.
- *setDate()*, ορίζει την ημερομηνία.
- *setHours()*, ορίζει την ώρα.
- *setMinutes()*, ορίζει τα λεπτά.
- *setMonth()*, ορίζει τον μήνα.
- *setSeconds()*, ορίζει τα δευτερόλεπτα.
- *setTime()*, ορίζει την πλήρη ώρα.
- *setYear()*, ορίζει το έτος.
- *toGMTString()*, μετατρέπει την ημερομηνία σε στοιχειοσειρά ώρας Greenwich.
- *toLocaleString()*, μετατρέπει την ημερομηνία σε στοιχειοσειρά.
- *UTC()*, επιστρέφει τα χιλιοστά του δευτερολέπτου από 1/1/1970.

Τρόποι για να εκτελέσετε σενάρια JavaScript

Υπάρχουν δύο περιπτώσεις στις οποίες ο browser σας εκτελεί τις εντολές που βρίσκει σε ένα σενάριο JavaScript:
Κάποια τμήματα του σεναρίου εκτελούνται με το φόρτωμα της ιστοσελίδας από τον Browser.
Κάποια τμήματα του σεναρίου εκτελούνται επειδή ο χρήστης πυροδότησε κάποια ενέργεια.
Υπάρχουν δύο πράγματα που ίσως θέλετε να να κάνετε όταν ο browser φορτώνει μια σελίδα. Πρώτον, μπορεί να θέλετε να κάνετε κάτι που ο χρήστης θα δει αμέσως, πχ. ένα πλαίσιο διαλόγου με ένα καλωσόρισμα προς το χρήστη . Δεύτερον, μπορεί να θέλετε να κάνετε κάτι που δεν θα είναι ορατό από το χρήστη, κάτι που θα ετοιμάζει ουσιαστικά το έδαφος για κάτι που άλλο που θα γίνει αργότερα. Μπορείτε π.χ να προσδιορίσετε λειτουργίες που πρόκειται να χρησιμοποιηθούν αργότερα.

Όπως έχουμε δύο περιπτώσεις εκτέλεσης σεναρίων, έτσι έχουμε και δύο τρόπους για να γράψουμε σενάρια. Μπορούμε να τα βάλουμε ανάμεσα σε ετικέτες <SCRIPT> ή </SCRIPT>, ή να τα τοποθετήσουμε μέσα σε ετικέτες HTML .

Απόκρυψη σεναρίων

Υπάρχει όμως ένα μικρό πρόβλημα με το σενάριο που γράψατε προηγουμένως. Συγκεκριμένοι browser που δεν υποστηρίζουν JavaScript, την επεξεργάζονται σαν απλό κείμενο και την παρουσιάζουν λίγο παράδοξα. Η διόρθωση γίνεται ως εξής :

Υπόδειξη :

```
<SCRIPT LANGUAGE = ΄΄JAVASCRIPT">
<!--
document.write(΄΄Το έγγραφο τροποποιήθηκε για τελευταία φορά :")
document.write(document.lastModified)
//-->
</SCRIPT>
```

Όπως παρατηρήσατε, προσθέσαμε δύο γραμμές, <!-- και //--> . Αυτές είναι οι ετικέτες σχολίου που χρησιμοποιούνται στην HTML. Οι browsers αγνοούν οτιδήποτε βρίσκεται μεταξύ των γραμμών αυτών.

Apache http WEB SERVER

Τι είναι ο Apache

Το έργο Apache είναι ένα λογισμικό το οποίο αναπτύχθηκε για να δημιουργήσει έναν εμπορικό ανοιχτού κώδικα web server .

Το έργο αυτό το διαχειρίζεται μια ομάδα εθελοντών από όλο τον κόσμο, γνώση με το όνομα **The Apache Group**.

Πως προέκυψε

Έως το 1995 ο πιο δημοφιλής web server ήταν ο http Daemon(Δαίμονας) ο οποίος είχε αναπτυχθεί από τον Rob Mc Cool στο Εθνικό Κέντρο Υπερυπολογιστικών Εφαρμογών (NCSA) του πανεπιστημίου του Illinois

Ωστόσο το έργο αυτό εγκαταλείφθηκε μόλις έφυγε ο Mc Cool.

Όμως αρκετοί Web Masters και προγραμματιστές είχαν αναπτύξει δικές τους επεκτάσεις για τον http Daemon οι οποίες ήταν ανοικτού κώδικα και περίμεναν να τις δουν στην πράξη.

Μερικοί από αυτούς συγκεντρώθηκαν για να συντονίσουν τις αλλαγές που είχαν κάνει μα τη μορφή «paches».

Χρησιμοποιώντας τον http Daemon του NCSA σαν βάση και προσθέτοντας όλα αυτά τα paches προέκυψε το έργο
A(που συμβολίζει το στερητικό Α- δηλαδή δίχως) **"pache"- APACHE** web server.

Γιατί είναι free

Το έργο Apache δημιουργήθηκε για να παράσχει μια ισχυρή, εμπορικής κυρίως φύσης υλοποίηση του HTTP πρωτοκόλλου.

Πρέπει να παραμείνει μια πλατφόρμα πάνω στην οποία ανεξάρτητοι webmasters αλλά και ιδρύματα θα δημιουργήσουν αξιόπιστα συστήματα τόσο για εμπορικούς όσο και για πιλοτικούς σκοπούς .

Οι άνθρωποι που το ανέπτυξαν πιστεύουν πως η δύναμη των on-line εφαρμογών θα παραμείνει πραγματική όσο υπάρχουν ελευθέρα εργαλεία πάνω στα οποία οι επιχ/σεις θα αναπτύσσουν τις ιδέες τους.

Όσο το web θα παραμένει "άκτιτο" – εφοδιαζόμενο με ελευθέρα εργαλεία , θα υπάρχουν ευκαιρίες τόσο για τις μεγάλες όσο και για τις μικρές επιχ/σεις να δείξουν την άξια τους επενδύοντας στο Διαδίκτυο.

Τέλος το Apache είναι free γιατί είναι μια οντότητα που αναπτύχθηκε χάρη στην αστείρευτη ανάγκη που ένιωθε μια ομάδα ανθρώπων για πρόσφορα στην κοινωνία.

Η Βάση Δεδομένων MySQL

Τι Είναι οι Βάσεις Δεδομένων (Databases)

Μια ***βάση δεδομένων*** *(database)* αποτελείται από έναν ή περισσότερους ***πίνακες*** *(tables)*, ο καθένας από τους οποίους περιέχει μια λίστα από κάποια πράγματα. Για μια βάση δεδομένων πελατών (clients), είναι φυσικό να ξεκινήσουμε μ' έναν πίνακα με όνομα *clients* που θα περιέχει μια λίστα από στοιχεία πελατών.

Ο κάθε πίνακας σε μια βάση δεδομένων περιέχει μια ή περισσότερες ***στήλες*** *(columns)* ή ***πεδία*** *(fields)*, όπου η κάθε στήλη περιέχει μια συγκεκριμένη πληροφορία για τον κάθε πελάτη που υπάρχει στην βάση δεδομένων (database).

Ο πίνακας clients μπορεί να περιέχει στήλες για τον κωδικό ενός πελάτη (ID), για το όνομά του (Name) καθώς και για την ημερομηνία γέννησής του (Date). Το κάθε ανέκδοτο που αποθηκεύουμε σ' αυτόν τον πίνακα λέμε ότι αποτελεί μια ***γραμμή*** *(row)* ή μια ***εγγραφή*** *(record)* του πίνακα. Για παράδειγμα, ας δούμε τον παρακάτω πίνακα :

ID	**Name**	**Date**
1	Αντωνιάδης	1970-04-01
2	Παπαδόπουλος	1968-02-22

Εκτός από τις στήλες για το όνομα του πελάτη *(Name)* και την ημερομηνία γέννησής του *(Date)*, υπάρχει και μια στήλη με όνομα *ID*, ο σκοπός της οποίας είναι να εκχωρήσει έναν μοναδικό αριθμό στον κάθε πελάτη έτσι ώστε να έχουμε έναν εύκολο τρόπο αναφοράς σ' αυτόν και να μπορούμε να τον ξεχωρίσουμε από τους άλλους πελάτες.

Σαν επισκόπηση, το παραπάνω είναι ένας πίνακας τριών στηλών που περιέχει δύο γραμμές ή καταχωρήσεις. Η κάθε γραμμή του πίνακα περιέχει έναν κωδικό (ID) αναγνώρισης του πελάτη, το όνομά του (text) καθώς και την ημερομηνία γέννησής του (date). Με βάση αυτήν την βασική ορολογία, είμαστε έτοιμοι να αρχίσουμε να χρησιμοποιούμε την MySQL.

Εκκίνηση (Logging onto) της MySQL

Το standard interface για να δουλέψουμε με τις βάσεις δεδομένων της MySQL είναι να συνδεθούμε με το λογισμικό του MySQL server και να δίνουμε μία εντολή την φορά. Για να κάνουμε αυτήν την σύνδεση με τον server, θα χρειασθούμε το *πρόγραμμα πελάτη (client program)* της MySQL.

Στο Linux, το πρόγραμμα αποκαλείται ***mysql*** και βρίσκεται εξ ορισμού στον κατάλογο */usr/local/mysql/bin*, ενώ στα Windows, το πρόγραμμα αποκαλείται ***mysql.exe*** και βρίσκεται εξ ορισμού στον κατάλογο *C:\mysql\bin*.

Υπάρχουν δύο τρόποι για να μπορέσουμε να συνδεθούμε με τον MySQL server. Ο πρώτος είναι να χρησιμοποιήσουμε το *telnet* για να συνδεθούμε (log into) στον server του Web host που μας φιλοξενεί και να δώσουμε την εντολή *mysql* από εκεί.
Ο δεύτερος είναι να φορτώσουμε (download) και να εγκαταστήσουμε το λογισμικό πελάτη (client software) της MySQL από το site *http://www.mysql.com/* στον δικό μας υπολογιστή και να το χρησιμοποιήσουμε για να συνδεθούμε με τον MySQL server.

Όποια μέθοδο κι αν επιλέξουμε και όποιο λειτουργικό σύστημα χρησιμοποιούμε, θα καταλήξουμε σε μια *γραμμή εντολών (command line)*, έτοιμοι να εκτελέσουμε το πρόγραμμα πελάτη της MySQL για να συνδεθούμε στον MySQL server. Πρέπει να γράψουμε τα εξής :

mysql -h <hostname> -u <username> -p

Θα πρέπει να αντικαταστήσουμε το <*hostname*> με το όνομα του host ή την IP διεύθυνση του υπολογιστή στον οποίο εκτελείται ο MySQL server. Αν εκτελούμε το πρόγραμμα πελάτη στον ίδιο υπολογιστή με τον server, μπορούμε να παραλείψουμε το τμήμα *-h <hostname>* της εντολής αντί να γράψουμε *-h localhost*, για παράδειγμα.
Το <*username*> πρέπει να είναι το δικό μας όνομα χρήστη στην MySQL.
Αν εγκαταστήσαμε εμείς οι ίδιοι τον MySQL server, αυτό θα είναι το *root*, ενώ αν χρησιμοποιούμε τον MySQL server του Web host που μας φιλοξενεί, αυτό θα πρέπει να είναι το όνομα χρήστη της MySQL που μας έχει δοθεί.

Το όρισμα *-p* λέει στο πρόγραμμα να ζητήσει από μας τον κωδικό εισόδου (password), το οποίο θα συμβεί μόλις δώσουμε την παραπάνω εντολή. Αν έχουμε εγκαταστήσει εμείς οι ίδιοι τον MySQL, αυτό το password θα είναι το root password που επιλέξαμε εμείς, ενώ αν χρησιμοποιούμε τον MySQL server του Web host που μας φιλοξενεί, αυτό θα πρέπει να είναι το password της MySQL που μας έχει δοθεί.
Αν τα γράψαμε όλα σωστά, το πρόγραμμα πελάτη της MySQL θα παρουσιάσει τον εαυτό του και θα εμφανίσει την γραμμή εντολών της MySQL, ως εξής :
mysql>

Τώρα, ο MySQL server είναι σε θέση να παρακολουθεί περισσότερες από μία βάσεις δεδομένων, που αυτό σημαίνει ότι Web host μπορεί να στήσει έναν μόνο MySQL server για να χρησιμοποιηθεί από πολλούς από τους συνδρομητές του.
Έτσι, το επόμενο βήμα μας θα πρέπει να είναι να επιλέξουμε την βάση δεδομένων με την οποία θα δουλέψουμε. Πρώτα απ' όλα, θα δούμε μια λίστα των βάσεων δεδομένων που υπάρχουν στον τρέχοντα server. Δίνουμε την επόμενη εντολή και μετά ENTER.

mysql> SHOW DATABASES;

Η MySQL θα εμφανίσει μια λίστα με τις βάσεις δεδομένων που υπάρχουν στον server, ως εξής :

```
Database
mysql
    test
2 rows in set (0.11 sec)
```

Ο MySQL server χρησιμοποιεί την πρώτη βάση δεδομένων, με όνομα *mysql*, για να μπορεί να παρακολουθεί τους χρήστες, τα συνθηματικά τους (passwords) καθώς και το τι επιτρέπεται να κάνουν. Θα αφήσουμε για λίγο αυτή την βάση δεδομένων.

Η δεύτερη βάση δεδομένων, με όνομα *test* αποτελεί ένα δείγμα βάσης δεδομένων. Η διαδικασία της διαγραφής στην MySQL αποκαλείται ***dropping*** *(απόρριψη)* και η εντολή για να διαγράψουμε μια βάση δεδομένων είναι η εξής :

mysql> DROP DATABASE test;

Αν δώσουμε αυτήν την εντολή και πατήσουμε Enter, η MySQL θα διαγράψει την βάση δεδομένων και θα εμφανίσει το μήνυμα *Query OK* σαν επιβεβαίωση. Επειδή αυτή η εντολή δεν εμφανίζει κάποιο μήνυμα προειδοποίησης, πρέπει να είμαστε πολύ προσεκτικοί όταν την δίνουμε.

Θα δούμε τώρα λίγα πράγματα για την *γραμμή εντολών (command line)* της MySQL. Όλες οι εντολές στην MySQL τελειώνουν με τον χαρακτήρα ; (semicolon). Έτσι, αν έχουμε ξεχάσει να κλείσουμε μια εντολή με τον χαρακτήρα ;, η MySQL θα νομίζει ότι δεν έχουμε τελειώσει με την εντολή αυτή και θα περιμένει να συνεχίσουμε να γράφουμε και στην επόμενη γραμμή :

mysql> SHOW
-> DATABASES;

Η MySQL δείχνει ότι περιμένει από μας να ολοκληρώσουμε την εντολή, αλλάζοντας την προτροπή (prompt) από *mysql>* σε *->*. Αυτό είναι βολικό όταν έχουμε να γράψουμε μακροσκελείς εντολές, καθώς μπορούμε να επεκτείνουμε τις εντολές μας σε πολλές γραμμές.

Για να ακυρώσουμε την τρέχουσα εντολή και να αρχίσουμε να την γράφουμε από την αρχή, γράφουμε τους χαρακτήρες ***c** και πατάμε ENTER, ως εξής :

mysql> DROP DATABASE\c
mysql>

Η MySQL θα αγνοήσει την εντολή που είχαμε ξεκινήσει και θα περιμένει να δώσουμε μια άλλη εντολή.
Τέλος, αν θέλουμε να εξέλθουμε από το πρόγραμμα πελάτη της MySQL, μπορούμε απλά να γράψουμε ***quit*** ή ***exit***. Είναι οι μόνες εντολές που δεν χρειάζονται τον χαρακτήρα ; (semicolon).
mysql> quit
Bye

Τι Είναι η SQL

Το σύνολο των εντολών που θα χρησιμοποιούμε από δω και πέρα για να λέμε στην MySQL τι να κάνει, αποτελεί μέρος ενός standard που αποκαλείται *Δομημένη Γλώσσα Ερωτημάτων (Structured Query Language)* ή ***SQL***. Οι εντολές της SQL αποκαλούνται επίσης και *ερωτήματα (queries)*.

Η SQL αποτελεί την standard γλώσσα για αλληλεπίδραση με τις περισσότερες βάσεις δεδομένων, έτσι ακόμα κι αν αλλάξουμε στο μέλλον από την MySQL σε μια βάση δεδομένων όπως την *Microsoft SQL Server*, θα διαπιστώσουμε ότι οι περισσότερες από τις εντολές είναι ολόιδιες.

Δεν πρέπει να συγχέουμε την SQL με την MySQL. Η MySQL είναι το λογισμικό του *διακομιστή βάσεων δεδομένων (database server software)* που χρησιμοποιούμε, ενώ η SQL είναι η γλώσσα που χρησιμοποιούμε για να αλληλεπιδράσουμε με την βάση δεδομένων.

Δημιουργία μιας Βάσης Δεδομένων (DataBase)

Η δημιουργία μιας βάσης δεδομένων είναι πολύ εύκολη υπόθεση :
mysql> CREATE DATABASE clients;
Τώρα που έχουμε μια βάση δεδομένων, πρέπει να ενημερώσουμε την MySQL ότι θέλουμε να την χρησιμοποιήσουμε. Η εντολή αυτή είναι η εξής :
mysql> USE clients;
Μπορούμε τώρα να αρχίσουμε να χρησιμοποιούμε την βάση δεδομένων.

Δημιουργία ενός Πίνακα (Table)

Η σύνταξη της εντολής SQL για την δημιουργία ενός πίνακα (table) είναι η εξής :
mysql> ***CREATE TABLE*** *<table name> (*
-> <column 1 name> <col. 1 type> <col. 1 details>,
-> <column 2 name> <col. 2 type> <col. 2 details>,
-> ...
->);
Όσον αφορά το παράδειγμα που είχαμε δει νωρίτερα με τον πίνακα Jokes, είχε τις εξής τρεις στήλες (columns) : ***ID*** (αριθμός, number), ***Name*** (το όνομα του πελάτη) και ***Date*** (την ημερομηνία γέννησης του πελάτη).

Η εντολή για να δημιουργήσουμε αυτόν τον πίνακα είναι η εξής :

mysql> CREATE TABLE Clients (
-> ID INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
-> Name TEXT,
-> Date DATE NOT NULL
->);

Ας το αναλύσουμε τώρα :
Η πρώτη γραμμή είναι αρκετά απλή : λέει ότι θέλουμε να δημιουργήσουμε έναν νέο πίνακα με όνομα *Clients*.

Η δεύτερη γραμμή λέει ότι θέλουμε μια στήλη (column) με όνομα *ID* που θα περιέχει μια ακέραια τιμή (integer, *INT*). Ακόμη, αυτή η στήλη δεν μπορεί να είναι κενή *(NOT NULL)*. Επίσης, αν δεν καθορίσουμε κάποια συγκεκριμένη τιμή, όταν κάνουμε μια νέα καταχώρηση, η MySQL θα επιλέξει η ίδια μια τιμή που θα είναι κατά ένα μεγαλύτερη από την μεγαλύτερη τιμή του πίνακα μέχρι τώρα *(AUTO_INCREMENT)*. Τέλος, αυτή η στήλη θα ενεργεί σαν ένα μοναδικό

αναγνωριστικό (unique identifier) για τις καταχωρήσεις του πίνακα, έτσι όλες οι τιμές αυτής της στήλης θα πρέπει να είναι μοναδικές *(PRIMARY KEY)*.

Η τρίτη γραμμή είναι πολύ απλή : λέει ότι θέλουμε μια στήλη με όνομα *Name* που θα περιέχει κείμενο (TEXT).

Η τέταρτη γραμμή ορίζει την τελευταία στήλη, με όνομα *Date*, η οποία θα περιέχει δεδομένα του τύπου DATE και η οποία δεν θα μπορεί να είναι κενή (NOT NULL).

Πρέπει να έχουμε υπόψη μας ότι, ενώ μπορούμε να γράψουμε τις εντολές της SQL με πεζά ή με κεφαλαία γράμματα, ένας MySQL server που εκτελείται σ' ένα σύστημα που είναι βασισμένο στο Unix, θα ξεχωρίζει τα πεζά από τα κεφαλαία γράμματα (case sensitive) όσον αφορά τα ονόματα των βάσεων δεδομένων και των πινάκων, εφόσον αυτά αντιστοιχούν σε καταλόγους (directories) και αρχεία (files) στον κατάλογο δεδομένων (data directory) της MySQL.

Αλλιώς, η MySQL δεν ξεχωρίζει καθόλου τα πεζά από τα κεφαλαία γράμματα (case insensitive) αλλά με μια εξαίρεση : τα ονόματα των πινάκων και των στηλών καθώς και τα άλλα ονόματα πρέπει να γράφονται ακριβώς το ίδιο όταν χρησιμοποιούνται περισσότερες από μία φορές στην ίδια εντολή.

Επίσης, εκχωρήσαμε έναν συγκεκριμένο τύπο δεδομένων σε κάθε στήλη που δημιουργήσαμε. Η ID θα περιέχει ακεραίους (integers), η Name θα περιέχει κείμενο (text) και η Date θα περιέχει ημερομηνίες (dates). Η MySQL απαιτεί να καθορίζουμε τον τύπο δεδομένων (data type) για κάθε στήλη από την αρχή.
Αν γράψουμε σωστά την παραπάνω εντολή, η MySQL θα εμφανίσει το μήνυμα *Query OK* και θα έχουμε έτσι δημιουργήσει τον πρώτο μας πίνακα (table).
Για να βεβαιωθούμε ότι πράγματι δημιουργήθηκε ο πίνακας, δίνουμε την εξής εντολή

mysql> SHOW TABLES;

Η απάντηση θα είναι ως εξής :
Tables in clients
Clients
1 row in set
Αυτή είναι η λίστα όλων των πινάκων που υπάρχουν στην βάση δεδομένων clinents και περιέχει έναν μόνο πίνακα : τον Clients. Ας ρίξουμε τώρα μια πιο αναλυτική ματιά στον πίνακα Clients :

mysql> DESCRIBE Clients;
Field | Type | Null | Key | Default
ID | int(11) | | PRI | 0 | ...
Name | text | YES | | NULL
Date | date | | | 0000-00-00
3 rows in set

Η παραπάνω εντολή εμφανίζει μια λίστα των στηλών (πεδίων) που υπάρχουν στον πίνακα. Όπως μπορούμε να δούμε, υπάρχουν τρεις στήλες σ' αυτόν τον πίνακα που εμφανίζονται σαν τρεις γραμμές στο αποτέλεσμα.

Για να διαγράψουμε έναν πίνακα στην MySQL, η εντολή είναι σχεδόν ολόιδια με την εντολή για την διαγραφή μιας βάσης δεδομένων :
*mysql> **DROP TABLE** <tableName>;*

Εισαγωγή Δεδομένων σε Πίνακα (Table)

Η εντολή για να εισάγουμε δεδομένα σε μια βάση δεδομένων αποκαλείται ***INSERT*** και υπάρχουν οι εξής δύο βασικές μορφές αυτής της εντολής :

```
mysql> INSERT INTO <table name> SET
-> columnName1 = value1,
-> columnName2 = value2,
-> ...
-> ;
mysql> INSERT INTO <table name>
-> (columnName1, columnName2, ...)
-> VALUES (value1, value2, ...);
```

Έτσι, για να προσθέσουμε έναν πελάτη στον πίνακα, μπορούμε να επιλέξουμε μια από τις εξής εντολές :

```
mysql> INSERT INTO Clients SET
-> Name = "Αβραμόπουλος",
-> Date = "1950-03-21";
mysql> INSERT INTO Clients
-> (Name, Date) VALUES (
-> "Δημητρίου",
-> "1960-07-14"
-> );
```

Πρέπει να έχουμε υπόψη μας ότι στην δεύτερη μορφή της εντολής *INSERT*, η σειρά με την οποία γράφουμε τις στήλες πρέπει να ταιριάζει με την σειρά με την οποία γράφουμε τις αντίστοιχες τιμές.

Εμφάνιση των Αποθηκευμένων Δεδομένων

Η εντολή για να δούμε τα δεδομένα που είναι αποθηκευμένα στους πίνακες μιας βάσης δεδομένων είναι η ***SELECT*** και αποτελεί την πιο πολύ χρησιμοποιούμενη και πιο πολύπλοκη εντολή της SQL.

Η επόμενη εντολή θα εμφανίσει ό,τι είναι αποθηκευμένο στον πίνακα Clients :
*mysql> SELECT * FROM Clients;*
Αν θέλουμε να εμφανισθούν οι τιμές ορισμένων μόνο στηλών, δίνουμε την εξής εντολή :
mysql> SELECT ID, Date FROM Clients;
Το αποτέλεσμα θα είναι ως εξής :

```
ID Date
 1  1950-03-21
 2  1960-07-14
2 rows in set (0.00 sec)
```

Εκτός από το να εμφανίσουμε τις στήλες που θέλουμε με την εντολή Select, μπορούμε να τροποποιήσουμε την εμφάνισή τους με συναρτήσεις (functions). Μια συνάρτηση, η *LEFT()*, εμφανίζει έναν μέγιστο καθορισμένο αριθμό χαρακτήρων για μια στήλη.

Για παράδειγμα, αν θέλουμε να δούμε μόνο τους 10 πρώτους χαρακτήρες της στήλης Name, μπορούμε να δώσουμε την εξής εντολή :

```
mysql> SELECT ID, LEFT(Name, 10), Date FROM Clients;
          ID LEFT(Name, 20) Date
          1 Αβραμόπουλ 1950-03-21
2 Δημητρίου 1960-07-14
        2 rows in set (0.05 sec)
```

Μια άλλη χρήσιμη συνάρτηση είναι η *COUNT()*, η οποία απλά μετράει τον αριθμό των επιστρεφόμενων αποτελεσμάτων. Έτσι, για παράδειγμα, αν θέλουμε να βρούμε πόσοι πελάτες υπάρχουν στον πίνακα, μπορούμε να χρησιμοποιήσουμε την εξής εντολή :

```
mysql> SELECT COUNT(*) FROM Clients;
        COUNT(*)
2
2 rows in set (0.06 sec)
```

Η Δήλωση WHERE

Χρησιμοποιώντας την δήλωση (clause) ***WHERE*** σε μια εντολή SELECT, μπορούμε να περιορίσουμε τα επιστρεφόμενα αποτελέσματα χρησιμοποιώντας κάποια συνθήκη, ως εξής :

```
mysql> SELECT COUNT(*) FROM Clients
-> WHERE Date >= "1950-01-01";
```

Το παραπάνω ερώτημα (query) θα μετρήσει τον αριθμό των πελατών που έχουν ημερομηνίες γέννησης μεγαλύτερες από ή ίσες από την 1η Ιανουαρίου 1950.
Μια άλλη παραλλαγή της δήλωσης WHERE που μας δίνει την δυνατότητα να αναζητήσουμε καταχωρήσεις που περιέχουν ένα συγκεκριμένο κομμάτι κειμένου, είναι η εξής :

```
mysql> SELECT Name FROM Clients
-> WHERE Name LIKE "%ίδη%";
```

Αυτό το ερώτημα (query) εμφανίζει το κείμενο όλων των ονομάτων των πελατών που περιέχουν το κείμενο *ίδη* στην στήλη Name. Η λέξη κλειδί (keyword) ***LIKE*** λέει στην MySQL ότι η συγκεκριμένη στήλη πρέπει να ταιριάζει με το δεδομένο υπόδειγμα, που σ' αυτήν την περίπτωση είναι το "%ίδη%".

Τα σύμβολα **%** εδώ σημαίνουν ότι το κείμενο *ίδη* μπορεί να έχει πριν και μετά ένα οποιοδήποτε κείμενο. Κάτι δηλαδή σαν τον γνωστό μας χαρακτήρα μπαλαντέρ (wildcard) * από άλλα προγράμματα.

Μπορούμε επίσης να συμπεριλάβουμε και συνθήκες στην δήλωση WHERE για να περιορίσουμε κι άλλο τα αποτελέσματα. Για παράδειγμα, για να εμφανίσουμε τους πελάτες που περιέχουν το κείμενο *ίδη* στο όνομά τους και που γεννήθηκαν μόνο τον *Απρίλιο του 1961*, μπορούμε να χρησιμοποιήσουμε το εξής ερώτημα :

```
mysql> SELECT Name FROM Clients WHERE
-> Name LIKE "%ίδη%" AND
-> Date >= "1961-04-01" AND
-> Date < "1961-05-01";
```

Τροποποίηση των Αποθηκευμένων Δεδομένων

Για να κάνουμε αλλαγές στις τιμές ενός πίνακα μιας βάσης δεδομένων, χρησιμοποιούμε την εντολή ***UPDATE***, η σύνταξη της οποίας είναι ως εξής :

```
mysql> UPDATE <tableName> SET
-> <col_name>=<new_value>, ...
-> WHERE <where clause>;
```

Έτσι, για παράδειγμα, αν θελήσουμε να αλλάξουμε την ημερομηνία γέννησης ενός πελάτη, θα πρέπει να δώσουμε την εξής εντολή :

```
mysql> UPDATE Clients SET Date="1960-04-01" WHERE ID=1;
```

Η στήλη ID είναι βολική σ' αυτήν την περίπτωση καθώς μας δίνει την δυνατότητα να διαχωρίσουμε εύκολα και σίγουρα τον πελάτη που θέλουμε να τροποποιήσουμε. Μπορούμε να χρησιμοποιήσουμε και την δήλωση WHERE, όπως στην επόμενη εντολή η οποία αλλάζει την ημερομηνία όλων των καταχωρήσεων πελατών που περιέχουν το κείμενο "*ίδη* ":

```
mysql> UPDATE Clients SET Date="1970-04-01"
-> WHERE Name LIKE "%ίδη%";
```

Διαγραφή Αποθηκευμένων Δεδομένων

Για να διαγράψουμε γραμμές (εγγραφές) στην SQL, η σύνταξη είναι η εξής :

```
mysql> DELETE FROM <tableName> WHERE <where clause>;
```

Έτσι, για παράδειγμα, για να διαγράψουμε όλους τους πελάτες που περιέχουν το κείμενο *ίδη* από τον πίνακα Clients, πρέπει να δώσουμε το επόμενο ερώτημα :

```
mysql> DELETE FROM Clients WHERE Name LIKE
"%ίδη%";
```

Η επόμενη εντολή θα αδειάσει τον πίνακα Clients με μιας :

```
mysql> DELETE FROM Clients
```

Η ΓΛΩΣΣΑ ΣΥΓΓΡΑΦΗΣ ΣΕΝΑΡΙΩΝ PHP

Τι Είναι η PHP

Η *PHP*, όπου τα αρχικά σημαίνουν *Hypertext PreProcessor*, είναι μια γλώσσα συγγραφής σεναρίων (scripting language) που ενσωματώνεται μέσα στον κώδικα της HTML και εκτελείται στην πλευρά του server (server-side scripting).

Ανταγωνιστικές της τεχνολογίας PHP είναι οι εξής γλώσσες προγραμματισμού : *ASP (Active Server Pages)* της εταιρείας Microsoft, *CFML (ColdFusion Markup Language)* της εταιρείας Allaire και *JSP (JavaServer Pages)* της εταιρείας Sun.

Το μεγαλύτερο μέρος της σύνταξής της, η PHP το έχει δανειστεί από την C, την Java και την Perl και διαθέτει και μερικά δικά της μοναδικά χαρακτηριστικά. Ο σκοπός της γλώσσας είναι να δώσει τη δυνατότητα στους web developers να δημιουργούν δυναμικά παραγόμενες ιστοσελίδες.

Ακολουθεί ένα εισαγωγικό παράδειγμα :

```
<html>
    <head>
        <title> Παράδειγμα </title>
    </head>
    <body>
        <?php echo "Γεια σας, είμαι ένα script της PHP!"; ?>
    </body>
</html>
```

Προσέξτε πόσο διαφέρει από ένα CGI script που γράφεται σ' άλλες γλώσσες, όπως η Perl ή η C, όπου αντί να γράψουμε ένα πρόγραμμα με πολλές εντολές για να δημιουργήσουμε κώδικα HTML, γράφουμε ένα HTML script με κάποιον ενσωματωμένο κώδικα για να κάνει κάτι, όπως στην συγκεκριμένη περίπτωση να εμφανίσει κάποιο κείμενο (μήνυμα). Ο κώδικας της PHP περικλείεται με ειδικά tags αρχής και τέλους για να μπορούμε να εισερχόμαστε και να εξερχόμαστε από το PHP mode.

Αυτό που ξεχωρίζει την PHP από μια γλώσσα όπως η JavaScript, η οποία εκτελείται στην πλευρά του χρήστη (client-side), είναι ότι ο κώδικάς της εκτελείται στον server. Αν είχαμε σ' έναν server ένα script παρόμοιο με το παραπάνω, ο χρήστης (client) θα λάμβανε το αποτέλεσμα της εκτέλεσης αυτού του script, χωρίς να είναι σε θέση να γνωρίζει ποιος μπορεί να είναι ο αρχικός κώδικας.

Μπορούμε ακόμη να ρυθμίσουμε (configure) τον web server ώστε να επεξεργάζεται όλα τα HTML αρχεία με την PHP και τότε δεν θα υπάρχει πράγματι κανένας τρόπος να μάθουν οι χρήστες τον κώδικά μας.

Τι Μπορεί να Κάνει η PHP

Στο πιο βασικό επίπεδο, η PHP μπορεί να κάνει ό,τι και τα άλλα προγράμματα της τεχνολογίας CGI, όπως επεξεργασία των δεδομένων μιας φόρμας, δημιουργία δυναμικού περιεχομένου ιστοσελίδων ή αποστολή και λήψη cookies.

Υπάρχουν τρεις κύριοι τομείς που χρησιμοποιείται ένα PHP script.

- Server-side scripting. Αυτό είναι το πιο παραδοσιακό και το κύριο πεδίο για την PHP. Χρειάζεστε τρία πράγματα για να δουλέψει αυτό. Τον PHP μεταγλωττιστή (parser) (CGI ή server module), ένα webserver (εξηπηρετητή σελίδων) και ένα web browser ("φυλλομετρητή"). Πρέπει να τρέξετε τον webserver, με μια συνδεδεμένη εγκατάσταση της PHP. Μπορείτε να προσπελάσετε τα αποτελέσματα του PHP προγράμματος με ένα web browser, βλέποντας την σελίδα PHP μέσα από τον server.
- Command line scripting. Μπορείτε να φτιάξετε ένα PHP script για να το τρέχετε χωρίς server ή browser. Χρειάζεστε μόνο τον PHP μεταγλωττιστή για να την χρησιμοποιήσετε με αυτό τον τρόπο. Αυτός ο τύπος είναι ιδανικός για script που εκτελούνται συχνά με τη χρήση της cron (σε Unix ή Linux) ή με τον Task Scheduler (στα Windows). Αυτά τα script μπορούν επίσης να χρησιμοποιηθούν για απλές εργασίες επεξεργασίες κειμένου.
- Εγγραφή client-side GUI εφαρμογών (Γραφικά περιβάλλοντα χρηστών). Η PHP ίσως να μην είναι η πιο καλή γλώσσα για να γράψει κανείς παραθυριακές εφαρμογές, αλλά αν ξέρετε PHP πολύ καλά και θέλετε να χρησιμοποιήσετε κάποια προχωρημένα χαρακτηριστικά της PHP στις client-side εφαρμογές σας, μπορείτε επίσης να χρησιμοποιήσετε το PHP-GTK για αυτού του είδους τα προγράμματα. Έχετε επίσης τη δυνατότητα να γράφετε cross-platform εφαρμογές με αυτό τον τρόπο. Το PHP-GTK είναι μια επέκταση της PHP και δεν συμπεριλαμβάνεται στην κύρια διανομή.

Η PHP μπορεί να χρησιμοποιηθεί σε όλα τα κύρια λειτουργικά συστήματα, συμπεριλαμβανομένου του Linux, πολλών εκδοχών του Unix (HP-UX, Solaris και OpenBSD), Microsoft Windows, Mac OS X, RISC OS και πιθανώς σε άλλα. Η PHP υποστηρίζει επίσης τους Apache, Microsoft Internet Information Server, Personal Web Server, Netscape και iPlanet servers, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd, και πολλούς άλλους webserver. Για την πλειοψηφία των server η PHP έχει ένα module, για τους υπόλοιπους η PHP μπορεί να λειτουργήσει ως ένας CGI επεξεργαστής.

Έτσι με την PHP έχετε την ελευθερία επιλογής ενός λειτουργικού συστήματος και ενός web server. Επιπλέον, έχετε επίσης την ελευθερία να χρησιμοποιήσετε

συναρτησιακό (procedural) ή αντικειμενοστρεφή (object oriented) προγραμματισμό ή μια ανάμειξη τους. Αν και η παρούσα έκδοση δεν υποστηρίζει όλα τα πρότυπα χαρακτηριστικά, μεγάλες βιβλιοθήκες κώδικα και μεγάλες εφαρμογές (συμπεριλαμβανομένης και της βιβλιοθήκης PEAR) είναι γραμμένες μόνο με αντικειμενοστρεφή κώδικα.

Με την PHP δεν είστε περιορισμένοι να εξάγετε HTML. Οι δυνατότητες της PHP συμπεριλαμβάνουν την εξαγωγή εικόνων, αρχείων PDF, ακόμη και ταινίες Flash (χρησιμοποιώντας τα libswf και Ming) παράγονται αμέσως. Μπορείτε επίσης να εξάγετε εύκολα οποιοδήποτε κείμενο όπως XHTML και οποιοδήποτε άλλο XML αρχείο. Η PHP μπορεί να δημιουργεί αυτόματα αυτά τα αρχεία και να τα αποθηκεύει στο σύστημα αρχείων, αντί να τα εκτυπώνει, αποτελώντας έτσι μια server-side cache για το δυναμικό σας περιεχόμενο.

Ίσως το δυνατότερο και πιο σημαντικό χαρακτηριστικό της PHP είναι η υποστήριξη που παρέχει σε μια ευρεία γκάμα από βάσεις δεδομένων. Έτσι, το να δημιουργήσουμε μια ιστοσελίδα που να παρέχει υποστήριξη σε βάσεις δεδομένων είναι απίστευτα απλό. Υποστηρίζει τις εξής βάσεις δεδομένων :

Adabas D	dBase	Empress	FilePro	Informix	InterBase	mSQL
MySQL	Oracle	PostgreSQL	Solid	Sybase	Velocis	Unix dbm

Η PHP παρέχει επίσης υποστήριξη για συνομιλία μ' άλλες υπηρεσίες, χρησιμοποιώντας πρωτόκολλα όπως τα IMAP, SNMP, NNTP, POP3 ή και το HTTP.

Μια Σύντομη Ιστορία της PHP

Η ιδέα για την δημιουργία της PHP ελήφθη το φθινόπωρο του 1994 από τον *Rasmus Lerdorf*. Οι πρώτες ανεπίσημες εκδόσεις (versions) της PHP χρησιμοποιήθηκαν στην αρχική του σελίδα (home page) για να μπορεί να παρακολουθεί αυτούς που έμπαιναν στην σελίδα. Η πρώτη έκδοση που δόθηκε για χρήση στο κοινό ήταν διαθέσιμη στις αρχές του 1995 με το όνομα *Personal Home Page Tools.*

Αποτελείτο από μια πολύ απλοϊκή μηχανή ανάλυσης (parser engine) η οποία καταλάβαινε λίγες μόνο ειδικές μακροεντολές (macros) και έναν αριθμό από utilities που βρίσκονταν σε κοινή χρήση στις home pages εκείνη την εποχή. Ένα guestbook, ένας μετρητής (counter) και κάποιο άλλο υλικό. Ο αναλυτής (parser) ξαναγράφηκε στα μέσα του 1995 και ονομάστηκε *PHP/FI Version 2.*

Το όνομα FI προέρχεται από ένα άλλο πακέτο που είχε γράψει ο Rasmus και το οποίο διερμήνευε (interpreted) τα δεδομένα από φόρμες της HTML. Συνδύασε τα εργαλεία scripts της Personal Home Page με τον *Form Interpreter* και πρόσθεσε υποστήριξη για mSQL. Έτσι γεννήθηκε η *PHP/FI*, η οποία αναπτύχθηκε αλματωδώς και διάφοροι χρήστες άρχισαν να συνεισφέρουν κώδικα σ' αυτήν.

Υπολογίζεται ότι μέχρι τα τέλη του 1996, η PHP/FI χρησιμοποιείτο σε τουλάχιστον 15.000 web sites σ' όλον τον κόσμο και στα μέσα του 1997 αυτός ο αριθμός είχε

ξεπεράσει τις 50.000. Στα μέσα του 1997 είχαμε επίσης μια αλλαγή στην ανάπτυξη της PHP.

Σταμάτησε να αποτελεί το αγαπημένο αντικείμενο του Rasmus και έγινε ο στόχος μιας πιο καλά οργανωμένης ομαδικής εργασίας. Ο αναλυτής (parser) ξαναγράφηκε από την αρχή από τους Zeev Suraski και Andi Gutmans και αυτός ο νέος parser αποτέλεσε τη βάση για την PHP Version 3. Ένα μεγάλο μέρος του utility code μεταφέρθηκε από την PHP/FI στην PHP3 και ένα μεγάλο μέρος του ξαναγράφηκε από την αρχή.

Σήμερα, η PHP/FI και η PHP3 έρχονται μ' έναν αριθμό εμπορικών προϊόντων όπως ο web server C2 StrongHold και το RedHat Linux. Σύμφωνα με μια συντηρητική εκτίμηση, η PHP χρησιμοποιείται από περισσότερα από 150.000 sites σ' όλον τον κόσμο.

Πώς να Ξεφύγουμε από την HTML

Υπάρχουν τέσσερις τρόποι για να μπορέσουμε να ξεφύγουμε από την HTML και να μπούμε στην μέθοδο συγγραφής κώδικα της PHP (PHP code mode) :

1ος τρόπος

```
<? echo ("Είναι η απλούστερη, μια εντολή επεξεργασίας SGML \n"); ?>
```

2ος τρόπος

```
<?php echo("Αν θέλουμε να εξυπηρετήσουμε XML έγγραφα \n"); ?>
```

3ος τρόπος

```
<script language="php">

    echo ("Σε μερικούς editors, όπως ο FrontPage, δεν αρέσουν οι εντολές επεξεργασίας");

</script>
```

4ος τρόπος

```
<% echo ("Μπορούμε να χρησιμοποιήσουμε και tags με στυλ ASP"); %>

<%= $variable;      # Είναι μια συντόμευση για το "<%echo .." %>
```

Ο πρώτος τρόπος είναι διαθέσιμος μόνο αν έχουμε ενεργοποιήσει τα σύντομα (short) tags. Αυτό μπορεί να γίνει με τη συνάρτηση *short_tags()*, ενεργοποιώντας το *short_open_tag configuration setting* στο αρχείο *config* της PHP ή μεταγλωττίζοντας την PHP με την επιλογή *--enable-short-tags option.*

Ο τέταρτος τρόπος είναι διαθέσιμος μόνο αν έχουν ενεργοποιηθεί τα tags με στυλ ASP με το *asp_tags configuration setting*. Η υποστήριξη για τα ASP-style tags προστέθηκε στην έκδοση 3.0.4.

Τερματισμός Εντολών

Οι εντολές στην PHP τερματίζονται με τον ίδιο τρόπο όπως στην C και την Perl, δηλ. μ' έναν χαρακτήρα ; (semicolon). Μπορούμε, όμως, να δηλώσουμε το τέλος μιας εντολής και με το tag κλεισίματος (closing tag) ?>. Έτσι, τα παρακάτω είναι ισοδύναμα :

```
<?php

        echo "This is a test";

?>
```

και

```
<?php echo "This is a test" ?>
```

Σχόλια (Comments)

Η PHP χρησιμοποιεί τον ίδιο τρόπο σχολιασμού όπως η C, η C++ και το Unix shell. Για παράδειγμα :

```
<?php

        echo "Αυτή είναι μια δοκιμή"; // Σχόλιο μίας γραμμής της C++

        /* Αυτό είναι ένα σχόλιο (comment) της C σε πολλές γραμμές

        και αυτή είναι μια άλλη γραμμή σχολίου */

        echo "Αυτή είναι άλλη μια δοκιμή";

        echo "Μια τελική δοκιμή"; # Σχόλιο της shell

?>
```

Τα σχόλια μίας γραμμής σχολιάζουν μέχρι το τέλος της γραμμής ή το τρέχον μπλοκ του PHP κώδικα, ανάλογα με το ποιο εμφανίζεται πρώτο.

```
        <h1> Αυτό είναι ένα <?# echo "απλό";?> παράδειγμα. </h1>

        <p> To header θα εμφανίσει το 'Αυτό είναι ένα παράδειγμα.' </p>
```

Πρέπει να είμαστε προσεκτικοί για να μην φωλιάζουμε τα σχόλια τύπου C.

```
<?php

    /*

    echo "Αυτή είναι μια δοκιμή";

    /* Αυτό το σχόλιο θα δημιουργήσει πρόβλημα */

    */

?>
```

Οι Τύποι Δεδομένων της PHP

Η PHP υποστηρίζει τους εξής τύπους δεδομένων :

- array
- floating-point numbers
- integer
- object
- string

Ο τύπος δεδομένων μιας μεταβλητής δεν ορίζεται συνήθως από τον προγραμματιστή αλλά αποφασίζεται την ώρα εκτέλεσης (runtime) από την PHP ανάλογα με το περιβάλλον (context) στο οποίο χρησιμοποιείται η μεταβλητή. Αν θέλουμε να κάνουμε μια μεταβλητή να μετατραπεί σ' έναν συγκεκριμένο τύπο, μπορούμε είτε να *μετατρέψουμε (cast)* τη μεταβλητή ή να χρησιμοποιήσουμε τη συνάρτηση *settype()* σ' αυτή.

Πρέπει να έχουμε υπόψη μας ότι μια μεταβλητή μπορεί να συμπεριφερθεί διαφορετικά σε συγκεκριμένες καταστάσεις, ανάλογα με το τι τύπο δεδομένων έχει εκείνη την στιγμή.

Κόλπα με τους Τύπους Δεδομένων (Type Juggling)

Η PHP δεν υποστηρίζει τον σαφή (explicit) ορισμό τύπων δεδομένων στις δηλώσεις μεταβλητών και αυτό γιατί ο τύπος μιας μεταβλητής καθορίζεται από το περιβάλλον (context) στο οποίο χρησιμοποιείται αυτή η μεταβλητη.

Αυτό σημαίνει ότι αν εκχωρήσουμε μια τιμή string σε μια μεταβλητή var, η var θα γίνει ένα string και αν αργότερα εκχωρήσουμε μια ακέραια τιμή στην var, αυτή θα γίνει ακέραια (integer).

Ένα παράδειγμα της αυτόματης μετατροπής τύπου στην PHP είναι ο τελεστής πρόσθεσης +. Αν κάποιος από τους τελεστέους (operands) είναι μια τιμή double, τότε όλοι οι τελεστέοι εκτιμούνται σαν τύπου doubles και το αποτέλεσμα θα είναι τύπου double.

Αλλιώς, οι τελεστέοι θα θεωρηθούν ότι είναι ακέραιοι (integers) και το αποτέλεσμα θα είναι επίσης integer. Αυτό βέβαια δεν αλλάζει τους τύπους δεδομένων των ίδιων των τελεστέων και η μόνη αλλαγή είναι στο πώς εκτιμούνται οι τελεστέοι.

```
$a = "0";                       //Η $a  είναι  string (ASCII 48)

$a ++;                          //Η $a είναι είναι το string "1" (ASCII 49)

$a += 1;                        //Η $a είναι τώρα  ακέραιος (2)

$a = $a + 1.3;                  //Η $a είναι  now a double (3.3)

$a = 5 + "10 Αυτοκίνητα ";      //Η $a είναι ακέραιος (15)

$a = 5 + "10 Μηχανάκια";        //Η $a είναι  ακέραιος (15)
```

Αν θέλουμε να δοκιμάσουμε κάποια από τις παραπάνω εκφράσεις, μπορούμε να την αντιγράψουμε και να εισάγουμε την παρακάτω γραμμή για να δούμε τι γίνεται :

```
echo "\$a==$a; type is " . gettype( $a ) . "<br>\n";
```

Η Μετατροπή Τύπων (Type Casting)

Η μετατροπή τύπων (type casting) στην PHP εργάζεται όπως και στην C : το όνομα του επιθυμητού τύπου γράφεται μέσα σε παρενθέσεις πριν από τη μεταβλητή η οποία θα μετατραπεί.

```
$foo = 10;                  // Το $foo είναι τύπου integer

$bar = (double) $foo;       // Το $bar είναι τύπου double
```

Οι μετατροπές (casts) που επιτρέπονται είναι οι εξής :

(int), (integer) - cast to integer

(real), (double), (float) - cast to double

(string) - cast to string

(array) - cast to array

(object) - cast to object

Μπορεί να υπάρχουν κενά και tabs μέσα στις παρενθέσεις :

```
$foo = (int) $bar;

$foo = ( int ) $bar;
```

Όταν κάνουμε μετατροπή από μια μεταβλητή scalar ή string σ' έναν πίνακα (array), η μεταβλητή θα γίνει το πρώτο στοιχείο του πίνακα :

```
$var = 'ciao';

$arr = (array) $var;

echo $arr[0];                    // εμφανίζει 'ciao'
```

Όταν κάνουμε μετατροπή από μια μεταβλητή scalar ή string σ' ένα αντικείμενο (object), η μεταβλητή θα γίνει μια ιδιότητα (attribute) του αντικειμένου και το όνομα (name) της ιδιότητας θα είναι τύπου scalar :

```
$var = 'ciao';

$obj = (object) $var;

echo $obj->scalar;  // εμφανίζει 'ciao'
```

Οι Μεταβλητές (Variables)

Οι *μεταβλητές (variables)* στην PHP παριστάνονται από το σύμβολο $ ακολουθούμενο από το όνομα της μεταβλητής. Τα ονόματα των μεταβλητών ξεχωρίζουν τα πεζά από τα κεφαλαία γράμματα (case-sensitive).

```
$var = "Bob";

$Var = "Joe";

echo "$var, $Var";   // εμφανίζει "Bob, Joe"
```

Στην PHP3, οι μεταβλητές πάντα εκχωρούνται *με τιμή (by value)*, δηλαδή όταν εκχωρούμε μια έκφραση σε μια μεταβλητή, η τιμή της αρχικής έκφρασης αντιγράφεται στη μεταβλητή προορισμού. Αυτό σημαίνει, για παράδειγμα, ότι αφού έχουμε εκχωρήσει την τιμή μιας μεταβλητής σε μια άλλη, η αλλαγή σε μια απ' αυτές τις μεταβλητές δεν θα επηρεάσει την άλλη.

Η PHP4 προσφέρει και έναν άλλον τρόπο για να εκχωρήσουμε τιμές σε μεταβλητές : *με αναφορά (by reference)*, δηλ. η νέα μεταβλητή αναφέρεται (references) ή αποτελεί ένα ψευδώνυμο (alias) ή δείχνει (points) στην αρχική μεταβλητή.

Οι αλλαγές στη νέα μεταβλητή επηρεάζουν και την αρχική και το αντίστροφο. Για να κάνουμε εκχώρηση με αναφορά, τοποθετούμε το σύμβολο *& (ampersand)* πριν από την αρχική μεταβλητή. Για παράδειγμα, ο επόμενος κώδικας εμφανίζει δύο φορές το μήνυμα *'My name is Bob'* :

```
<?php
```

```
$foo = 'Bob';                    // Εκχώρηση της τιμής 'Bob' στην $foo

$bar = &$foo;                    // Αναφορά στην $foo μέσω της $bar.

$bar = "My name is $bar";        // Η $bar αλλάζει

echo $foo;                       // Η $foo αλλάζει επίσης

echo $bar;

?>
```

Κάτι σημαντικό που πρέπει να σημειώσουμε είναι ότι μόνο ονοματισμένες μεταβλητές (named variables) μπορούν να εκχωρηθούν με αναφορά (by reference).

```
<?php

$foo = 25;

$bar = &$foo;        // Έγκυρη εκχώρηση

$bar = &(24 * 7);    // Μη έγκυρη εκχώρηση

function test() {

        return 25;

}

$bar = &test();      // Μη έγκυρη

?>
```

Η Εμβέλεια των Μεταβλητών

Η εμβέλεια (scope) μιας μεταβλητής είναι το περιβάλλον (context) μέσα στο οποίο ορίζεται. Οι περισσότερες από τις PHP μεταβλητές έχουν μία μόνο περιοχή εμβέλειας. Για παράδειγμα :

```
$a = 1;

include "b.inc";
```

Εδώ, η μεταβλητή $a θα είναι διαθέσιμη μέσα στο συμπεριλαμβανόμενο (included) b.inc script. Όμως, στις οριζόμενες από τον προγραμματιστή συναρτήσεις (user-defined functions) υπάρχει μια τοπική εμβέλεια. Μια μεταβλητή που χρησιμοποιείται μέσα σε μια συνάρτηση είναι εξ ορισμού περιορισμένη στην τοπική εμβέλεια αυτής της συνάρτησης. Για παράδειγμα :

```
$a = 1;              /* καθολική εμβέλεια */

Function Test () {

        echo $a;     /* αναφέρεται στην τοπική μεταβλητή */

}

Test ();
```

Αυτό το script δεν θα εμφανίσει κάποια έξοδο επειδή η εντολή echo αναφέρεται σε μια τοπική μεταβλητή (local variable) $a, η οποία δεν έχει αποκτήσει τιμή μέσα στην εμβέλειά της. Αυτό είναι διαφορετικό από τη γλώσσα C στο ότι οι καθολικές μεταβλητές (global variables) της C είναι αυτόματα διαθέσιμες στις συναρτήσεις εκτός κι αν επικαλύπτονται σαφώς από μια τοπική δήλωση.

Αυτό μπορεί να προκαλέσει προβλήματα στο ότι μπορεί κάποιος άθελά του να αλλάξει μια καθολική μεταβλητή. Στην PHP οι καθολικές μεταβλητές πρέπει να δηλωθούν σαν global μέσα σε μια συνάρτηση αν πρόκειται να τις χρησιμοποιήσουμε μέσα σ' αυτή τη συνάρτηση. Ακολουθεί ένα παράδειγμα :

```
$a = 1;

$b = 2;

Function Sum () {

        global $a, $b;

        $b = $a + $b;

}

Sum ();

echo $b;
```

Το παραπάνω script θα εμφανίσει το 3. Δηλώνοντας τις $a και $b σαν global μέσα στη συνάρτηση, όλες οι αναφορές και στις δύο μεταβλητές θα αφορούν τις καθολικές τιμές. Δεν υπάρχει κάποιος περιορισμός στον αριθμό των καθολικών μεταβλητών που μπορεί να χειριστεί μια συνάρτηση.

Ένας δεύτερος τρόπος για να έχουμε πρόσβαση σε μεταβλητές σε καθολική εμβέλεια είναι να χρησιμοποιήσουμε τον ειδικό πίνακα που ορίζεται στην PHP με όνομα *$GLOBALS*. Έτσι, το προηγούμενο παράδειγμα θα μπορεί να ξαναγραφεί ως εξής :

```
$a = 1;

$b = 2;
```

```
Function Sum () {

        $GLOBALS["b"] = $GLOBALS["a"] + $GLOBALS["b"];

}

Sum ();

echo $b;
```

Ο πίνακας *$GLOBALS* είναι ένας associative πίνακας με το όνομα της καθολικής μεταβλητής να αποτελεί το key και τα περιεχόμενα αυτής της μεταβλητής να αποτελούν την τιμή του στοιχείου του πίνακα.

Ένα άλλο σημαντικό χαρακτηριστικό είναι οι στατικές μεταβλητές. Μια *στατική μεταβλητή (static variable)* υπάρχει μόνο στην τοπική εμβέλεια μιας συνάρτησης αλλά δεν χάνει την τιμή της όταν η εκτέλεση του προγράμματος εγκαταλείπει τη συνάρτηση. Ας δούμε το επόμενο παράδειγμα :

```
Function Test () {

        $a = 0;

        echo $a;

        $a++;

}
```

Αυτή η συνάρτηση δεν είναι και τόσο χρήσιμη εφόσον κάθε φορά που καλείται καταχωρεί στο $a το 0 και εκτυπώνει το "0". Η εντολή $a++, η οποία αυξάνει την τιμή της μεταβλητής, δεν κάνει τίποτα εφόσον μόλις τελειώνει η συνάρτηση, εξαφανίζεται η μεταβλητή $a.

Για να δημιουργήσουμε μια χρήσιμη συνάρτηση μέτρησης η οποία δεν θα χάνει τον έλεγχο της τρέχουσας μέτρησης, θα πρέπει να δηλώσουμε τη μεταβλητή $a σαν *static* :

```
Function Test () {

        static $a = 0;

        echo $a;

        $a++;

}
```

Τώρα, κάθε φορά που καλείται η συνάρτηση Test(), θα εκτυπώνει την τιμή της $a και θα την αυξάνει.

Οι στατικές μεταβλητές παρέχουν επίσης έναν τρόπο για να ασχοληθούμε τις αναδρομικές συναρτήσεις. Μια *αναδρομική συνάρτηση (recursive function)* είναι αυτή που καλεί τον εαυτό της. Η επόμενη συνάρτηση μετράει αναδρομικά έως το 10 και χρησιμοποιεί τη στατική μεταβλητή $count για να ξέρει πότε να σταματήσει :

```
Function Test () {
        static $count = 0;
        $count++;
        echo $count;
        if ($count < 10) {
                Test ();
        }
        $count- -;
}
```

Μεταβλητές Μεταβλητές

Μερικές φορές είναι βολικό να μπορούμε να έχουμε μεταβλητά ονόματα μεταβλητών (variable variable names), δηλ. ένα όνομα μεταβλητής το οποίο μπορεί να ορισθεί και να χρησιμοποιηθεί δυναμικά. Όπως γνωρίζουμε, μια κανονική μεταβλητή ορίζεται με μια εντολή σαν την εξής :

```
$a = "hello";
```

Μια μεταβλητή μεταβλητή αποκτά την τιμή μιας μεταβλητής και την αντιμετωπίζει σαν το όνομα μιας μεταβλητής. Στο παραπάνω παράδειγμα, το hello, μπορεί να χρησιμοποιηθεί σαν το όνομα μιας μεταβλητής χρησιμοποιώντας δύο σύμβολα $, ως εξής :

```
$$a = "world";
```

Σ’ αυτό το σημείο έχουμε ορίσει δύο μεταβλητές και τις έχουμε αποθηκεύσει στο συμβολικό δένδρο της PHP : η $a με περιεχόμενο "hello" και η $hello με περιεχόμενο "world". Συνεπώς, η επόμενη εντολή :

```
echo "$a ${$a}";
```

παράγει την ίδια ακριβώς έξοδο με την :

echo "$a $hello";

Δηλαδή και οι δύο παράγουν το : *hello world*.

Για να μπορέσουμε να χρησιμοποιήσουμε μεταβλητές μεταβλητές με πίνακες (arrays), θα πρέπει να λύσουμε ένα πρόβλημα ασάφειας. Δηλαδή, αν γράψουμε $$a[1], τότε ο αναλυτής (parser) θα πρέπει να γνωρίζει αν σκοπεύαμε να χρησιμοποιήσουμε το $a[1] σαν μια μεταβλητή ή αν θέλαμε να είναι το $$a η μεταβλητή και μετά το [1] ο δείκτης (index) απ' αυτή τη μεταβλητή. Η σύνταξη για να επιλύσουμε αυτήν την αμφιβολία είναι : *${$a[1]}* για την πρώτη περίπτωση και *${$a}[1]* για τη δεύτερη.

Μεταβλητές Εκτός της PHP

Όταν υποβάλλεται μια φόρμα σ' ένα PHP script, όλες οι μεταβλητές αυτής της φόρμας γίνονται αυτόματα διαθέσιμες στο script από την PHP. Για παράδειγμα, ας δούμε την εξής φόρμα :

```
<form action="foo.php3" method="post">

    Όνομα : <input type="text" name="name"><br>

    <input type="submit">

</form>
```

Όταν υποβληθεί η φόρμα, η PHP θα δημιουργήσει μια μεταβλητή με όνομα *$name*, η οποία θα περιέχει την τιμή που καταχωρήθηκε στο πεδίο *name* της φόρμας.

Η PHP καταλαβαίνει επίσης τους πίνακες στο περιβάλλον των μεταβλητών φόρμας αλλά μόνο σε μία διάσταση. Μπορούμε, για παράδειγμα, να ομαδοποιήσουμε σχετικές μεταβλητές μαζί ή να χρησιμοποιήσουμε αυτό το χαρακτηριστικό για να ανακτήσουμε τιμές από μια λίστα πολλαπλής επιλογής (multiple select input) :

```
<form action="array.php" method="post">

    Όνομα : <input type="text" name="personal[name]"><br>

    Email : <input type="text" name="personal[email]"><br>

    Μπύρα : <br>

    <select multiple name="beer[]">

        <option value="Fix"> Fix

        <option value="Heineken"> Heineken

        <option value="Pilsen"> Pilsen
```

```
</select>

<input type="submit">

</form>
```

Αν το χαρακτηριστικό *track_vars* της PHP είναι ενεργοποιημένο, είτε με τη ρύθμιση σύνθεσης (configuration setting) *track_vars* ή με την οδηγία (directive) *<?php_track_vars?>*, τότε οι μεταβλητές που υποβάλλονται με τις μεθόδους POST ή GET θα βρίσκονται επίσης στους global associative πίνακες (arrays) *$HTTP_POST_VARS* και *$HTTP_GET_VARS*.

Οι Μεταβλητές Image Submit

Όταν υποβάλλουμε μια φόρμα, μπορούμε να χρησιμοποιήσουμε μια εικόνα (image) αντί για το στάνταρτ πλήκτρο submit, μ' ένα tag σαν το :

```
<input type=image src="image.gif" name="sub">
```

Όταν ο χρήστης κάνει κλικ κάπου πάνω στην εικόνα, η φόρμα θα σταλεί στον server με δύο επιπλέον μεταβλητές, τις *sub_x* και *sub_y*, οι οποίες περιέχουν τις συντεταγμένες (coordinates) του κλικ που έκανε ο χρήστης μέσα στην εικόνα.

Τα HTTP Cookies

Η PHP υποστηρίζει τα HTTP cookies όπως ορίζεται από τις προδιαγραφές της Netscape. Τα cookies είναι ένας μηχανισμός για να αποθηκεύονται δεδομένα στον απομακρυσμένο φυλλομετρητή και έτσι να μπορούμε να παρακολουθούμε ή να αναγνωρίζουμε τους χρήστες. Μπορούμε να ορίσουμε τα cookies με τη συνάρτηση *SetCookie()*.

Τα cookies αποτελούν μέρος του HTTP header, έτσι η συνάρτηση SetCookie() πρέπει να κληθεί πριν σταλεί κάποια έξοδος στον φυλλομετρητή. Αυτός είναι ο ίδιος περιορισμός που ισχύει και για τη συνάρτηση Header(). Τα cookies που στέλνονται σε μας από τον client θα μετατραπούν αυτόματα σε μια μεταβλητή της PHP όπως συμβαίνει με τα δεδομένα των μεθόδων GET και POST.

Αν θελήσουμε να εκχωρήσουμε πολλαπλές τιμές σ' ένα μόνο cookie, απλά προσθέτουμε τα σύμβολα [ ] στο όνομα του cookie, ως εξής :

```
SetCookie ("MyCookie[]", "Testing", time()+3600);
```

Ένα cookie θα αντικαταστήσει ένα ήδη υπάρχον με το ίδιο όνομα στον φυλλομετρητή μας εκτός κι αν διαφέρουν η διαδρομή (path) ή το domain. Έτσι, για μια εφαρμογή shopping cart μπορεί να θέλουμε να έχουμε έναν μετρητή (counter) και να το μεταβιβάσουμε αυτό ως εξής :

```
$Count++;

SetCookie ("Count", $Count, time()+3600);

SetCookie ("Cart[$Count]", $item, time()+3600);
```

Οι Μεταβλητές Περιβάλλοντος (Environment)

Η PHP κάνει αυτόματα διαθέσιμες τις μεταβλητές περιβάλλοντος (environment) σαν κανονικές PHP μεταβλητές.

```
echo $HOME;

/* Δείχνει τη μεταβλητή environment HOME, αν έχει ορισθεί */
```

Εφόσον οι πληροφορίες που έρχονται μέσω των μηχανισμών GET, POST και Cookie δημιουργούν επίσης αυτόματα μεταβλητής PHP, είναι μερικές φορές καλύτερο να διαβάσουμε ρητά μια μεταβλητή από το environment για να είμαστε σίγουροι ότι λαμβάνουμε τη σωστή έκδοση.

Η συνάρτηση *getenv()* μπορεί να χρησιμοποιηθεί γι' αυτόν τον σκοπό. Μπορούμε επίσης να ορίσουμε μια μεταβλητή environment με τη συνάρτηση *putenv()*.

Οι Τελείες (Dots) στις Εισερχόμενες Μεταβλητές

Τυπικά, η PHP δεν αλλάζει τα ονόματα των μεταβλητών όταν αυτά μεταβιβάζονται σ' ένα script. Όμως, θα πρέπει να σημειωθεί ότι η τελεία (dot) δεν αποτελεί έναν έγκυρο χαρακτήρα σ' ένα όνομα μεταβλητής της PHP, ως εξής :

```
$varname.ext;          /* μη έγκυρο όνομα μεταβλητής */
```

Τώρα, αυτό που βλέπει ο αναλυτής (parser) είναι μια μεταβλητή με όνομα $varname ακολουθούμενη από τον τελεστή ένωσης string και από το ext που δεν ταιριάζει με κάποια γνωστή λέξη κλειδί.

Γι' αυτόν τον λόγο, είναι σημαντικό να σημειώσουμε ότι η PHP αντικαθιστά αυτόματα τις τελείες στα εισερχόμενα ονόματα μεταβλητών με τον χαρακτήρα _ (underscore).

Καθορισμός των Τύπων Μεταβλητών

Επειδή η PHP καθορίζει τους τύπους των μεταβλητών και τους μετατρέπει όπως χρειάζεται, δεν είναι πάντα σίγουρο τι τύπο δεδομένων έχει μια δεδομένη μεταβλητή σε κάποια δεδομένη χρονική στιγμή.

Η PHP περιέχει αρκετές συναρτήσεις που μπορούμε να χρησιμοποιήσουμε για να βρούμε τον τύπο δεδομένων μιας μεταβλητής. Αυτές είναι οι *gettype(), is_long(), is_double(), is_string(), is_array()* και *is_object()*.

Οι Εκφράσεις (Expressions)

Οι *εκφράσεις (expressions)* είναι από τους σημαντικότερους θεμέλιους λίθους της PHP. Στην PHP, σχεδόν ο,τιδήποτε γράφουμε αποτελεί μια έκφραση. Σαν έναν πολύ απλοϊκό αλλά ακριβή ορισμό για μια έκφραση μπορούμε να πούμε ότι «είναι ο,τιδήποτε έχει μια τιμή».

Οι βασικότερες μορφές εκφράσεων είναι οι σταθερές (constants) και οι μεταβλητές (variables). Για παράδειγμα, όταν γράφουμε *$a = 5*, εκχωρούμε το 5 στο $a. Το 5 είναι μια έκφραση (expression) με την τιμή 5 και σ' αυτήν την περίπτωση το 5 είναι μια ακέραια σταθερά.

Μετά απ' αυτήν την εκχώρηση, θα αναμέναμε η τιμή της $a να είναι ίση με 5, έτσι αν γράψουμε *$b = $a*, θα περιμέναμε να συμπεριφερθεί σαν να είχαμε γράψει *$b = 5*. Μ' άλλα λόγια, το $a είναι μια έκφραση (expression) με την τιμή 5 επίσης.

Λίγο περισσότερα σύνθετα παραδείγματα για τις εκφράσεις είναι οι συναρτήσεις (functions), όπως για παράδειγμα η ακόλουθη συνάρτηση :

```
function foo () {

        return 5;

}
```

Οι συναρτήσεις είναι εκφράσεις με την τιμή της τιμής επιστροφής τους (return value). Έτσι, εφόσον η συνάρτηση foo() επιστρέφει το 5, η τιμή της έκφρασης foo() είναι 5. Συνήθως, βέβαια οι συναρτήσεις δεν επιστρέφουν απλά μια στατική τιμή αλλά κάνουν και υπολογισμούς.

Η PHP υποστηρίζει τους εξής τρεις τύπους τιμών scalar : ακέραιες τιμές (integer), τιμές κινητής υποδιαστολής (floating point) και αλφαριθμητικές τιμές (string) values. Οι τιμές scalar είναι τιμές που δεν μπορούμε να διασπάσουμε σε μικρότερα κομμάτια, σ' αντίθεση με τους πίνακες (arrays), για παράδειγμα.

Η PHP υποστηρίζει επίσης δύο σύνθετους (composite, non-scalar) τύπους : τους πίνακες (arrays) και τα αντικείμενα (objects). Ο καθένας απ' αυτούς τους τύπους τιμών μπορεί να εκχωρηθεί σε μεταβλητές ή να επιστραφεί από συναρτήσεις.

Η PHP αντιμετωπίζει τις εκφράσεις με τον ίδιο τρόπο που τις αντιμετωπίζουν πολλές από τις υπόλοιπες γλώσσες. Η PHP είναι μια προσανατολισμένη σε εκφράσεις (expression-oriented) γλώσσα, με την έννοια ότι σχεδόν ο,τιδήποτε αποτελεί μια έκφραση.

Στο παράδειγμα που είδαμε προηγουμένως, το *$a = 5* είναι μια έκφραση που έχει την τιμή 5. Έτσι, το να γράψουμε κάτι σαν το *$b = ($a = 5)* είναι το ίδιο με το να γράψουμε *$a = 5; $b = 5;*. Εφόσον οι εκχωρήσεις γίνονται από δεξιά προς τα αριστερά, μπορούμε να γράψουμε επίσης *$b = $a = 5*.

Ένα άλλο καλό παράδειγμα είναι οι pre- και post- increment and decrement. Οι χρήστες της PHP/FI 2 και πολλών άλλων γλωσσών ίσως να είναι εξοικειωμένοι με την γραφή variable++ και variable--. Πρόκειται για τους τελεστές *αύξησης (increment)* και *μείωσης (decrement)*.

Στην PHP/FI 2, η εντολή $a++ δεν έχει καμία τιμή καθώς δεν αποτελεί έκφραση και έτσι δεν μπορούμε να την εκχωρήσουμε ή να την χρησιμοποιήσουμε κατά κάποιον τρόπο. Η PHP βελτιώνει τις δυνατότητες για αύξηση/μείωση (increment/decrement) περιλαμβάνοντας κι αυτές τις εκφράσεις επίσης, όπως στην C.

Το *pre-increment*, που γράφεται ως *++$variable* είναι ίσο με την τιμή της μεταβλητής αυξημένης κατά ένα, δηλ. η PHP αυξάνει την τιμή της μεταβλητής πριν χρησιμοποιήσει την τιμή της.

Το *post-increment*, που γράφεται ως *$variable++* είναι ίσο με την αρχική τιμή της $variable, πριν αυτή αυξηθεί, δηλ. η PHP αυξάνει την μεταβλητή αφού έχει χρησιμοποιήσει την τιμή της.

Ένας πολύ κοινός τύπος εκφράσεων είναι οι εκφράσεις σύγκρισης (comparison expressions), οι οποίες αποτιμούνται σε 0 ή 1, που σημαίνουν FALSE ή TRUE αντίστοιχα.

Η PHP υποστηρίζει τους τελεστές σύγκρισης >, >=, ==, !=, < και <=. Αυτές οι εκφράσεις χρησιμοποιούνται συνήθως μέσα σε εκτελέσεις υπό συνθήκη (conditional execution), όπως είναι οι εντολές if.

Το τελευταίο παράδειγμα εκφράσεων που θα δούμε είναι οι συνδυασμένες εκφράσεις τελεστή και εκχώρησης. Γνωρίζουμε ήδη ότι αν θέλουμε να αυξήσουμε το $a κατά 1, μπορούμε να γράψουμε απλά $a++ ή ++$a.

Αλλά, αν θέλουμε να προσθέσουμε περισσότερο από το 1, όπως για παράδειγμα το 3; Θα μπορούσαμε να γράψουμε το $a++ πολλές φορές, αλλά αυτό δεν είναι προφανώς κάτι αποδοτικό.

Η πρόσθεση του 3 στην τρέχουσα τιμή του $a μπορεί να γραφεί σαν *$a += 3*, που σημαίνει *«πάρε την τιμή της $a, πρόσθεσε το 3 σ' αυτήν και εκχώρησε το αποτέλεσμα πίσω ξανά στην $a»*. Υπάρχουν και οι τελεστές *$a -= 5*, δηλ. αφαίρεση του 5 από την τιμή της $a, *$b *= 7*, δηλ. πολλαπλασιασμός της τιμής της $b με το 7 κ.ά.

Υπάρχει μια ακόμα περίεργη έκφραση, ο τριαδικός τελεστής υπό συνθήκη (ternary conditional operator) :

$first ? $second : $third

Αν η τιμή της πρώτης υποέκφρασης είναι true, δηλ. όχι μηδενική, τότε αποτιμάται η δεύτερη υποέκφραση και αυτό είναι και το αποτέλεσμα της έκφρασης υπό συνθήκη (conditional expression). Αλλιώς, αποτιμάται η τρίτη υποέκφραση και αυτή είναι η τιμή.

Το ακόλουθο παράδειγμα αναφέρεται στα παραπάνω :

```
function double($i) {
   return $i*2;
}
$b = $a = 5;  /* εκχωρείται η τιμή 5 στις μεταβλητές $a και $b */
$c = $a++;
/* post-increment, εκχωρείται η αρχική τιμή της $a (5) στην $c */
$e = $d = ++$b;
/* pre-increment, εκχωρείται η αυξημένη τιμή της $b (6) στις $d και $e
και σ' αυτό το σημείο, οι $d και $e είναι ίσες με 6 */
$f = double($d++);
/* εκχωρείται η διπλή τιμή της $d πριν από την αύξησή της, δηλ. 2*6 = 12 στην $f */
$g = double(++$e);
/* εκχωρείται η διπλή τιμή της $e μετά από την αύξησή της, δηλ. 2*7 = 14 στην $g */
$h = $g += 10;
/* πρώτα, η $g αυξάνεται κατά 10 και τελειώνει με την τιμή 24, η τιμή
της εκχώρησης (24) εκχωρείται μετά στην $h και η $h τελειώνει με την τιμή 24 επίσης */
```

Η PHP δεν διαθέτει έναν λογικό τύπο δεδομένων (boolean) και η αληθής τιμή των εκφράσεων στην PHP υπολογίζεται με παρόμοιο τρόπο με την Perl. Αυτό σημαίνει ότι μια μη μηδενική αριθμητική τιμή είναι TRUE και το 0 είναι FALSE.

Οι αρνητικές τιμές δεν είναι ίσες με μηδέν και έτσι θεωρούνται TRUE. Το άδειο string και το string "0" είναι FALSE και όλα τα άλλα strings είναι TRUE. Με τις μη

scalar τιμές (πίνακες και αντικείμενα), αν μια τιμή δεν περιέχει κάποιο στοιχείο θεωρείται FALSE, αλλιώς θεωρείται TRUE

Η Συνάρτηση require()

Η εντολή *require()* αντικαθιστά τον εαυτό της μ' ένα συγκεκριμένο αρχείο, όπως ακριβώς δουλεύει δηλαδή η εντολή *#include* στην C.

Κάτι σημαντικό που πρέπει να έχουμε υπόψη μας είναι ότι όταν ένα αρχείο χρησιμοποιείται σε μια από τις συναρτήσεις include() ή require(), η ανάλυση (parsing) ξεφεύγει από τον έλεγχο της PHP, πηγαίνει στον έλεγχο της HTML στην αρχή του αρχείου και επανέρχεται στον έλεγχο της PHP ξανά στο τέλος.

Γι' αυτόν τον λόγο, ο κώδικας που υπάρχει μέσα στο αρχείο και ο οποίος πρέπει να εκτελεσθεί σαν κώδικας της PHP πρέπει να περικλείεται με έγκυρα tags αρχής και τέλους της PHP.

Η require() δεν είναι ουσιαστικά μια συνάρτηση της PHP και αποτελεί περισσότερο μια δομή της γλώσσας. Υπόκειται σε μερικούς διαφορετικούς ρόλους απ' ό,τι οι συναρτήσεις. Για παράδειγμα, η require() δεν υπόκειται σε δομές ελέγχου (control structures) και ακόμη, δεν επιστρέφει κάποια τιμή.

Σ' αντίθεση με την include(), η require() θα διαβασθεί πάντα στο αρχείο (target file), ακόμη κι αν η γραμμή στην οποία βρίσκεται δεν εκτελείται ποτέ. Αν θέλουμε να συμπεριλάβουμε ένα αρχείο υπό συνθήκη, πρέπει να χρησιμοποιήσουμε τη συνάρτηση *include()*.

Η εντολή υπό συνθήκη δεν θα επηρεάσει την require(). Όμως, αν η γραμμή στην οποία βρίσκεται η require() δεν εκτελεσθεί, δεν θα εκτελεσθεί ούτε ο κώδικας που υπάρχει στο αρχείο (target file).

Παρόμοια, οι δομές βρόχου δεν επηρεάζουν τη συμπεριφορά της require(). Αν και ο κώδικας που περιέχεται στο αρχείο (target file) υπόκειται ακόμα στον βρόχο, η ίδια η require() εκτελείται μία μόνο φορά.

Αυτό σημαίνει ότι δεν μπορούμε να τοποθετήσουμε μια εντολή require() μέσα σε μια δομή βρόχου και να αναμένουμε να συμπεριλάβει τα περιεχόμενα ενός διαφορετικού αρχείου σε κάθε επανάληψη. Για να το κάνουμε αυτό, χρησιμοποιούμε μια εντολή include().

```
require( 'header.inc' );
```

Πρέπει να έχουμε υπόψη μας ότι και η include() και η require() στην ουσία τραβούν τα περιεχόμενα του αρχείου (target file) στο ίδιο το καλών αρχείο script (calling script file) και δεν καλούν το target μέσω HTTP ή με κάτι παρόμοιο.

Έτσι, όποια μεταβλητή έχει ορισθεί στην εμβέλεια στην οποία λαμβάνει χώρα η συμπερίληψη θα είναι διαθέσιμη αυτόματα μέσα στο συμπεριλαμβανόμενο αρχείο, εφόσον έχει γίνει ουσιαστικά ένα μέρος του καλούντος αρχείου.

```
require( "file.inc?varone=1&vartwo=2" ); /* Δεν θα δουλέψει */

$varone = 1;

$vartwo = 2;

require( "file.inc" );

/* Οι $varone και $vartwo θα είναι διαθέσιμες στο file.inc */
```

Η Συνάρτηση include()

Η εντολή *include()* περιλαμβάνει και αποτιμά το καθορισμένο αρχείο. Κάτι σημαντικό που πρέπει να γνωρίζουμε για το πώς λειτουργεί αυτό, είναι ότι όταν ένα αρχείο γίνεται include() ή require(), η ανάλυση ξεφεύγει από τον έλεγχο της PHP και πηγαίνει στον έλεγχο της HTML στην αρχή του αρχείου (target file) και επανέρχεται ξανά στο τέλος.

Γι' αυτόν τον λόγο, ο κώδικας που υπάρχει μέσα στο αρχείο target και ο οποίος πρέπει να εκτελεσθεί σαν κώδικας της PHP πρέπει να περικλείεται με έγκυρα tags αρχής και τέλους της PHP.

Αυτό συμβαίνει κάθε φορά που συναντάται η εντολή include(), έτσι μπορούμε να χρησιμοποιήσουμε μια εντολή include() μέσα σε μια δομή βρόχου για να συμπεριλάβουμε έναν αριθμό από διαφορετικά αρχεία.

```
$files = array ('first.inc', 'second.inc', 'third.inc');

for ($i = 0; $i < count($files); $i++) {

        include $files[$i];

}
```

Η include() διαφέρει από την require() στο ότι η εντολή include αποτιμάται κάθε φορά που συναντάται (και μόνο όταν εκτελείται), ενώ η εντολή require() αντικαθίσταται από το αρχείο required όταν συναντάται για πρώτη φορά, αν τα περιεχόμενα του αρχείου θα αποτιμηθούν ή όχι. Για παράδειγμα, αν βρίσκεται μέσα σε μια εντολή if η συνθήκη της οποίας έχει αποτιμηθεί σε false.

Επειδή η include() αποτελεί μια ειδική δομή γλώσσας, πρέπει να την περικλείσουμε σ' ένα μπλοκ εντολών αν βρίσκεται μέσα σ' ένα μπλοκ υπό συνθήκη (conditional block).

```
/* Αυτό είναι ΛΑΘΟΣ και δεν θα δουλέψει όπως αναμένεται */

if ($condition)

        include($file);
```

```
else

        include($other);

 /* Αυτό είναι ΣΩΣΤΟ */

if ($condition) {

        include($file);

} else {

        include($other);

}
```

Και στην PHP3 και στην PHP4, είναι δυνατό να εκτελέσουμε μια εντολή return μέσα σ' ένα αρχείο που έχει γίνει include(), ώστε να τερματίσουμε την επεξεργασία σ' αυτό το αρχείο και να επιστρέψουμε στο script που το κάλεσε. Υπάρχουν, όμως, μερικές διαφορές στον τρόπο που δουλεύει αυτό.

Η πρώτη είναι ότι στην PHP3, το return δεν μπορεί να εμφανισθεί μέσα σ' ένα μπλοκ εκτός κι αν είναι ένα function block, όπου το return εφαρμόζεται σ' εκείνη τη συνάρτηση και όχι σ' ολόκληρο το αρχείο. Στην PHP4, όμως, δεν υπάρχει αυτός ο περιορισμός.

Επίσης, η PHP4 μάς δίνει τη δυνατότητα να επιστρέψουμε τιμές από τα αρχεία που έχουν γίνει include(). Μπορούμε να αντιμετωπίσουμε την τιμή μιας κλήσης στην include() όπως και με μια κανονική συνάρτηση. Αυτό δημιουργεί ένα parse error στην PHP3.

Ακολουθεί ένα παράδειγμα με την include() στην PHP3 και την PHP4. Υποθέτουμε ότι το αρχείο test.inc βρίσκεται στον ίδιο κατάλογο με το κύριο αρχείο :

```
<?php

        echo "Before the return <br>\n";

        if ( 1 ) {

                return 27;

        }

        echo "After the return <br>\n";

?>
```

Υποθέτουμε ότι το κύριο αρχείο (main.html) περιέχει τα εξής :

```
<?php

        $retval = include( 'test.inc' );

        echo "File returned : '$retval'<br>\n";

?>
```

Όταν κληθεί η main.html στην PHP3, θα δημιουργηθεί ένα λάθος ανάλυσης (parse error) στη γραμμή 2 και αυτό γιατί δεν μπορούμε να πάρουμε την τιμή μιας συνάρτησης include() στην PHP3. Στην PHP4, όμως, το αποτέλεσμα θα είναι :

Before the return

File returned : '27'

Τώρα, ας υποθέσουμε ότι το main.html έχει τροποποιηθεί ώστε να περιέχει τα εξής :

```
<?php

        include( 'test.inc' );

        echo "Back in main.html<br>\n";

?>
```

Στην PHP4, η έξοδος θα είναι :

Before the return

Back in main.html

Όμως, η PHP3 θα δώσει την εξής έξοδο :

Before the return

27Back in main.html

Parse error: parse error in

/home/torben/public_html/phptest/main.html on line 5

Το παραπάνω parse error είναι ένα αποτέλεσμα του γεγονότος ότι η εντολή return περικλείεται σ' ένα non-function block μέσα στο αρχείο test.inc. Αν μετακινηθεί η εντολή return έξω από το μπλοκ, η έξοδος θα είναι η εξής :

Before the return

27Back in main.html

Κλάσσεις και Αντικείμενα

Μια *τάξη (class)* είναι μια συλλογή από μεταβλητές και από συναρτήσεις που εφαρμόζονται σ' αυτές τις μεταβλητές. Για να ορίσουμε μια τάξη χρησιμοποιούμε την εξής σύνταξη :

```
<?php

    class Cart {

        var $items;    // Τα items στο shopping cart

        // Προσθέτουμε $num προϊόντα του $artnr στο cart

        function add_item ($artnr, $num) {

            $this->items[$artnr] += $num;

        }

        // Αφαιρούμε $num προϊόντα του $artnr από το cart

        function remove_item ($artnr, $num) {

            if ($this->items[$artnr] > $num) {

                $this->items[$artnr] -= $num;

                return true;

            } else {

                return false;

            }

        }

    }

?>
```

Αυτό ορίζει μια τάξη με όνομα *Cart* η οποία αποτελείται από έναν πίνακα προϊόντων στο cart (καλάθι αγορών) και δύο συναρτήσεις για να μπορούμε να προσθέσουμε και να αφαιρέσουμε προϊόντα από το cart.

Οι τάξεις είναι τύποι και μπορούμε να δημιουργήσουμε μια μεταβλητή ενός συγκεκριμένου τύπου με τον τελεστή *new*, ως εξής :

```
$cart = new Cart;

$cart->add_item("10", 1);
```

αραπάνω κώδικας δημιουργεί ένα αντικείμενο με όνομα *$cart* από την κλάσση Cart. Η συνάρτηση add_item() αυτού του αντικειμένου καλείται για να προσθέσει ένα προϊόν με κωδικό αριθμό 10 στο cart.

κλάσσεις μπορεί να είναι επεκτάσεις (extensions) άλλων τάξεων. Η προκύπτουσα τάξη έχει όλες τις μεταβλητές και τις συναρτήσεις της βασικής τάξης και ό,τι προσθέσουμε εμείς. Αυτό μπορούμε να το κάνουμε με τη λέξη κλειδί *extends*. Δεν υποστηρίζεται η πολλαπλή κληρονομικότητα (multiple inheritance).

```
class Named_Cart extends Cart {

    var $owner;


    function set_owner ($name) {

        $this->owner = $name;

    }

}
```

Ο παραπάνω κώδικας ορίζει μια τάξη με όνομα *Named_Cart* που έχει όλες τις μεταβλητές και τις συναρτήσεις της τάξης Cart συν μια επιπλέον μεταβλητή με όνομα *$owner* και μια επιπλέον συνάρτηση με όνομα *set_owner()*. Μπορούμε τώρα να ορίσουμε και να μάθουμε τον ιδιοκτήτη (owner) ενός cart.

```
$ncart = new Named_Cart;              // Δημιουργία ενός named cart

$ncart->set_owner ("kris");           // Όνομα του ιδιοκτήτη του cart

print $ncart->owner;

// εκτύπωση του ονόματος του ιδιοκτήτη του cart

$ncart->add_item ("10", 1);

// μια συνάρτηση που την έχει κληρονομήσει από το cart
```

Μέσα στις συναρτήσεις μιας τάξης, η μεταβλητή *$this* σημαίνει το ίδιο το αντικείμενο. Μπορούμε να χρησιμοποιήσουμε τη σύνταξη *$this->something* για να έχουμε πρόσβαση σε μια οποιαδήποτε μεταβλητή ή συνάρτηση με όνομα *something* του τρέχοντος αντικειμένου.

Οι *δημιουργοί (constructors)* είναι συναρτήσεις σε μια τάξη που καλούνται αυτόματα όταν δημιουργούμε ένα νέο στιγμιότυπο (instance) μιας τάξης. Μια συνάρτηση γίνεται δημιουργός (constructor) όταν έχει το ίδιο όνομα με την τάξη.

```
class Auto_Cart extends Cart {

        function Auto_Cart () {

                $this->add_item ("10", 1);

        }

}
```

Ο παραπάνω κώδικας δημιουργεί μια τάξη με όνομα *Auto_Cart* που είναι μια επέκταση της τάξης Cart συν έναν δημιουργό (constructor) ο οποίος αρχικοποιεί το cart μ' ένα στοιχείο του προϊόντος που έχει κωδικό αριθμό 10 κάθε φορά που δημιουργείται ένα νέο Auto_Cart με τον τελεστή new. Οι δημιουργοί μπορούν επίσης να λάβουν ορίσματα και αυτά τα ορίσματα μπορεί να είναι προαιρετικά.

```
class Constructor_Cart extends Cart {

        function Constructor_Cart ($item = "10", $num = 1) {

                $this->add_item ($item, $num);

        }

}

// Shop the same old boring stuff.

$default_cart   = new Constructor_Cart;

// Shop for real ...

$different_cart = new Constructor_Cart ("20", 17);
```

Δημιουργία Εικόνων Gif

Η PHP δεν περιορίζεται στο να δημιουργεί μόνο μια έξοδο κώδικα της HTML. Μπορεί επίσης να χρησιμοποιηθεί για να δημιουργήσει αρχεία εικόνων GIF ή και ροές (streams) εικόνων GIF. Θα χρειασθεί να μεταγλωττίσουμε την PHP με τη βιβλιοθήκη GD των συναρτήσεων εικόνας για να μπορέσει να δουλέψει αυτό.

```
<?php

        Header("Content-type: image/gif");
```

```
    $string=implode($argv," ");

    $im = imagecreatefromgif("images/button1.gif");

    $orange = ImageColorAllocate($im, 220, 210, 60);

    $px = (imagesx($im)-7.5*strlen($string))/2;

    ImageString($im, 3, $px, 9, $string, $orange);

    ImageGif($im);

    ImageDestroy($im);

?>
```

Αυτό το παράδειγμα θα κληθεί από μια σελίδα μ' ένα tag σαν το εξής :

```
<img src="button.php3?text">
```

Το παραπάνω script *button.php3* λαμβάνει μετά αυτό το string "text" και το τοποθετεί στην κορυφή μιας εικόνας βάσης που είναι η "images/ button1.gif" και εξάγει την εικόνα που δημιουργείται.

Αυτό αποτελεί έναν πολύ βολικό τρόπο για να αποφύγουμε τη σχεδίαση νέων εικόνων για πλήκτρα εντολής κάθε φορά που θέλουμε να αλλάξουμε το κείμενο ενός πλήκτρου. Με τη μέθοδο αυτή δημιουργούνται δυναμικά.

Επικύρωση (Authentication) του HTTP με την PHP

Η επικύρωση (authentication) του HTTP στην PHP είναι διαθέσιμη μόνο όταν εκτελείται σαν ένα Apache module και δεν είναι συνεπώς διαθέσιμη στο CGI. Σ' ένα script της PHP σ' ένα Apache module, μπορούμε να χρησιμοποιήσουμε τη συνάρτηση Header() για να στείλουμε ένα μήνυμα "Authentication Required" στον φυλλομετρητή του χρήστη (πελάτη) για να εμφανίσει (pop up) ένα παράθυρο εισόδου Username/Password.

Αφού ο χρήστης έχει καταχωρήσει ένα username και ένα password, θα κληθεί ξανά το URL που περιέχει το PHP script με τις μεταβλητές *$PHP_AUTH_USER*, *$PHP_AUTH_PW* και $*PHP_AUTH_TYPE* που περιέχουν αντίστοιχα το user name, το password και τον τύπο επικύρωσης (authentication type).

Ένα παράδειγμα script το οποίο θα επέβαλε την επικύρωση του πελάτη (client authentication) σε μια σελίδα, είναι το εξής :

```
<?php

    if(!isset($PHP_AUTH_USER)) {
```

```
        Header("WWW-Authenticate : Basic realm=\"My Realm\"");

        Header("HTTP/1.0 401 Unauthorized");

        echo "Κείμενο που θα σταλεί αν πατηθεί το Cancel\n";

        exit;

    } else {

        echo "Hello $PHP_AUTH_USER <P>";

        echo "Καταχωρήσατε το $PHP_AUTH_PW για password <P>";

    }

?>
```

Αντί να εκτυπώσουμε απλά τα *$PHP_AUTH_USER* και *$PHP_AUTH_ PW*, θα θέλαμε πιθανώς να ελέγξουμε αν είναι έγκυρα τα username και password. Αυτό μπορεί να γίνει στέλνοντας ένα ερώτημα (query) σε μια βάση δεδομένων ή αναζητώντας τον χρήστη σ' ένα αρχείο dbm.

Για να μπορέσουμε να εμποδίσουμε κάποιον από το να γράψει ένα script που να αποκαλύπτει το password μιας σελίδας που έχει επικυρωθεί (authenticated) μέσω ενός παραδοσιακού εξωτερικού μηχανισμού, οι μεταβλητές PHP_AUTH δεν θα ορισθούν αν είναι ενεργοποιημένη η εξωτερική επικύρωση (external authentication) γι' αυτή τη συγκεκριμένη σελίδα.

Σ' αυτήν την περίπτωση, η μεταβλητή *$REMOTE_USER* μπορεί να χρησιμοποιηθεί για να αναγνωρισθεί ο εξωτερικά επικυρωμένος χρήστης.

Παρ' όλα αυτά, όμως, τα παραπάνω δεν αποτρέπουν κάποιον που ελέγχει ένα μη επικυρωμένο (non-authenticated) URL από το να κλέβει passwords από επικυρωμένα URLs στον ίδιο server.

Ακολουθεί ένα παράδειγμα επικύρωσης (authentication) HTTP που επιβάλει ένα νέο name/password.

```
<?php

    function  authenticate()  {

Header( "WWW-authenticate:  basic  realm='Test  Authentication System'");

        Header( "HTTP/1.0  401  Unauthorized");

        echo  "Πρέπει να καταχωρήσετε ένα έγκυρο login  ID και

         password για να έχετε πρόσβαση σ' αυτά τα στοιχεία\n";
```

```
        exit;
    }
    if(!isset($PHP_AUTH_USER) || ($SeenBefore == 1 &&
        !strcmp($OldAuth, $PHP_AUTH_USER)) ) {
        authenticate();
    }
    else {
        echo "Welcome: $PHP_AUTH_USER<BR>";
        echo "Old: $OldAuth";
        echo "<FORM ACTION=\"$PHP_SELF\"
            METHOD=POST>\n";
        echo "<INPUT TYPE=HIDDEN NAME=\"SeenBefore\"
            VALUE=\"1\">\n";
        echo "<INPUT TYPE=HIDDEN NAME=\"OldAuth\"
            VALUE=\"$PHP_AUTH_USER\">\n";
        echo "<INPUT TYPE=Submit
            VALUE=\"Re Authenticate\">\n";
        echo "</FORM>\n";
    }
?>
```

Τα Cookies

Η PHP υποστηρίζει καθαρά τα HTTP cookies, τα οποία είναι ένας μηχανισμός αποθήκευσης δεδομένων στον απομακρυσμένο φυλλομετρητή για να μπορούμε έτσι να παρακολουθούμε ή να αναγνωρίζουμε τους χρήστες. Μπορούμε να ορίσουμε cookies με τη συνάρτηση *setcookie()*.

Τα cookies αποτελούν μέρος της επικεφαλίδας (header) του HTTP και έτσι η συνάρτηση *setcookie()* πρέπει να κληθεί πριν σταλεί κάποια έξοδος στον φυλλομετρητή. Αυτός είναι ο ίδιος περιορισμός που έχει και η συνάρτηση *header()*.

Τα cookies που στέλνονται σε μας από τον πελάτη (client) μετατρέπονται αυτόματα σε μια μεταβλητή της PHP όπως ακριβώς συμβαίνει με τις μεθόδους GET και POST. Αν θελήσουμε να εκχωρήσουμε πολλαπλές τιμές σ' ένα μόνο cookie, προσθέτουμε τα [ ] στο όνομα του cookie.

Uploads με τη Μέθοδο POST

Η PHP μπορεί να λάβει uploads αρχείων από έναν συμβατό φυλλομετρητή με το RFC-1867, όπως είναι ο Netscape Navigator και ο Microsoft Internet Explorer. Με τη δυνατότητα αυτή μπορεί κάποιος να κάνει upload και κείμενο (text) και δυαδικά αρχεία (binary files).

Με τις συναρτήσεις για επικύρωση της PHP και χειρισμό αρχείων, έχουμε πλήρη έλεγχο για το ποιος έχει το δικαίωμα να κάνει upload και το τι πρέπει να γίνει με το αρχείο αφού έχει γίνει upload.

Μπορούμε να δημιουργήσουμε μια φόρμα ειδικά για upload αρχείων, ως εξής :

```
<FORM ENCTYPE="multipart/form-data" ACTION="_URL_"
        METHOD=POST>
    <INPUT TYPE="hidden" name="MAX_FILE_SIZE"
        value="1000">
    Send this file : <INPUT NAME="userfile" TYPE="file">
    <INPUT TYPE="submit" VALUE="Send File">
</FORM>
```

Το *_URL_* πρέπει να δείχνει σ' ένα αρχείο της PHP, ενώ το κρυμμένο πεδίο *MAX_FILE_SIZE* πρέπει να προηγείται από το πεδίο file input και η τιμή του είναι το μέγιστο αποδεκτό μέγεθος αρχείου (maximum filesize) σε bytes.

Σ' αυτό το αρχείο προορισμού, ορίζονται οι ακόλουθες μεταβλητές για ένα επιτυχημένο upload :

- *$userfile*, είναι το προσωρινό όνομα αρχείου με το οποίο αποθηκεύθηκε το αρχείο που έγινε upload στο μηχάνημα του server.
- *$userfile_name*, είναι το αρχικό όνομα του αρχείου στο σύστημα του αποστολέα.

- *$userfile_size*, είναι το μέγεθος του αρχείου που γίνεται upload σε bytes.
- *$userfile_type*, είναι ο τύπος mime του αρχείου, όπως *image/gif*.

Πρέπει να έχουμε υπόψη μας ότι το τμήμα *$userfile* των παραπάνω μεταβλητών είναι το ίδιο με το όνομα του πεδίου INPUT που έχει τύπο *TYPE=file* στη φόρμα upload, όπως *userfile* στο παραπάνω παράδειγμα.

Τα αρχεία εξ ορισμού αποθηκεύονται στον προκαθορισμένο προσωρινό κατάλογο του server (temporary directory). Αυτό μπορεί να αλλάξει ορίζοντας τη μεταβλητή περιβάλλοντος *TMPDIR* στο περιβάλλον στο οποίο εκτελείται η PHP. Αν την ορίσουμε χρησιμοποιώντας τη συνάρτηση *putenv()* μέσα από ένα script της PHP δεν θα δουλέψει.

Το script της PHP το οποίο λαμβάνει το αρχείο που γίνεται upload θα εφαρμόσει όποια λογική είναι απαραίτητη για να καθορίσει το τι πρέπει να γίνει με το αρχείο που γίνεται upload. Μπορούμε, για παράδειγμα, να χρησιμοποιήσουμε τη μεταβλητή *$file_size* για να απορρίψουμε όλα τα αρχεία που είναι είτε πάρα πολύ μικρά ή πάρα πολύ μεγάλα.

Μπορούμε να χρησιμοποιήσουμε τη μεταβλητή *$file_type* για να απορρίψουμε όλα τα αρχεία που δεν ταιριάζουν με κάποια συγκεκριμένα κριτήρια. Θα πρέπει είτε να διαγράψουμε το αρχείο από τον προσωρινό κατάλογο ή να το μετακινήσουμε κάπου αλλού ή να το μετονομάσουμε.

Uploading Πολλών Αρχείων

Μπορούμε να κάνουμε upload πολλά αρχεία ταυτόχρονα και να έχουμε τις πληροφορίες αυτόματα οργανωμένες σε πίνακες (arrays). Για να γίνει αυτό, πρέπει να χρησιμοποιήσουμε την ίδια σύνταξη υποβολής πίνακα στη φόρμα της HTML όπως κάνουμε με τις λίστες επιλογής και τα πλαίσια ελέγχου :

```
<form action="file-upload.html" method="post"

    enctype="multipart/form-data">

    Send these files : <br>

    <input name="userfile[]" type="file"><br>

    <input name="userfile[]" type="file"><br>

    <input type="submit" value="Send files">

</form>
```

Όταν υποβάλλεται η παραπάνω φόρμα, θα σχηματισθούν οι πίνακες *$userfile*, *$userfile_name* και *$userfile_size*. Ο καθένας απ' αυτούς θα είναι ένας πίνακας με αριθμητικούς δείκτες.

Για παράδειγμα, ας υποθέσουμε ότι υποβάλλονται τα αρχεία */home/test/review.html* και */home/test/xwp.out*. Σ' αυτήν την περίπτωση, το $userfile_name[0] θα περιέχει την τιμή review.html και το $userfile_name[1] θα περιέχει την τιμή xwp.out. Παρόμοια, το $userfile_size[0] θα περιέχει το μέγεθος αρχείου του review.html κοκ.

Υποστήριξη για τη Μέθοδο PUT

Η PHP παρέχει υποστήριξη για τη μέθοδο PUT του HTTP που χρησιμοποιείται από πελάτες (clients) όπως ο Netscape Composer και ο W3C Amaya. Οι αιτήσεις (requests) της PUT είναι πολύ απλούστερες από το upload ενός αρχείου και μοιάζουν ως εξής :

PUT /path/filename.html HTTP/1.1

Αυτό σημαίνει συνήθως ότι ο απομακρυσμένος πελάτης θα ήθελε να αποθηκεύσει τα περιεχόμενα ως εξής : /path/filename.html. Δεν αποτελεί προφανώς μια καλή ιδέα για το Apache ή την PHP να επιτρέπουν στον οποιονδήποτε να επικαλύπτει αρχεία στο δικό μας web tree.

Έτσι, για να χειριστούμε μια τέτοια αίτηση (request) πρέπει πρώτα να πούμε στον web server ότι θέλουμε ένα συγκεκριμένο script της PHP να χειριστεί την αίτηση. Στο Apache το κάνουμε αυτό με την οδηγία (directive) *Script*, που μπορεί να τοποθετηθεί οπουδήποτε στο αρχείο σύνθεσης (configuration file) του Apache :

Script PUT /put.php3

Το παραπάνω λέει στο Apache να στείλει όλες τις αιτήσεις PUT για τα URIs που ταιριάζουν στο περιεχόμενο με το script put.php3. Αυτό προϋποθέτει, φυσικά, ότι έχουμε ενεργοποιήσει την PHP για την επέκταση .php3 και ότι είναι ενεργή η PHP.

Μέσα στο αρχείο put.php3, θα γράψουμε κάτι σαν το εξής :

```
<? copy($PHP_UPLOADED_FILE_NAME,
        $DOCUMENT_ROOT.$REQUEST_URI); ?>
```

Το παραπάνω θα αντιγράψει το αρχείο στην τοποθεσία που ζητείται από τον απομακρυσμένο πελάτη. Θα θέλουμε πιθανώς να κάνουμε κάποιους ελέγχους ή/και να επικυρώσουμε τον χρήστη πριν να λάβει χώρα αυτή η αντιγραφή του αρχείου.

Το μόνο τρυκ εδώ είναι ότι όταν η PHP βλέπει μια αίτηση με την μέθοδο PUT, αποθηκεύει το αρχείο που γίνεται upload σ' ένα προσωρινό αρχείο όπως ακριβώς εκείνα που χειρίζονται από την μέθοδο POST.

Όταν τελειώσει η αίτηση (request), διαγράφεται αυτό το προσωρινό αρχείο. Έτσι, το script της PHP που θα γράψουμε για την μέθοδο PUT πρέπει να αντιγράψει κάπου αλλού αυτό το αρχείο.

Το όνομα αυτού του προσωρινού αρχείου βρίσκεται στην μεταβλητή *$PHP_PUT_FILENAME* και μπορούμε να βρούμε το προτεινόμενο όνομα για το αρχείο προορισμού στην *$REQUEST_URI*, που είναι αυτό που καθόρισε ο απομακρυσμένος χρήστης, αλλά δεν είμαστε υποχρεωμένοι να το χρησιμοποιήσουμε. Θα μπορούσαμε, για παράδειγμα, να αντιγράψουμε όλα τα αρχεία που γίνονται upload σ' έναν ειδικό κατάλογο *uploads*.

Χρήση Απομακρυσμένων Αρχείων

Μπορούμε να ανοίξουμε ένα αρχείο σ' έναν απομακρυσμένο web server, αν αναλύσουμε (parse) την έξοδο για τα δεδομένα που θέλουμε και μετά να χρησιμοποιήσουμε αυτά τα δεδομένα σ' ένα ερώτημα μιας βάσης δεδομένων (database query) ή να τα εξάγουμε στο δικό μας website.

Το επόμενο παράδειγμα εμφανίζει τον τίτλο (title) μιας απομακρυσμένης σελίδας.

```
<?php

        $file = fopen("http://www.php.net/", "r");

        if (!$file) {

                echo "<p>Unable to open remote file.\n";

                exit;

        }

        while (!feof($file)) {

                $line = fgets($file, 1024);

                /* This only works if the title and its tags are on one line. */

                if (eregi("<title>(.*)</title>", $line, $out)) {

                        $title = $out[1];

                        break;

                }

        }

        fclose($file);

?>
```

Το επόμενο παράδειγμα αποθηκεύει δεδομένα σ’ έναν απομακρυσμένο server.

```
<?php
        $file = fopen("ftp://ftp.php.net/incoming/outputfile", "w");
        if (!$file) {
                echo "<p>Unable to open remote file for writing.\n";
                exit;
        }
        /* Εδώ καταχωρούμε τα δεδομένα */
        fputs($file, "$HTTP_USER_AGENT\n");
        fclose($file);
?>
```

ΠΑΡΑΡΤΗΜΑ Α

ΕΓΚΑΤΑΣΤΑΣΕΙΣ ΕΡΓΑΛΕΙΩΝ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ ΓΙΑ ΤΗΝ ΑΝΑΠΤΥΞΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Παρακάτω θα γίνει μια παρουσίαση της εγκατάστασης των εργαλείων που χρησιμοποιήθηκαν για την υλοποιήσει της εφαρμογής τα οποία είναι:

- Η γλώσσα προγραμματισμού διαδικτιακών εφαρμογών PHP
- Το σύστημα διαχείρισης βάσεων δεδομένων MySQL
- Ο Apache Web Server
- Ένα πακέτο λογισμικού που μπορεί να βοηθήσει κάποιον ώστε να στήσει γρήγορα και εύκολα τα εργαλεία PHP- MySQL- και Apache Web Server με το όνομα PHP DEV.

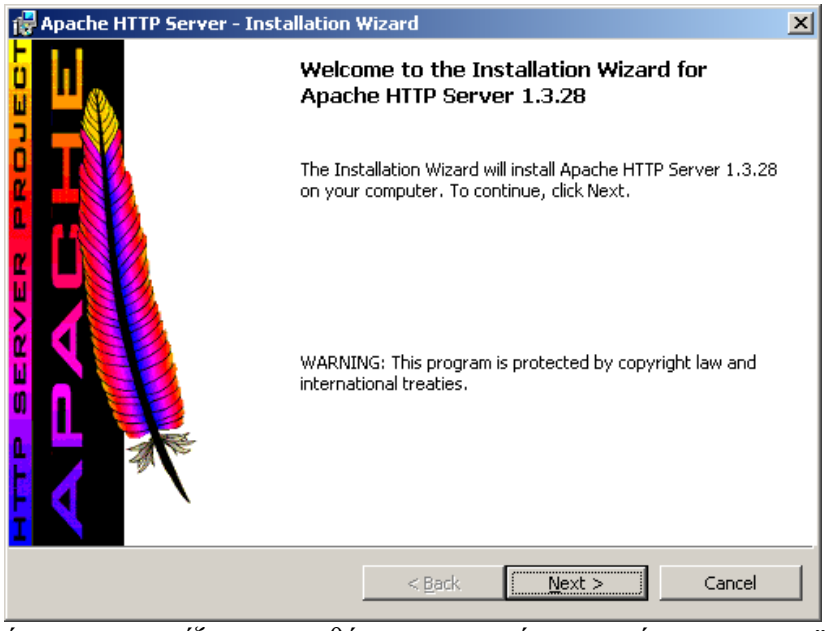
Εγκατάσταση Apache HTTP Web Server

Αφού τοποθετήσουμε το cd που περιέχει το αρχείο ***Apache HTTP Web Server.exe*** κάνουμε διπλό κλικ πάνω του και μας εμφανίζετε ο Installation Wizard ο οποίος και θα μας βοηθήσει να εγκαταστήσουμε στο σύστημα μας το οποίο σαφώς και χρησιμοποιεί λειτουργικό σύστημα Windows τον Apache HTTP Web Server.

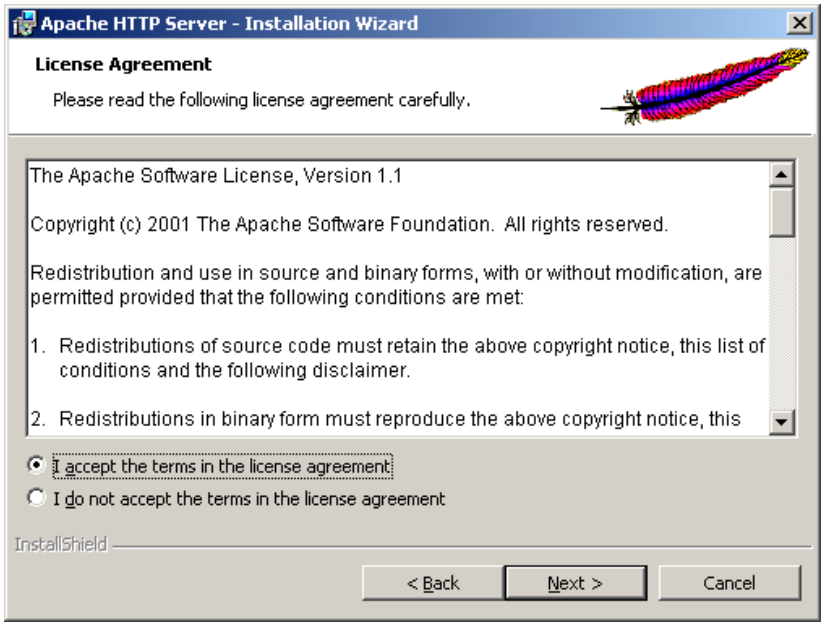
Τα βήματα που ακολουθούμε κατά την εγκατάσταση φαίνονται στις παρακάτω εικόνες.

Αυτή είναι η πρώτη εικόνα που μας εμφανίζετε μόλις κάνουμε διπλό κλικ στο αρχείο

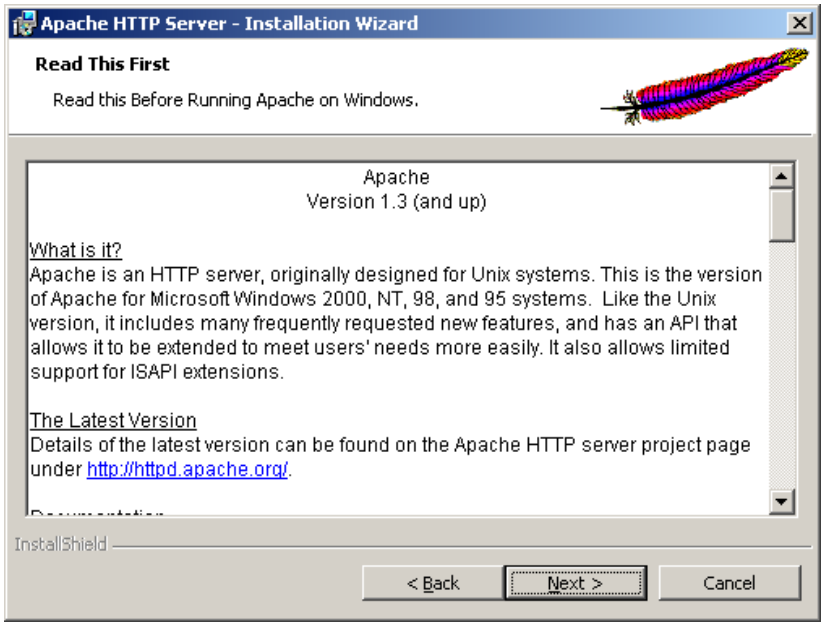
Apache HTTP Web Server.exe η οθόνη αυτή είναι ουσιαστικά μια οθόνη εισαγωγική και εμείς απλός πατάμε το κουμπί Next.



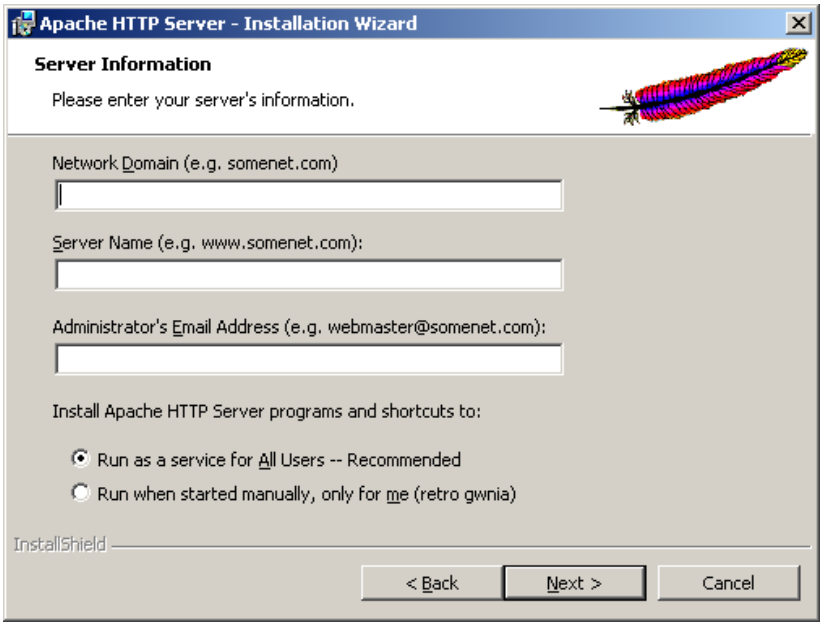
Στη συνεχεία μας εμφανίζετε μια οθόνη με τους όρους χρήσης του προϊόντος τους οποίους και αποδεχόμαστε πατώντας την επιλογή I accert the terms in the licence agreement και έπειτα πατάμε το κουμπί Next.



Στη συνεχεία μας εμφανίζετε μια οθόνη που μας λέει κάποιες πληροφορίες για τον Apache HTTP Web Server γενικά αλλά και κάποιες πληροφορίες για τη συγκεκριμένη έκδοση του την προσπερνάμε και αυτήν πατώντας το κουμπί Next.

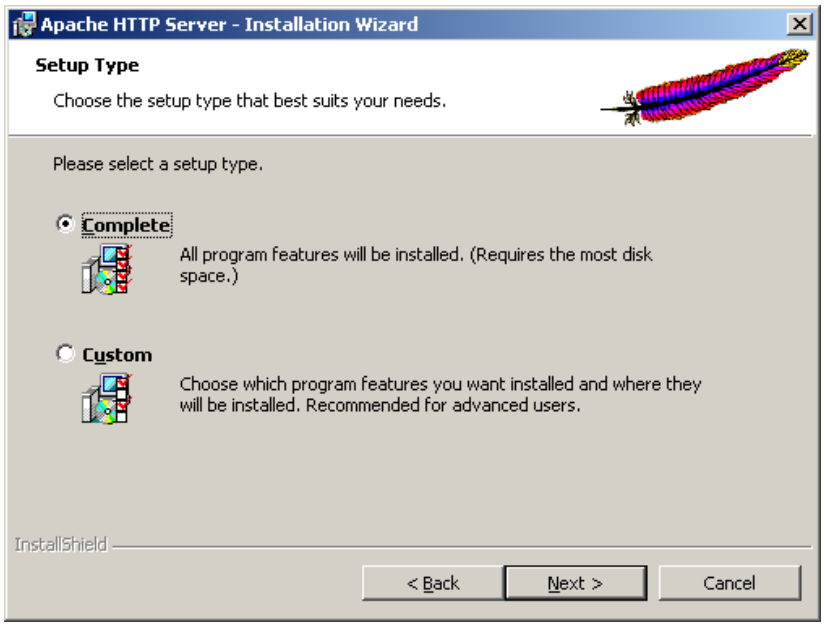


Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία υποχρεωτικά πρέπει να γράψουμε ένα Domain Name, ένα Server Name και ένα email το οποίο είναι το email του Administrator του συστήματος.
Μετά πρέπει να αποφασίσουμε αν θέλουμε η εγκατάσταση του Server να εξυπηρετεί όλους τους χρηστές η μόνο τον Administrator.
Μετά πατάμε το κουμπί Next.

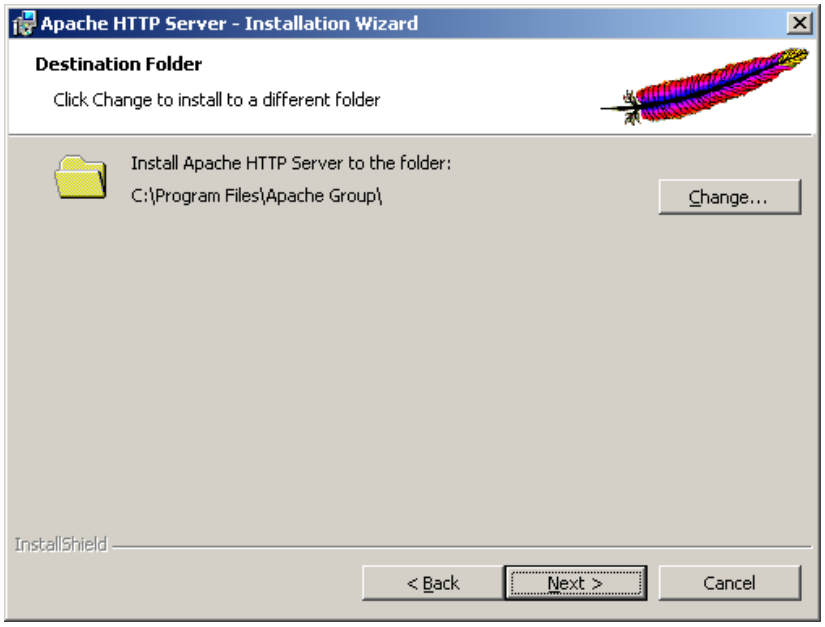


Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία πρέπει να επιλέξουμε αν η εγκατάσταση θα είναι με όλα τα στοιχεία του Apache ή αν εμείς θα επιλέξουμε ποια

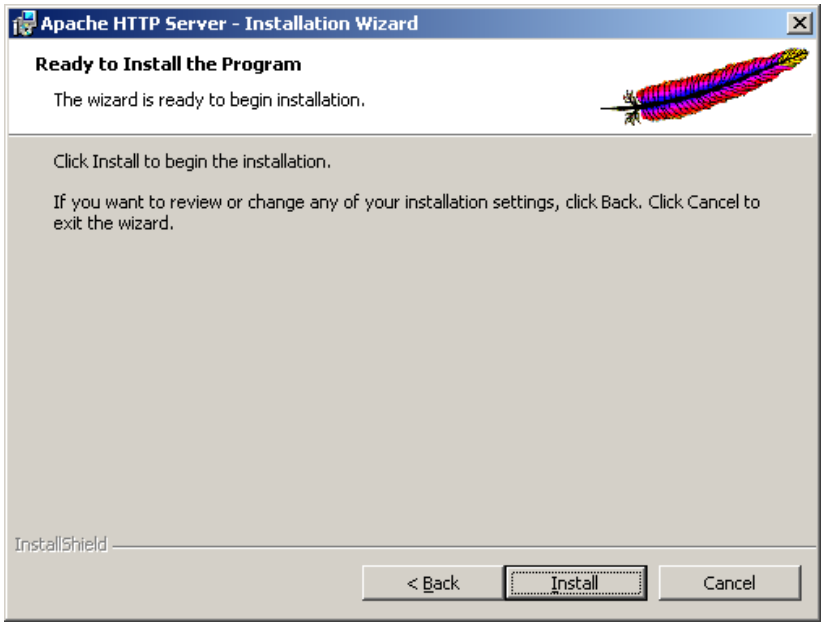
στοιχεία του Apache θα εγκαταστήσουμε και ποια όχι ,επιλέγουμε συνήθως το complete. Μετά πατάμε το κουμπί Next.



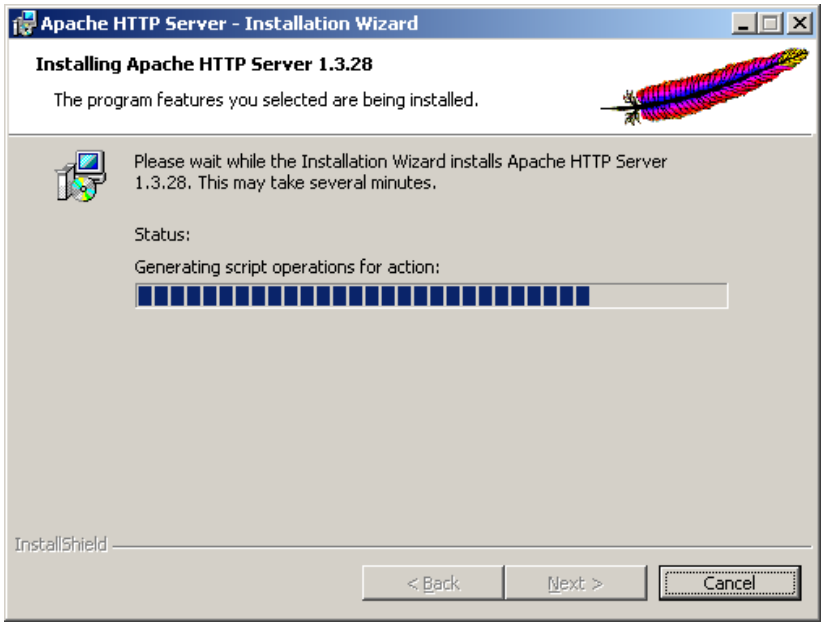
Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία θέτουμε τη διαδρομή στο δίσκο μας όπου θα γίνει η εγκατάσταση και πατάμε το κουμπί Next.



Φτάνοντας σε αυτή την οθόνη πλέον είμαστε έτοιμοι για να κάνουμε την εγκατάσταση έχοντας επιλέξει τις κατάλληλες ρυθμίσεις προηγουμένως , γι'αυτό τώρα πατάμε το κουμπί Install.

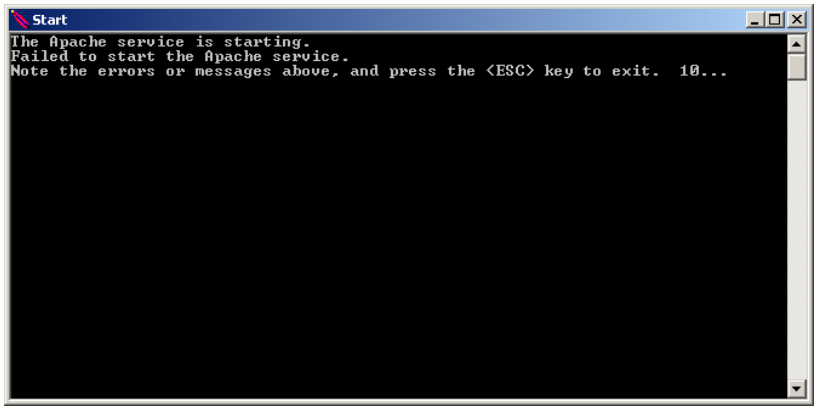


Σ'αυτή την οθόνη παρακολουθούμε την πρόοδο της εγκατάστασης .



Αφού τελειώσει η εγκατάσταση ακολουθούμε τα εξής βήματα πηγαίνουμε στην επιφάνεια εργασίας των Windows και ακολουθούμε τα εξής βήματα ΕΝΑΡΞΗ-ΠΡΟΓΡΑΜΜΑΤΑ-Apache Group-Apache-Start.

Αυτό το κάνουμε για να ξεκινήσουμε τον **Apache HTTP Web Server** σαν υπηρεσία του συστήματος μας.
Αν όλα πήγαν καλά θα δούμε το εξής DOS παράθυρο.



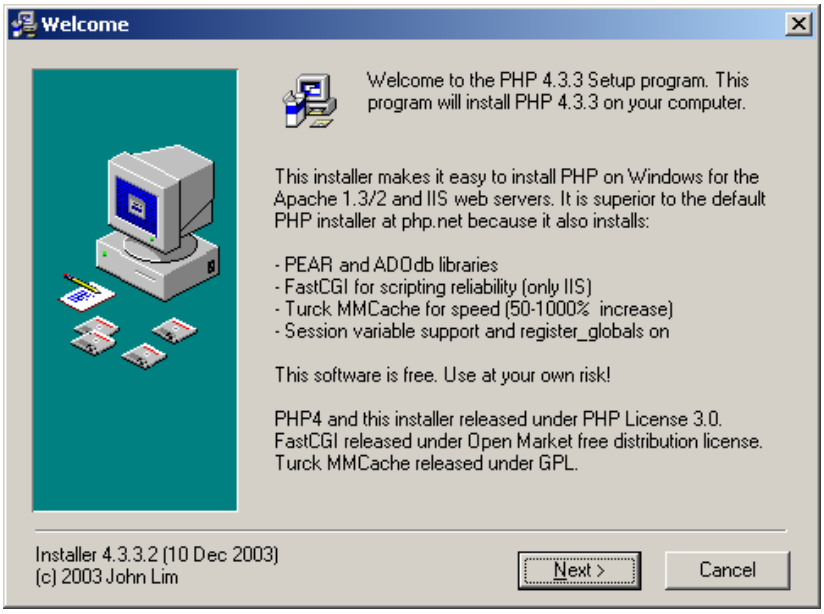
Εγκατάσταση PHP4.3.3

Αφού τοποθετήσουμε το cd που περιέχει το αρχείο **PHP 4.3.3.*exe*** κάνουμε διπλό κλικ πάνω του και μας εμφανίζετε ο Installation Wizard ο οποίος και θα μας βοηθήσει να εγκαταστήσουμε στο σύστημα μας το οποίο σαφώς και χρησιμοποιεί λειτουργικό σύστημα Windows την PHP 4.3.3.

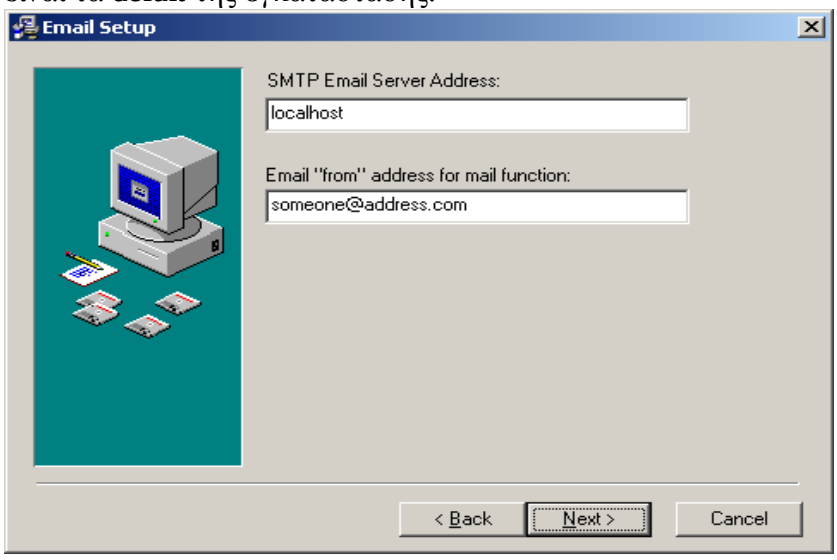
Τα βήματα που ακολουθούμε κατά την εγκατάσταση φαίνονται στις παρακάτω εικόνες.

Αυτή είναι η πρώτη εικόνα που μας εμφανίζετε μόλις κάνουμε διπλό κλικ στο αρχείο

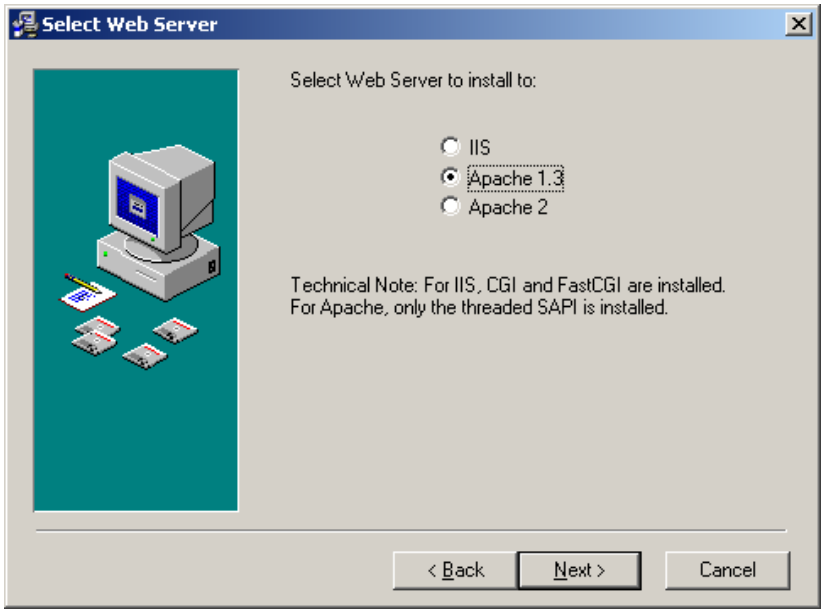
PHP 4.3.3.*exe* η οθόνη αυτή είναι ουσιαστικά μια οθόνη εισαγωγική και εμείς απλός πατάμε το κουμπί Next.



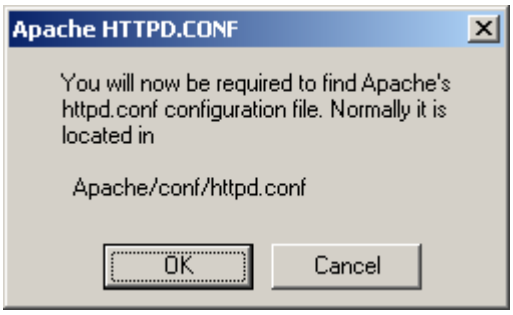
Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία πρέπει να βάλουμε το όνομα του Mail Server , αν χρησιμοποιούμε και τη διεύθυνση του τα ονόματα που βλέπετε παρακάτω είναι τα defult της εγκατάστασης.



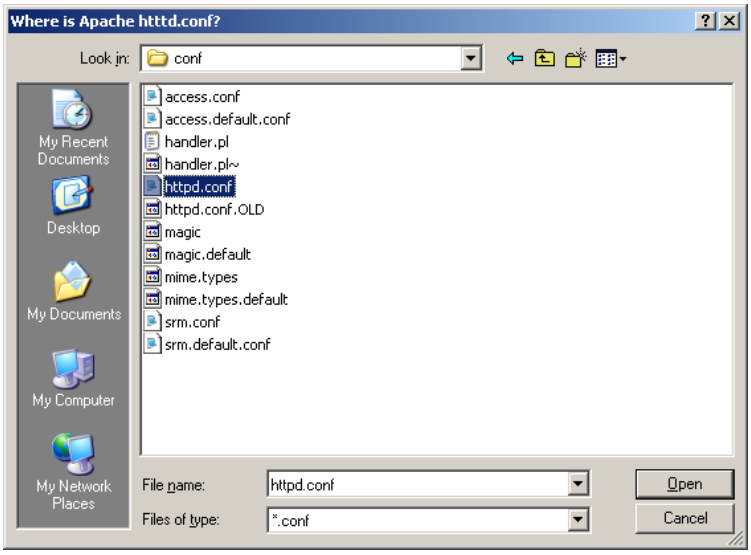
Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία πρέπει να επιλέξουμε ποιον Web Server χρησιμοποιούμε αφού κάνουμε την επιλογή μας πατάμε το κουμπί Next.



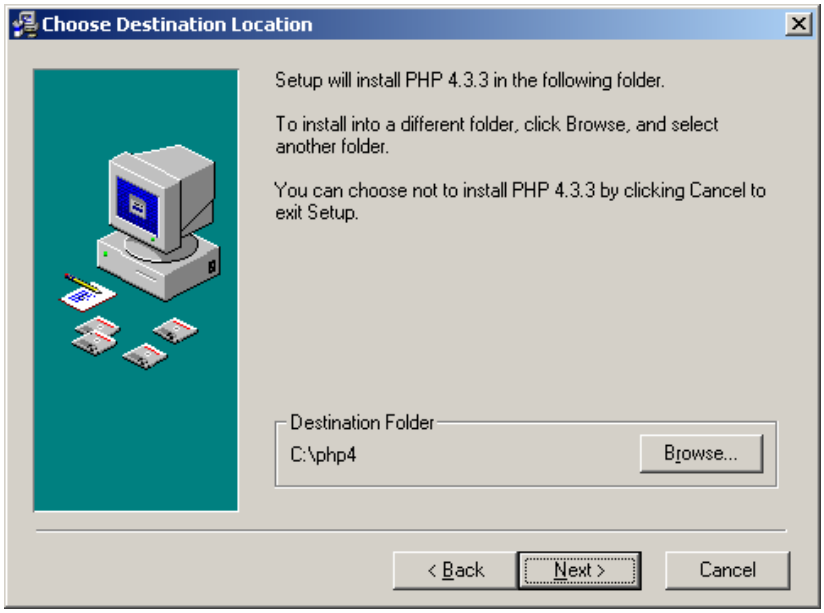
Αφού επιλέξαμε σαν Web Server μας τον Apache 1.3 ,ο Wizard μας ζητά να του δείξουμε που είναι το αρχείο **http.conf** δηλαδή που είναι το αρχείο με την παραμετροποίηση του Apache 1.3 ώστε να ξέρει η PHP που να πάει και να προσθέσει ή να αντλήσει δικές της πληροφορίες στον Server.



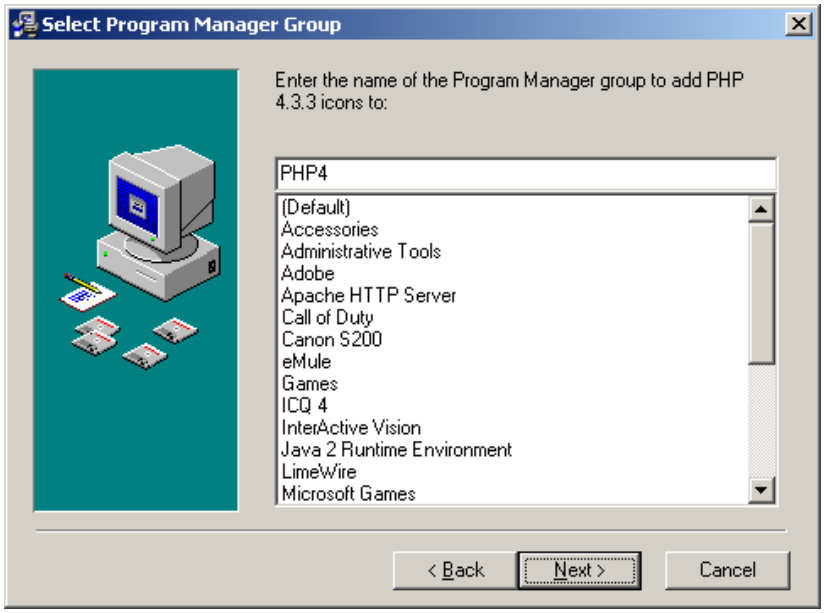
Έτσι ψάχνουμε και βρίσκουμε το αρχείο
.



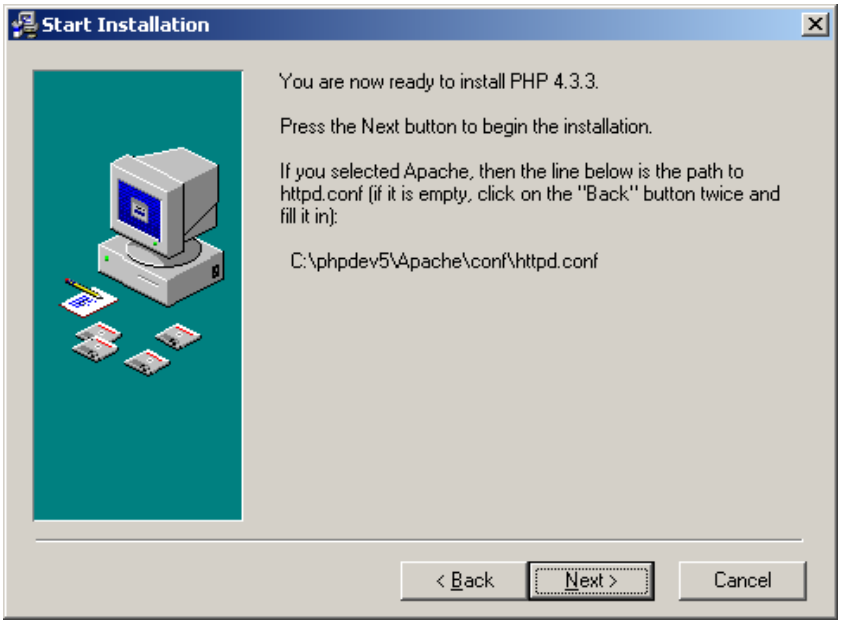
Αφού βρω το αρχείο τότε μόνο ο Wizard μας επιτρέπει να ορίσουμε τη διαδρομή στο δίσκο που θα αποθηκευτούν τα αρχεία της PHP.
Διαλέγουμε τη διαδρομή και πατάμε το κουμπί Next.



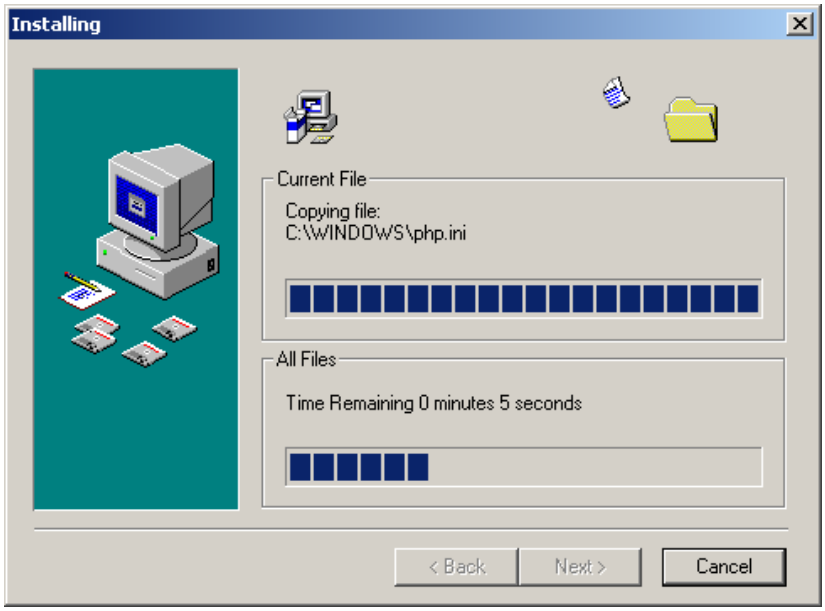
Μετά μας λέει πως θέλουμε να ονομάσουμε το φάκελο που θα περιέχονται τα αρχεία της PHP , διαλέγουμε το όνομα που θέλουμε και πατάμε το κουμπί Next.



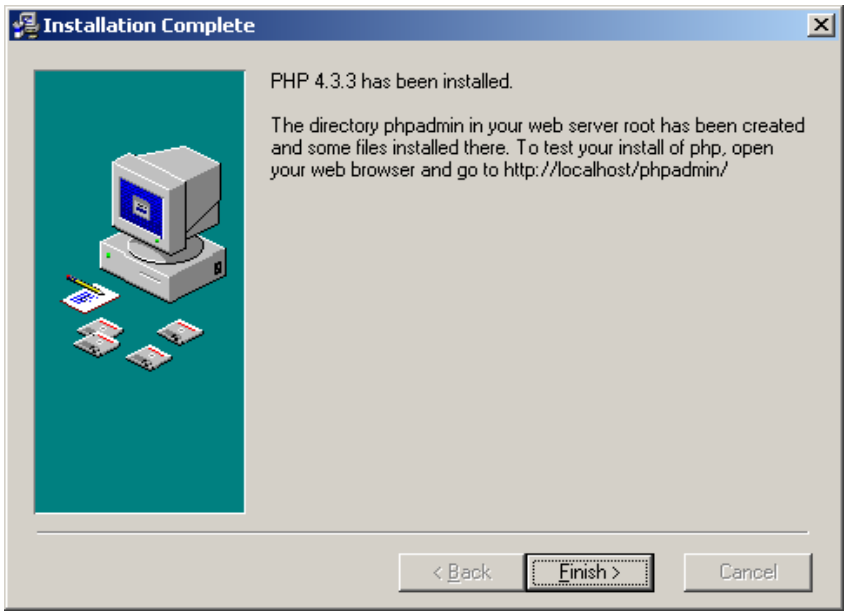
Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία πρέπει απλός να πατήσουμε το κουμπί Install για να αρχίσει η εγκατάσταση.



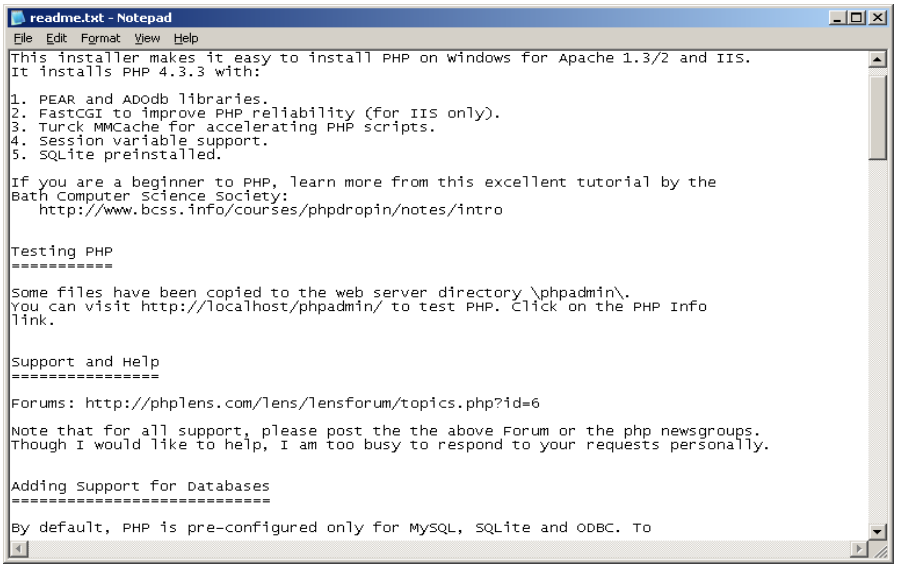
Στην επόμενη οθόνη παρακολουθούμε την πρόοδο της εγκατάστασης .



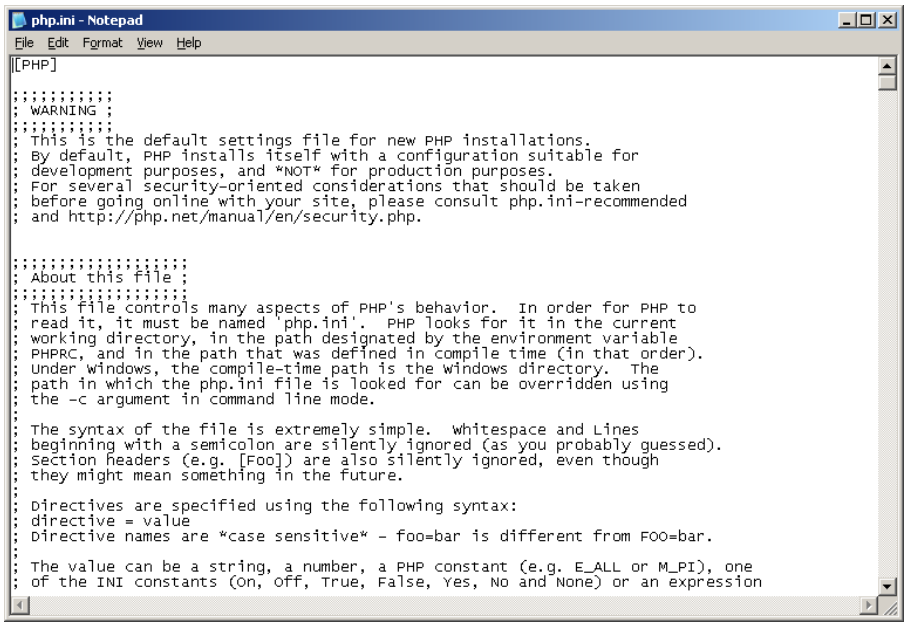
Αν έχουν πάει όλα καλά μας εμφανίζετε η παρακάτω οθόνη



Μετά μας εμφανίζετε το αρχείο redme.txt που μας αναφέρει όλες τις λεπτομέρειες της έκδοσης αυτής της PHP.



Και τέλος το αρχείο php.ini το οποίο ουσιαστικά είναι το αρχείο παραμετροποίησης της PHP.



Εγκατάσταση του Συστήματος Διαχείρισης Βάσεων Δεδομένων MySQL

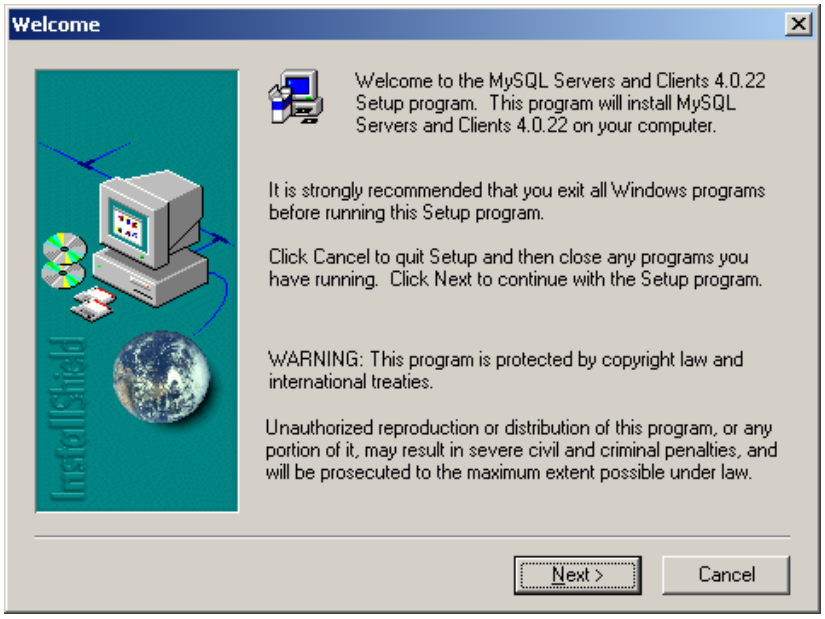
Αφού τοποθετήσουμε το cd που περιέχει το αρχείο ***MySQL.exe*** κάνουμε διπλό κλικ πάνω του και μας εμφανίζετε ο Installation Wizard ο οποίος και θα μας βοηθήσει να εγκαταστήσουμε στο σύστημα μας το οποίο σαφώς και χρησιμοποιεί λειτουργικό σύστημα Windows το Σύστημα Διαχείρισης Βάσεων Δεδομένων MySQL Server.

.

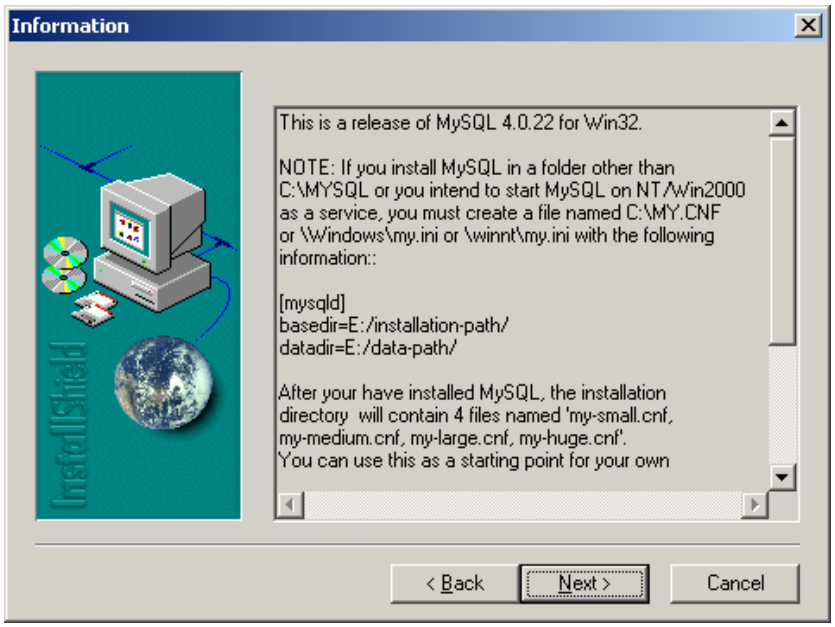
Τα βήματα που ακολουθούμε κατά την εγκατάσταση φαίνονται στις παρακάτω εικόνες.

Αυτή είναι η πρώτη εικόνα που μας εμφανίζετε μόλις κάνουμε διπλό κλικ στο αρχείο

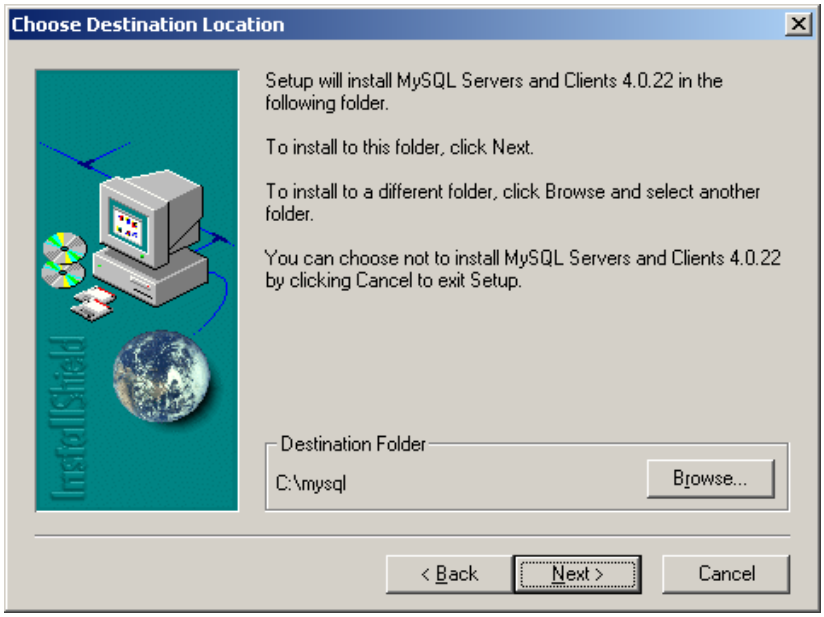
MySQL.exe η οθόνη αυτή είναι ουσιαστικά μια οθόνη εισαγωγική και εμείς απλός πατάμε το κουμπί Next.



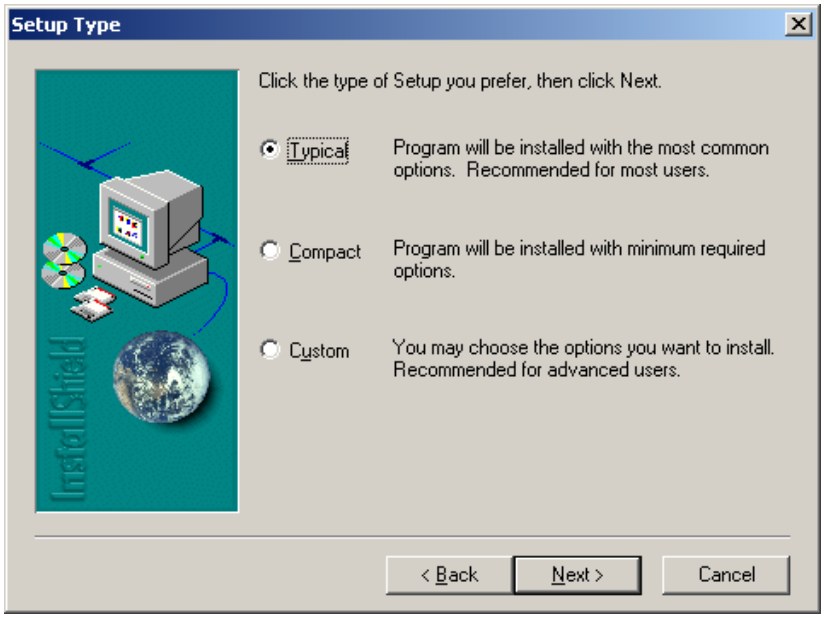
Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία βλέπουμε διάφορες πληροφορίες για την έκδοση του MySQL Server που θέλουμε να εγκαταστήσουμε.



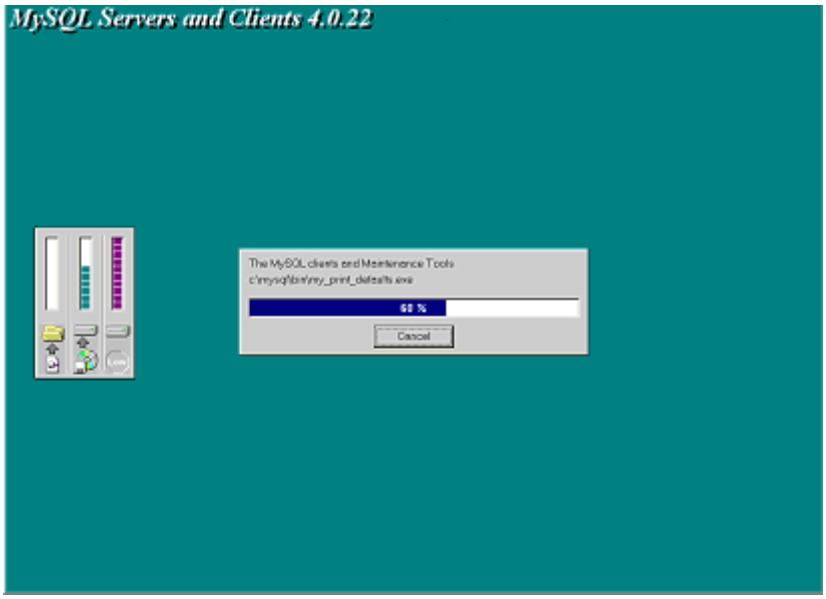
Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία θέτουμε τη διαδρομή στο δίσκο μας όπου θα γίνει η εγκατάσταση και πατάμε το κουμπί Next.



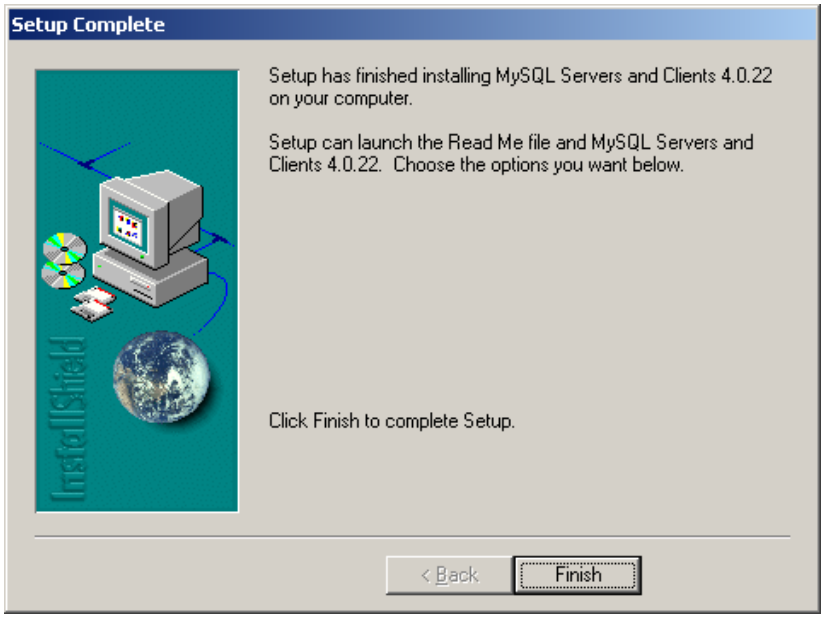
Στη συνεχεία μας εμφανίζετε μια οθόνη στην οποία πρέπει να επιλέξουμε αν η εγκατάσταση θα είναι με όλα τα τυπικά στοιχεία του MySQL Server αν θα εγκατασταθεί ο MySQL Server με ελάχιστες απαιτήσεις ,ή αν εμείς θα επιλέξουμε ποια στοιχεία του MySQL Server θα εγκαταστήσουμε και ποια όχι ,επιλέγουμε συνήθως το Typical. Μετά πατάμε το κουμπί Next.



Στην επόμενη οθόνη παρακολουθούμε την πρόοδο της εγκατάστασης .

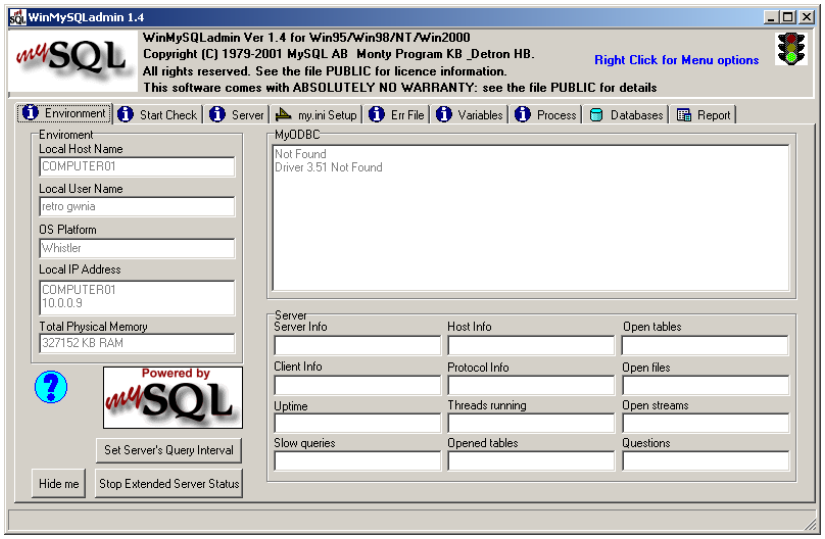


Αν έχουν πάει όλα καλά μας εμφανίζετε η παρακάτω οθόνη.

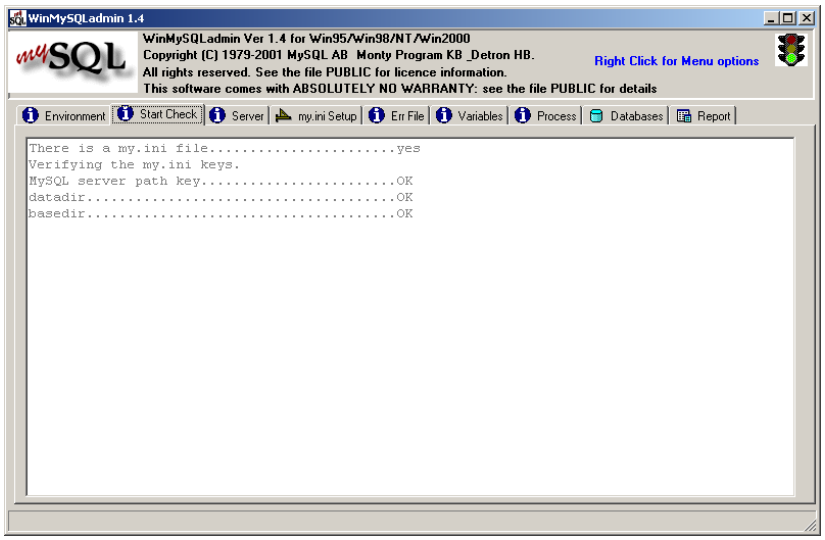


Αφού τώρα καταφέρομε να εγκαταστήσουμε τον MySQL Server πρέπει να τον ξεκινήσουμε κιόλας , για να γίνει αυτό πηγαίνουμε στο φάκελο που έχουμε κάνει την εγκατάσταση, στη συγκεκριμένη περίπτωση στον C:\mysql πάμε στον υποφάκελο \bin και κάνουμε διπλό κλικ στο αρχείο MysqlStart.

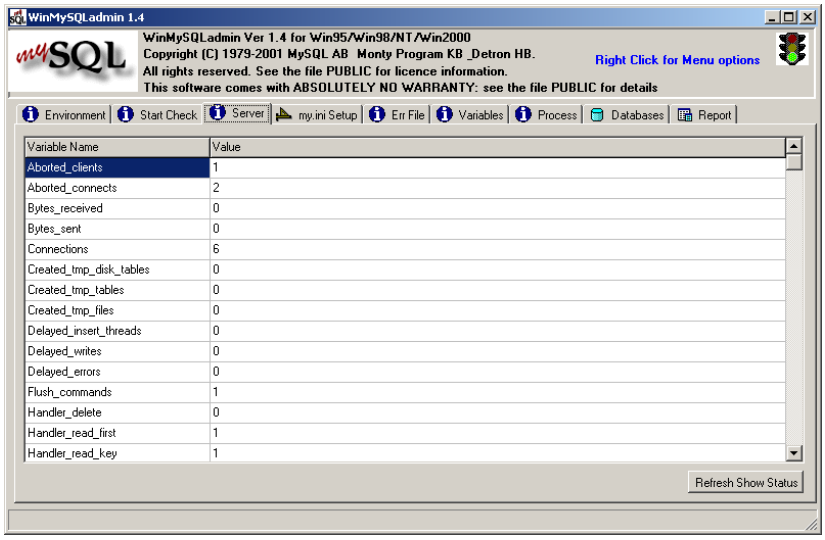
Ένα από τα σπουδαιότερα εργαλεία που μας παρέχει η MySQL για την διαχείριση των βάσεων δεδομένων μας είναι μια εφαρμογή που εγκαθίσταται μαζί με τον MySQL Server και λέγετε WinMySQLadmin, στις παρακάτω εικόνες κάνουμε μια μικρή παρουσίαση του συγκεκριμένου εργαλείου.
Για να χρησιμοποιήσουμε το συγκεκριμένο εργαλείο , πρέπει πρώτα να το ξεκινήσουμε , για να γίνει αυτό πηγαίνουμε στο φάκελο που έχουμε κάνει την εγκατάσταση, στη συγκεκριμένη περίπτωση στον C:\mysql πάμε στον υποφάκελο \bin και κάνουμε διπλό κλικ στο αρχείο WinMySQLadmin.
Η πρώτη εικόνα ,πλήκτρο Environment του WinMySQLadmin μας δίνει γενικές πληροφορίες για το σύστημα μας καθώς και για τους Server που έχουμε εγκατεστημένους.



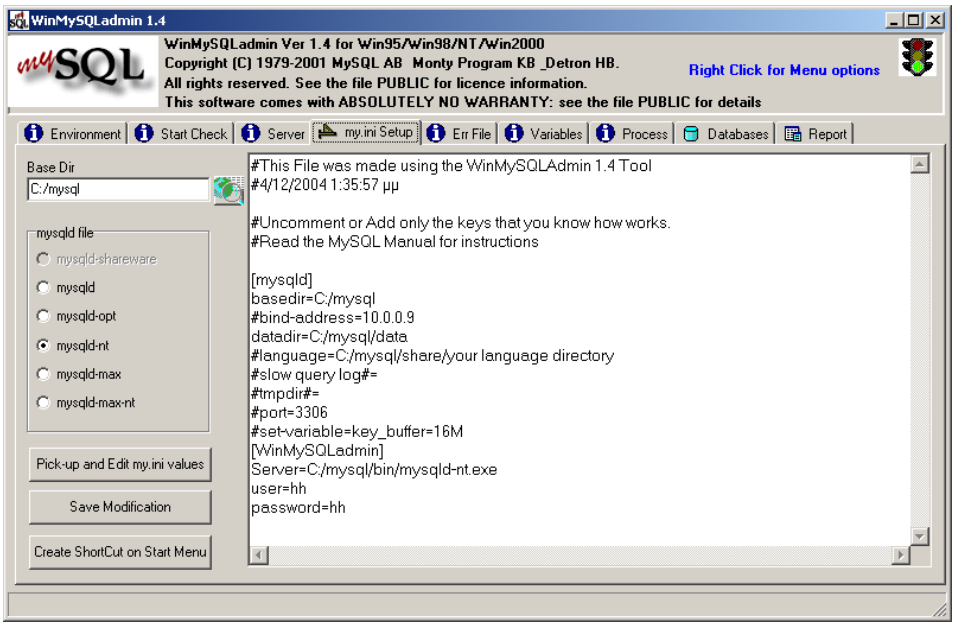
Σ'αυτή την εικόνα ,πλήκτρο Start Check γίνετε ένας έλεγχος του MySQL Server και περνούμε κάποια μηνύματα όπως π.χ για το αρχείο **.ini** του MySQL Server και αλλά.



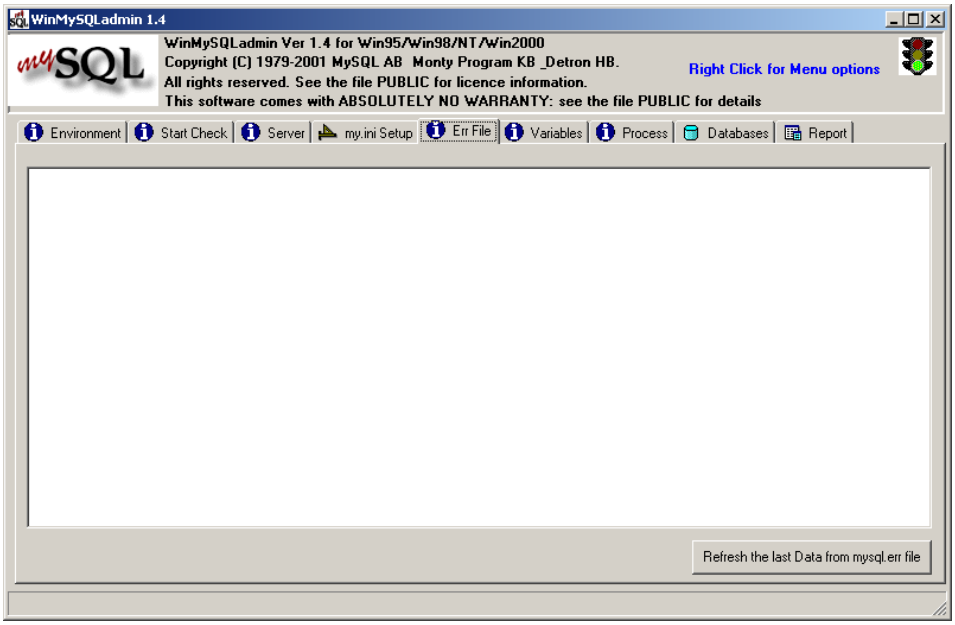
Σ'αυτή την εικόνα ,πλήκτρο Server παίρνουμε πληροφορίες για κάποιες μεταβλητές του MySQL Server και σαφώς μπορούμε να τις τροποποιήσουμε .



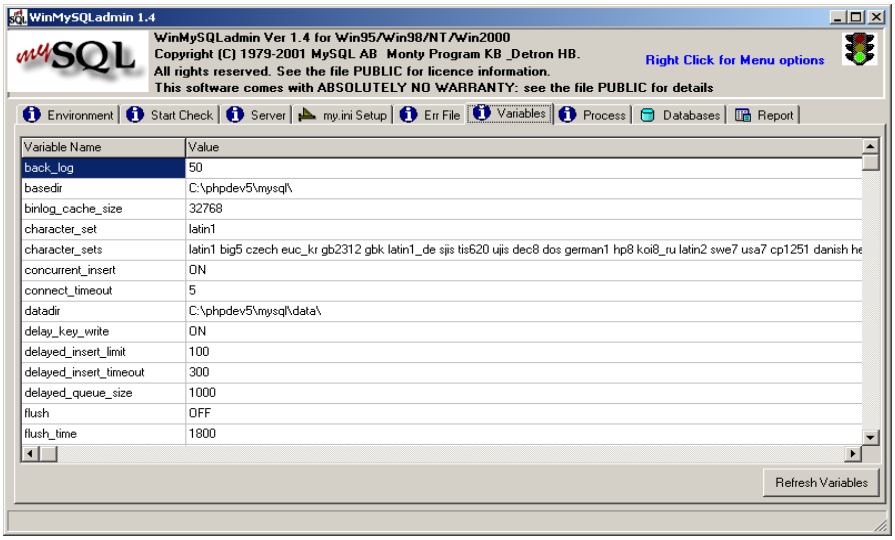
Σ'αυτή την εικόνα, πλήκτρο **my.iniSetup** ουσιαστικά τροποποιούμε το αρχείο παραμετροποίησης **.ini** του MySQL Server



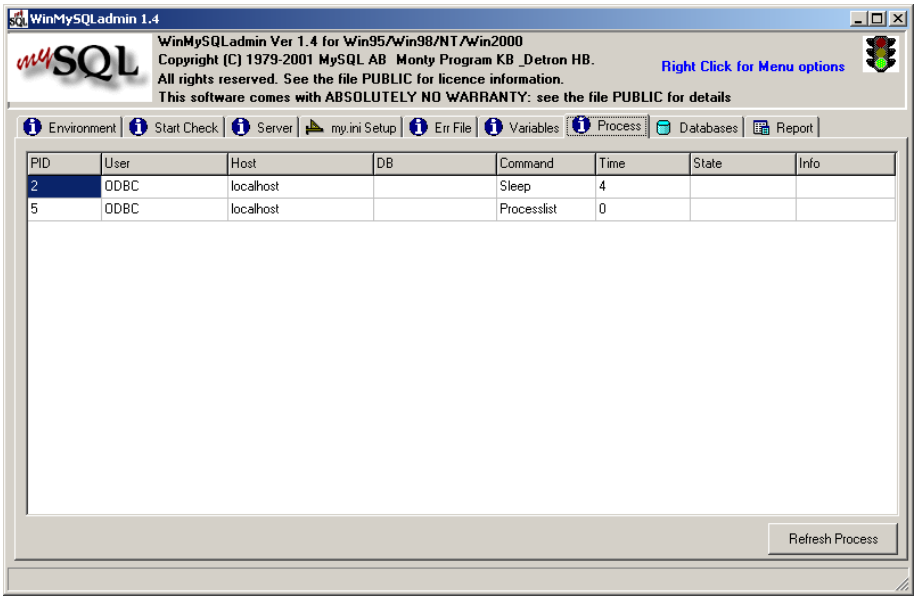
Σ'αυτή την εικόνα, πλήκτρο Err File , εμφανίζονται διάφορα λάθη τα οποία γίνονται και στα οποία εμπλέκεται ο MySQL Server.



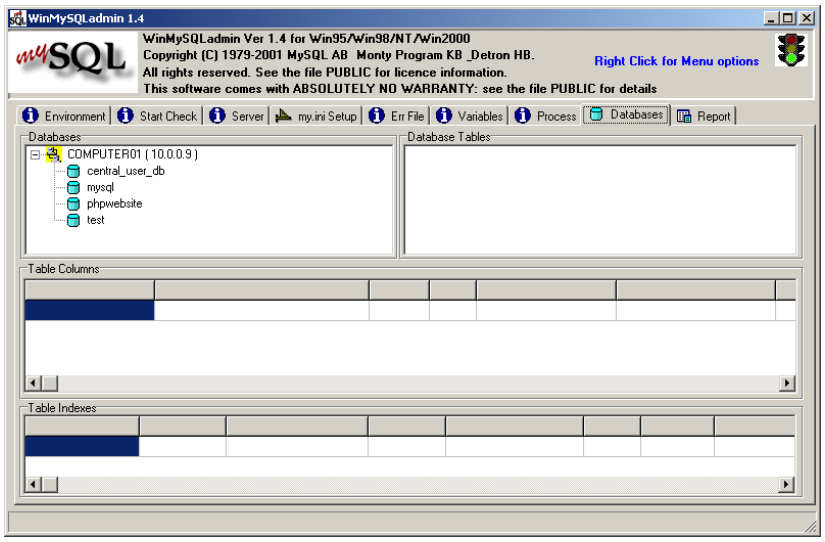
Σ'αυτή την εικόνα ,πλήκτρο Variables παίρνουμε πληροφορίες για κάποιες μεταβλητές του MySQL Server και σαφώς μπορούμε να τις τροποποιήσουμε .



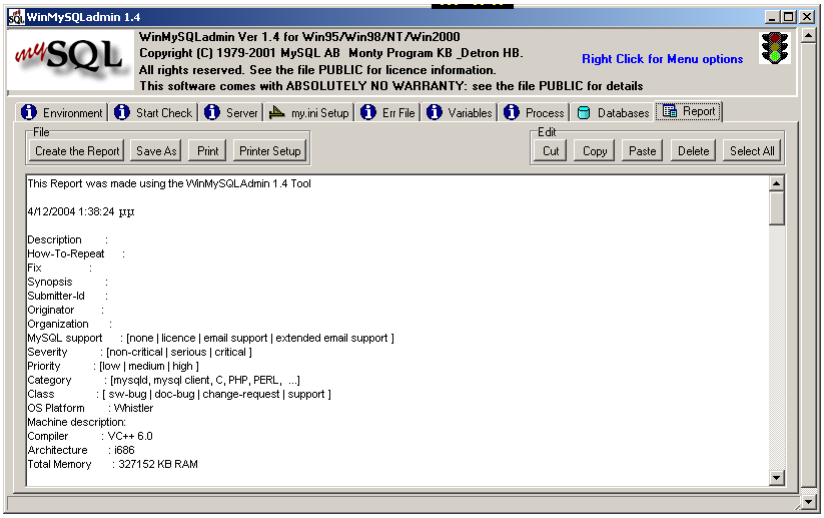
Σ'αυτή την εικόνα ,πλήκτρο Process παίρνουμε πληροφορίες για την εμπλοκή του MySQL Server στη λειτουργία διαφόρων άλλων διαδικασιών.



Σ’αυτή την εικόνα ,πλήκτρο Databases βλέπουμε τις διάφορες βάσεις δεδομένων τις όποιες έχουμε δημιουργήσει στον MySQL Server και σαφέστατα μπορούμε να τις τροποποιήσουμε.



Τέλος σ’αυτή την εικόνα ,πλήκτρο Report ο WinMySQLadmin δημιουργεί μια έκθεση με όλες τις παραμέτρους και τις πληροφορίες που χρειάζονται σε έναν Administrator για την κατάσταση του MySQL Server.



Εγκατάσταση του λογισμικού αυτόματης εγκατάστασης των PHP,MySQL KAI Apache phpdev5beta3

Το πακέτο λογισμικού ***phpdev5beta3*** είναι ένα αυτοματοποιημένο πακέτο εγκατάστασης και αυτόματης παραμετροποίησης των PHP,MySQL και Apache για χρηστές όχι και τόσο έμπειρους με αυτά τα εργαλεία, ουσιαστικά αυτό που έκανε το PHP Group που δημιούργησε το ***phpdev5beta3*** είναι να ενοποιήσει της τεχνολογίες PHP,MySQL και Apache κάτω από το περιβάλλον AnalogX το οποίο είναι ένα γραφικό περιβάλλον διαχειρίσεις των παραπάνω τεχνολογιών.

Αφού τοποθετήσουμε το cd που περιέχει το αρχείο ***phpdev5beta3.exe*** κάνουμε διπλό κλικ πάνω του και μας εμφανίζετε ο Installation Wizard ο οποίος και θα μας βοηθήσει να εγκαταστήσουμε στο σύστημα μας το οποίο σαφώς και χρησιμοποιεί λειτουργικό σύστημα Windows το Σύστημα Διαχείρισης Βάσεων Δεδομένων MySQL Server.

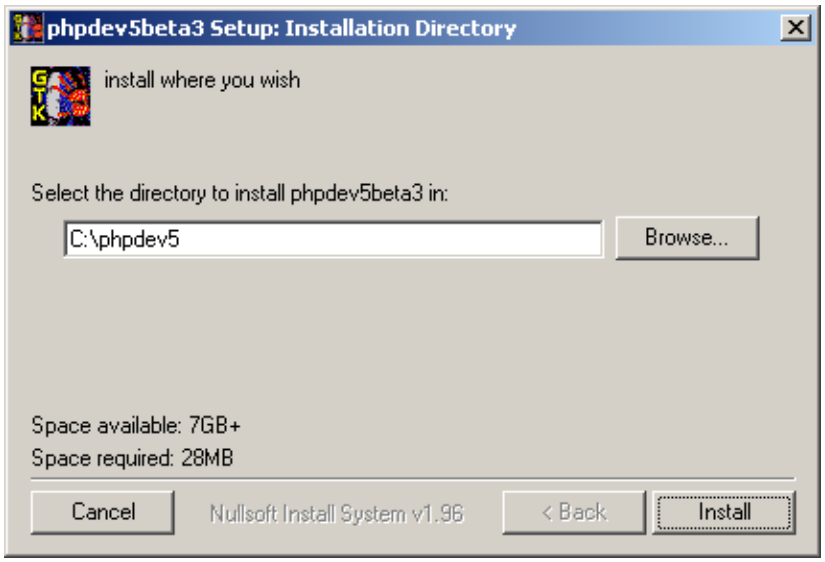
.

Τα βήματα που ακολουθούμε κατά την εγκατάσταση φαίνονται στις παρακάτω εικόνες.

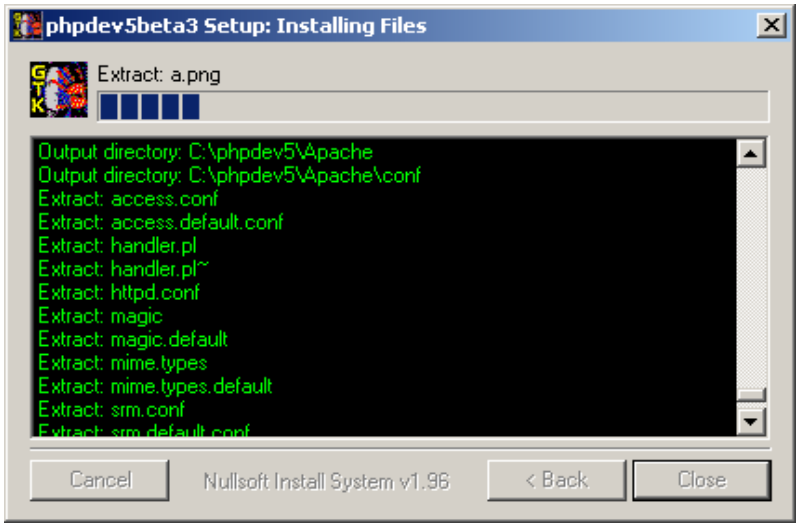
Αυτή είναι η πρώτη εικόνα που μας εμφανίζετε μόλις κάνουμε διπλό κλικ στο αρχείο

MySQL.exe η οθόνη αυτή μας παροτρύνει να επιλέξουμε τη διαδρομή στο δίσκο μας όπου θέλουμε να εγκατασταθεί η ***phpdev5beta3.***

Αφού επιλέξουμε τη διαδρομή πατάμε το πλήκτρο Install.

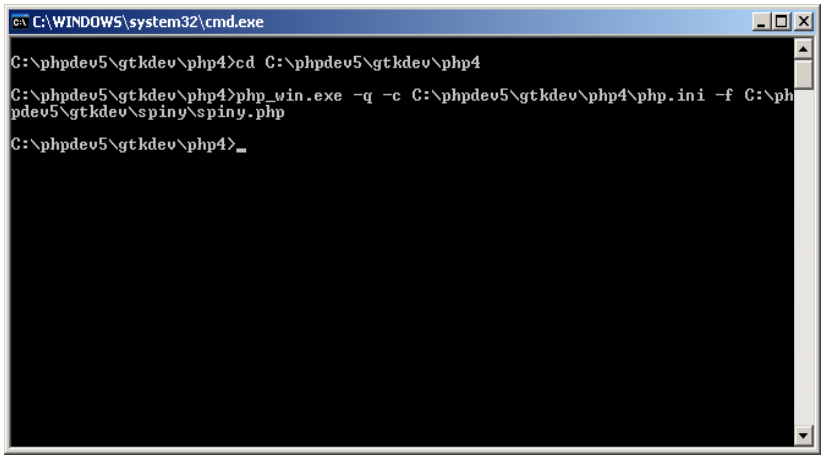


Σ'αυτή την εικόνα γίνετε αποσυμπίεση των αρχείων του προγράμματος για να ακολουθήσει η εγκατάσταση τους .



Αυτή είναι η οθόνη της ***phpdev5beta3*** στην οποία μπορούμε να επιλέξουμε να την εγκαταστήσουμε , ή να διαμορφώσουμε τον mail και proxy Server μας , αν έχουμε . Εμείς επιλέγουμε το πλήκτρο Install phpdev5 now.

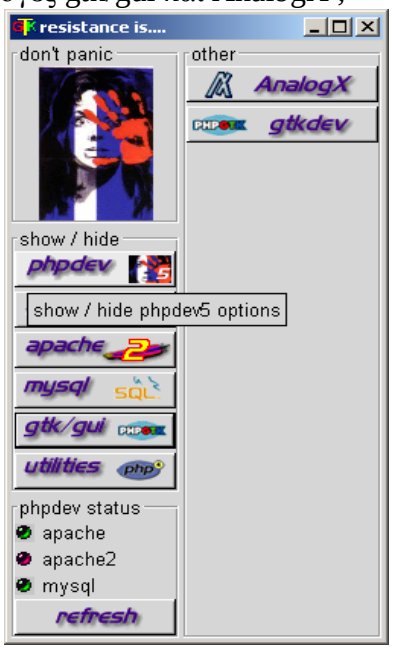
Αν η εγκατάσταση γίνει σωστά θα εμφανιστεί στην οθόνη μας το παρακάτω DOS παράθυρο.



1) 2) 3)

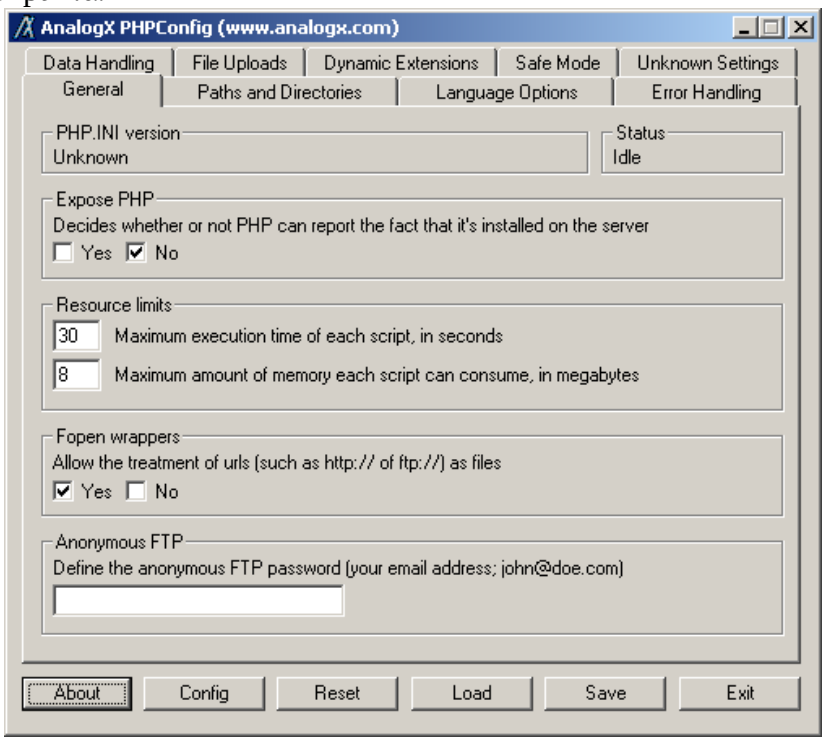
1)Η βασική κονσόλα της ***phpdev5beta3*** όταν είναι ανοιχτές οι υπηρεσίες Apache και MySql τα λαμπάκια κάτω είναι πράσινα.
2)Το μενού που εμφανίζετε όταν πατάμε το πλήκτρο apache.
3) Το μενού που εμφανίζετε όταν πατάμε το πλήκτρο mysql.

Αν ακολουθήσουμε τις επιλογές gtk/gui και AnalogX ,

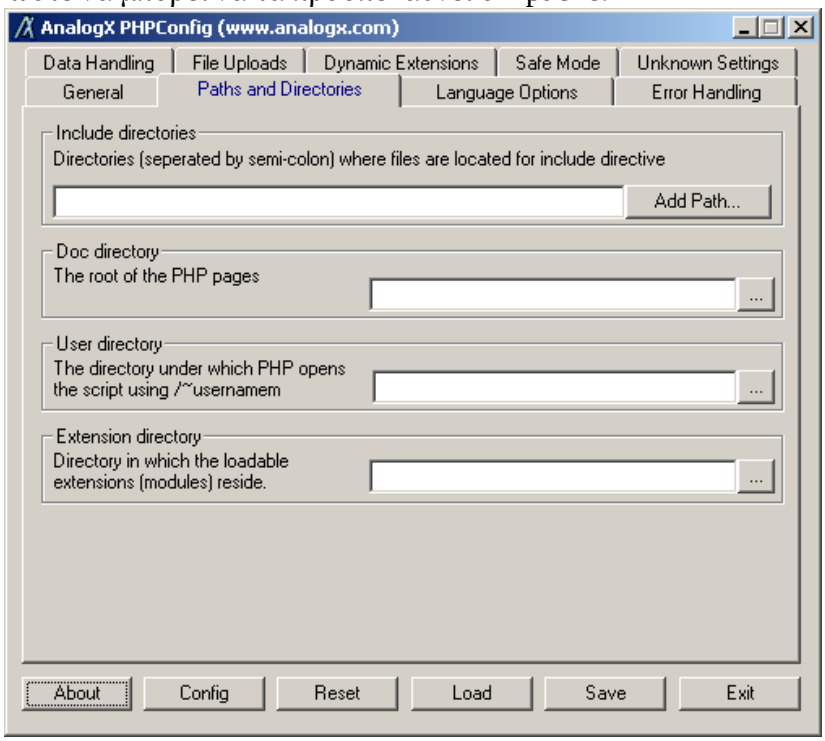


τότε μεταφερόμαστε στο γραφικό περιβάλλον διαχειρίσεις το οποίο είναι το ακόλουθο.

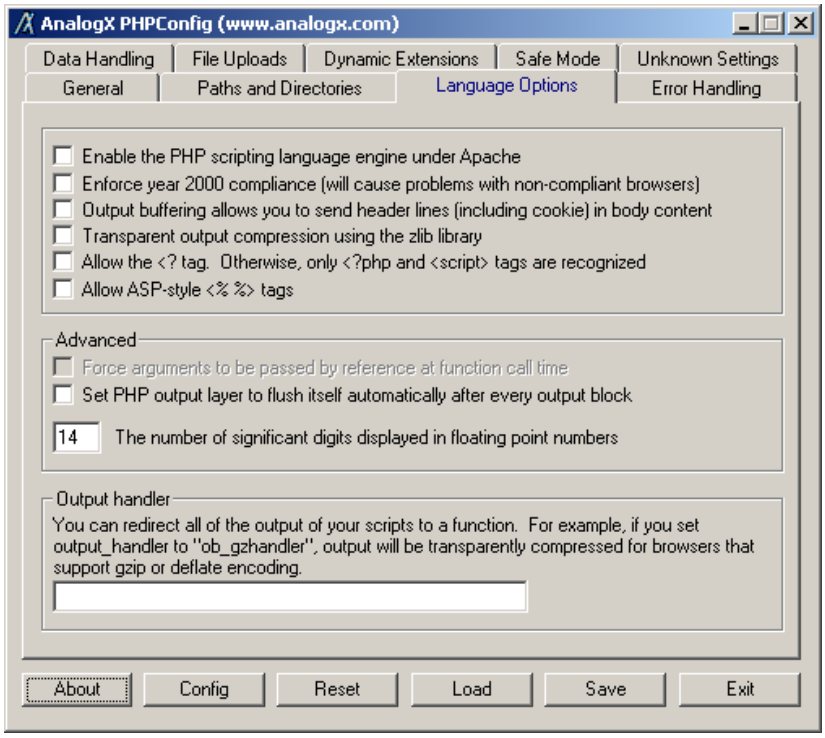
Σ'αυτή την εικόνα ,πλήκτρο General βλέπουμε διάφορες πληροφορίες όπως την έκδοση του αρχείου PHP.ini το ποσά δευτερόλεπτα μπορεί να εκτελείτε στον Server ένα PHP script κ.α.



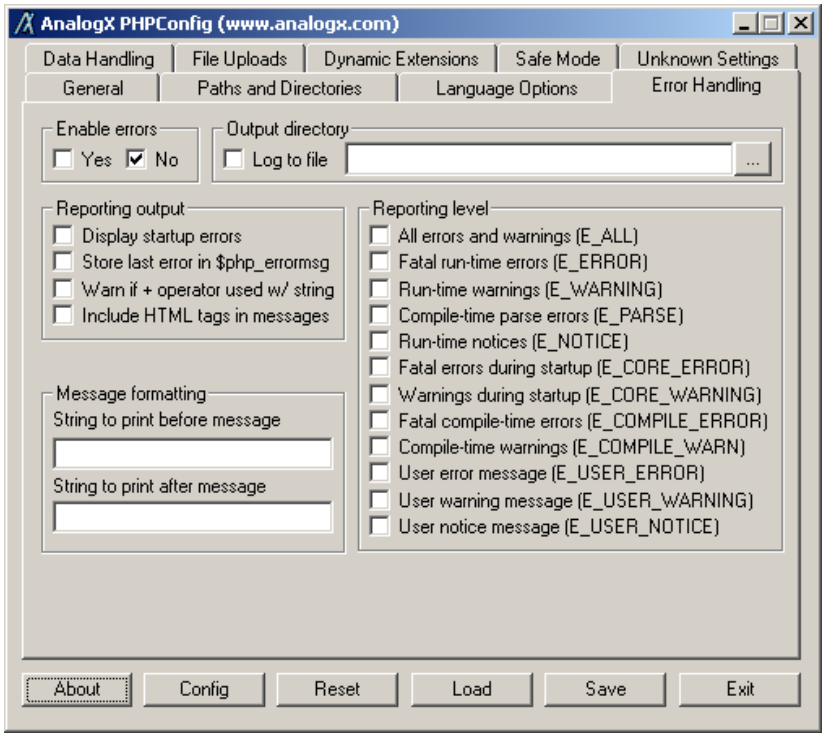
Σ'αυτή την εικόνα ,πλήκτρο Paths and Directories μας δίνετε η δυνατότητα να ορίσουμε τον document root το φάκελο δηλαδή που πρέπει να τοποθετούμε τα PHP script μας , ώστε να μπορεί να τα προσπελαύνει ο Apache.



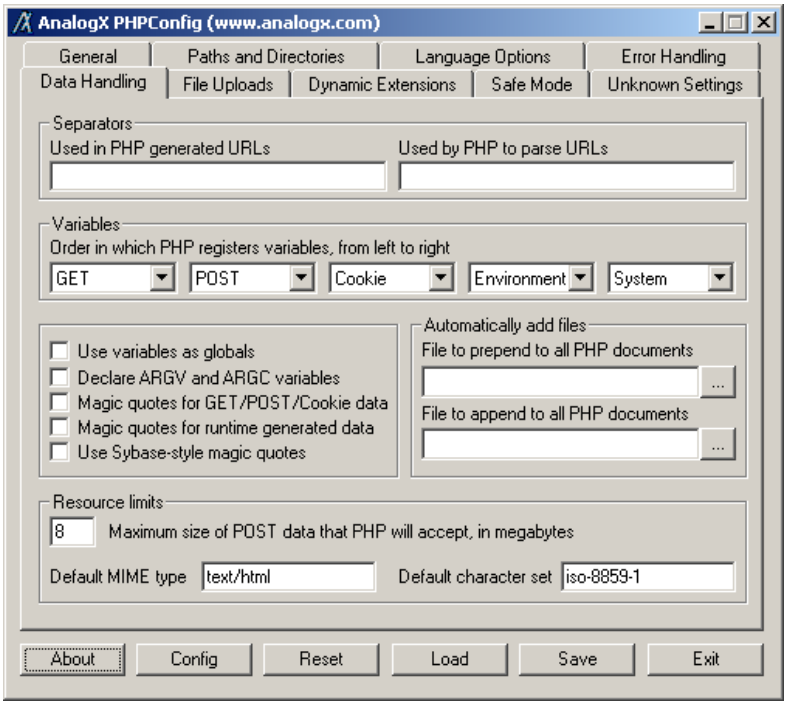
Σ'αυτή την εικόνα ,πλήκτρο Paths and Directories μας δίνετε η δυνατότητα να ορίσουμε τη μορφής scripts θα καταλαβαίνει η μηχανή της PHP π.χ αν τα scripts θα ξεκινούν με το κλασσικό <?php>,ή με το <?>, ή ακόμη και με το στυλ της ASP το <%>.



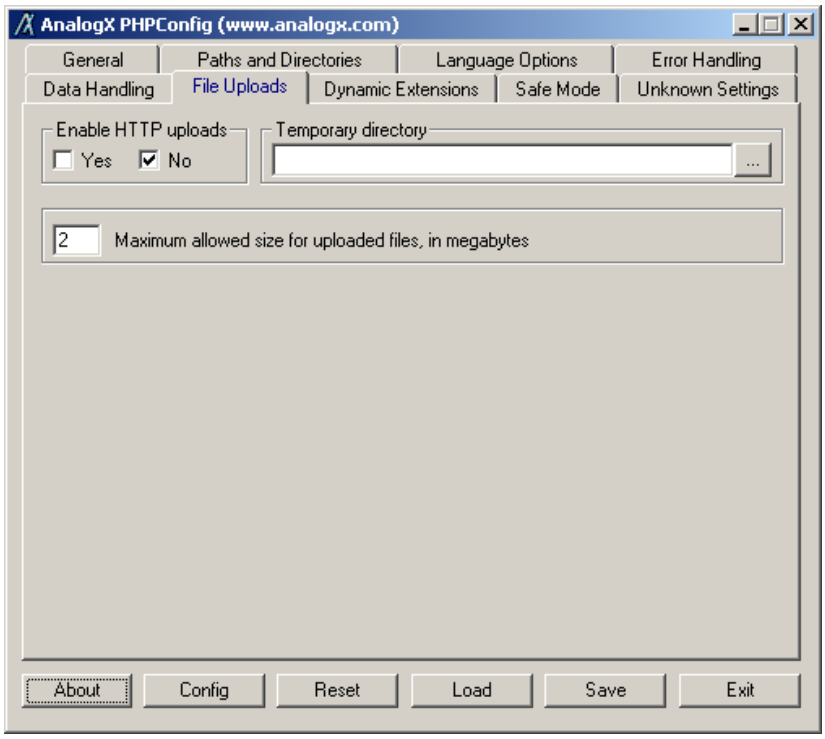
Σ'αυτή την εικόνα, πλήκτρο ErrorHandling,εμφανίζονται διάφορα λάθη τα οποία γίνονται και στα οποία εμπλέκεται η PHP από εδώ μπορούμε να ορίσουμε αν θέλουμε να ειδοποιούμαστε και για ποια από αυτά τα λάθη , ή αν απλός θα προσπερνιούνται, και τα scripts θα συνεχίζουν να εκτελούνται με ότι ρίσκο συνεπάγεται αυτό.



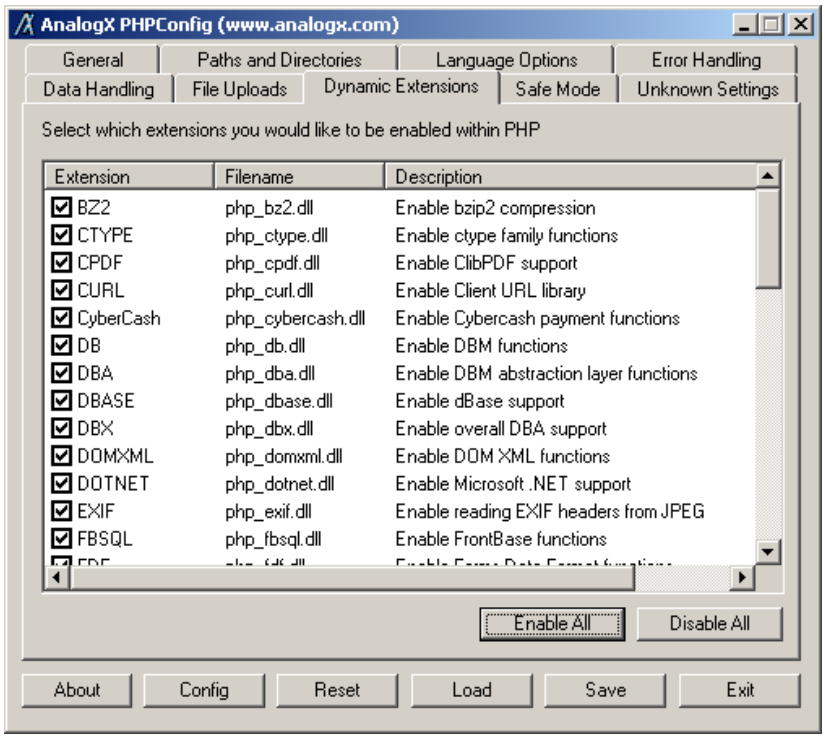
Σ'αυτή την εικόνα, πλήκτρο Data Handing, μας δίνετε η δυνατότητα να επηρεάσουμε την μέθοδο με την οποία θα περνάν τα δεδομένα από τον Server και να ορίσουμε ακόμη και ιεραρχία στις μεθόδους Αποστόλης και λήψης των δεδομένων τροποποιώντας κάποιες μεταβλητές της PHP.



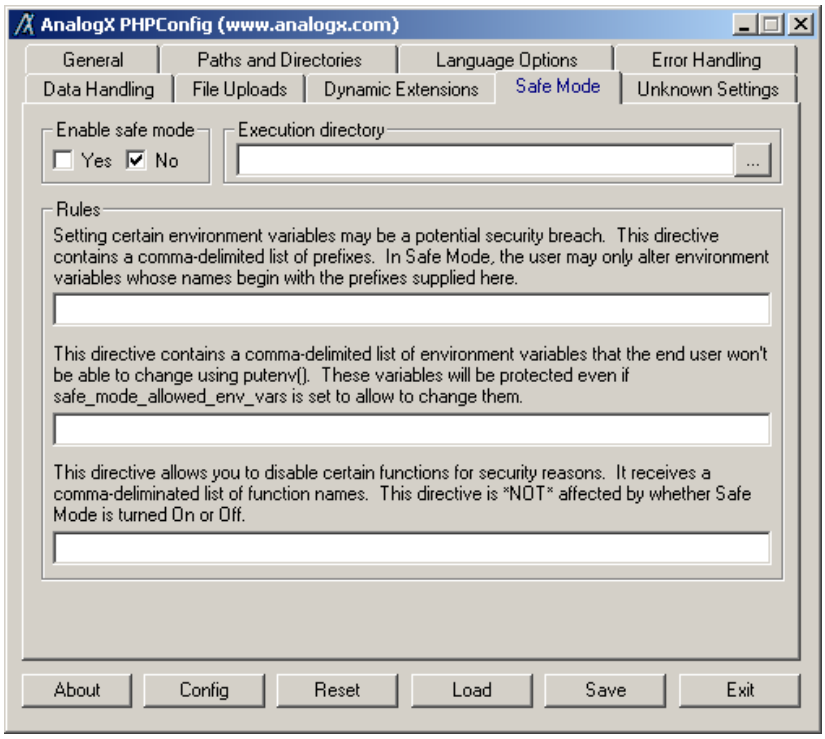
Σ'αυτή την εικόνα, πλήκτρο File Uploads, μας δίνετε η δυνατότητα να ορίσουμε το πρωτόκολλο το οποίο θα επηρεάζει την Αποστόλη και λήψη των δεδομένων και τον φάκελο στον οποίο αυτά θα αποθηκεύονται προσωρινά μέχρι να υποστούν επεξεργασία από τον Server, έτσι ώστε αν κάτι δεν πάει καλά στην επικοινωνία με τον Server να μη χαθούν τα δεδομένα που αποστάλθηκαν.



Σ'αυτή την εικόνα, πλήκτρο Dynamic Extensions, μας δίνετε η δυνατότητα να ορίσουμε τι τύπους αρχεία βάσεων δεδομένων θα υποστηρίζονται από την PHP, ώστε να περνούν στον Server για επεξεργασία.

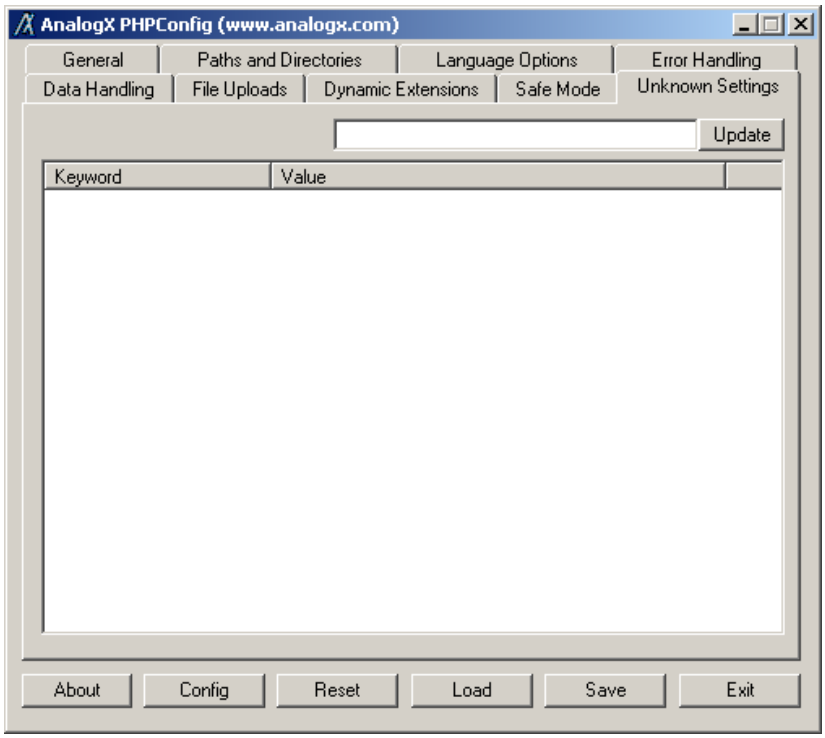


Σ'αυτή την εικόνα, πλήκτρο Safe Mode, μας δίνετε η δυνατότητα να ορίσουμε το φάκελο στον οποίο θα εκτελείτε σε λειτουργία Safe Mode ο Apache αν για κάποιο λόγο διακοπεί η λειτουργία του και χρειαστεί να επανεκκινήση.



Σ'αυτή την εικόνα, πλήκτρο Safe Mode, μας δίνετε η δυνατότητα να ορίσουμε κάποιες άχρηστες προς το παρόν ρυθμίσεις για την εφαρμογή μας οι οποίες όμως μελλοντικά μπορεί να υποστηριχτούν π.χ μπορεί τώρα η εφαρμογή μας να εξυπηρετείτε μόνο από το HTTP πρωτόκολλο ,μπορεί όμως εδώ να ορίσουμε και κάποιο άγνωστο προς το παρόν πρωτόκολλο όπως το FTP για μελλοντική χρήση

.



ΠΑΡΑΡΤΗΜΑ Β

Η ΕΦΑΡΜΟΓΗ ΚΑΙ Ο ΚΩΔΙΚΑΣ ΤΗΣ

Παρακάτω θα γίνει μια παρουσίαση της εφαρμογής που αναπτύχθηκε στα πλαίσια αυτής της εργασίας , επίσης θα παρατεθεί ο κώδικας της .

Η εφαρμογή αφορά ένα On Line Μάθημα ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ το οποίο φυσικά μπορεί να λειτουργήσει υπό πραγματικές συνθήκες .

Έμπνευση για τη δημιουργία αυτής της εφαρμογής υπήρξε η μη λειτουργία ενός τέτοιου site στο τμήμα για το μάθημα του Ηλεκτρονικού Εμπορίου στο οποίο διδάσκονται και οι τεχνολογίες που χρησιμοποιήθηκαν για την κατασκευή του.

Το site του μαθήματος αποτελείται από 4 κύρια μέρη:

- Την αρχική σελίδα η οποία φορτώνεται και πρώτη και είναι εισαγωγική.
- Την σελίδα σημειώσεις όπου ο φοιτητής έχει τη δυνατότητα να μάθει για τις διδάσκουσες τεχνολογίες .
- Την σελίδα Downloads όπου ο φοιτητής έχει τη δυνατότητα να κατεβάσει χρήσιμα προγράμματα και ebooks για την ευκολότερη κατανόηση του μαθήματος .
- Την σελίδα όπου ο φοιτητής έχει τη δυνατότητα να κατεβάσει της ασκήσεις του εξάμηνου αλλά και αφού τις λύσει να τις ανεβάσει (upload) στον Server.

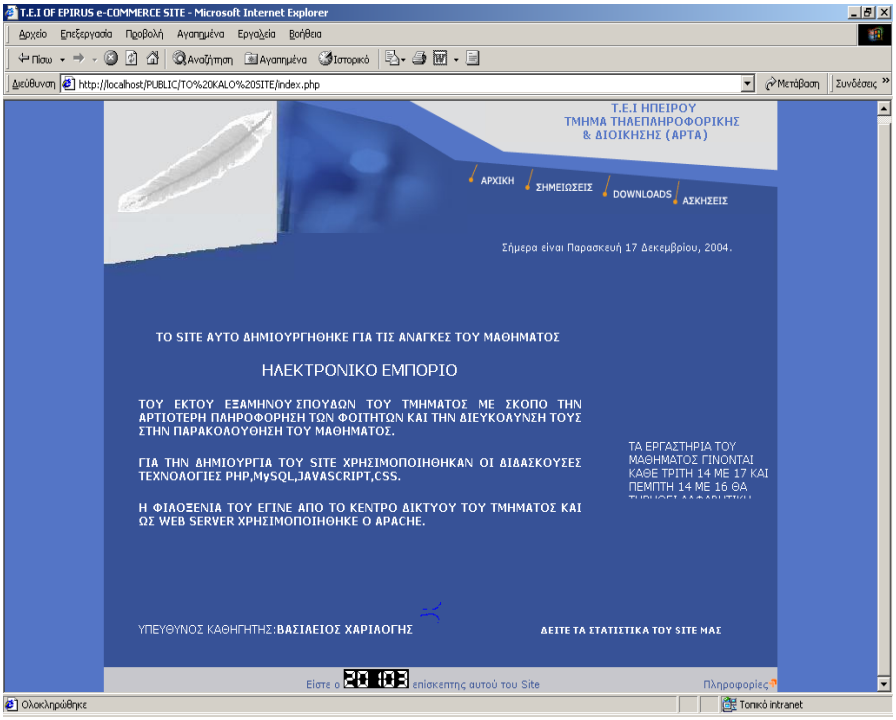
Για την κατασκευή του site χρησιμοποιήθηκαν οι τεχνολογίες HTML PHP,MySQL,CSS και javascript.

Εκτός από τα 4 κύρια μέρη της η εφαρμογή αποτελείται και από κάποια αλλά μέρη τα οποία κυρίως υλοποιούνται από PHP scripts αυτά είναι:

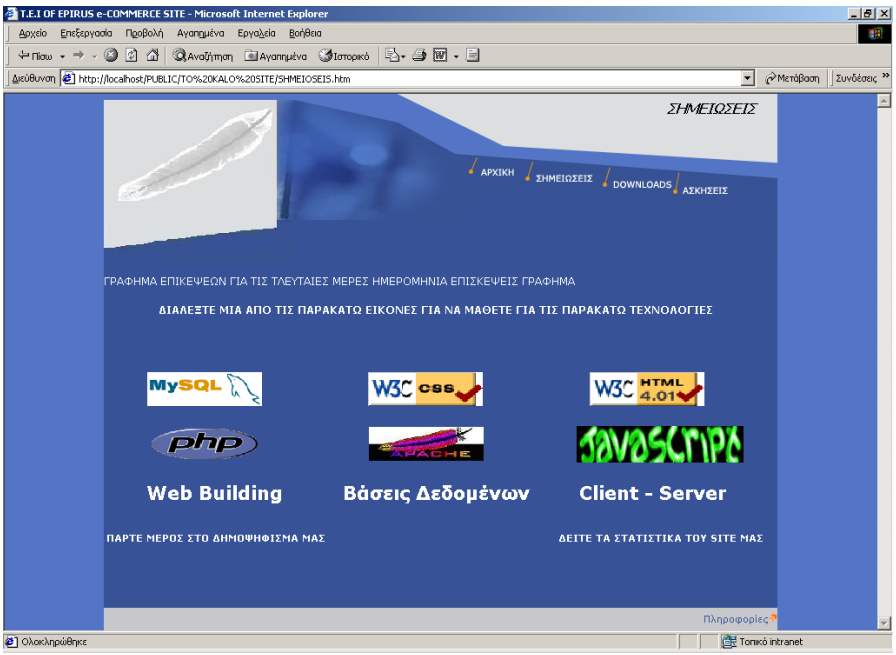
- Ένα PHP script το οποίο μας παρέχει στατιστικά τόσο με τη μορφή πινάκων , όσο και με τη μορφή γραφήματος για την επισκεψημότητα της σελίδας.
- Ένα PHP script το οποίο μας παρέχει εν μετρητή για το πόσοι επισκέπτες έχουν επισκεφτεί το site μας .
- Ένα PHP script το οποίο μας παρέχει τη δυνατότητα να ανεβάζουμε αρχεία στον Server.
- Ένα script (javascript) το οποίο μας παρέχει τη δυνατότητα να δημιουργήσουμε ένα κυλιόμενο πίνακα ανακοινώσεων.
- Ένα script (javascript) το οποίο μας παρέχει τη δυνατότητα να εμφανίζουμε την τρέχουσα μέρα και ημερομηνία.
- Ένα PHP script το οποίο μας παρέχει τη δυνατότητα να διεξάγουμε ένα δημοψήφισμα.
- Ένα PHP script το οποίο μας παρέχει τη δυνατότητα ένα Guest Book για το site μας .

Παρακάτω θα παρατεθούν με τη μορφή εικόνων οι σελίδες του site που αφορούν το μάθημα του ηλεκτρονικού εμπορίου.

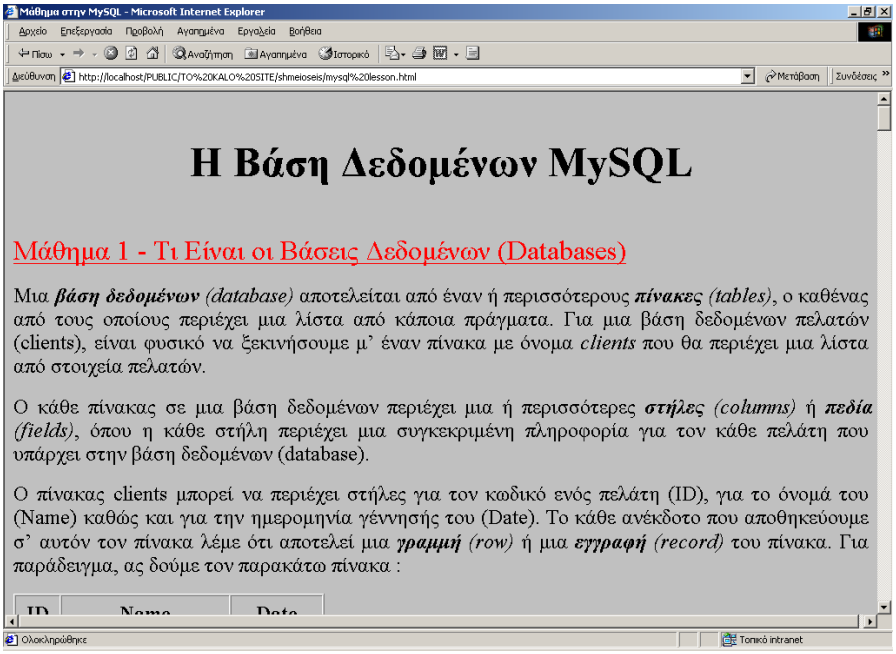
Αυτή είναι η αρχική σελίδα του site



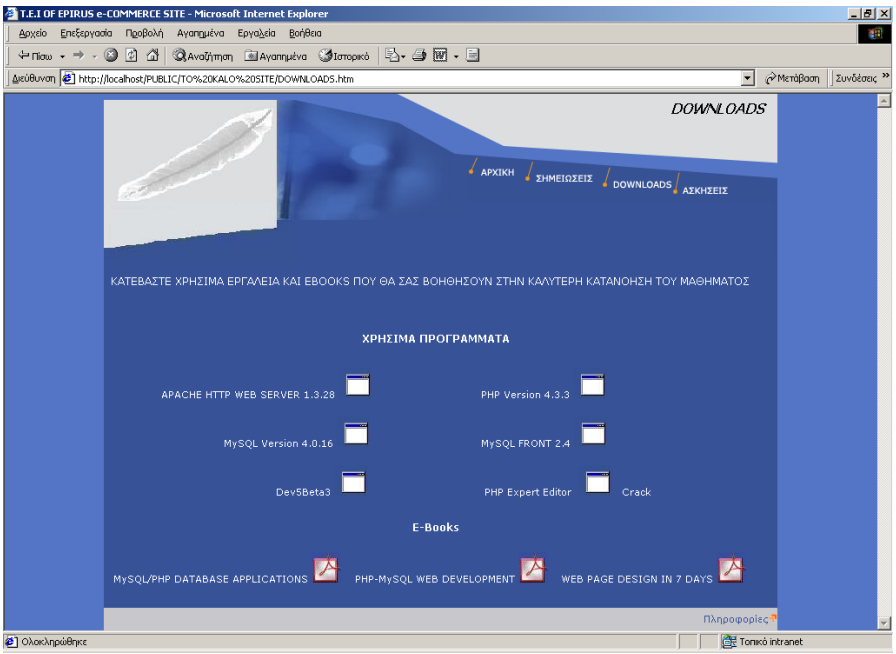
Αυτή είναι η σελίδα που αφορά τις σημειώσεις τις οποίες μπορούν να διαβάζουν οι φοιτητές που παρακολουθούν το μάθημα



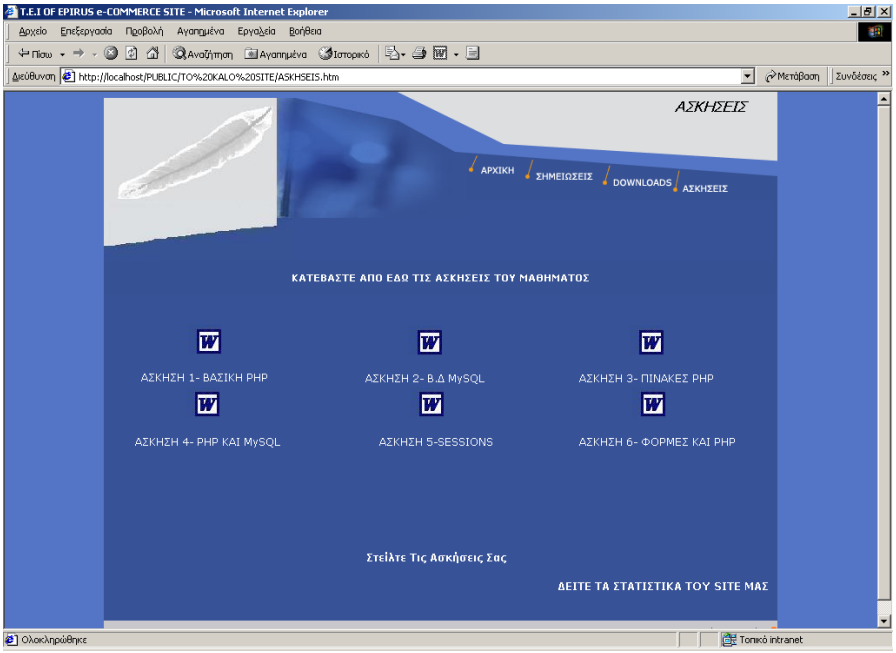
Ενδεικτικά παρατίθεται η σελίδα για ένα κομμάτι των σημειώσεων που παρέχονται στο site και αφορά την MySQL.



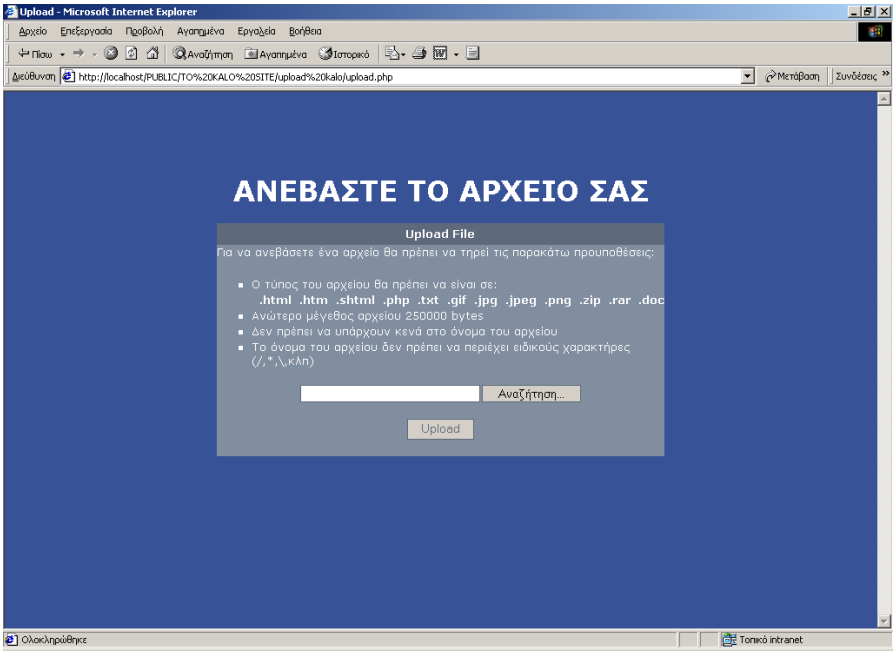
Αυτή είναι η σελίδα που δίνει τη δυνατότητα στους φοιτητές να κατεβάσουν τα απαραίτητα προγράμματα για το μάθημα , καθώς και κάποια ebooks για την βαθύτερη κατανόηση της διδακτέας ύλης .



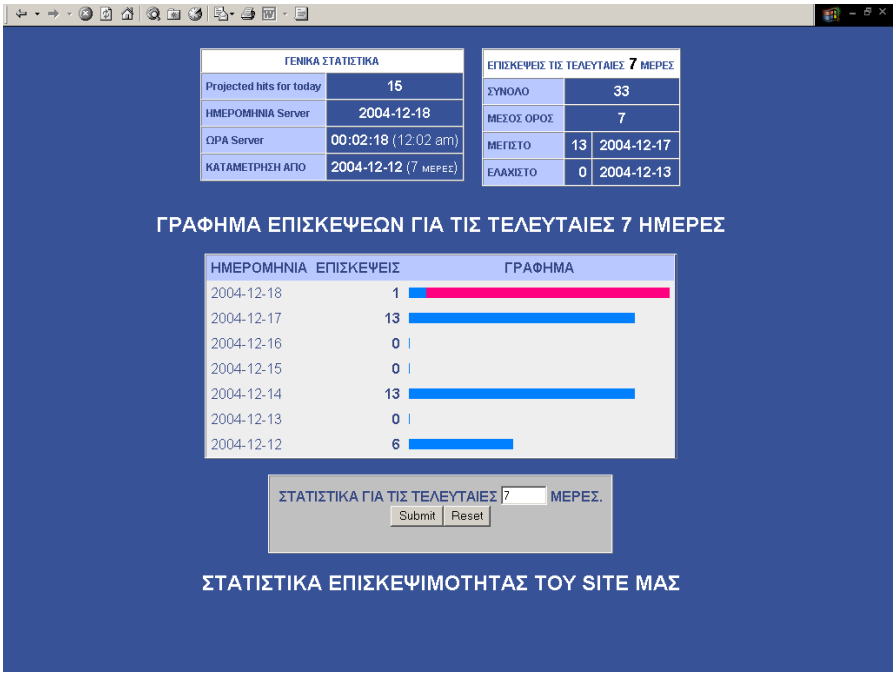
Αυτή είναι η σελίδα που δίνει τη δυνατότητα στους φοιτητές να κατεβάσουν τις ασκήσεις που πρέπει να λύσουν κατά τη διάρκεια του εξαμήνου από εδώ υπάρχει και το link για να κάνουν upload των εργασιών τους όταν τις λύσουν.



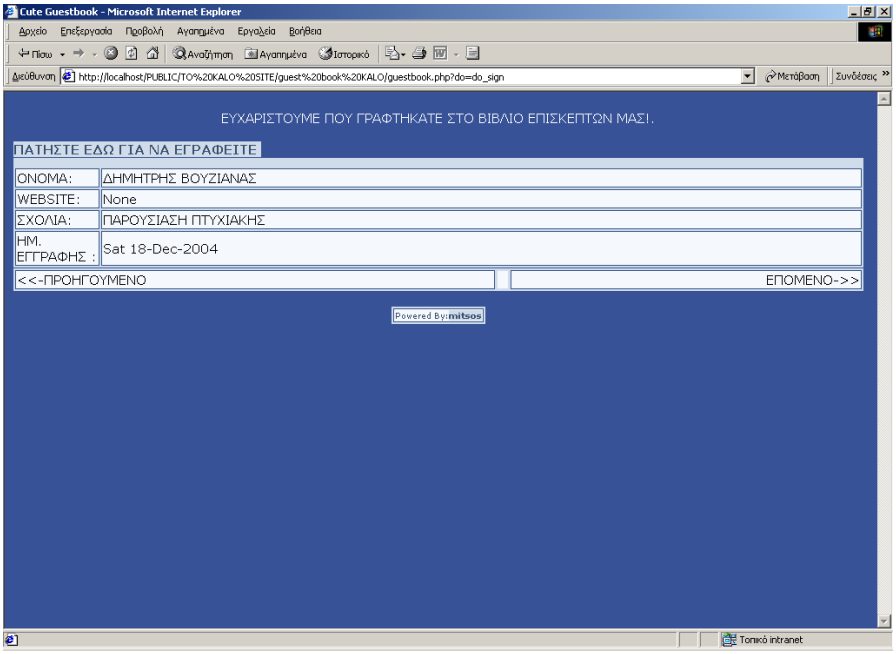
Αυτή είναι η σελίδα που δίνει τη δυνατότητα στους φοιτητές να ανεβάσουν τις ασκήσεις τους στον server.



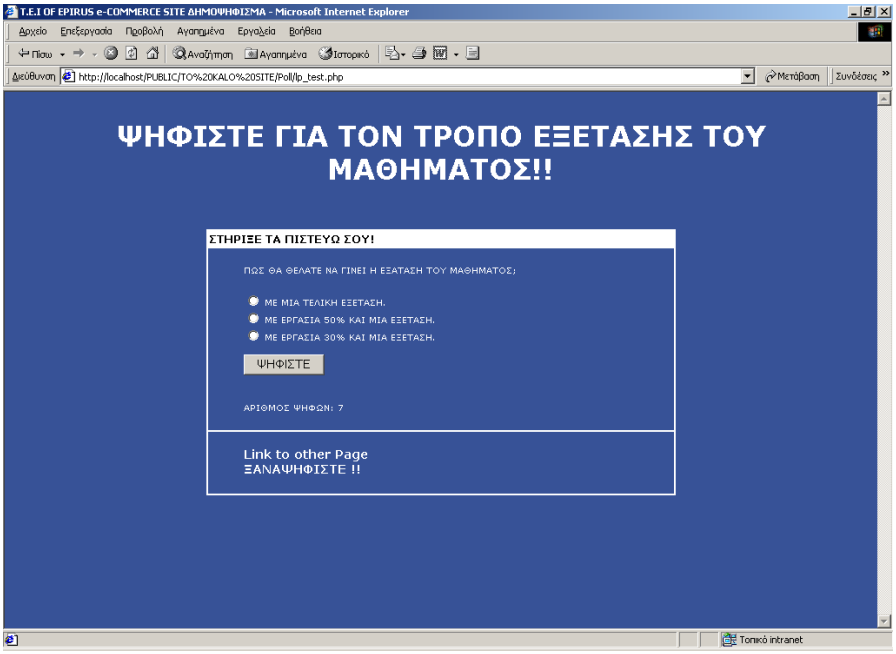
Αυτή είναι η σελίδα που δίνει τη δυνατότητα στους επισκέπτες του site να δουν κάποια στατιστικά στοιχεία γι'άυτο.



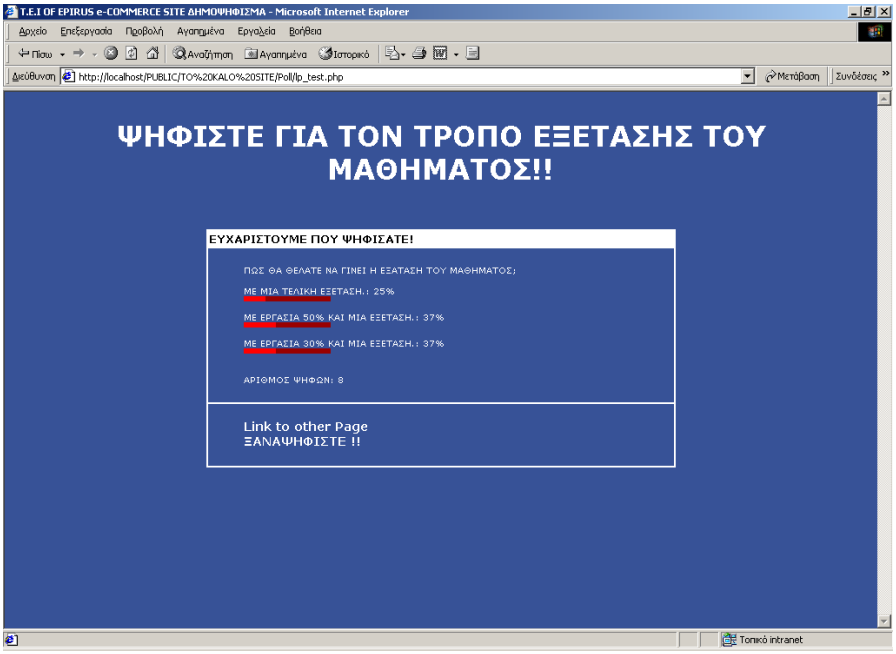
Αυτή είναι η σελίδα που δίνει τη δυνατότητα στους επισκέπτες του site να εγγραφούν σε ένα guest book και να γράψουν κάποιες παρατηρήσεις τους .



Αυτή είναι η σελίδα που δίνει τη δυνατότητα στους επισκέπτες του site να ψηφίσουν σε ένα on line δημοψήφισμα που αφορά το πώς θα γίνετε η εξέταση του μαθήματος .



Τα αποτελέσματα του δημοψηφίσματος φαίνονται , ως εξής .



Αν και η γραφική αναπαράσταση είναι αυτή που έχει άξια στις πραγματικές web εφαρμογές , δεν πρέπει ποτέ να ξεχνάμε πως φτιάχνονται τα τόσο όμορφα γραφικά και οι τόσο εξελιγμένες υπηρεσίες των σύγχρονων web site, όλα αυτά φυσικά φτιάχνονται με κώδικα γι΄ αυτό και παρακάτω παρατήθονται τα αρχεία του κώδικα της εφαρμογής.

ΤΟ ΑΡΧΕΙΟ INDEX.PHP

```
<?php include "fstats.php" ?>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//GR">
<html>
<head>
<title>T.E.I OF EPIRUS e-COMMERCE SITE</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<meta name="keywords" content="Keywords here">
<meta name="description" content="Description here">
<meta name="Author" content="MyFreeTemplates.com">
<META NAME="robots" CONTENT="index, follow"> <!-- (Robot commands: All, None, Index, No
Index, Follow, No Follow) -->
<META NAME="revisit-after" CONTENT="30 days">
<META NAME="distribution" CONTENT="global">
<META NAME="rating" CONTENT="general">
<META NAME="Content-Language" CONTENT="english">


<script language="JavaScript" type="text/JavaScript" src="images/myfreetemplates.js"></script>
<link href="images/myfreetemplates.css" rel="stylesheet" type="text/css">
<style type="text/css">
<!--
a:link {
        color: #FFFFFF;
        text-decoration: none;
}
a:visited {
        text-decoration: none;
        color: #FFFFFF;
}
a:hover {
        text-decoration: underline;
        color: #FFFFFF;
}
a:active {
        text-decoration: none;
}
.style1 {color: #375297}
-->
</style></head>

<body background="images/pixi_lightblue.gif" leftmargin="0" topmargin="0" marginwidth="0"
marginheight="0" onLoad="MM_preloadImages('images/btn_main-over.gif','images/btn_overview-
over.gif','images/btn_services-over.gif','images/products_btn-over.gif')">
<table width="775" height="100%" border="0" align="center" cellpadding="0" cellspacing="0"
background="images/pixi_darkblue.gif">
  <tr>
    <td height="7" colspan="2" valign="top"> <TABLE width="775" BORDER=0 CELLPADDING=0
CELLSPACING=0>
      <TR>
        <TD width="336"><IMG SRC="images/pixi_lightblue.gif" WIDTH=336 HEIGHT=7></TD>
        <TD width="16"><IMG SRC="images/topangle.gif" WIDTH=16 HEIGHT=7></TD>
```

```
      <TD background="images/pixi_light_grey.gif"><IMG SRC="images/spacer.gif" WIDTH=423
HEIGHT=7></TD>
     </TR>
    </TABLE></td>
  </tr>
  <tr>
   <td height="118" colspan="2" valign="top"> <table border="0" cellpadding="0" cellspacing="0"
id="navtable">
     <tr>
      <td width="352" rowspan="3"><TABLE width="352" BORDER=0 CELLPADDING=0
CELLSPACING=0>
        <TR>
         <TD><img src="images/mainpic1.jpg" width="219" height="118"></TD>
         <TD><IMG SRC="images/toptreatment1.jpg" WIDTH=133 HEIGHT=118></TD>
        </TR>
       </TABLE></td>
      <td width="423" background="images/pixi_light_grey.gif"> <TABLE BORDER=0
CELLPADDING=0 CELLSPACING=0>
        <TR>
         <TD width="87"><IMG SRC="images/topcurve2.gif" WIDTH="87" HEIGHT="43"></TD>
         <TD width="190" background="images/pixi_light_grey.gif"><IMG
SRC="images/spacer.gif" WIDTH=190 HEIGHT=43></TD>
         <TD width="146" background="images/pixi_light_grey.gif"><img
src="images/toplogoASKHSEIS.gif" width="146" height="43"></TD>
        </TR>
       </TABLE></td>
     </tr>
     <tr>
      <td width="423" background="images/pixi_light_grey.gif"> <TABLE width="100%"
BORDER=0 CELLPADDING=0 CELLSPACING=0>
        <TR>
         <TD width="277"><IMG SRC="images/topcurve3.gif" WIDTH=277 HEIGHT=20></TD>
         <TD width="146" background="images/pixi_light_grey.gif"><IMG
SRC="images/spacer.gif" WIDTH=146 HEIGHT=20></TD>
        </TR>
       </TABLE></td>
     </tr>
     <tr>
      <td width="423"> <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
        <TR>
         <TD width="66"><IMG SRC="images/toptreatment1a.jpg" WIDTH=66
HEIGHT=55></TD>
         <TD width="72"><A HREF="index.php"><IMG SRC="images/btn_main.gif"
NAME="btn_main" WIDTH=72 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_main','','images/btn_main-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
         <TD width="89"><A HREF="SHMEIOSEIS.htm"><IMG SRC="images/btn_overview.gif"
NAME="btn_overview" WIDTH=89 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_overview','','images/btn_overview-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
         <TD width="81"><A HREF="DOWNLOADS.htm"><IMG SRC="images/btn_services.gif"
NAME="btn_services" WIDTH=81 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_services','','images/btn_services-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
         <TD width="72"> <A HREF="ASKHSEIS.htm"><IMG SRC="images/products.gif"
NAME="products" WIDTH=72 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('products','','images/products_btn-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
         <TD width="43"><IMG SRC="images/topnavend.gif" WIDTH=43 HEIGHT=55></TD>
        </TR>
       </TABLE></td>
```

```
      </tr>
    </table></td>
  </tr>
  <tr>
   <td height="69" colspan="2" valign="top">
    <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
     <TR>
      <TD ROWSPAN=2><IMG SRC="images/mainpic2.jpg" WIDTH=219 HEIGHT=69></TD>
      <TD><IMG SRC="images/toptreatment2.jpg" WIDTH=199 HEIGHT=20></TD>
      <TD width="357" height="69" ROWSPAN=2 bgcolor="#375297"> </TD>
      <TD ROWSPAN=2> </TD>
     </TR>
     <TR>
      <TD width="199" height="49" bgcolor="#375297"> </TD>
     </TR>
    </TABLE>
   </td>
  </tr>
  <tr>
   <td width="775" valign="top"> <div align="center"><br>
     <strong>&Kappa;&Alpha;&Tau;&Epsilon;&Beta;&Alpha;&Sigma;&Tau;&Epsilon;
&Alpha;&Pi;&Omicron; &Epsilon;&Delta;&Omega; &Tau;&Iota;&Sigma;
&Alpha;&Sigma;&Kappa;&Eta;&Sigma;&Epsilon;&Iota;&Sigma; &Tau;&Omicron;&Upsilon;
&Mu;&Alpha;&Theta;&Eta;&Mu;&Alpha;&Tau;&Omicron;&Sigma; </strong><br>
     <br>

   </div>
    <TABLE width="774" BORDER=0 CELLPADDING=0 CELLSPACING=0>
     <TR>
      <TD width="765" height="250" valign="top" bgcolor="#375297"><table width="766"
height="156" border="0" bordercolor="#FF0000">
        <tr>
         <td width="226" height="62" align="center">       
          <p><a href="ASKHSEIS%20FOITHTON/ASKHSH%201.doc"><img
src="images/word2.jpg" width="38" height="34" border="0"></a></p>
          <p align="center">A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 1-
&Beta;&Alpha;&Sigma;&Iota;&Kappa;&Eta; PHP </p></td>
         <td width="226" align="center" bordercolor="#FF0000"><p> </p>
          <p><a href="ASKHSEIS%20FOITHTON/ASKHSH%202.doc"><img
src="images/word2.jpg" width="38" height="34" border="0"></a></p>
          <p align="center">&Alpha;&Sigma;&Kappa;&Eta;&Sigma;&Eta; 2- B.&Delta; MySQL
</p></td>
         <td width="226" align="center"><p> </p>
          <p><a href="ASKHSEIS%20FOITHTON/ASKHSH%203.doc"><img
src="images/word2.jpg" width="38" height="34" border="0"></a></p>
          <p>A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 3-
&Pi;&Iota;&Nu;&Alpha;&Kappa;&Epsilon;&Sigma; PHP </p></td>
        </tr>
        <tr>
         <td height="62" align="center"><p
align="left">            &
nbsp;          <a
href="ASKHSEIS%20FOITHTON/ASKHSH%204.doc"><img src="images/word2.jpg" width="38"
height="34" border="0"></a> </p>
          <p align="center">A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 4- PHP KAI
MySQL    </p></td>
         <td bordercolor="#FF0000"><div
align="left">            &
nbsp;          <a
href="ASKHSEIS%20FOITHTON/ASKHSH%205.doc"><img src="images/word2.jpg" width="38"
height="34" border="0"></a>
```

```
            <p align="center">A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 5-SESSIONS</p>
          </div></td>
          <td><div
align="left">            &
nbsp;          <a
href="ASKHSEIS%20FOITHTON/ASKHSH%206.doc"><img src="images/word2.jpg" width="38"
height="34" border="0"></a>
            <p align="center">A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 6-
&Phi;&Omicron;&Rho;&Mu;&Epsilon;&Sigma; &Kappa;&Alpha;&Iota; PHP </p>
          </div></td>
        </tr>
        <tr>
          <td height="62"> </td>
          <td bordercolor="#FF0000"> </td>
          <td> </td>
        </tr>
      </table>
            <p> </p>
            <p align="center"><a
href="upload%20kalo/upload.php"><strong>&Sigma;&tau;&epsilon;&#943;&lambda;&tau;&epsilon;
&Tau;&iota;&sigmaf; &Alpha;&sigma;&kappa;&#942;&sigma;&epsilon;&iota;&sigmaf;
&Sigma;&alpha;&sigmaf;</strong></a></p>
            <p align="right"><strong><a href="fstats-
view.php">&Delta;&Epsilon;&Iota;&Tau;&Epsilon; &Tau;&Alpha;
&Sigma;&Tau;&Alpha;&Tau;&Iota;&Sigma;&Tau;&Iota;&Kappa;&Alpha;
&Tau;&Omicron;&Upsilon; SITE MA&Sigma;</a></strong></p>
            <p> </p></TD>

      </TR>
   </TABLE>  </tr>

  <tr>
    <td height="26" colspan="2" valign="top"> <TABLE width="775" BORDER=0
CELLPADDING=0 CELLSPACING=0>
      <TR>
        <TD width="53" height="26" bgcolor="#C7C7CC"> </TD>
        <TD width="722" bgcolor="#C7C7CC" ><div align="right"><span
class="style1">&Pi;&lambda;&eta;&rho;&omicron;&phi;&omicron;&rho;&#943;&epsilon;&sigmaf;
</span><a href="info.htm"><img src="images/mftico.gif"
alt="&Pi;&lambda;&eta;&rho;&omicron;&phi;&omicron;&rho;&#943;&epsilon;&sigmaf;..."
width="11" height="11" border="0"></a> </div></TD>
      </TR>
    </TABLE></td>
  </tr>
</table>
</body>
</html>
```

ΤΟ ΑΡΧΕΙΟ ASKHSEIS .HTML

```
<?php include "fstats.php" ?>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//GR">
<html>
<head>
<title>T.E.I OF EPIRUS e-COMMERCE SITE</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<meta name="keywords" content="Keywords here">
<meta name="description" content="Description here">
<meta name="Author" content="MyFreeTemplates.com">
```

```
<META NAME="robots" CONTENT="index, follow"> <!-- (Robot commands: All, None, Index, No Index, Follow, No Follow) -->
<META NAME="revisit-after" CONTENT="30 days">
<META NAME="distribution" CONTENT="global">
<META NAME="rating" CONTENT="general">
<META NAME="Content-Language" CONTENT="english">


<script language="JavaScript" type="text/JavaScript" src="images/myfreetemplates.js"></script>
<link href="images/myfreetemplates.css" rel="stylesheet" type="text/css">
<style type="text/css">
<!--
a:link {
	color: #FFFFFF;
	text-decoration: none;
}
a:visited {
	text-decoration: none;
	color: #FFFFFF;
}
a:hover {
	text-decoration: underline;
	color: #FFFFFF;
}
a:active {
	text-decoration: none;
}
.style1 {color: #375297}
-->
</style></head>

<body background="images/pixi_lightblue.gif" leftmargin="0" topmargin="0" marginwidth="0" marginheight="0" onLoad="MM_preloadImages('images/btn_main-over.gif','images/btn_overview-over.gif','images/btn_services-over.gif','images/products_btn-over.gif')">
<table width="775" height="100%" border="0" align="center" cellpadding="0" cellspacing="0" background="images/pixi_darkblue.gif">
  <tr>
    <td height="7" colspan="2" valign="top"> <TABLE width="775" BORDER=0 CELLPADDING=0 CELLSPACING=0>
      <TR>
        <TD width="336"><IMG SRC="images/pixi_lightblue.gif" WIDTH=336 HEIGHT=7></TD>
        <TD width="16"><IMG SRC="images/topangle.gif" WIDTH=16 HEIGHT=7></TD>
        <TD background="images/pixi_light_grey.gif"><IMG SRC="images/spacer.gif" WIDTH=423 HEIGHT=7></TD>
      </TR>
    </TABLE></td>
  </tr>
  <tr>
    <td height="118" colspan="2" valign="top"> <table border="0" cellpadding="0" cellspacing="0" id="navtable">
      <tr>
        <td width="352" rowspan="3"><TABLE width="352" BORDER=0 CELLPADDING=0 CELLSPACING=0>
          <TR>
            <TD><img src="images/mainpic1.jpg" width="219" height="118"></TD>
            <TD><IMG SRC="images/toptreatment1.jpg" WIDTH=133 HEIGHT=118></TD>
          </TR>
        </TABLE></td>
        <td width="423" background="images/pixi_light_grey.gif"> <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
          <TR>
```

```
            <TD width="87"><IMG SRC="images/topcurve2.gif" WIDTH="87" HEIGHT="43"></TD>
            <TD width="190" background="images/pixi_light_grey.gif"><IMG
SRC="images/spacer.gif" WIDTH=190 HEIGHT=43></TD>
            <TD width="146" background="images/pixi_light_grey.gif"><img
src="images/toplogoASKHSEIS.gif" width="146" height="43"></TD>
          </TR>
        </TABLE></td>
      </tr>
      <tr>
       <td width="423" background="images/pixi_light_grey.gif"> <TABLE width="100%"
BORDER=0 CELLPADDING=0 CELLSPACING=0>
          <TR>
            <TD width="277"><IMG SRC="images/topcurve3.gif" WIDTH=277 HEIGHT=20></TD>
            <TD width="146" background="images/pixi_light_grey.gif"><IMG
SRC="images/spacer.gif" WIDTH=146 HEIGHT=20></TD>
          </TR>
        </TABLE></td>
      </tr>
      <tr>
       <td width="423"> <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
          <TR>
            <TD width="66"><IMG SRC="images/toptreatment1a.jpg" WIDTH=66
HEIGHT=55></TD>
            <TD width="72"><A HREF="index.php"><IMG SRC="images/btn_main.gif"
NAME="btn_main" WIDTH=72 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_main','','images/btn_main-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
            <TD width="89"><A HREF="SHMEIOSEIS.htm"><IMG SRC="images/btn_overview.gif"
NAME="btn_overview" WIDTH=89 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_overview','','images/btn_overview-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
            <TD width="81"><A HREF="DOWNLOADS.htm"><IMG SRC="images/btn_services.gif"
NAME="btn_services" WIDTH=81 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_services','','images/btn_services-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
            <TD width="72"> <A HREF="ASKHSEIS.htm"><IMG SRC="images/products.gif"
NAME="products" WIDTH=72 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('products','','images/products_btn-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
            <TD width="43"><IMG SRC="images/topnavend.gif" WIDTH=43 HEIGHT=55></TD>
          </TR>
        </TABLE></td>
      </tr>
    </table></td>
  </tr>
  <tr>
   <td height="69" colspan="2" valign="top">
    <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
      <TR>
       <TD ROWSPAN=2><IMG SRC="images/mainpic2.jpg" WIDTH=219 HEIGHT=69></TD>
       <TD><IMG SRC="images/toptreatment2.jpg" WIDTH=199 HEIGHT=20></TD>
       <TD width="357" height="69" ROWSPAN=2 bgcolor="#375297"> </TD>
       <TD ROWSPAN=2> </TD>
      </TR>
      <TR>
       <TD width="199" height="49" bgcolor="#375297"> </TD>
      </TR>
    </TABLE>
   </td>
  </tr>
  <tr>
```

```
  <td width="775" valign="top"> <div align="center"><br>
    <strong>&Kappa;&Alpha;&Tau;&Epsilon;&Beta;&Alpha;&Sigma;&Tau;&Epsilon;
&Alpha;&Pi;&Omicron; &Epsilon;&Delta;&Omega; &Tau;&Iota;&Sigma;
&Alpha;&Sigma;&Kappa;&Eta;&Sigma;&Epsilon;&Iota;&Sigma; &Tau;&Omicron;&Upsilon;
&Mu;&Alpha;&Theta;&Eta;&Mu;&Alpha;&Tau;&Omicron;&Sigma; </strong><br>
    <br>

  </div>
   <TABLE width="774" BORDER=0 CELLPADDING=0 CELLSPACING=0>
    <TR>
     <TD width="765" height="250" valign="top" bgcolor="#375297"><table width="766"
height="156" border="0" bordercolor="#FF0000">
      <tr>
       <td width="226" height="62" align="center">       
        <p><a href="ASKHSEIS%20FOITHTON/ASKHSH%201.doc"><img
src="images/word2.jpg" width="38" height="34" border="0"></a></p>
        <p align="center">A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 1-
&Beta;&Alpha;&Sigma;&Iota;&Kappa;&Eta; PHP </p></td>
       <td width="226" align="center" bordercolor="#FF0000"><p> </p>
        <p><a href="ASKHSEIS%20FOITHTON/ASKHSH%202.doc"><img
src="images/word2.jpg" width="38" height="34" border="0"></a></p>
        <p align="center">&Alpha;&Sigma;&Kappa;&Eta;&Sigma;&Eta; 2- B.&Delta; MySQL
</p></td>
       <td width="226" align="center"><p> </p>
        <p><a href="ASKHSEIS%20FOITHTON/ASKHSH%203.doc"><img
src="images/word2.jpg" width="38" height="34" border="0"></a></p>
        <p>A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 3-
&Pi;&Iota;&Nu;&Alpha;&Kappa;&Epsilon;&Sigma; PHP </p></td>
      </tr>
      <tr>
       <td height="62" align="center"><p
align="left">            &
nbsp;         <a
href="ASKHSEIS%20FOITHTON/ASKHSH%204.doc"><img src="images/word2.jpg" width="38"
height="34" border="0"></a> </p>
        <p align="center">A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 4- PHP KAI
MySQL    </p></td>
       <td bordercolor="#FF0000"><div
align="left">            &
nbsp;          <a
href="ASKHSEIS%20FOITHTON/ASKHSH%205.doc"><img src="images/word2.jpg" width="38"
height="34" border="0"></a>
         <p align="center">A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 5-SESSIONS</p>
       </div></td>
       <td><div
align="left">            &
nbsp;          <a
href="ASKHSEIS%20FOITHTON/ASKHSH%206.doc"><img src="images/word2.jpg" width="38"
height="34" border="0"></a>
        <p align="center">A&Sigma;&Kappa;&Eta;&Sigma;&Eta; 6-
&Phi;&Omicron;&Rho;&Mu;&Epsilon;&Sigma; &Kappa;&Alpha;&Iota; PHP </p>
       </div></td>
      </tr>
      <tr>
       <td height="62"> </td>
       <td bordercolor="#FF0000"> </td>
       <td> </td>
      </tr>
     </table>
       <p> </p>
```

```
            <p align="center"><a
href="upload%20kalo/upload.php"><strong>&Sigma;&tau;&epsilon;&#943;&lambda;&tau;&epsilon;
&Tau;&iota;&sigmaf; &Alpha;&sigma;&kappa;&#942;&sigma;&epsilon;&iota;&sigmaf;
&Sigma;&alpha;&sigmaf;</strong></a></p>
            <p align="right"><strong><a href="fstats-
view.php">&Delta;&Epsilon;&Iota;&Tau;&Epsilon; &Tau;&Alpha;
&Sigma;&Tau;&Alpha;&Tau;&Iota;&Sigma;&Tau;&Iota;&Kappa;&Alpha;
&Tau;&Omicron;&Upsilon; SITE MA&Sigma;</a></strong></p>
            <p> </p></TD>

      </TR>
   </TABLE>  </tr>

  <tr>
    <td height="26" colspan="2" valign="top"> <TABLE width="775" BORDER=0
CELLPADDING=0 CELLSPACING=0>
      <TR>
        <TD width="53" height="26" bgcolor="#C7C7CC"> </TD>
        <TD width="722" bgcolor="#C7C7CC" ><div align="right"><span
class="style1">&Pi;&lambda;&eta;&rho;&omicron;&phi;&omicron;&rho;&#943;&epsilon;&sigmaf;
</span><a href="info.htm"><img src="images/mftico.gif"
alt="&Pi;&lambda;&eta;&rho;&omicron;&phi;&omicron;&rho;&#943;&epsilon;&sigmaf;..."
width="11" height="11" border="0"></a> </div></TD>
      </TR>
     </TABLE></td>
  </tr>
</table>
</body>
</html>
```

ΤΟ ΑΡΧΕΙΟ DOWNLOADS.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//GR">
<html>
<head>
<title>T.E.I OF EPIRUS e-COMMERCE SITE</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<meta name="keywords" content="Keywords here">
<meta name="description" content="Description here">
<meta name="Author" content="MyFreeTemplates.com">
<META NAME="robots" CONTENT="index, follow"> <!-- (Robot commands: All, None, Index, No
Index, Follow, No Follow) -->
<META NAME="revisit-after" CONTENT="30 days">
<META NAME="distribution" CONTENT="global">
<META NAME="rating" CONTENT="general">
<META NAME="Content-Language" CONTENT="english">


<script language="JavaScript" type="text/JavaScript" src="images/myfreetemplates.js"></script>
<link href="images/myfreetemplates.css" rel="stylesheet" type="text/css">
<style type="text/css">
<!--
a:link {
            color: #FFFFFF;
            text-decoration: none;
}
a:visited {
            text-decoration: none;
            color: #FFFFFF;
```

```
}
a:hover {
	text-decoration: underline;
}
a:active {
	text-decoration: none;
}
.style1 {color: #375297}
-->
</style></head>

<body background="images/pixi_lightblue.gif" leftmargin="0" topmargin="0" marginwidth="0"
marginheight="0" onLoad="MM_preloadImages('images/btn_main-over.gif','images/btn_overview-
over.gif','images/btn_services-over.gif','images/products_btn-over.gif')">
<table width="775" height="100%" border="0" align="center" cellpadding="0" cellspacing="0"
background="images/pixi_darkblue.gif">
  <tr>
   <td height="7" colspan="2" valign="top"> <TABLE width="775" BORDER=0 CELLPADDING=0
CELLSPACING=0>
      <TR>
        <TD width="336"><IMG SRC="images/pixi_lightblue.gif" WIDTH=336 HEIGHT=7></TD>
        <TD width="16"><IMG SRC="images/topangle.gif" WIDTH=16 HEIGHT=7></TD>
        <TD background="images/pixi_light_grey.gif"><IMG SRC="images/spacer.gif" WIDTH=423
HEIGHT=7></TD>
      </TR>
    </TABLE></td>
  </tr>
  <tr>
   <td height="118" colspan="2" valign="top"> <table border="0" cellpadding="0" cellspacing="0"
id="navtable">
      <tr>
        <td width="352" rowspan="3"><TABLE width="352" BORDER=0 CELLPADDING=0
CELLSPACING=0>
          <TR>
            <TD><IMG SRC="images/mainpic1.jpg" WIDTH=219 HEIGHT=118></TD>
            <TD><IMG SRC="images/toptreatment1.jpg" WIDTH=133 HEIGHT=118></TD>
          </TR>
        </TABLE></td>
        <td width="423" background="images/pixi_light_grey.gif"> <TABLE BORDER=0
CELLPADDING=0 CELLSPACING=0>
          <TR>
            <TD width="87"><IMG SRC="images/topcurve2.gif" WIDTH="87" HEIGHT="43"></TD>
            <TD width="190" background="images/pixi_light_grey.gif"><IMG
SRC="images/spacer.gif" WIDTH=190 HEIGHT=43></TD>
            <TD width="146" background="images/pixi_light_grey.gif"><img
src="images/toplogoDOWNLOADS.gif" width="146" height="43"></TD>
          </TR>
        </TABLE></td>
      </tr>
      <tr>
        <td width="423" background="images/pixi_light_grey.gif"> <TABLE width="100%"
BORDER=0 CELLPADDING=0 CELLSPACING=0>
          <TR>
            <TD width="277"><IMG SRC="images/topcurve3.gif" WIDTH=277 HEIGHT=20></TD>
            <TD width="146" background="images/pixi_light_grey.gif"><IMG
SRC="images/spacer.gif" WIDTH=146 HEIGHT=20></TD>
          </TR>
        </TABLE></td>
      </tr>
      <tr>
        <td width="423"> <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
```

```
          <TR>
            <TD width="66"><IMG SRC="images/toptreatment1a.jpg" WIDTH=66
HEIGHT=55></TD>
            <TD width="72"><A HREF="index.php"><IMG SRC="images/btn_main.gif"
NAME="btn_main" WIDTH=72 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_main','','images/btn_main-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
            <TD width="89"><A HREF="SHMEIOSEIS.htm"><IMG SRC="images/btn_overview.gif"
NAME="btn_overview" WIDTH=89 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_overview','','images/btn_overview-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
            <TD width="81"><A HREF="DOWNLOADS.htm"><IMG SRC="images/btn_services.gif"
NAME="btn_services" WIDTH=81 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_services','','images/btn_services-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
            <TD width="72"> <A HREF="ASKHSEIS.htm"><IMG SRC="images/products.gif"
NAME="products" WIDTH=72 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('products','','images/products_btn-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
            <TD width="43"><IMG SRC="images/topnavend.gif" WIDTH=43 HEIGHT=55></TD>
          </TR>
        </TABLE></td>
      </tr>
    </table></td>
  </tr>
  <tr>
    <td height="69" colspan="2" valign="top">
      <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
        <TR>
          <TD ROWSPAN=2><IMG SRC="images/mainpic2.jpg" WIDTH=219 HEIGHT=69></TD>
          <TD><IMG SRC="images/toptreatment2.jpg" WIDTH=199 HEIGHT=20></TD>
          <TD width="357" height="69" ROWSPAN=2 bgcolor="#375297"> </TD>
          <TD ROWSPAN=2> </TD>
        </TR>
        <TR>
          <TD width="199" height="49" bgcolor="#375297"> </TD>
        </TR>
      </TABLE>
    </td>
  </tr>
  <tr>
    <td width="775" valign="top">
      <TABLE width="774" BORDER=0 align="center" CELLPADDING=0 CELLSPACING=0>
        <TR>
          <TD width="765" height="250" valign="top" bgcolor="#375297"><p><br>
            &Kappa;&Alpha;&Tau;&Epsilon;&Beta;&Alpha;&Sigma;&Tau;&Epsilon;
&Chi;&Rho;&Eta;&Sigma;&Iota;&Mu;&Alpha;
&Epsilon;&Rho;&Gamma;&Alpha;&Lambda;&Epsilon;&Iota;&Alpha;  ΚΑΙ
&Epsilon;&Beta;&Omicron;&Omicron;&Kappa;S &Pi;&Omicron;&Upsilon; &Theta;&Alpha;
&Sigma;&Alpha;&Sigma; &Beta;&Omicron;&Eta;&Theta;&Eta;&Sigma;&Omicron;&Upsilon;&Nu;
&Sigma;&Tau;&Eta;&Nu; &Kappa;&Alpha;&Lambda;&Upsilon;&Tau;&Epsilon;&Rho;&Eta;
&Kappa;&Alpha;&Tau;&Alpha;&Nu;&Omicron;&Eta;&Sigma;&Eta; &Tau;&Omicron;&Upsilon;
&Mu;&Alpha;&Theta;&Eta;&Mu;&Alpha;&Tau;&Omicron;&Sigma;</p>
          <p align="center"> </p>
          <p align="center"><strong>X&Rho;&Eta;&Sigma;&Iota;&Mu;&Alpha;
&Pi;&Rho;&Omicron;&Gamma;&Rho;&Alpha;&Mu;&Mu;&Alpha;&Tau;&Alpha; </strong></p>
<table width="670" border="0" align="center">
            <tr>
              <td width="357" height="50" align="left">    APACHE HTTP
WEB SERVER 1.3.28 <a href="downloads/apache_1.3.28-win32-x86-no_src.msi"><img
src="images/exe.jpg" width="38" height="35" border="0"></a></td>
```

```
            <td width="297">     PHP Version 4.3.3 <a
href="downloads/install-php4(H%20KALH).exe"><img src="images/exe.jpg" width="38"
height="35" border="0"></a></td>
          </tr>
          <tr>
            <td height="58">
              
        MySQL Version 4.0.16 <a
href="downloads/MYSQL_4_0_16_WIN.ZIP"><img src="images/exe.jpg" width="38" height="35"
border="0"></a></td>
            <td>     MySQL FRONT 2.4 <a href="downloads/MySQL-
Front_2.4_Setup.exe"><img src="images/exe.jpg" width="38" height="35" border="0"></a></td>
          </tr>
          <tr>
            <td
height="49">            
              
           Dev5Beta3 <a
href="downloads/dev5beta3.exe"><img src="images/exe.jpg" width="38" height="35"
border="0"></a></td>
            <td>      PHP Expert Editor  <a
href="downloads/setup.exe"><img src="images/exe.jpg" width="38" height="35"
border="0"></a>   <a
href="downloads/PHP%20Expert%20Editor%203.0%20crack/phpxedit.exe">Crack</a></td>
          </tr>
        </table>
        <p align="center"><strong>E-Books </strong></p>
        <table width="768" border="0">
          <tr>
            <td width="277">  MySQL/PHP DATABASE APPLICATIONS <a
href="downloads/EBOOKS/MySQL-PHP%20Database%20Applications(decrypted).pdf"><img
src="images/acrobat.jpg" width="35" height="35" border="0"></a></td>
            <td width="230">PHP-MySQL WEB DEVELOPMENT <a
href="downloads/EBOOKS/PHP%20and%20MySQL%20Web%20Dev(decrypted).pdf"><img
src="images/acrobat.jpg" width="35" height="35" border="0"></a></td>
            <td width="236">WEB PAGE DESIGN IN 7 DAYS <a
href="downloads/EBOOKS/Web%20page%20design%20in%207%20days.pdf"><img
src="images/acrobat.jpg" width="35" height="35" border="0"></a></td>
          </tr>
        </table>          <p align="left"><strong></strong></p></TD>

    </TR>
  </TABLE>
 </tr>

 <tr>
  <td height="26" colspan="2" valign="top"> <TABLE width="775" BORDER=0
CELLPADDING=0 CELLSPACING=0>
    <TR>
      <TD width="53" height="26" bgcolor="#C7C7CC"> </TD>
      <TD width="722" bgcolor="#C7C7CC" ><div align="right"><span
class="style1">&Pi;&lambda;&eta;&rho;&omicron;&phi;&omicron;&rho;&#943;&epsilon;&sigmaf;
</span><a href="info.htm"><img src="images/mftico.gif" alt="Πληροφορίες..." width="11"
height="11" border="0"></a> </div></TD>
    </TR>
  </TABLE></td>
 </tr>
</table>
</body>
</html>
```

ΤΟ ΑΡΧΕΙΟ SHMEIOSEIS.HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//GR">
<html>
<head>
<title>T.E.I OF EPIRUS e-COMMERCE SITE</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<meta name="keywords" content="Keywords here">
<meta name="description" content="Description here">
<META NAME="robots" CONTENT="index, follow"> <!-- (Robot commands: All, None, Index, No
Index, Follow, No Follow) -->
<META NAME="revisit-after" CONTENT="30 days">
<META NAME="distribution" CONTENT="global">
<META NAME="rating" CONTENT="general">
<META NAME="Content-Language" CONTENT="english">


<script language="JavaScript" type="text/JavaScript" src="images/myfreetemplates.js"></script>
<link href="images/myfreetemplates.css" rel="stylesheet" type="text/css">
<style type="text/css">
<!--
a:link {
	color: #FFFFFF;
	text-decoration: none;
}
a:visited {
	text-decoration: none;
	color: #FFFFFF;
}
a:hover {
	text-decoration: underline;
	color: #FFFFFF;
}
a:active {
	text-decoration: none;
}
.style1 {color: #375297}
.style2 {
	color: #D4D0C8;
	font-weight: bold;
}
.style3 {color: #ed981b}
-->
</style></head>

<body background="images/pixi_lightblue.gif" leftmargin="0" topmargin="0" marginwidth="0"
marginheight="0" onLoad="MM_preloadImages('images/btn_main-over.gif','images/btn_overview-
over.gif','images/btn_services-over.gif','images/products_btn-over.gif')">
<table width="775" height="100%" border="0" align="center" cellpadding="0" cellspacing="0"
background="images/pixi_darkblue.gif">
  <tr>
    <td height="7" colspan="2" valign="top"> <TABLE width="775" BORDER=0 CELLPADDING=0
CELLSPACING=0>
      <TR>
        <TD width="336"><IMG SRC="images/pixi_lightblue.gif" WIDTH=336 HEIGHT=7></TD>
        <TD width="16"><IMG SRC="images/topangle.gif" WIDTH=16 HEIGHT=7></TD>
        <TD background="images/pixi_light_grey.gif"><IMG SRC="images/spacer.gif" WIDTH=423
HEIGHT=7></TD>
      </TR>
```

```
    </TABLE></td>
  </tr>
  <tr>
   <td height="118" colspan="2" valign="top"> <table border="0" cellpadding="0" cellspacing="0"
id="navtable">
      <tr>
       <td width="352" rowspan="3"><TABLE width="352" BORDER=0 CELLPADDING=0
CELLSPACING=0>
          <TR>
           <TD><IMG SRC="images/mainpic1.jpg" WIDTH=219 HEIGHT=118></TD>
           <TD><IMG SRC="images/toptreatment1.jpg" WIDTH=133 HEIGHT=118></TD>
          </TR>
         </TABLE></td>
       <td width="423" background="images/pixi_light_grey.gif"> <TABLE BORDER=0
CELLPADDING=0 CELLSPACING=0>
          <TR>
           <TD width="87"><IMG SRC="images/topcurve2.gif" WIDTH="87" HEIGHT="43"></TD>
           <TD width="190" background="images/pixi_light_grey.gif"><IMG
SRC="images/spacer.gif" WIDTH=190 HEIGHT=43></TD>
           <TD width="146" background="images/pixi_light_grey.gif"><img
src="images/toplogoSHMEIOSEIS.gif" width="146" height="43"></TD>
          </TR>
         </TABLE></td>
      </tr>
      <tr>
       <td width="423" background="images/pixi_light_grey.gif"> <TABLE width="100%"
BORDER=0 CELLPADDING=0 CELLSPACING=0>
          <TR>
           <TD width="277"><IMG SRC="images/topcurve3.gif" WIDTH=277 HEIGHT=20></TD>
           <TD width="146" background="images/pixi_light_grey.gif"><IMG
SRC="images/spacer.gif" WIDTH=146 HEIGHT=20></TD>
          </TR>
         </TABLE></td>
      </tr>
      <tr>
       <td width="423"> <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
          <TR>
           <TD width="66"><IMG SRC="images/toptreatment1a.jpg" WIDTH=66
HEIGHT=55></TD>
           <TD width="72"><A HREF="index.php"><IMG SRC="images/btn_main.gif"
NAME="btn_main" WIDTH=72 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_main','','images/btn_main-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
           <TD width="89"><A HREF="SHMEIOSEIS.htm"><IMG SRC="images/btn_overview.gif"
NAME="btn_overview" WIDTH=89 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_overview','','images/btn_overview-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
           <TD width="81"><A HREF="DOWNLOADS.htm"><IMG SRC="images/btn_services.gif"
NAME="btn_services" WIDTH=81 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('btn_services','','images/btn_services-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
           <TD width="72"> <A HREF="ASKHSEIS.htm"><IMG SRC="images/products.gif"
NAME="products" WIDTH=72 HEIGHT=55 BORDER=0
onMouseOver="MM_swapImage('products','','images/products_btn-over.gif',1)"
onMouseOut="MM_swapImgRestore()"></A></TD>
           <TD width="43"><IMG SRC="images/topnavend.gif" WIDTH=43 HEIGHT=55></TD>
          </TR>
         </TABLE></td>
      </tr>
    </table></td>
  </tr>
```

```
 <tr>
  <td height="69" colspan="2" valign="top">
   <TABLE BORDER=0 CELLPADDING=0 CELLSPACING=0>
    <TR>
     <TD ROWSPAN=2><IMG SRC="images/mainpic2.jpg" WIDTH=219 HEIGHT=69></TD>
     <TD><IMG SRC="images/toptreatment2.jpg" WIDTH=199 HEIGHT=20></TD>
     <TD width="357" height="69" ROWSPAN=2 bgcolor="#375297"> </TD>
     <TD ROWSPAN=2> </TD>
    </TR>
    <TR>
     <TD width="199" height="49" bgcolor="#375297"> </TD>
    </TR>
   </TABLE>
  </td>
 </tr>
 <tr>
  <td width="775" valign="top">
   <TABLE width="774" BORDER=0 CELLPADDING=0 CELLSPACING=0>
    <TR>
     <TD width="765" height="250" valign="top" bgcolor="#375297"><p><br>
     &Gamma;&Rho;&Alpha;&Phi;&Eta;&Mu;&Alpha;
&Epsilon;&Pi;&Iota;&Kappa;&Epsilon;&Psi;&Epsilon;&Omega;&Nu; &Gamma;&Iota;&Alpha;
&Tau;&Iota;&Sigma; &Tau;&Lambda;&Epsilon;&Upsilon;&Tau;&Alpha;&Iota;&Epsilon;&Sigma;
&Mu;&Epsilon;&Rho;&Epsilon;&Sigma;
&Eta;&Mu;&Epsilon;&Rho;&Omicron;&Mu;&Eta;&Nu;&Iota;&Alpha;
&Epsilon;&Pi;&Iota;&Sigma;&Kappa;&Epsilon;&Psi;&Epsilon;&Iota;&Sigma;
&Gamma;&Rho;&Alpha;&Phi;&Eta;&Mu;&Alpha; </p>
      <p align="center"><strong>&Delta;&Iota;&Alpha;&Lambda;&Epsilon;&Xi;&Tau;&Epsilon;
&Mu;&Iota;&Alpha; &Alpha;&Pi;&Omicron;
&Tau;&Iota;&Sigma; &Pi;&Alpha;&Rho;&Alpha;&Kappa;&Alpha;&Tau;&Omega;
&Epsilon;&Iota;&Kappa;&Omicron;&Nu;&Epsilon;&Sigma; &Gamma;&Iota;&Alpha;
&Nu;&Alpha; &Mu;&Alpha;&Theta;&Epsilon;&Tau;&Epsilon; &Gamma;&Iota;&Alpha;
&Tau;&Iota;&Sigma; &Pi;&Alpha;&Rho;&Alpha;&Kappa;&Alpha;&Tau;&Omega;
&Tau;&Epsilon;&Chi;&Nu;&Omicron;&Lambda;&Omicron;&Gamma;&Iota;&Epsilon;&Sigma;
</strong></p>
     <p> </p>      <table width="766" height="156" bordercolor="#375297">
      <tr>
       <td width="226" height="62" align="center"><a
href="shmeioseis/mysql%20lesson.html"><img src="images/MYSQL%20PIC.gif" width="132"
height="39" border="0"></a></td>
       <td width="226" align="center" bordercolor="#FF0000"><a
href="shmeioseis/&#924;&#940;&#952;&#951;&#956;&#945;%20&#947;&#953;&#945;%20&#964;
&#951;&#957;%20&#932;&#949;&#967;&#957;&#959;&#955;&#959;&#947;&#943;&#945;%20C
SS.htm"><img src="images/vcss.png" width="132" height="39" border="0"></a></td>
       <td width="226" align="center"><img src="images/valid-html401.png" width="132"
height="39"></td>
      </tr>
      <tr>
       <td height="62"
align="left">             <
a href="shmeioseis/php%20language.htm"><img src="images/php%20PIC.gif" width="132"
height="39" border="0"></a></td>

       <td                                                    align="left"
bordercolor="#FF0000">        &
nbsp;   <a
href="shmeioseis/APACHE%20WEB%20SERVER.htm"><img
src="images/APACHE.jpg" width="132" height="39" border="0"></a></td>
```

```
          <td>        <a
href="shmeioseis/JAVASCRIPT%20LESSONS.htm"><img src="images/javascript2.gif" width="192"
height="42" border="0"></a></td>
        </tr>
        <tr>
          <td height="62"><h2 align="center"><a
href="shmeioseis/WebBuilding%20lesson.html">Web Building</a> </h2></td>
          <td bordercolor="#FF0000"><div align="center">
            <h2><a
href="shmeioseis/databases%20lesson.htm">&Beta;&#940;&sigma;&epsilon;&iota;&sigmaf;
&Delta;&epsilon;&delta;&omicron;&mu;&#941;&nu;&omega;&nu; </a></h2>
          </div></td>
          <td><h2>     <a href="shmeioseis/client-
server%20lesson.htm">Client - Server</a></h2></td>
        </tr>
       </table>
        <table width="762" bordercolor="#FF0000">
          <tr>
            <td><strong>&Pi;&Alpha;&Rho;&Tau;&Epsilon;
&Mu;&Epsilon;&Rho;&Omicron;&Sigma; &Sigma;&Tau;&Omicron;
&Delta;&Eta;&Mu;&Omicron;&Psi;&Eta;&Phi;&Iota;&Sigma;&Mu;&Alpha;
&Mu;&Alpha;&Sigma; </strong></td>
            <td><div align="right" class="style2"><a href="fstats-view.php"
class="style3">&Delta;&Epsilon;&Iota;&Tau;&Epsilon; &Tau;&Alpha;
&Sigma;&Tau;&Alpha;&Tau;&Iota;&Sigma;&Tau;&Iota;&Kappa;&Alpha;
&Tau;&Omicron;&Upsilon; SITE &Mu;&Alpha;&Sigma; </a></div></td>
          </tr>
        </table>          <p align="left"> </p>
        <p> </p>         </TD>

      </TR>
    </TABLE>
  </tr>

  <tr>
    <td height="26" colspan="2" valign="top"> <TABLE width="775" BORDER=0
CELLPADDING=0 CELLSPACING=0>
      <TR>
        <TD width="53" height="26" bgcolor="#C7C7CC"> </TD>
        <TD width="722" bgcolor="#C7C7CC" ><div align="right"><span
class="style1">&Pi;&lambda;&eta;&rho;&omicron;&phi;&omicron;&rho;&#943;&epsilon;&sigmaf;
</span><a href="info.htm"><img src="images/mftico.gif"
alt="&Pi;&lambda;&eta;&rho;&omicron;&phi;&omicron;&rho;&#943;&epsilon;&sigmaf;..."
width="11" height="11" border="0"></a> </div></TD>
      </TR>
    </TABLE></td>
  </tr>
</table>
</body>
</html>
```

ΤΟ ΑΡΧΕΙΟ COUNTER.PHP

```
<?php
        // ΑΝΟΙΓΜΑ ΤΟΥ ΑΡΧΕΙΟΥ

        $file = "stats";
        if(!($fp = fopen($file , "r")))
                die ("Could not open the requested file!  You may need to CHMOD 777 it!");
```

```
$count = trim(fgets($fp));
$selection = trim(fgets($fp));

fclose($fp);

$count++;

$fp = fopen($file , "w");

fwrite($fp, $count);
fwrite($fp, "\n");
fwrite($fp, $selection);

fclose($fp);
$count_image = "<img src='images/".$selection."/";
for($i = 0; $i < strlen($count); $i++){
        $temp = $count_image.substr($count, $i, 1).".gif' />";
        print $temp;

}



// ΑΥΞΗΣΗ ΤΟΥ COUNTER

//ΕΓΓΡΑΦΗ ΣΤΟ ΑΡΧΕΙΟ

// Loop ΜΕΧΡΙ ΝΑ ΤΥΠΩΣΕΙ ΤΙΣ ΕΙΚΟΝΕΣ ΤΟΥ COUNTER


?>
```

ΤΟ ΑΡΧΕΙΟ FSTATS.PHP

```
<?


require "config.php";

$date = date("Y") . "-" . date("m") . "-" . date("d");

$strsql="SELECT * FROM stats_day where date='$date'";

$xConn=mysql_connect($host,$uid,$pwd);
mysql_select_db($db,$xConn);

if (!($result = mysql_query($strsql)))
{
  $strError0="Open Stats2 Database Failed<br>";
  $strError1="Error Number: ".mysql_errno().$BR;
  $strError2="Error Description: ".mysql_error.$BR;
  $strError5="Query: ".$strsql.$BR.$BR."strWhere: ".$strWhere.$BR;
  $strError0=$BR.$strError0.$strError1.$strError2.$strError5;
  echo $strError0;
  break;
}
else
```

```
{
  $intTotalRecs=mysql_num_rows($result);
  if ($intTotalRecs>0) mysql_query("UPDATE stats_day SET hits = hits + 1 WHERE date='$date'");
  else mysql_query("INSERT INTO stats_day SET date='$date',hits='1'");
}

mysql_close();

?>
```

ΤΟ ΑΡΧΕΙΟ FSTATSVIEW.PHP

```
<?


require "config.php";

if(isset($_GET['last_x_days'])) $last_x_days=$_GET['last_x_days']; # register globals off?
if (isset($last_x_days))
{
  $last_x_days=round($last_x_days);
  if ($last_x_days>0) $limit=$last_x_days;
}


if ($no_cache == 1)
{
  # no cache
  header("Expires: Tue, 17 Jun 2003 01:00:00 GMT");                    # ΗΜΕΡΟΜΗΝΙΑ
  header("Last-Modified: " . gmdate("D, d M Y H:i:s") . " GMT");       #ΠΑΝΤΑ ΤΡΟΠΟΠΟΙΗΜΕΝΗ
  header("Cache-Control: no-store, no-cache, must-revalidate");       # HTTP/1.1
  header("Cache-Control: post-check=0, pre-check=0", false);
  header("Pragma: no-cache");                                         # HTTP/1.0
}

function diff_of_days($start_date,$end_date)
{
       // Gets the difference in days between two dates.(YYYY-MM-DD)
      // total numbers of days in range = diff_of_days($start_date,$end_date) + 1
       $s_parts = explode("-",$start_date);
       $e_parts = explode("-",$end_date);
       $s_timestamp = mktime(0,0,0,$s_parts[1],$s_parts[2],$s_parts[0]);
       $e_timestamp = mktime(0,0,0,$e_parts[1],$e_parts[2],$e_parts[0]);
       $difference = abs($e_timestamp - $s_timestamp)/86400;
       return($difference);
}

function date_x_days_ago($start_date,$xdays)
{
       // Gets the date x days before $start_date. (YYYY-MM-DD)
       $s_parts = explode("-",$start_date);
       $e_timestamp = mktime(0,0,0,$s_parts[1],$s_parts[2]-$xdays,$s_parts[0]);
       $end_date = date("Y-m-d",$e_timestamp);
       return($end_date);
}

$today = date("Y-m-d");
$time1 = date("H:i:s");
```

```
$time2 = date("g:i a");


?><html><head>
<title>ΣΤΑΤΙΣΤΙΚΑ SITE ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ</title>
<meta http-equiv="Content-Style-Type" content="text/css">
<style type="text/css">
<!--
BODY {
	COLOR: #344684;
	FONT-FAMILY: Arial, Helvetica, sans-serif;
	SCROLLBAR-FACE-COLOR: #0080FF;
	SCROLLBAR-HIGHLIGHT-COLOR: #FFFFFF;
	SCROLLBAR-SHADOW-COLOR: #5b6bb6;
	SCROLLBAR-3DLIGHT-COLOR: #000000;
	SCROLLBAR-ARROW-COLOR: #fdfc65;
	SCROLLBAR-TRACK-COLOR: #B9C9FF;
	SCROLLBAR-DARKSHADOW-COLOR: #000000;
	background-color: #375297;
}
A:hover { FONT-FAMILY: Arial, Helvetica, sans-serif; COLOR: #FF0000; FONT-WEIGHT: bold;
TEXT-DECORATION: underline
}
.style5 {font-size: 12px; font-weight: bold; }
.style6 {
	font-size: 18px;
	font-weight: bold;
}
.style10 {font-size: 12px; font-weight: bold; color: #375297; }
.style11 {color: #FFFFFF}
.style15 {
	font-size: 24px;
	font-weight: bold;
	color: #FFFFFF;
}
.style16 {font-size: 10px}
.style17 {color: #000000}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"></head>
<body>

<?


do {	// ΞΕΚΙΝΑ Η MAIN
============================================================



$xConn=mysql_connect($host,$uid,$pwd);
mysql_select_db($db,$xConn);


# find first date recorded
$strsql="SELECT * FROM stats_day ORDER BY date LIMIT 0, 1";
if (!($result = mysql_query($strsql)))
{
  $strError0="Open Stats Database Failed<br>";
  $strError1="Error Number: ".mysql_errno().$BR;
  $strError2="Error Description: ".mysql_error.$BR;
```

```
  $strError5="Query: ".$strsql.$BR.$BR."strWhere: ".$strWhere.$BR;
  $strError0=$BR.$strError0.$strError1.$strError2.$strError5;
  echo $strError0;
  break;
}
$intTotalRecs=mysql_num_rows($result);
if (($intTotalRecs==0))
  { ?>
<P align=center class="style11"><font size=5>No days recorded in database.</font>
<p class="style11">
<? break;
  }
$row=mysql_fetch_row($result);
$first_day=$row[0];
$ndays = diff_of_days($first_day,$today)+1;
$ndaysf = number_format($ndays);


if ($limit>$ndays) $limit=$ndays;
$start_date = date_x_days_ago($today,$limit-1);
if (($start_date < $first_day)) $start_date = $first_day;


$strsql="SELECT * FROM stats_day WHERE date >= '$start_date' ORDER BY date DESC LIMIT 0,
$limit";
if (!($result = mysql_query($strsql)))
{
  $strError0="Open Stats Database Failed<br>";
  $strError1="Error Number: ".mysql_errno().$BR;
  $strError2="Error Description: ".mysql_error.$BR;
  $strError5="Query: ".$strsql.$BR.$BR."strWhere: ".$strWhere.$BR;
  $strError0=$BR.$strError0.$strError1.$strError2.$strError5;
  echo $strError0;
  break;
}
$intTotalRecs=mysql_num_rows($result);
if (($intTotalRecs==0))
  { ?>
<P align=center class="style11"><font size=5>No non-zero days recorded in database within specified
range.</font>
<p class="style11">
<P align=center><span class="style11"><b>&Omicron;&Iota;
&Epsilon;&Pi;&Iota;&Sigma;&Kappa;&Epsilon;&Psi;&Epsilon;&Iota;&Sigma;
&Kappa;&Alpha;&Tau;&Alpha;&Mu;&Epsilon;&Tau;&Rho;&Omicron;&Upsilon;&Nu;&Tau;&Alp
ha;&Iota; &Alpha;&Pi;&Omicron;</b></span> <font color=blue><b><? echo $first_day;
?></b></font> <span class="style11">(<? echo $ndaysf; ?>
&Mu;&Epsilon;&Rho;&Epsilon;&Sigma;)</span></p>
<P align=center class="style6"> </p>
<? break;
  }


$x=0;
$high_hits=0;
$total_hits=0;
$high_day="";
$newest_date="";
$newest2_date="";

while(($row=mysql_fetch_row($result)))
{
```

```
  $date[$x]=$row[0];
  $hits[$x]=$row[1];
  $total_hits+=$row[1];
  if($x==0) { $newest_date = $row[0]; $newest_hits=$row[1]; }
  if($x==1) { $newest2_date = $row[0]; $newest2_hits=$row[1]+$newest_hits; }
  if($row[1]>$high_hits) { $high_hits=$row[1]; $high_day=$row[0];}
  if (!isset($low_hits)) { $low_hits=$row[1];  $low_day=$row[0]; }
  if($row[1]<$low_hits)  { $low_hits=$row[1];  $low_day=$row[0];}
  $x++;
}


$number_days=$x;



# fill in array with days that had 0 hits and therefore not recorded
$date[$x]="0000-00-00";                // ΔΗΜΙΟΥΡΓΗΣΕ ΤΙΣ ΝΕΕΣ ΤΙΜΕΣ ΤΟΥ ΠΙΝΑΚΑ ΓΙΑ ΝΑ
                                       / /ΑΠΟΦΥΓΟΥΜΕ ΤΑ ΛΑΘΗ
$hits[$x]=0;
if($number_days==$limit) {$date2=$date; $hits2=$hits;} else
{
$currentday=$today;
$currentindex=0;
for ($x=0; $x<$limit; $x++)
{
  $date2[$x]=$currentday;
  if ($date[$currentindex]==$currentday)
  {
    $hits2[$x]=$hits[$currentindex];
    $currentindex++;
  }
  else
  {
    $hits2[$x]=0;
  }
  if (!isset($low_hits)) { $low_hits=$hits2[0];  $low_day=$date2[0]; }
  if($hits2[$x]<=$low_hits)  { $low_hits=$hits2[$x];  $low_day=$date2[$x];}
  $currentday = date_x_days_ago($currentday,1);
}
}



$date=$date2;
$hits=$hits2;
$number_days=$limit;
$today = date("Y") . "-" . date("m") . "-" . date("d");


$proj_hits = 0;
$proj_hitsf = 0;
if($newest_date==$today)
{
  $mins = date("i") + date("G")*60.0;        # ΤΑ ΛΕΠΤΑ
  $proj_hits = $newest_hits * (1440.0/$mins);
  if(isset($newest2_hits))
    $proj_hits = $newest_hits + ( (1440.0-$mins)*($newest2_hits/($mins+1440.0)) );
  $proj_hits = round($proj_hits);
  $proj_hitsf = number_format($proj_hits);
}
```

```
$average_hits = ($total_hits + $proj_hits - $hits2[0]) / $number_days;
$average_hits =  round($average_hits,2);
$total_hitsf   = number_format($total_hits);
$high_hitsf    = number_format($high_hits);
$low_hitsf     = number_format($low_hits);
$average_hitsf = number_format($average_hits);
if($high_hits==0) $high2=1;
else
{
  $high2=$high_hits;
  if ($proj_hits>$high2) $high2=$proj_hits;
}



?>

<div align="left"></div>
<div align=center><center>
<table border=0 cellpadding=10 cellspacing=0><tr><td valign=top>

<div align=center><center>
<table border=1 cellpadding=5 cellspacing=0>
<tr bgColor="#FFFFFF">
<th colspan=2 align=center><span class="style10">&Gamma;&Epsilon;&Nu;&Iota;&Kappa;&Alpha;
&Sigma;&Tau;&Alpha;&Tau;&Iota;&Sigma;&Tau;&Iota;&Kappa;&Alpha; </span></th>
</tr>
<tr>
<td bgcolor="#B9C9FF"><span class="style5">Projected hits for today</span></td>
<td align=center><font color=WHITE><b><? echo $proj_hitsf; ?></b></font>
</td>
</tr>
<tr>
<td bgcolor="#B9C9FF"><span
class="style5">&Eta;&Mu;&Epsilon;&Rho;&Omicron;&Mu;&Eta;&Nu;&Iota;&Alpha; Server
</span></td>
<td align=center><font color=WHITE><b><? echo $today; ?></b></font></td>
</tr>
<tr>
<td bgcolor="#B9C9FF"><span class="style5">&Omega;&Rho;&Alpha; Server </span></td>
<td align=center><font color=WHITE><b><? echo $time1; ?></b></font><b> </b><span
class="style11">(<? echo $time2; ?>)</span></td>
</tr>
<tr>
<td bgcolor="#B9C9FF"><span
class="style5">&Kappa;&Alpha;&Tau;&Alpha;&Mu;&Epsilon;&Tau;&Rho;&Eta;&Sigma;&Eta;
&Alpha;&Pi;&Omicron;</span></td>
<td align=center><font color=WHITE><b><? echo $first_day; ?></b></font><span class="style11">
(<? echo $ndaysf; ?> <span
class="style16">&Mu;&Epsilon;&Rho;&Epsilon;&Sigma;</span>)</span></td>
</tr>
</table>
</center></div>

</td><td>

<div align=center><center>
<table border=1 cellpadding=5 cellspacing=0>
<tr bgcolor="#FFFFFF">
<th colspan=3 align=center><span
class="style10">&Epsilon;&Pi;&Iota;&Sigma;&Kappa;&Epsilon;&Psi;&Epsilon;&Iota;&Sigma;
```

```
&Tau;&Iota;&Sigma;
&Tau;&Epsilon;&Lambda;&Epsilon;&Upsilon;&Tau;&Alpha;&Iota;&Epsilon;&Sigma;</span><font
color="#B9C9FF"> <font color="BLACK" size="+1"><? echo $number_days; ?></font>
</font><span class="style10">&Mu;&Epsilon;&Rho;&Epsilon;&Sigma;</span> </th>
</tr>
<tr>
<td bgcolor="#B9C9FF"><span
class="style5">&Sigma;&Upsilon;&Nu;&Omicron;&Lambda;&Omicron;</span></td>
<td colspan=2 align=center><font color=WHITE><b><? echo $total_hitsf; ?></b></font></td>
</tr>
<tr>
<td bgcolor="#B9C9FF"><span class="style5">&Mu;&Epsilon;&Sigma;&Omicron;&Sigma;
&Omicron;&Rho;&Omicron;&Sigma; </span></td>
<td colspan=2 align=center><font color=WHITE><b><? echo $average_hitsf; ?></b></font></td>
</tr>
<tr>
<td bgcolor="#B9C9FF"><span
class="style5">&Mu;&Epsilon;&Gamma;&Iota;&Sigma;&Tau;&Omicron;</span></td>
<td align=right><font color=WHITE><b><? echo $high_hitsf; ?></b></font></td>
<td align=center><font color=WHITE><b><? echo $high_day; ?></b></font>
</td>
</tr>
<tr>
<td bgcolor="#B9C9FF"><span
class="style5">&Epsilon;&Lambda;&Alpha;&Chi;&Iota;&Sigma;&Tau;&Omicron;</span></td>
<td align=right><font color=WHITE><b><? echo $low_hitsf; ?></b></font></td>
<td align=center><font color=WHITE><b><? echo $low_day; ?></b></font></td>
</tr>
</table></center></div>

</td></tr>
</table></center></div>


<?


echo "
<h1 align=center><font size=5 color=#FFFFFF face=Arial>ΓΡΑΦΗΜΑ ΕΠΙΣΚΕΨΕΩΝ ΓΙΑ ΤΙΣ
ΤΕΛΕΥΤΑΙΕΣ $limit ΗΜΕΡΕΣ</font></h1><p>
<center>

<table border=1 cellPadding=0 cellSpacing=0><tr><td bgColor=\"#EEEEEE\">
<table border=0 cellPadding=5 cellSpacing=0>

<tr bgColor=\"#B9C9FF\"><th
align=center>&Eta;&Mu;&Epsilon;&Rho;&Omicron;&Mu;&Eta;&Nu;&Iota;&Alpha;</th><th
align=center>ΕΠΙΣΚΕΨΕΙΣ</th><th align=center>ΓΡΑΦΗΜΑ</th></tr>
";

for ($x=0; $x<$number_days; $x++)                    # ΔΕΙΧΝΕΙ ΤΟ ΓΡΑΦΗΜΑ
{
$width=$bar_width * ($hits[$x] / $high2);
$width=round($width);
$hitsf = number_format($hits[$x]);
$projbar="";
if ($proj_hits>0 && $x==0)
{
  $width2 = $bar_width * (($proj_hits-$hits[$x])/ $high2);
  $width2 = round($width2);
  if ($width2>0)
```

```
   $projbar = "<img src=\"$bar2_image\" border=0 width=$width2 height=$bar_height alt=\"Projected:
$proj_hits\">";
}
echo "<tr>
<td>$date[$x]</td>
<td align=right><b>$hitsf</b></td>
<td><img src=\"$bar_image\" border=0 width=$width height=$bar_height>$projbar</td>
</tr>\n";
}

echo "\n</table><td></tr></table>\n\n</center>\n\n";



} while(0);

mysql_close();

?>

<br>

<div align=center><center>
<table border=1 cellspacing=0 cellpadding=10 bgcolor="#C0C0C0"><tr><td>
<form method="GET">
<p align=center><b>&Sigma;&Tau;&Alpha;&Tau;&Iota;&Sigma;&Tau;&Iota;&Kappa;&Alpha;
&Gamma;&Iota;&Alpha; &Tau;&Iota;&Sigma;
&Tau;&Epsilon;&Lambda;&Epsilon;&Upsilon;&Tau;&Alpha;&Iota;&Epsilon;&Sigma;
   <input type="text" name="last_x_days" size=5 value="<? echo $limit?>">
   &Mu;&Epsilon;&Rho;&Epsilon;&Sigma;.<br>
<input type="submit" value="Submit"><input type="reset" value="Reset"></b></p>
</form></td></tr>
</table></center></div>

<p class="style11">

<p align=center><span
class="style15">&Sigma;&Tau;&Alpha;&Tau;&Iota;&Sigma;&Tau;&Iota;&Kappa;&Alpha;
&Epsilon;&Pi;&Iota;&Sigma;&Kappa;&Epsilon;&Psi;&Iota;&Mu;&Omicron;&Tau;&Eta;&Tau;&Al
pha;&Sigma; TOY SITE MA&Sigma;</span><font size=2><br>
</font>
<p class="style17">
</body>
</html>
```

ΤΟ ΑΡΧΕΙΟ LPTEST.PHP

```
<?php

include("lp_source.php");

?>

<html>

<head>
```

```
<title>T.E.I OF EPIRUS e-COMMERCE SITE ΔΗΜΟΨΗΦΙΣΜΑ</title>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-9"><style type="text/css">
<!--
a:link {
	color: #FFFFFF;
	text-decoration: none;
}
a:visited {
	text-decoration: none;
	color: #FFFFFF;
}
a:hover {
	text-decoration: underline;
}
a:active {
	text-decoration: none;
	color: #FFFFFF;
}
.style3 {font-size: 14px}
-->
</style>
<basefont size="2" face="Verdana">

</head>

<body bgcolor="#375297" text="#FFFFFF">

<center>

<br>

<h1>ΨΗΦΙΣΤΕ ΓΙΑ ΤΟΝ ΤΡΟΠΟ ΕΞΕΤΑΣΗΣ ΤΟΥ ΜΑΘΗΜΑΤΟΣ!! </h1>

<br>

<br>

<table border=0 width=540 bgcolor=#FFFFFF>

	<tr>

	  <td width="534"><font size=2 color=#000000>

	    <strong><?php echo($mainstr); ?></strong>

	</td>

   </tr>

   <tr>

	<td bgcolor="#375297">

	  <font size="1" color="#FFFFFF">

	  <br>

	  <blockquote>

	  <?php echo($question); ?><br><br>
```

```
                <?php

                if($votingstep==1) { echo($step1str); }

                if($votingstep==2) { echo($step2str); }

                if($votingstep==3) { echo($step3str); }

                ?><br>

                ΑΡΙΘΜΟΣ ΨΗΦΩΝ: <?php echo($totalvotes); ?>

    </blockquote></td>

  </tr>

  <tr>

    <td bgcolor="#375297">

      <font size="2" color="#FFFFFF">        <blockquote>
        <h5><br>
            <span class="style3"><a href="lp_silly.php">Link to other Page</a><br>

          <a href="lp_recookie.php">ΞΑΝΑΨΗΦΙΣΤΕ !!</a></span></h5>
      </blockquote>

    </td>

  </tr>

</table>

</center>
</body>

</html>
```

ΤΟ ΑΡΧΕΙΟ LPSETTINGS.PHP

```
<?php

//ΣΕΤΑΡΙΣΜΑ ΤΟΥ ΔΗΜΟΨΗΦΙΣΜΑΤΟΣ



$callingfile="lp_test.php";

// ΑΡΧΕΙΟ ΠΟΥ ΚΡΑΤΟΥΝΤΑΙ ΟΙ ΨΗΦΟΙ
$filename = "lplist.txt";

// password
 $pwd="mitsos";

//ΧΡΟΝΟΣ ΜΕΤΑΞΥ ΤΩΝ ΨΗΦΩΝ
$time_between_votes = 30000;

//ΠΛΑΤΟΣ ΓΡΑΦΗΜΑΤΟΣ
$graph_width = 100;
```

```
// ΥΨΟΣ ΓΡΑΦΗΜΑΤΟΣ
$graph_height = 6;

// ΜΥΝΗΜΑΤΑ
$message1="ÓÔÇÑÉÎÅ ÔÁ ÐÉÓÔÅÕÙ ÓÏÕ!";
$message2="ÅÕ×ÁÑÉÓÔÏÕÌÅ ÐÏÕ ØÇÖÉÓÁÔÅ!";
$message3="ÔÁ ÁÐÏÔÅËÅÓÌÁÔÁ ÔÏÕ ÄÇÌÏØÇÖÉÓÌÁÔÏÓ";


$vote_str="&Psi;&Eta;&Phi;&Iota;&Sigma;&Tau;&Epsilon;";


$buttonstyle="";
?>
```

ΤΟ ΑΡΧΕΙΟ LPRECOOKIE.PHP

```
<?php



if (isset($HTTP_COOKIE_VARS["votingstep"])) {

      setcookie("votingstep","1",time()+1);

}



?>



<html>

<head>

<title>T.E.I OF EPIRUS e-COMMERCE SITE ΔΗΜΟΨΗΦΙΣΜΑ</title>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"><style type="text/css">
<!--
body {
      background-color: #375297;
}
.style2 {font-size: 16px}
a:link {
      color: #FFFFFF;
      text-decoration: none;
}
a:visited {
      text-decoration: none;
      color: #FFFFFF;
}
a:hover {
      text-decoration: underline;
      color: #FFFFFF;
}
a:active {
      text-decoration: none;
      color: #FFFFFF;
```

```
}
.style4 {font-size: 10px}
-->
</style>
<basefont size="2" face="Verdana">

</head>

<body text="#FFFFFF">

<center>
  <h2>&Xi;&Alpha;&Nu;&Alpha;&Psi;&Eta;&Phi;&Iota;&Sigma;&Tau;&Epsilon;!! <span
class="style2"><br>

   </span><br>
</h2>
  <table width=461 height="389" border=0 bgcolor=#FFFFFF>

      <tr>

         <td width="455" height="21"><font size=2 color=#000000>


<strong>&Xi;&Alpha;&Nu;&Alpha;&Psi;&Eta;&Phi;&Iota;&Sigma;&Tau;&Epsilon;!!</strong>

      </td>

   </tr>

   <tr>

      <td bgcolor="#375297">

         <font size="2" color="#FFFFFF">         <div align="center">
           <h3>&Eta;
&Pi;&Rho;&Omicron;&Eta;&Gamma;&Omicron;&Upsilon;&Mu;&Epsilon;&Nu;&Eta;
&Psi;&Eta;&Phi;&Omicron;&Sigma; &Sigma;&Alpha;&Sigma; &Epsilon;&Chi;&Epsilon;&Iota;
&Kappa;&Alpha;&Tau;&Omega;&Chi;&Epsilon;&Iota;&Rho;&Omega;&Theta;&Epsilon;&Iota;&nb
sp;&Omicron;&Mu;&Omega;&Sigma;   </h3>
           <h3> &Mu;&Pi;&Omicron;&Rho;&Epsilon;&Iota;&Tau;&Epsilon; &Nu;&Alpha;
&Xi;&Alpha;&Nu;&Alpha;&Psi;&Eta;&Phi;&Iota;&Sigma;&Epsilon;&Tau;&Epsilon; !!<br>
           </h3>
         </div>
     <blockquote>
       <div align="center"><span class="style4">  * H
&Lambda;&Upsilon;&Sigma;&Eta; &Alpha;&Upsilon;&Tau;&Eta;
&Pi;&Rho;&Omicron;&Tau;&Epsilon;&Iota;&Nu;&Epsilon;&Tau;&Epsilon;
&Mu;&Omicron;&Nu;&Omicron; &Alpha;&Nu;
&Nu;&Omicron;&Mu;&Iota;&Zeta;&Epsilon;&Tau;&Epsilon; &Pi;&Omega;&Sigma; &Eta;
&Pi;&Rho;&Omicron;&Eta;&Gamma;&Omicron;&Upsilon;&Mu;&Epsilon;&Nu;&Eta;
 &Alpha;&Pi;&Alpha;&Nu;&Tau;&Eta;&Sigma;&Eta; &Sigma;&Alpha;&Sigma;
&Delta;&Epsilon;&Nu; &Sigma;&Alpha;&Sigma;
&Epsilon;&Kappa;&Phi;&Rho;&Alpha;&Zeta;&Epsilon;&Iota;
&Pi;&Rho;&Alpha;&Gamma;&Mu;&Alpha;&Tau;&Iota;&Kappa;&Alpha;
&Alpha;&Lambda;&Lambda;&Iota;&Omega;&Sigma; &Tau;&Alpha;
&Alpha;&Pi;&Omicron;&Tau;&Epsilon;&Lambda;&Epsilon;&Sigma;&Mu;&Alpha;&Tau;&Alpha;
&Pi;&Omicron;&Upsilon; &Theta;&Alpha;
&Epsilon;&Xi;&Alpha;&Chi;&Theta;&Omicron;&Upsilon;&Nu; &Theta;&Alpha;
&Epsilon;&Iota;&Nu;&Alpha;&Iota;
&Pi;&Lambda;&Alpha;&Sigma;&Mu;&Alpha;&Tau;&Iota;&Kappa;&Alpha;!! </span></div>
     </blockquote></td>
```

```
   </tr>

   <tr>

      <td height="107" bgcolor="#375297">

         <font size="2" color="#FFFFFF">                  <blockquote>
           <div align="center"><a
href="lp_test.php">&Epsilon;&Pi;&Iota;&Sigma;&Tau;&Rho;&Omicron;&Phi;&Eta;
 &Sigma;&Tau;&Omicron;
&Delta;&Eta;&Mu;&Omicron;&Psi;&Eta;&Phi;&Iota;&Sigma;&Mu;&Alpha;!! </a></div>
     </blockquote>      </td>

   </tr>
</table>

</center>
</body>

</html>
```

ΤΟ ΑΡΧΕΙΟ LP_ADMIN.PHP

```
<?php include("lp_settings.inc"); ?>

<html>

<head>

<title>T.E.I OF EPIRUS e-COMMERCE SITE ΔΗΜΟΨΗΦΙΣΜΑ</title>

<meta http-equiv="Content-Type" content="text/html; charset=windows-1253"><style
type="text/css">
<!--
body {
      background-color: #375297;
}
a:link {
      color: #FFFFFF;
      text-decoration: none;
}
a:visited {
      text-decoration: none;
      color: #FFFFFF;
}
a:hover {
      text-decoration: underline;
      color: #FFFFFF;
}
a:active {
      text-decoration: none;
      color: #FFFFFF;
}
-->
</style><basefont size="2" face="Verdana">

</head>
```

```
<body text="#FFFFFF">

<center>

<br>

<h1> Admin Center ΓΙΑ ΤΟ ΔΗΜΟΨΗΦΙΣΜΑ ΤΟΥ</h1>
<h1> ΗΛΕΚΤΡΟΝΙΚΟΥ ΕΜΠΟΡΙΟΥ</h1>
<br>

<br>

<?php


function getIP() {

	if (getenv("HTTP_CLIENT_IP")) $ip = getenv("HTTP_CLIENT_IP");

	else if(getenv("HTTP_X_FORWARDED_FOR")) $ip =
getenv("HTTP_X_FORWARDED_FOR");

	else if(getenv("REMOTE_ADDR")) $ip = getenv("REMOTE_ADDR");

	else $ip = "UNKNOWN";

return $ip;

}


function write2log ($linetoadd) {

	$rightnow=date("F j, Y, g:i a");

	$fplog=fopen('lp_log.dat', "a");

	fputs($fplog, getIP()."|".$rightnow."|".$linetoadd."\n");

	fclose($fplog);

}


import_request_variables("p", "p_");


if (!isset($p_adminstep)) {

	$adminstep=0;

	} else { $adminstep = $p_adminstep; }


if ($adminstep==0) {
```

```
write2log("Enters admincenter");

$adminstep0str='<form action="lp_admin.php" method="post" name="form0">Admin
Password:<input type="hidden" name="adminstep" value="'.($adminstep+1).'"><input
type="Password" name="pwd" size="20"><br><input type="Submit" value="OK"></form>';

}


if ($adminstep==1) {

if ($p_pwd == $pwd) {

$adminstep1str='<form action="lp_admin.php" method="post" name="form1">ΑΡΙΘΜΟΣ
ΑΠΑΝΤΗΣΕΩΝ:<input type="hidden" name="adminstep" value="'.($adminstep+1).'"><input
type="Text" name="numofitems" size="5"><br>ΕΡΩΤΗΣΗ:   <input type="Text"
name="question" size="35"><br><input type="Submit" value="OK"></form>';

}

else {

$adminstep1str='ΛΑΘΟΣ Password, ΠΡΟΣΠΑΘΗΣΕ ΞΑΝΑ...<br><br><a
href="lp_admin.php">Back</a>';

write2log("WRONG PASSWORD ENTERED IN ADMINCENTER");

}



}



if($adminstep==2) :

$adminstep2str='<form action="lp_admin.php" method="post" name="form2"><input
type="hidden" name="question" value="'.$p_question.'"><input type="hidden" name="adminstep"
value="'.($adminstep+1).'">';

if (isset($p_numofitems)) {

$hector=$p_numofitems+0;

if($p_numofitems<=0) {

$adminstep=0;

$errtext="Ο ΑΡΙΘΜΟΣ ΤΩΝ ΑΠΑΝΤΗΣΕΩΝ ΠΡΕΠΕΙ ΝΑ ΕΙΝΑΙ ΑΚΕΡΑΙΟΣ
ΑΡΙΘΜΟΣ ΚΑΙ ΜΕΓΑΛΥΤΕΡΟΣ ΤΟΥ ΜΗΔΕΝΟΣ!!";

$hector=-1;

}

}



$in=0;
```

```
        while($hector > 0) {

                $adminstep2str=$adminstep2str.'ΑΠΑΝΤΗΣΗ '.($in+1).'<br><input type="Text" size="25"
name="item['.$in.']"><br><br>';

                $in++;

                $hector=$hector-1;

        }

        $adminstep2str=$adminstep2str.'<input type="hidden" name="numofitems"
value='.$p_numofitems.'><input type="Submit" value="OK"></form>';

endif;

if($adminstep==3) {

        $fp=fopen($filename, "w");

        $hector=$p_numofitems+0;

        $in=0;

        $linetoadd=stripslashes($p_question.":");

        fputs($fp, $linetoadd);

        while($hector > 0) {

                $linetoadd=stripslashes($p_item[$in]).":0:";

                fputs($fp, $linetoadd);

                $in++;

                $hector=$hector-1;

        }

        fclose($fp);

        write2log("New poll was created with question: ".stripslashes($p_question));

}


?>

<table border=0 width=400 bgcolor=#FFFFFF>

        <tr>

           <td><font size=2 color=#000000>

                <strong> ΒΗΜΑ  <?php echo($adminstep); ?></strong>

        </td>
```

```
   </tr>

   <tr>

      <td bgcolor="#375297">

         <font size="2" color="#FFFFFF">

         <br>

         <blockquote>

         <?php if($adminstep==0) {

               echo($adminstep0str);

         }

         if($adminstep==1) {


               echo($adminstep1str);

         }

         if($adminstep==2) {

               echo($adminstep2str);

         }

         if($adminstep==3) {

               echo('ΤΟ ΑΡΧΕΙΟ ΔΗΜΙΟΥΡΓΗΘΗΚΕ... ΤΟ ΔΗΜΟΨΗΦΙΣΜΑ ΜΠΟΡΕΙ ΝΑ
ΔΟΚΙΜΑΣΤΕΙ!<br><br><a href="lp_test.php">ΔΟΚΙΜΑΣΤΕ ΤΟ
ΔΗΜΟΨΗΦΙΣΜΑ!!</a><br><br><br><font size="1"><a href="lp_admin.php?adminstep=1">ΠΙΣΩ
ΣΤΟ ΒΗΜΑ 1</a> ΠΡΟΕΙΔΟΠΟΙΗΣΗ:ΟΛΑ ΤΑ ΔΕΔΟΜΕΝΑ ΜΠΟΡΕΙ ΝΑ ΧΑΘΟΥΝ!');

         }

            ?>

     </blockquote></td>

   </tr>

</table>

</center>
</body>

</html>
```

ΤΟ ΑΡΧΕΙΟ GUESTBOOK.PHP

```
<html>
<head>
<SCRIPT language=JavaScript>
function textCounter(field, countfield, maxlimit) {
if (field.value.length > maxlimit)
field.value = field.value.substring(0, maxlimit);
```

```
else
countfield.value = maxlimit - field.value.length;
}
</SCRIPT>
<script language="JavaScript">
<!--
function emo(emo){
document.all.comment.value += emo;
document.all.comment.focus();
return;
}
//-->
</script>
<style>
BODY{
	FONT-SIZE: 15px; COLOR: #000; FONT-FAMILY: Verdana, Tahoma, Arial, sans-serif
BORDER-RIGHT: #c2cfdf 1px solid; PADDING-RIGHT: 6px; PADDING-TOP: 6px;
BACKGROUND-COLOR: #375297}
TABLE {
	FONT-SIZE: 11px; COLOR: #000; FONT-FAMILY: Verdana, Tahoma, Arial, sans-serif
}
TR {
	BORDER-RIGHT: #345487 1px solid; PADDING-RIGHT: 6px; BORDER-TOP: #345487 1px
solid; PADDING-LEFT: 6px; PADDING-BOTTOM: 6px; BORDER-LEFT: #345487 1px solid;
PADDING-TOP: 6px; BORDER-BOTTOM: #345487 1px solid; BACKGROUND-COLOR: #f5f9fd
}
TD {
	BORDER-RIGHT: #345487 1px solid; BORDER-TOP: #345487 1px solid; BORDER-LEFT:
#345487 1px solid; BORDER-BOTTOM: #345487 1px solid; BACKGROUND-COLOR: #f5f9fd
}
	TEXTAREA	{font-family:Verdana;font-size:10pt;background:#f5f9fd;color:#000000;}
input
{
	BORDER-RIGHT: #345487 1px solid; BORDER-TOP: #345487 1px solid; BORDER-LEFT:
#345487 1px solid; BORDER-BOTTOM: #345487 1px solid; BACKGROUND-COLOR: #f5f9fd
}
	A:link  {
	FONT-WEIGHT: bold; FONT-SIZE: 15px; COLOR: #3a4f6c; TEXT-DECORATION: none
}
A:visited {
	FONT-WEIGHT: bold; FONT-SIZE: 15px; COLOR: #3a4f6c; TEXT-DECORATION: none
}
A:active {
	COLOR: #000; TEXT-DECORATION: underline
}
A:hover {
	COLOR: #465584; TEXT-DECORATION: underline
}
TABLE{
	PADDING-RIGHT: 7px; MARGIN-TOP: 1px; PADDING-LEFT: 7px; FONT-SIZE: 15px;
PADDING-BOTTOM: 7px; PADDING-TOP: 7px; BACKGROUND-COLOR: #d1dceb
}
.gmenu{
	PADDING-RIGHT: 7px; MARGIN-TOP: 1px; PADDING-LEFT: 7px; FONT-SIZE:
10px;PADDING-BOTTOM: 7px; PADDING-TOP: 7px; BACKGROUND-COLOR: #d1dceb
}
	.small{
	FONT-SIZE: 10px; BACKGROUND-COLOR: #d1dceb
}

</style>
```

```
<title>Cute Guestbook</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"></head>
<body>

<?php
extract($HTTP_GET_VARS);
extract($HTTP_POST_VARS);
include "conf.txt";
function smiles(){
?>
<a href="javascript:emo(' :) ')"><img src="images/smile.gif" border=0 alt=":)"></a>
   <a href="javascript:emo(' :(' ) "><img src="images/sad.gif" border=0 alt=":("></a>
   <a href="javascript:emo(' :P ')"><img src="images/tongue.gif" border=0 alt=":P"></a>
   <a href="javascript:emo(' :o ')"><img src="images/scared.gif"  border=0 alt=":o"></a>
   <a href="javascript:emo(' :C ')"><img src="images/mad.gif" border=0 alt=":C"></a>
   <a href="javascript:emo(' ;Y ')"><img src="images/thumbsup.gif" border=0 alt=";Y"></a>
      <a href="javascript:emo(' :O) ')"><img src="images/clown.gif"border=0 alt=":0"></a>
   <a href="javascript:emo(' :D ')"><img src="images/laugh.gif" border=0 alt=":D"></a>
   <a href="javascript:emo(' :/ ')"><img src="images/sweatdrop.gif" border=0 alt=":/"></a>
   <a href="javascript:emo(' :s ')"><img src="images/shocking.gif" border=0 alt=":s"></a>
   <a href="javascript:emo(' :r ')"><img src="images/naughty.gif" border=0 alt=":r"></a>
   <a href="javascript:emo(' ;) ')"><img src="images/rolleyes.gif"border=0 alt=";)"></a>
   <a href="javascript:emo(' :H) ')"><img src="images/heart.gif"border=0 alt=":H)"></a>
   <a href="javascript:emo(' :i) ')"><img src="images/info.gif"border=0 alt=":i)"></a>
   <a href="javascript:emo(' :Q) ')"><img src="images/question.gif"border=0 alt=":Q)"></a>
      <?php
}

$day =date("D d");
$month =date("M");
$year =date("Y");
$dt="$day-$month-$year";
$copyright="<FONT SIZE=1 COLOR=#3a4f6c>Powered By:</font><A HREF=http://www.cgixp.tk
target=new><font class=small>mitsos</font></A></FONT>";
if($do=="do_sign"){
if ($name!="" && $comment !=""){
$name = str_replace ("|~!|","", $name);
$comment = str_replace ("\n", "<br>", $comment);
$comment = str_replace ("|~!|", "-", $comment);
$comment = str_replace ("<img ","", $comment);
$comment = str_replace ("<a ","", $comment);
$comment = str_replace (":)","<img src=images/smile.gif>", $comment);
$comment = str_replace (":(","<img src=images/sad.gif>", $comment);
$comment = str_replace (":P","<img src=images/tongue.gif>", $comment);
$comment = str_replace (":o","<img src=images/scared.gif>", $comment);
$comment = str_replace (":C","<img src=images/mad.gif>", $comment);
$comment = str_replace (";)","<img src=images/rolleyes.gif>", $comment);
$comment = str_replace (";Y","<img src=images/thumbsup.gif>", $comment);
$comment = str_replace (":D","<img src=images/laugh.gif>", $comment);
$comment = str_replace (":s","<img src=images/shocking.gif>", $comment);
$comment = str_replace (":r","<img src=images/naughty.gif>", $comment);
$comment = str_replace (":/","<img src=images/sweatdrop.gif>", $comment);
$comment = str_replace (":O)","<img src=images/clown.gif>", $comment);
$comment = str_replace (":H)","<img src=images/heart.gif>", $comment);
$comment = str_replace (":i)","<img src=images/info.gif>", $comment);
$comment = str_replace (":Q)","<img src=images/question.gif>", $comment);
$name = stripslashes ($name);
$comment = stripslashes ($comment);
$lis="0";
$user=file("badwords.txt");
for($x=0;$x<sizeof($user);$x++) {
```

```
$comment = str_replace($temp[0],"-",$comment);
$jemp = explode(";",$user[$x]);
$opp[$x] = "$jemp[0];";
$list[$lis] = $opp[$x];
$lis++;
}
if(sizeof($list) != "0") {
for($y=0;$y<sizeof($list);$y++) {
$temp = explode(";",$list[$y]);
$temq=ucwords($temp[0]);
$temr=ucfirst($temp[0]);
$tems=strtoupper($temp[0]);
$comment = str_replace ($temp[0],"-", $comment);
$comment = str_replace ($tep,"-", $comment);
$comment = str_replace ($temq,"-", $comment);
$comment = str_replace ($temr,"-", $comment);
$comment = str_replace ($tems,"-", $comment);
$name = str_replace ($temp[0],"-",$name);
$name = str_replace ($temq,"-",$name);
$name = str_replace ($temr,"-",$name);
$name = str_replace ($tems,"-",$name);
}
}
$fp = fopen ("guestbook.txt", "a+");
fwrite ($fp, $name);
fwrite ($fp, "|~!|");
fwrite ($fp, $email);
fwrite ($fp, "|~!|");
fwrite ($fp, $website);
fwrite ($fp, "|~!|");
fwrite ($fp, $comment);
fwrite ($fp, "|~!|");
fwrite ($fp, $dt);
fwrite ($fp, "|~!|");
fwrite ($fp, "\n");
fclose ($fp);
print  "<CENTER><FONT
COLOR=WHITE>&Epsilon;&Upsilon;&Chi;&Alpha;&Rho;&Iota;&Sigma;&Tau;&Omicron;&Upsil
on;&Mu;&Epsilon; &Pi;&Omicron;&Upsilon;
&Gamma;&Rho;&Alpha;&Phi;&Tau;&Eta;&Kappa;&Alpha;&Tau;&Epsilon;
&Sigma;&Tau;&Omicron; &Beta;&Iota;&Beta;&Lambda;&Iota;&Omicron;
&Epsilon;&Pi;&Iota;&Sigma;&Kappa;&Epsilon;&Pi;&Tau;&Omega;&Nu;
&Mu;&Alpha;&Sigma;!.<FONT COLOR=red></CENTER><BR>";
}else{ echo"<CENTER><FONT
COLOR=WHITE>&Sigma;&Upsilon;&Mu;&Pi;&Lambda;&Eta;&Rho;&Omega;&Sigma;&Tau;&Ep
silon; &Tau;&Alpha; &Kappa;&Alpha;&Tau;&Alpha;&Lambda;&Lambda;&Eta;&Lambda;&Alpha;
&Pi;&Epsilon;&Delta;&Iota;&Alpha;!!!<FONT COLOR=red></CENTER><BR><BR>";}
}
$latest_rev = $maximum;
$lp=file("guestbook.txt");
$abd = count($lp);
//echo "$abd";
if ($abd > $latest_rev) {
$folename = "guestbook.txt";
$fd = fopen ($folename, "r");
$puff = fread ($fd, filesize($folename));
fclose ($fd);
$last_file = fopen ("temp.txt", "a+");
fwrite ($last_file, $puff);
fclose($last_file);
$fz = fopen ("guestbook.txt", "w");
```

```
fwrite ($fz, "");
fclose($fz);
$lines14 = file("temp.txt");
$a1 = count($lines14);
$u = $a1 - $maximum;
for($i1 = $a1; $i1 >= $u ;$i1--){
$msg_old =  $lines14[$i1] . $msg_old;
}
$fz = fopen ("guestbook.txt", "a+");
fwrite ($fz,$msg_old);
fclose($fz);
}
$fz = fopen ("temp.txt", "w");
fwrite ($fz, "");
fclose($fz);
$user1=file("guestbook.txt");
if($action==""){
echo "<a href=?action=sign class=gmenu>&Pi;&Alpha;&Tau;&Eta;&Sigma;T&Epsilon;
&Epsilon;&Delta;&Omega; &Gamma;&Iota;&Alpha; &Nu;&Alpha;
E&Gamma;&Rho;&Alpha;&Phi;&Epsilon;&Iota;TE   </a>";
echo "<table width=100%><td>";
$tlis = 0;
      for($mp=sizeof($user1)-1;$mp>=0;$mp--){
            $mop = explode("|~!|",$user1[$mp]);
            $opp[$mp] = "$mop[0]|~!|$mop[1]|~!|$mop[2]|~!|$mop[3]|~!|$mop[4]|~!|$mop[5]|~!|";
            $list123[$tlis] = $opp[$mp];
            $tlis++;
      }
$latest_max = sizeof($list123);
$u = $a - $latest_max;
if (is_file("guestbook.txt"))
{
$s=sizeof($list123);
if ($page==" or !$page) { $page=1; }
$end=$listings_per_page*$page;
$start=$end-$listings_per_page;
if ($start<>'0') {
$new_page=$page-1;
$prev="<a href='?page=$new_page' ><<-
&Pi;&Rho;&Omicron;&Eta;&Gamma;&Omicron;&Upsilon;&Mu;&Epsilon;&Nu;&Omicron;</a>";
}
else {
      $prev="<<-
&Pi;&Rho;&Omicron;&Eta;&Gamma;&Omicron;&Upsilon;&Mu;&Epsilon;&Nu;&Omicron;";
}

if ($end<$s) {
      $new_page1=$page+1;
      $next="<a href='?page=$new_page1'
>&Epsilon;&Pi;&Omicron;&Mu;&Epsilon;&Nu;&Omicron;->></a>";
}
else {
      $next="&Epsilon;&Pi;&Omicron;&Mu;&Epsilon;&Nu;&Omicron;->>";
}
$today = time();
for ($i=$start; $i<$end; $i++){
if(substr($list123[$i], 1, 5 )!= "")
{
$p=explode('|~!|', $list123[$i]);
if ($p[2]=="http://"){
$link="None";
```

```
}else{
$link="<a href=$p[2] target=balnk>$p[2]</a>";
}if($p[1]==""){
$nm=$p[0];
}else{ $nm="<a href=mailto:$p[1]>$p[0]</a>"; }
echo "<tr><tr><tr><td
width=10%>&Omicron;&Nu;&Omicron;&Mu;&Alpha;:</td><td>$nm</td></tr><tr><td>WEBSITE:
</td><td>$link</td></tr><tr><td>&Sigma;&Chi;&Omicron;&Lambda;&Iota;&Alpha;:</td><td>$p[3
]</td></tr></tr><tr><td>&Eta;&Mu;.
&Epsilon;&Gamma;&Gamma;&Rho;&Alpha;&Phi;&Eta;&Sigma;
:</td><td>$p[4]</td></tr></tr></tr>";
}
}
$pages=$s/$listings_per_page;
$pages1=round($pages, 2);
$p= explode(".", $pages1);
$pcount=count($p);
$ext=$p[$pcount-2];
if ($ext!=0) {
      $num=$p[0]+1;
}

else {
      $num=$p[0];
}
echo "</table><table ALIGN='CENTER' width='100%'><tr><td align='left'>$prev</td><td
align='center'>";
for ($i=1; $i<=$num; $i++) {
      if ($i==$page) {
echo " <B>$i</B> ";

      }
      else {
echo " <a href='?page=$i'>$i</a> ";
}
}
echo "</td><td align='right'>$next</td></tr></table>";
}
}
if ($action=="sign"){
echo"<table align= 'center'><font size=3
COLOR=#FFFFFF>&Epsilon;&Gamma;&Gamma;&Rho;&Alpha;&Phi;&Epsilon;&Iota;&Tau;&Epsi
lon; &Sigma;&Tau;&Omicron; GUEST BOOK MA&Sigma;</font></td></table>";
echo "<table align='center' ><form method=post action=?do=do_sign><tr><td>*
ONOMA:</td><td><input type=text name=name maxlength=28
size=28></td></tr><tr><td>Email:</td><td><input type=text name=email maxlength=35
size=28></td></tr><tr><td>WEBSITE:</td><td><input type=text name=website maxlength=80
size=28 value=http://></td></tr><tr><td></td><td>";
smiles();
echo"</td></tr><tr><td>* &Sigma;&Chi;&Omicron;&Lambda;&Iota;&Alpha; :</td><td><textarea
name=comment rows=5 cols=45
onkeydown=textCounter(this.form.comment,this.form.descriptionleft,250);
onkeyup=textCounter(this.form.comment,this.form.descriptionleft,250);></textarea><BR>&Chi;&Alp
ha;&Rho;&Alpha;&Kappa;&Tau;&Eta;&Rho;&Epsilon;&Sigma; &Pi;&Omicron;&Upsilon;
&Mu;&Epsilon;&Nu;&Omicron;&Upsilon;&Nu; :<INPUT style='BORDER-RIGHT: 0px; BORDER-
TOP:0px;BORDER-LEFT:0px; BORDER-BOTTOM:0px;       FONT-SIZE: 15px; COLOR: #000;
FONT-FAMILY:Arial' maxLength=3 name=descriptionleft readOnly size=3 tabIndex=250
value=250> </td></tr><tr><td></td><td align=center><input type=submit value=Submit> <input
type=button value=Cancel onclick=javascript:history.back()> * =
&Alpha;&Pi;&Alpha;&Iota;&Tau;&Epsilon;&Iota;&Tau;&Alpha;&Iota;</td></tr></form></table>";
}
```

```
echo "<BR><table align=center class=small><tr class=small><td>$copyright</td></tr></tr></table>";
?>
</body>
</html>
```

ΤΟ ΑΡΧΕΙΟ UPLOAD.PHP

```
<?
switch($upload) {
default:
include "config.php";
echo "
<html>

<head>
<title>Upload</title>
</head>

<body topmargin=\"10\" leftmargin=\"0\" bgcolor=\"#375297\" link=\"#white\"
vlink=\"#white\" alink=\"#white\" text=\"#FFFFFF\" style=\"font-family: Verdana;
font-size: 8pt; color: #FFFFFF\">



<div align=\"center\">
  <center><br><br><br><br><br><br><h1>ΑΝΕΒΑΣΤΕ ΤΟ ΑΡΧΕΙΟ ΣΑΣ</h1>
  <table border=\"0\" cellpadding=\"0\" cellspacing=\"0\" style=\"border-collapse:
collapse\" bordercolor=\"#111111\" width=\"500\" id=\"AutoNumber1\">
    <tr>
      <td bgcolor=\"#5E6A7B\" height=\"25\">
      <p align=\"center\"><font size=\"2\"><b>Upload File</b></font></td>
    </tr>
    <tr>
      <td bgcolor=\"#818EA0\"><font size=\"2\">Για να ανεβάσετε ένα αρχείο θα
πρέπει να τηρεί τις παρακάτω προυποθέσεις:</font><ul type=\"square\">
        <li><font size=\"2\">Ο τύπος του αρχείου θα πρέπει να είναι σε: <b>";
        if (($extensions == "") or ($extensions == " ") or ($ext_count == "0") or
($ext_count == "") or ($limit_ext != "yes") or ($limit_ext == "")) {
          echo "any extension";
        } else {
        $ext_count2 = $ext_count+1;
        for($counter=0; $counter<$ext_count; $counter++) {
           echo "  $extensions[$counter]";
        }
        }
        if (($limit_size == "") or ($size_limit != "yes")) {
           $limit_size = "any size";
        } else {
           $limit_size .= " bytes";
        }
        echo"</b></font></li>
        <li><font size=\"2\">Ανώτερο μέγεθος αρχείου $limit_size</font></li>
```

```
      <li><font size=\"2\">Δεν πρέπει να υπάρχουν κενά στο όνομα του
αρχείου</font></li>
      <li><font size=\"2\">Το όνομα του αρχείου δεν πρέπει να περιέχει ειδικούς
χαρακτήρες
      (/,*,\,κλπ)</font><BR>
      </li>
     </ul>
     <form method=\"POST\" action=\"$PHP_SELF?upload=doupload\"
enctype=\"multipart/form-data\">
<p align=\"center\">
<input type=file name=file size=30 style=\"font-family: v; font-size: 10pt; color:
#5E6A7B; border: 1px solid #5E6A7B; padding-left: 4; padding-right: 4; padding-
top: 1; padding-bottom: 1\"><br>
<br>
<button name=\"submit\" type=\"submit\" style=\"font-family: v; font-size: 10pt;
color: #5E6A7B; border: 1px solid #5E6A7B; padding-left: 4; padding-right: 4;
padding-top: 1; padding-bottom: 1\">Upload</button>
</p>
</form>
     <p>
</td>
   </tr>
     </table>
  </center>
</div>


</body>

</html>";
break;
case "doupload":
include "config.php";
$endresult = "<font size=\"2\">Το αρχείο εστάλη</font>";
if ($file_name == "") {
$endresult = "<font size=\"2\">Λάθος!! Δεν επιλέξατε κάποιο αρχείο</font>";
}else{
if(file_exists("$absolute_path/$file_name")) {
$endresult = "<font size=\"2\">Λάθος!! Το αρχείο υπάρχει ήδη</font>";
} else {
if (($size_limit == "yes") && ($limit_size < $file_size)) {
$endresult = "<font size=\"2\">Λάθος!! Το αρχείο έχει μεγάλο μέγεθος</font>";
} else {
$ext = strrchr($file_name,'.');
if (($limit_ext == "yes") && (!in_array($ext,$extensions))) {
$endresult = "<font size=\"2\">Λάθος!! Το αρχείο αυτού του τύπου δεν είναι
αποδεκτό</font>";
}else{
@copy($file, "$absolute_path/$file_name") or $endresult = "<font
size=\"2\">Λάθος!! Το αρχείο δεν εστάλη</font>";
}
```

```
}
}
}
echo "
<html>

<head>
<title>Upload ΑΣΚΗΣΕΩΝ</title>
</head>

<body topmargin=\"10\" leftmargin=\"0\" bgcolor=\"#5475C6\" link=\"#818EA0\"
vlink=\"#5C697A\" alink=\"#818EA0\" text=\"#FFFFFF\" style=\"font-family:
Verdana; font-size: 8pt; color: #FFFFFF\">

<div align=\"center\">
  <center>
  <table border=\"0\" cellpadding=\"0\" cellspacing=\"0\" style=\"border-collapse:
collapse\" bordercolor=\"#111111\" width=\"400\" id=\"AutoNumber1\">
    <tr>
      <td bgcolor=\"#5E6A7B\" height=\"25\">
      <p align=\"center\"><font size=\"2\"><b>Upload File</b></font></td>
    </tr>
    <tr>
      <td bgcolor=\"#818EA0\">
      <center> $endresult </center>
        </td>
    </tr>

  </table>
  </center>
</div>


</body>

</html>";
break;
}
?>
```

Εκτός από τα scripts της PHP και τον HTML κώδικα η εφαρμογή είχε και την υποστήριξη μιας βάσης δεδομένων τα MySQL που χρησιμοποιήθηκαν για τη δημιουργία της βάσης δεδομένων είναι τα παρακάτω:

ΓΙΑ ΤΟΝ ΠΙΝΑΚΑ ΤΟΝ ΣΤΑΤΙΣΤΙΚΩΝ

```
CREATE TABLE stats_day (
   date date DEFAULT '0000-00-00' NOT NULL,
   hits mediumint(8) unsigned DEFAULT '0' NOT NULL,
   PRIMARY KEY (date)
);
```

ΒΙΒΛΙΟΓΡΑΦΙΑ – ΠΗΓΕΣ INTERNET

ΒΙΒΛΙΟΓΡΑΦΙΑ

-ΑΝΑΠΤΥΞΗ WEB ΕΦΑΡΜΟΓΩΝ ΜΕ PHP ΚΑΙ MySQL,
LUKE WELLING, LAURA THOMSON,
ΕΚΔΟΣΕΙΣ Μ. ΓΚΙΟΥΡΔΑΣ

-ΜΑΘΕΤΕ PHP,MySQL ΚΑΙ APACHE ΣΕ 24 ΩΡΕΣ,
ΕΚΔΟΣΕΙΣ Μ. ΓΚΙΟΥΡΔΑΣ

-PHP-ΟΔΗΓΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ,
STERLING HUGHES,
ΕΚΔΟΣΕΙΣ Μ. ΓΚΙΟΥΡΔΑΣ

-PHP MAGAZINE ΤΕΥΧΟΣ ΙΑΝΟΥΑΡΙΟΥ 2004,
ΕΚΔΟΣΕΙΣ S&S

ΠΗΓΕΣ INTERNET

-WWW.PHPNET.COM
-WWW.HOTSCRIPTS.COM
-WWW.APACHE.ORG
-WWW.W3C.ORG
-WWW.DIDE.FLO.SCH.GR
-WWW.PDPLAB.GR
-WWW.IT.UOM.GR/PROJECT/CLIENT_SERVER
-WWW.JUPITERIMAGE.COM
-WWW.GRJAVA.COM
-WWW.WEBGRP.CEID.UPATRAS.GR
-WWW.HOWSTUFFWORKS.COM
-WWW.ANNAEVI.COM
-WWW.LEARNTHENET.COM
-WWW.PCMAG.GR
-WWW.IN.GR/RAM