ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ
ΤΜΗΜΑ ΔΙΟΙΚΗΣΗΣ ΕΠΙΧΕΙΡΗΣΕΩΝ (ΠΑΤΡΑ)

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη Εφαρμογής Παγκόσμιου Ιστού για την Εξ' Αποστάσεως Εκμάθηση Javascript & PHP.

ΣΠΟΥΔΑΣΤΗΣ:Κωνσταντίνος Κολέτσος

Επιβλέπων καθηγητής: Κωνσταντίνος Γιωτόπουλος

Πάτρα – Μάιος 2015

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή

Πάτρα, ……..

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. ……

2. ……

3. ……

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένεια μου για την βοήθεια και την στήριξη που μου έδειξαν σε όλη την διάρκεια των σπουδών μου, παρέχοντας μου ηθική και οικονομική στήριξη, συνέβαλαν κι αυτοί από την μεριά τους για να πετύχω τους στόχους που είχα θέσει.

Επίσης, θα ήθελα να εκφράσω ιδιαίτερες ευχαριστίες στον επιτηρητή – καθηγητή μου Κύριο Κωνσταντίνο Γιωτόπουλο, στον Κύριο. Ιωάννη Τζήμα, καθώς επίσης και στον Κύριο. Σωτήρη Χριστοδούλου, για την εμπιστοσύνη που μου έδειξαν παρέχοντας το θέμα της εργασίας, καθώς και για την βοήθεια που μου παρείχαν κατά την διάρκεια της εκπόνηση της.

Ακόμα θα ήθελα να ευχαριστήσω τον Δημήτρη Διαμαντή, προσωπικό φίλο και συμφοιτητή ο οποίος μοιράστηκε μαζί μου τις γνώσεις του για το ψηφιακό design και την δημιουργία γραφικών, δίνοντας μου έτσι την δυνατότητα να προσφέρω πιο πλήρες αποτελέσματα στην εκπόνηση της εργασίας αυτής.

Τέλος θα ήθελα να ευχαριστήσω το διοικητικό και διδακτικό προσωπικό που πολλές φορές με βοήθησαν σε αρκετές και δύσκολες στιγμές κατά την διάρκεια των σπουδών μου. Η βοήθεια τους στο να ξεπεράσω τις δυσκολίες μου, με εξόπλισε με τα απαραίτητα εφόδια σε επίπεδο γνώσεων, ώστε να μπορώ να αντιμετωπίζω τις επαγγελματικές προκλήσεις που πρόκειται να συναντήσω στην υπόλοιπη ζωή μου.

Περίληψη

Σκοπός της πτυχιακής εργασίας είναι η δημιουργία ενός διαδικτυακού τόπου στον οποίο οι επόμενες γενιές φοιτητών θα αποκτήσουν γνώσεις τόσο θεωρητικές όσο και πρακτικές πάνω στην ανάπτυξη και κατασκευή ιστοσελίδων καθώς επίσης και άλλων γλωσσών προγραμματισμού. Ο διαδικτυακός τόπος αυτός θα είναι μορφοποιημένος και σχεδιασμένος έτσι ώστε να είναι όσο το δυνατόν εύχρηστος προς τους άπειρους χρήστες παρέχοντας μόνο τις απαραίτητες πληροφορίες που χρειάζονται, αποφεύγοντας έτσι την απόσπαση της προσοχής τους σε περιττές πληροφορίες. Ο ιστότοπος αυτός θα περιέχει μια σειρά από οδηγούς εκμάθησης για διάφορες γλώσσες προγραμματισμού σε μορφή άρθρων. Η ανάπτυξη και η κατασκευή του διαδικτυακού ιστότοπου θα πραγματοποιηθεί με την χρήση του Joomla.

Abstract

The aim of the thesis is to create a website where the next generations of students will acquire knowledge both theoretical and practical on the development and manufacturing sites as well as other programming languages. The website that will be formatted and designed so as to be as convenient as possible for inexperienced users by providing only the necessary information they need, thus avoiding the distraction unnecessary information. This site contains a series of tutorials for various programming languages in the form of articles. The development and construction of the web site will take place with the use of Joomla.

ΠΕΡΙΕΧΟΜΕΝΑ

Λίστα Σχημάτων

Λίστα Πινάκων

ΛΙΣΤΑ ΕΙΚΟΝΩΝ

ΕΙΣΑΓΩΓΗ

Ορισμός προβλήματος

Τα τελευταία χρόνια το ιντερνετ προσφέρει στους χρήστες του ένα νέο φάσμα δυνατοτήτων επικοινωνίας, διείσδυσης και πληροφόρησης. Βασικότερος παράγοντας ήταν και είναι οι ραγδαίες τεχνολογικές εξελίξεις, με σημαντικότερη καινοτομία το Διαδίκτυο, που διευκόλυνε την πρόσβαση μέσω φυλλομετρητών, με υποστήριξη γραφικών και επέτρεψε στην γρήγορη και εύκολη πλοήγηση. Η ανάγκη των χρηστών του Διαδικτύου για πληροφόρηση έφτασε στα μέγιστα δυνατά της όρια, όταν στην αγορά εμφανίστηκαν και τα πρώτα smart phones με γρήγορους επεξεργαστές και δυνατότητα προβολής προχωρημένων γραφικών. Έκτοτε το διαδίκτυο επεκτάθηκε τόσο σε δυνατότητες όσο και σε πλατφόρμες.

Σήμερα η δυνατότητα της δικτύωσης για έναν άνθρωπο έχει αποτελέσει ένα από τα βασικότερα αγαθά καθώς καλύπτει ανάγκες όπως επικοινωνίας, μάθηση, διερεύνηση , διασκέδαση. Σε αυτή την πτυχιακή εργασία εστιάζουμε στην μάθηση μέσω του διαδικτύου (ή αλλιώς e-learning). Σκοπός μας είναι η ανάπτυξη εφαρμογής παγκόσμιου ιστού για την εξ’ αποστάσεως εκμάθηση των χρηστών της σε διάφορες γλώσσες προγραμματισμού και κυρίως γλώσσες όπως η Javascript και η PHP που χρησιμοποιούνται για την ανάπτυξη τέτοιων εφαρμογών. Θέλοντας να καλύψουμε το κενό της δωρεάν ελλιπής ηλεκτρονικής μάθησης στα ελληνικά για τις γλώσσες προγραμματισμού, εστιάζουμε στο να προσφέρουμε ποιοτικούς οδηγούς εκμάθησης των γλωσσών, βασισμένους σε συγκεκριμένη ύλη, παρέχοντας τα κατάλληλα εφόδια για το χτίσιμο της γνώσης που ένας χρήστης της εφαρμογής επιζητά.

Δομή της πτυχιακής εργασίας

Η παρούσα πτυχιακή εργασία χωρίζεται σε δυο μέρη. Το πρώτο είναι το θεωρητικό μέρος, και το περιεχόμενο του εστιάζει κυρίως στις βασικές έννοιες του Διαδικτύου και των συστημάτων διαχείρισης περιεχομένου. Δίνεται ιδιαίτερη έμφαση σε έννοιες όπως το Διαδίκτυο, η ιστοσελίδα, το όνομα τομέα, η φιλοξενία ιστοσελίδων, ο εξυπηρετητής ιστού και ο φυλλομετρητής. Επίσης παρουσιάζονται αναλυτικά ο ορισμός ενός συστήματος διαχείρισης περιεχομένου, καθώς επίσης τα είδη του.

Μελετώνται με ακρίβεια τα δημοφιλέστερα συστήματα, και αποσαφηνίζονται τυχών τεχνικοί όροι για την δημιουργία μιας ιστοσελίδας βασιζόμενη σε ένα τέτοιο σύστημα.

Στο δεύτερο μέρος επιχειρείται η αναπαράσταση από το μηδέν, της κατασκευής μιας ιστοσελίδας με σκοπό την εξ' αποστάσεως εκμάθηση των επισκεπτών της. Η κατασκευή γίνεται βασιζόμενη στην χρήση ενός ελεύθερου και ανοικτού κώδικα συστήματος διαχείρισης περιεχομένου, το Joomla. Με τον κατάλληλο συνδυασμό προτύπου, πρόσθετων και ρυθμίσεων, δημιουργούμε άρθρα –σελίδες, εισάγουμε έξυπνα ενθέματα, δημιουργούμε τα κατάλληλα μενού, και κάνουμε τις απαραίτητες παρεμβάσεις με την χρήση γλωσσών όπως HTML και CSS.

Η συνεισφορά της πτυχιακής εργασίας

Η πτυχιακή έρευνα επιχειρεί να αναδείξει τις δυνατότητες της εξ' αποστάσεως εκμάθησης που προσφέρει μια ιστοσελίδα εμπλουτισμένη με το κατάλληλο περιεχόμενο. Τα οφέλη από μια τέτοια διαδικασία προκύπτουν στην εκπαίδευση φοιτητών και αυτόνομων ατόμων που ενδιαφέρονται να μάθουν για τον προγραμματισμό και γενικά για την απόκτηση αυτοδίδακτης γνώσης. Έτσι η ιστοσελίδα αυτή επιχειρεί να γίνει ένα χρήσιμο εγχειρίδιο εκμάθησης για κάθε αρχάριο και μη.

1 ΒΑΣΙΚΕΣ ΕΝΝΟΙΑ ΔΙΑΔΙΚΤΥΟΥ

Ξεκινώντας την εργασία σε αυτό το πρώτο κεφάλαιο θα αναφερθούμε στις βασικές έννοιες του διαδικτύου. Θα μάθουμε τι είναι το Διαδίκτυο, τι είναι οι ιστοσελίδες, το όνομα τομέα, η φιλοξενία ιστοσελίδων καθώς και πως αυτές λειτουργούν μέσω των εξυπηρετητών και των φυλλομετρητών από την πλευρά των επισκεπτών.

1.1 Διαδίκτυο

Το Διαδίκτυο (Internet) είναι παγκόσμιο σύστημα διασυνδεδεμένων δικτύων υπολογιστών, οι οποίοι χρησιμοποιούν καθιερωμένη ομάδα πρωτοκόλλων, η οποία συχνά αποκαλείται «TCP/IP» (αν και αυτή δεν χρησιμοποιείται από όλες τις υπηρεσίες του Διαδικτύου) για να εξυπηρετεί εκατομμύρια χρηστών καθημερινά σε ολόκληρο τον κόσμο. Οι διασυνδεδεμένοι ηλεκτρονικοί υπολογιστές ανά τον κόσμο, οι οποίοι βρίσκονται σε ένα κοινό δίκτυο επικοινωνίας, ανταλλάσσουν μηνύματα

(πακέτα) με τη χρήση διαφόρων πρωτοκόλλων (τυποποιημένοι κανόνες επικοινωνίας), τα οποία υλοποιούνται σε επίπεδο υλικού και λογισμικού. Το κοινό αυτό δίκτυο καλείται Διαδίκτυο.

Το Διαδίκτυο είναι ένα επικοινωνιακό δίκτυο που επιτρέπει την ανταλλαγή δεδομένων μεταξύ οποιοδήποτε διασυνδεδεμένου υπολογιστή. Η τεχνολογία του είναι κυρίως βασισμένη στην διασύνδεση επιμέρους δικτύων ανά τον κόσμο με πολυάριθμα πρωτόκολλα επικοινωνίας Στην πιο εξειδικευμένη και περισσότερο χρησιμοποιούμενη μορφή του, με τον όρο Διαδίκτυο, περιγράφεται το παγκόσμιο πλέγμα διασυνδεδεμένων υπολογιστών και των υπηρεσιών και πληροφοριών που παρέχει στους χρήστες του. Το Διαδίκτυο χρησιμοποιεί μεταγωγή πακέτων και τη στοίβα πρωτοκόλλων. Σήμερα, ο όρος διαδίκτυο κατέληξε να αναφέρεται στο παγκόσμιο αυτό δίκτυο. Για να ξεχωρίζει, το παγκόσμιο αυτό δίκτυο γράφεται με κεφαλαίο το αρχικό «Δ». Η τεχνική της διασύνδεσης δικτύων μέσω μεταγωγής πακέτων και της στοίβας πρωτοκόλλων ονομάζεται Διαδικτύωση.[1]

1.2 Ιστοσελίδα

World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .

What's out there?
Pointers to the world's online information, subjects , W3 servers, etc.
Help
on the browser you are using
Software Products
A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)
Technical
Details of protocols, formats, program internals etc
Bibliography
Paper documentation on W3 and references.
People
A list of some people involved in the project.
History
A summary of the history of the project.
How can I help ?
If you would like to support the web..
Getting code
Getting the code by anonymous FTP , etc.

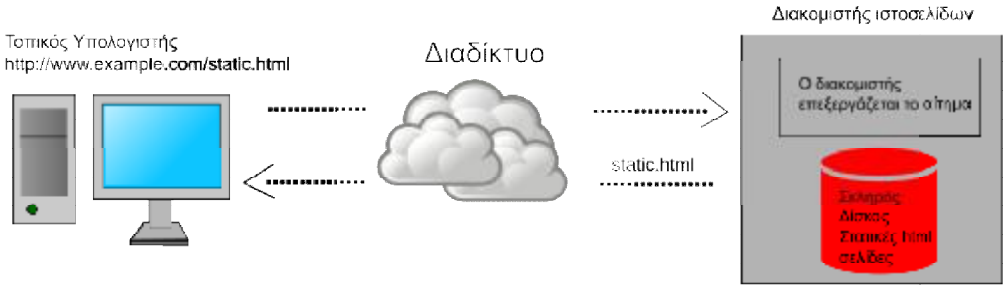
Εικόνα 1.1 Η πρώτη ιστοσελίδα στον κόσμο

Ιστοσελίδα (web page) είναι ένα είδος εγγράφου του παγκόσμιου ιστού (WWW) που περιλαμβάνει πληροφορίες με την μορφή κειμένου, υπερκειμένου, εικόνας, βίντεο και ήχου.

Πολλές ιστοσελίδες μαζί συνθέτουν έναν ιστότοπο (εναλλακτικές ονομασίες: ιστοχώρος ή δικτυακός τόπος, αγγλ. web site ή Internet site). Οι σελίδες ενός ιστοτόπου εμφανίζονται κάτω από το ίδιο όνομα χώρου (domain) π.χ. microsoft.com. Οι ιστοσελίδες αλληλοσυνδέονται και μπορεί ο χρήστης να μεταβεί από τη μία στην άλλη κάνοντας «κλικ», επιλέγοντας δηλαδή συνδέσμους που υπάρχουν στο κείμενο ή στις φωτογραφίες της ιστοσελίδας. Οι σύνδεσμοι προς άλλες σελίδες εμφανίζονται συνήθως υπογραμμισμένοι και με μπλε χρώμα για να είναι γρήγορα ξεκάθαρο στον επισκέπτη ότι πρόκειται για σύνδεσμο προς άλλη ιστοσελίδα, χωρίς όμως πάντα να είναι αυτό απαραίτητο.

Η κατασκευή ιστοσελίδων είναι κάτι που μπορεί να γίνει πολύ εύκολα με προγράμματα που κυκλοφορούν ελεύθερα, αλλά υπάρχουν και αυτοματοποιημένοι μηχανισμοί κατασκευής ιστοσελίδων που επιτρέπουν σε απλούς χρήστες να δημιουργήσουν εύκολα και γρήγορα προσωπικές ή και εμπορικές ιστοσελίδες. Από την άλλη μεριά υπάρχουν και πολλές εταιρίες, που εξειδικεύονται στη δημιουργία ελκυστικών και λειτουργικών ιστοσελίδων που έχουν σαν στόχο να οδηγήσουν τους επισκέπτες στην αγορά κάποιου προϊόντος, στην επικοινωνία με τον ιδιοκτήτη του ιστοτόπου ή απλά στο ανέβασμα του εταιρικού προφίλ μιας επιχείρησης [2].

1.2.1 Στατική ιστοσελίδα



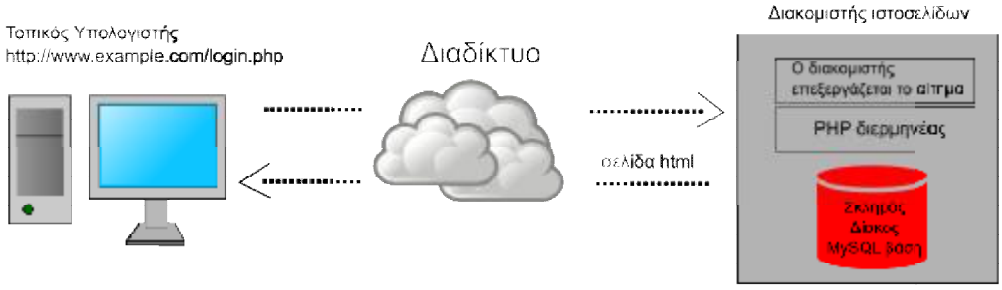
Σχήμα 1.1 Στατική Ιστοσελίδα

Είναι μια ιστοσελίδα της οποίας το περιεχόμενο μεταδίδεται στον χρήστη που την επισκέπτεται στην ακριβώς ίδια μορφή που είναι αποθηκευμένη στον εξυπηρετητή που την διαθέτει. Οι στατικές ιστοσελίδες μπορούν κάθε μια τους χωριστά να αποτελούν και από ένα διαφορετικό αρχείο htm ή html. Τα περιεχόμενα τους

εμφανίζουν εξίσου με τον ίδιο τρόπο σε όλους τους χρήστες που επισκέπτονται έναν ιστότοπο.

Βασικό τους πλεονέκτημα είναι ότι δεν χρειάζονται προγραμματιστικές δεξιότητες για να δημιουργηθούν καθώς επίσης μπορούν εύκολα να αντιγραφούν, καθώς όπως είπαμε κάθε μια ιστοσελίδα είναι και ένα αρχείο html. Δεν χρειάζονται ειδικό λογισμικό σε στον εξυπηρετητή που φιλοξενούνται προκειμένου να δει κάποιος το περιεχόμενο τους. Οι σελίδες αυτές είναι κατευθείαν διαθέσιμες στον φυλλομετρητή με τον οποίο ο χρήστης επιλέγει να τις προσπελάσει ,είτε βρίσκονται σε κάποιον απομακρυσμένο διακομιστή είτε αποθηκευμένες σε κάποιο τοπικό μέσο, usb, cd ή dvd. Εν αντιθέσει το βασικότερο τους μειονέκτημα είναι ότι δεν προσφέρουν διαδραστικότητα με το χρήστη καθώς επίσης ένας μεγάλος αριθμός στατικών ιστοσελίδων αποτελεί δύσκολη περίπτωση διαχείρισης χωρίς αυτόματα εργαλεία [3].

1.2.2 Δυναμική ιστοσελίδα



Σχήμα 1.2 Δυναμική Ιστοσελίδα

Είναι ένα είδος ιστοσελίδας της οποίας το περιεχόμενο δεν μεταδίδεται στον χρήστη όπως είναι αποθηκευμένο σε κάποιον εξυπηρετητή, αλλά πριν μεταδοθεί παράγεται δυναμικά και ύστερα αποστέλλεται το αποτέλεσμα. Οι δυναμικές ιστοσελίδες στηρίζονται σε κάποιο σενάριο εντολών προκειμένου να παράξουν δυναμικά το περιεχόμενο τους , όπως οι γλώσσες PHP, JSP κ.α.

Δεν υπάρχει ακριβής χρονική τοποθέτηση για το πότε άρχισαν να χρησιμοποιούνται, ωστόσο η ιδέα δημιουργίας δυναμικών ιστοσελίδων υπήρχε και πριν την ανάπτυξη των πρώτων στατικών ιστοσελίδων. Σήμερα οι δυναμικές ιστοσελίδες αποτελούν το

δομικό στοιχείων της νέας γενιάς του παγκόσμιου ιστού όπου πλέον η πληροφορία διαμοιράζεται και διαχέεται σε πολλαπλές ιστοσελίδες.

Το βασικό πλεονέκτημα τους είναι η διαδραστικότητα με τον χρήστη. Το περιεχόμενο τους αλλάζει με βάση το ποίος τις επισκέπτεται και έτσι διαφορετικές μορφές της ίδιας ιστοσελίδας παρέχονται δυναμικά σε διαφορετικούς χρήστες. Αυτό τις καθιστά πιο λειτουργικές σε σχέση με τις στατικές ιστοσελίδες.

Άλλο ενα πλεονέκτημα τους ειναι η ευκολία ενημέρωσης, καθώς στις περισσότερες περιπτώσεις οι δυναμικές ιστοσελίδες αντλούν περιεχόμενο απο κάποια βαση δεδομένων, έτσι αλλάζοντας η προσθέτοντας περιεχόμενο μέσα σε αυτήν η μορφή των ιστοσελίδων αυτών προσθέτει η αλλάζει ανάλογα το περιχόμενο με βάση το τι έχουμε πράξει στην βάση. Αποφεύγουμε με αυτόν τον τρόπο να επεξεργαζόμαστούμε πλήθος αρχείων html που θα χρειαζόταν να επεξεργαστούμε εαν διαθέταμε στατικές ιστοσελίδες. Χάρις αυτή την δυνατότητα τους στην ενημέρωση του υλικού η ανανέωση του περιεχομένου και η προβολή του μας καθιστά ενα σημείο ανανεώσιμων ενδιαφέροντων για του χρήστες που μας επισκέπτονται, καθώς για ότι αλλάξει θα επιστρέψουν στην ιστοσελίδα μας για να ενημερωθούν.

Μειονεκτήματα των δυναμικών ιστοσελίδων μεταξύ άλλων είναι η ανάπτυξη τους όπου είναι πολύ πιο βραδύτερη και πολύ πιο ακριβή σε κόστος, καθώς επίσης το ίδιο σχετικά με το κόστος ισχύει και για την φιλοξενία τους σε κάποιον εξυπηρετητή, λόγω μεγάλων απαιτήσεων τους [4].

1.3 Όνομα τομέα

Όνομα χώρου ή τομέα ή περιοχής (domain name) στο Διαδίκτυο είναι ένας περιορισμένος τομέας των διεθνών πόρων του Συστήματος Ονομάτων Χώρου (DNS) ο οποίος εκχωρείται για αποκλειστική χρήση σε ένα φυσικό ή νομικό πρόσωπο. Το όνομα τομέα / χώρου δεν ανήκει στο πρόσωπο που του έχει εκχωρηθεί αλλά έχει μόνο την αποκλειστική δυνατότητα χρήσης του για όσο διάστημα έχει καταβάλει τα τέλη κατοχύρωσης. Ένα όνομα χώρου μπορεί να έχει διάφορες καταλήξεις όπως .com, .eu, .gr, .net, .org, .info, .biz, .de, .it, .es κ.λ.π., ανάλογα με τη χρήση και τη χώρα προέλευσής του.

Στα ονόματα χώρου επιτρέπεται μόνο η χρήση αλφαριθμητικών στοιχείων και παυλών. Για τα ονόματα χώρου με κατάληξη .gr υπάρχουν απαγορευμένες κατηγορίες. Αν ένα όνομα χώρου θεωρείται κοινόχρηστο ή γεωγραφικός όρος εκχωρείται μόνο στους αντίστοιχους οργανισμούς τοπικής αυτοδιοίκησης ανεξάρτητα από τον τρόπο γραφής του με λατινικά στοιχεία. Επίσης δεν επιτρέπεται η εκχώρηση ονομάτων χώρου με κατάληξη .gr που αποτελούν λέξεις κλειδιά στο Διαδίκτυο.

Τα κατοχυρωμένα ονόματα χώρου είναι συνήθως τα ονόματα των τριών ή τεσσάρων πρώτων επιπέδων. Τα υπόλοιπα ονόματα χώρου δεν χρειάζονται κατοχύρωση. Στα ονόματα χώρου κάθε τελεία δείχνει την αλλαγή επιπεδου ή αρχή ενός υποσυνόλου - υποτομέα και το σύνολο - χώρος που περιλαμβάνει όλα τα σύνολα είναι η πιο δεξιά τελεία που συνήθως παραλείπεται. Οι λύτες είναι το λογισμικό που μας βοήθα να χρησιμοποιήσουμε τα ονόματα χώρου. Οι λύτες διαβάζουν τα ονόματα του DNS από δεξιά προς τα αριστερά.

Για παράδειγμα όταν γράφουμε το όνομα «DNS.example.wikipedia.www.el.ipduh.com» εννοούμε «DNS.example.wikipedia.www.el.ipduh.com.». Η τελική τελεία είναι το σύνολο που περιλαμβάνει όλο το σύστημα και το υποσύνολο που ονομάζεται «com.». Το σύνολο «com.» περιλαμβάνει το σύνολο «ipduh.com.», το σύνολο «ipduh.com.» περιλαμβάνει το «el.ipduh.com.», το σύνολο «el.ipduh.com.» περιλαμβάνει το σύνολο «www.el.ipduh.com.» κ.ο.κ [5].

1.4 Φιλοξενία ιστοσελίδων

Η φιλοξενία ιστοσελίδων (αγγλικά: web hosting) είναι ένα μια διαδικτυακή υπηρεσία που επιτρέπει σε ιδιώτες και εταιρείες να διαθέτουν μία ιστοσελίδα συνεχώς αναρτημένη στο Διαδίκτυο, χωρίς να χρειάζεται να επιβαρύνονται με το κόστος του ανάλογου εξοπλισμού (π.χ. εξυπηρετητές) ή την ανάγκη εξυπηρέτησης μεγάλου αριθμού εξωτερικών συνδέσεων και εύρους σύνδεσης (bandwidth). Αυτό το αναλαμβάνουν οι εταιρίες φιλοξενίας ιστοσελίδων (web hosts) που προσφέρουν χώρο στον διακομιστή τους καθώς και μέρος της σύνδεσής τους στο δίκτυο.

Ο όρος Web Hosting αναφέρεται στη διαδικασία με την οποία ο ιδιοκτήτης μίας ιστοσελίδας ενοικιάζει χώρο σε υπολογιστές (διακομιστές) για να τοποθετήσει τα αρχεία του ή και την ηλεκτρονική αλληλογραφία του. Τα αρχεία αυτά, που στοιχειοθετούν την ιστοσελίδα του, προσφέρονται μέσω ασφαλούς δικτύου αδιάλειπτης παροχής στους επισκέπτες.

Η διαχείριση για ένα ιστότοπο από τον ιδιοκτήτη του, μπορεί να γίνει μέσω προγραμμάτων απομακρυσμένης σύνδεσης ή από τον περιηγητή ιστού (browser) μέσω πρόσβασης σε έναν πίνακα ελέγχου (control panel), το οποίο δίνει τη δυνατότητα διαχείρισης των emails, των αρχείων, των στατιστικών επισκεψιμότητας του ιστότοπου, των εγκατεστημένων εφαρμογών και διαθέσιμων τεχνολογιών κ.α.

Ο ιδιοκτήτης του ιστότοπου μπορεί να ανεβάζει τα αρχεία του μέσω προγράμματος (FTP client) στο διακομιστή φιλοξενίας, να διαχειρίζεται τους λογαριασμούς ηλεκτρονικής αλληλογραφία (email accounts) και να εγκαθιστά τις επιθυμητές διαδικτυακές εφαρμογές στον ιστότοπό του (ιστολόγιο / blog, forum, βιβλίο επισκεπτών κλπ). Μερικοί από αυτούς τους πίνακες ελέγχου φιλοξενίας είναι το Plesk, το Cpanel, το Webmin κ.α.

Από τις αρχές της δεκαετίας του '90 πολλές εταιρείες ξεκίνησαν να δραστηριοποιούνται στον τομέα της παροχής φιλοξενίας, καθώς η ανάγκη για σταθερή και συνεχή διαδικτυακή παρουσία άρχισε να γίνεται επιβεβλημένη Τη δεκαετία του 2000 η βιομηχανία του web hosting γνώρισε τεράστια άνθηση πρώτα στις Η.Π.Α. κι έπειτα και στην Ευρώπη, ακολουθώντας την μεγάλη ζήτηση για υπηρεσίες φιλοξενία από ιδιώτες κι επιχειρήσεις, για τη στέγαση της ιστοσελίδας τους [6].

1.5 Εξυπηρετητής ιστού

Εξυπηρετητής ή διακομιστής (αγγλ.: server) είναι υλικό ή / και λογισμικό που αναλαμβάνει την παροχή διάφορων υπηρεσιών, «εξυπηρετώντας» αιτήσεις άλλων προγραμμάτων, γνωστούς ως πελάτες (clients) που μπορούν να τρέχουν στον ίδιο υπολογιστή ή σε σύνδεση μέσω δικτύου. Όταν ένας υπολογιστής εκτελεί κυρίως τέτοια προγράμματα εξυπηρετητές συνεχόμενα, 24 ώρες την ημέρα, τότε μπορούμε να αναφερθούμε σε όλον τον υπολογιστή ως εξυπηρετητή, αφού αυτή είναι η κύρια

λειτουργία του. Παρομοίως, ως πελάτη μπορούμε να θεωρήσουμε είτε κάποιο λογισμικό που επικοινωνεί και υποβάλει αιτήματα στον εξυπηρετητή, είτε σε όλο τον υπολογιστή όταν ο εξυπηρετητής είναι άλλος υπολογιστής και οι 2 υπολογιστές είναι συνδεδεμένοι σε ένα δίκτυο.

Η επικοινωνία μεταξύ πελάτη και εξυπηρετητή γίνεται μέσω ενός τοπικού δικτύου, ή ακόμα και μέσω του Διαδικτύου. σε μεγάλα δίκτυα όπου ο εξυπηρετητής αναλαμβάνει πολλές εξυπηρετήσεις είναι συνήθως υπολογιστής που διαφέρει ως προς τη σύνθεσή του από άλλους κοινούς υπολογιστές, μιας και οι δυνατότητες του είναι σαφώς αναβαθμισμένες. Κύρια χαρακτηριστικά ενός εξυπηρετητή είναι οι επεξεργαστές που υποστηρίζει και χρησιμοποιεί για την επεξεργασία των δεδομένων που δέχεται, οι γρήγοροι και μεγάλης χωρητικότητας σκληροί δίσκοι αλλά και οι ταχύτατες μνήμες που υποστηρίζει. Συνήθως συνοδεύεται από σύστημα διπλής τροφοδοσίας (dual power supply) και από συσκευή αδιάλειπτης παροχής ενέργειας (UPS), για μεγαλύτερη αξιοπιστία και σιγουριά στις παρεχόμενες υπηρεσίες του [7].

1.6 Φυλλομετρητής

Εικόνα 1.2 Διάσιμοι φυλλομετρητές ιστού

Ένας Web browser (φυλλομετρητής ιστοσελίδων, πλοηγός Web, πρόγραμμα περιήγησης Web ή περιηγητής Ιστού) είναι ένα λογισμικό που επιτρέπει στον χρήστη του να προβάλλει, και να αλληλεπιδρά με, κείμενα, εικόνες, βίντεο, μουσική, παιχνίδια και άλλες πληροφορίες συνήθως αναρτημένες σε μια ιστοσελίδα ενός ιστότοπου στον Παγκόσμιο Ιστό ή σε ένα τοπικό δίκτυο. Το κείμενο και οι εικόνες σε μια ιστοσελίδα μπορεί να περιέχουν υπερσυνδέσμους προς άλλες ιστοσελίδες του

ίδιου ή διαφορετικού ιστότοπου. Ο Web browser επιτρέπει στον χρήστη την γρήγορη και εύκολη πρόσβαση σε πληροφορίες που βρίσκονται σε διάφορες ιστοσελίδες και ιστότοπους εναλλάσσοντας τις ιστοσελίδες μέσω των υπερσυνδέσμων. Οι φυλλομετρητές χρησιμοποιούν τη γλώσσα μορφοποίησης HTML για την προβολή των ιστοσελίδων, για αυτό η εμφάνιση μιας ιστοσελίδας μπορεί να διαφέρει ανάλογα με τον browser.

Οι πλοηγοί Web ουσιαστικά αποτελούν λογισμικό πελάτη του δικτυακού πρωτοκόλλου επιπέδου εφαρμογών HTTP. Για κάθε browser διατίθενται, επίσης, και αρκετά πρόσθετα στοιχεία («add-ons» ή «plug-ins»), με στόχο την επαύξηση των δυνατοτήτων τους, τη βελτίωση της χρηστικότητας τους και την προστασία του χρήστη σε θέματα ασφάλειας [8].

2 ΣΥΣΤΗΜΑΤΑ ΔΙΑΧΕΙΡΗΣΗΣ ΠΕΡΙΕΧΟΜΕΝΟΥ

Στο δεύτερο κεφάλαιο θα αναφερθούμε με τα συστήματα διαχείρησης περιεχομένου, με τα είδη των συστημάτων αυτών, καθώς επίσης και με τα πιο δημοφιλέστερα συτήματα ανοικτού κώδικα συγκρίνοντας τα μεταξύ τους.

2.1 Ορισμός συστήματος διαχείρησης περιεχομένου

Μέχρι πριν από λίγα χρόνια, ο μόνος τρόπος για να διατηρήσει μια εταιρία το site της ενημερωμένο ήταν να συνάψει συμβόλαιο με μια εταιρία παροχής υπηρεσιών συντήρησης. Τα τελευταία χρόνια, όμως, οι ίδιες οι εταιρίες παροχής τέτοιων υπηρεσιών προσφέρουν μια πολλά υποσχόμενη εναλλακτική λύση. Πολλές από αυτές έχουν αναπτύξει ειδικά συστήματα, τα οποία μειώνουν το χρόνο και το κόστος λειτουργίας ενός δικτυακού τόπου.

Ο όρος Content Management Systems (CMS, Συστήματα Διαχείρισης Περιεχομένου) αναφέρεται στις εφαρμογές που επιτρέπουν στον πελάτη να διαχειρίζεται το δικτυακό του περιεχόμενο, όπως κείμενα, εικόνες, πίνακες κ.λπ., με εύκολο τρόπο, συνήθως παρόμοιο με αυτόν της χρήσης ενός κειμενογράφου. Οι εφαρμογές διαχείρισης περιεχομένου επιτρέπουν την αλλαγή του περιεχόμενου χωρίς να είναι απαραίτητες ειδικές γνώσεις σχετικές με τη δημιουργία ιστοσελίδων ή γραφικών, καθώς συνήθως

τα κείμενα γράφονται μέσω κάποιων online WYSIWYG ("What You See Is What You Get") html editors, ειδικών δηλαδή κειμενογράφων, παρόμοιων με το MS Word, που επιτρέπουν τη μορφοποίηση των κειμένων όποτε υπάρχει ανάγκη.

Οι αλλαγές του site μπορούν να γίνουν από οποιονδήποτε υπολογιστή που είναι συνδεμένος στο Διαδίκτυο, χωρίς να χρειάζεται να έχει εγκατεστημένα ειδικά προγράμματα επεξεργασίας ιστοσελίδων, γραφικών κ.λπ. Μέσω ενός απλού φυλλομετρητή ιστοσελίδων (browser), ο χρήστης μπορεί να συντάξει ένα κείμενο και να ενημερώσει άμεσα το δικτυακό του τόπο.
Αυτό που αποκαλούμε πολλές φορές "δυναμικό περιεχόμενο" σε ένα website δεν είναι άλλο παρά οι πληροφορίες που παρουσιάζονται στο site και μπορούν να αλλάξουν από τους ίδιους τους διαχειριστές του μέσω κάποιας εφαρμογής, η οποία ουσιαστικά μπορεί να εισάγει (προσθέτει), διορθώνει και να διαγράφει εγγραφές σε πίνακες βάσεων δεδομένων, όπου τις περισσότερες φορές καταχωρούνται όλες αυτές οι πληροφορίες.
Αυτό σημαίνει ότι δεν χρειάζεται να δημιουργηθούν πολλές ξεχωριστές ιστοσελίδες για την παρουσίαση των πληροφοριών στο site, αλλά αρκεί ένας ενιαίος σχεδιασμός στα σημεία όπου θέλουμε να εμφανίζεται το περιεχόμενό μας, καθώς και να υπάρχει ο ειδικός σε κάποια συγκεκριμένη γλώσσα προγραμματισμού (ASP, PHP, Coldfusion, Perl, CGI κ.λπ.), ο οποίος αναλαμβάνει να εμφανίσει τις σωστές πληροφορίες στις σωστές θέσεις.

Έτσι, για το δικτυακό τόπο μιας εφημερίδας π.χ., που απαιτεί εύλογα καθημερινή ενημέρωση αλλά δεν χρησιμοποιεί κάποιο σύστημα Content Management, θα πρέπει ο υπεύθυνος για το σχεδιασμό του (designer) να δημιουργήσει μια σελίδα με τα γραφικά, την πλοήγηση και το περιβάλλον διεπαφής (interface) του website, ο υπεύθυνος ύλης να τοποθετήσει το περιεχόμενο στα σημεία της ιστοσελίδας που θέλει, και να ενημερωθούν οι σύνδεσμοι των υπόλοιπων σελίδων ώστε να συνδέονται με την καινούργια. Αφού την αποθηκεύσει, πρέπει να την ανεβάσει στο website μαζί με τις υπόλοιπες ιστοσελίδες που άλλαξαν.

Αντιθέτως, αν ο δικτυακός τόπος λειτουργεί με χρήση κάποιου συστήματος CM, το μόνο που έχει να κάνει ο διαχειριστής του είναι να ανοίξει τη σχετική φόρμα

εισαγωγής νέου άρθρου στη διαχειριστική εφαρμογή του website και να γράψει ή να επικολλήσει (copy-paste) τα στοιχεία που επιθυμεί. Αυτόματα, μετά την καταχώριση γίνονται από το ίδιο το σύστημα διαχείρισης περιεχομένου όλες οι απαραίτητες ενέργειες, ώστε το άρθρο να είναι άμεσα διαθέσιμο στους επισκέπτες και όλοι οι σύνδεσμοι προς αυτό ενημερωμένοι.

Με την αυξητική τάση χρήσης των CMS στην Ελλάδα και το εξωτερικό, γίνεται εμφανές ότι το μέλλον του Διαδικτύου σε ό,τι αφορά περιεχόμενο και πληροφορίες που πρέπει να ανανεώνονται τακτικά, ανήκει στα προγράμματα διαχείρισης περιεχομένου, αφού προσφέρουν πολλά πλεονεκτήματα, ταχύτητα και ευκολίες στη χρήση τους [9].

2.2 Είδη συστημάτων διαχείρησης περιεχομένου

Τα Content Management Systems διακρίνονται σε ορισμένες κατηγορίες ανάλογα με ορισμένα βασικά χαρακτηριστικά τα οποία παρουσιάζουν. Μπορούν, λοιπόν, να κατηγοριοποιηθούν ανάλογα με το είδος του παρόχου τους και ανάλογα με το που βρίσκεται ο χώρος αποθήκευσης και διαχείρισης της βάσης δεδομένων και του CMS [10].

ASP και Licensed : με βάση το χώρο αποθήκευσης και διαχείρισης

- Στα Application Service Provider (ASP) CMS, δηλαδή Υποστήριξης Παρόχου Υπηρεσίας, ο κατασκευαστής τους φιλοξενεί όλα τα δεδομένα και το λογισμικό στους server της εταιρίας του. Με αυτόν τον τρόπο απαλείφονται τα έξοδα για μία ακριβή αγορά λογισμικού και hardware του συστήματος, που θα φιλοξενεί το CMS. Παράλληλα μειώνονται και οι ανάγκες για τεχνικούς πόρους, όπως για παράδειγμα για συντηρητές του δικτύου των υπολογιστών. Τέλος, βασικότερο πλεονέκτημα ενός τέτοιου είδους συστήματος είναι η συνεχής εξέλιξη, καθώς ο πάροχος προωθεί διαρκώς νέες λειτουργίες του προϊόντος και ανανεώσεις στου πελάτες του, προσφέροντας έτσι το χαρακτηριστικό της άμεσης ανανέωσης και πρωτοπορίας της ιστοσελίδας.
- Στα CMS με παροχή άδειας (Licensed), ο πάροχος του πουλάει το προϊόν, δηλαδή παρέχει άδεια χρήσης του, δεν εμπλέκεται στην όλη διαδικασία

λειτουργίας του και ο χρήστης είναι πλέον υπεύθυνος, ώστε να το εγκαταστήσει, να το ρυθμίσει και να το συντηρήσει. Διαχειριστής σε αυτήν την περίπτωση είναι το τεχνικό τμήμα του οργανισμού. Η προσέγγιση αυτών των CMS εξασφαλίζει ότι φιλοξενείς και διαχειρίζεσαι τα δικά σου δεδομένα. Επίσης, τα Licensed είναι ιδανικά για οργανισμούς, οι οποίοι διατηρούν ήδη στις εγκαταστάσεις του κάποιο είδος υπηρεσίας παρόμοιας, όπως για παράδειγμα το σύστημα Διαχείρισης Εξυπηρέτησης Πελατών (CRM), οπότε θα ήταν πιο φθηνό να συντηρούν ταυτόχρονα και ένα CMS.

Commercial, Open source, Managed Open Source : Με βάση το είδος του παρόχου

- Commercial: πρόκειται για λογισμικό, που προέρχεται είτε από κερδοσκοπικές είτε από μη κερδοσκοπικές εταιρίες. Οι πάροχοι αυτοί αναπτύσσουν κατά κύριο λόγο το λογισμικό, το οποίο στην συνέχεια πουλάνε και υποστηρίζουν τεχνικά. Στην σημερινή εποχή, οι εμπορικές αυτές λύσεις είναι πιο συχνές από τις ελεύθερες λύσεις των open source CMS.
- Open Source: πρόκειται για μία λύση CMS, που δημιουργείται και συντηρείται από έναν ανεπίσημο και ανιδιοτελή συνεργάτη μίας κοινότητας χρηστών. Στην συνέχεια, το λογισμικό αυτό διανέμεται για συγκεκριμένο σκοπό στα μέλη αυτής της κοινότητας. Για αυτά τα ανοιχτά λογισμικά θα πρέπει σαφώς στο κόστος τους να συμπεριληφθεί και τα έξοδα τεχνικής υποστήριξης τους, τα οποία σαφώς και είναι αυξημένα σε αυτό το μοντέλο. Ακόμη, θα πρέπει να προστεθεί το εσωτερικό hardware και λογισμικό και το τεχνικό προσωπικό που χρειάζεται για να συντηρηθεί αυτό το σύστημα, όπως είναι για παράδειγμα οι προγραμματιστές, οι οποίοι εγκαθιστούν τις ανανεώσεις και εξελίσσουν τις λειτουργίες του προγράμματος.
- Managed Open Source: πρόκειται για έναν συνδυασμό της εμπορικής και της ελεύθερης προσέγγισης, όπου ένας πάροχος υιοθετεί μία open- source λύση σαν την βασική του πλατφόρμα και στην συνέχεια προσφέρει την λύση αυτή σε άλλους σε συνδυασμό με συμπληρωματικές υπηρεσίες τεχνικής υποστήριξης. Αυτή η λύση ουσιαστικά σχεδόν δεν υπάρχει σήμερα στην

κοινότητα των μη-κερδοσκοπικών παρόχων. Παρόλα αυτά, καθώς οι λύσεις open-source ωριμάζουν, οι ειδικοί περιμένουν ότι θα εμφανιστούν πολύ πιο έντονα.Όσον αφορά τη διάκριση των CMS σε σχέση με τον τρόπο παράδοσης τους, έχουμε δύο μορφές λογισμικού. Υπάρχουν εκατοντάδες επιλογές από CMS και των δύο κατηγοριών και η κάθε μία από αυτές διαφέρει στην υλοποίηση, στο κόστος και στην εξυπηρέτηση.

2.3 Τα δημοφιλέστερα συστήματα διαχείρησης περιεχομένου

Το WordPress, το Joomla και το Drupal είναι τα τρία πιο δημοφιλή συστήματα διαχείρισης περιεχομένου (CMS) σε απευθείας σύνδεση. Και τα τρία είναι open source και χτισμένα σε PHP + MySQL. Και τα τρία διαφέρουν σημαντικά όσον αφορά τις δυνατότητες , την ικανότητα , την ευελιξία και την ευκολία χρήσης . Παρακάτω , θα ρίξουμε μια ματιά σε μερικά από τα πλεονεκτήματα και τα μειονεκτήματα της κάθε μία από αυτές τις λύσεις CMS [11].

2.3.1 Drupal

Το Drupal είναι ο παππούς των συστημάτων CMS σε αυτόν τον κατάλογο - κυκλοφόρησε για πρώτη φορά στις αρχές του 2001. Όπως το WordPress και το Joomla, το Drupal είναι open-source και βασίζεται σε PHP - MySQL. Το Drupal είναι εξαιρετικά ισχυρό και προγραματιστικά - φιλικό. Ας εξετάσουμε μερικά πλεονεκτήματα και τα μειονεκτήματα του Drupal:

Πλεονεκτήματα του Drupal

- Εξαιρετικά Ευέλικτο: Θέλετε ένα απλό blog με μια στατική σελίδα μπροστά; Το Drupal μπορεί να το χειριστεί αυτό . Θέλετε ένα ισχυρό backend που μπορεί να υποστηρίξει εκατοντάδες χιλιάδες σελίδες με εκατομμύρια χρήστες κάθε μήνα; Σίγουρα, το Drupal μπορεί να κάνει και αυτό. Το λογισμικό είναι ισχυρό και ευέλικτο.
- Προγραματιστικά Φιλικό: Η βασική εγκατάσταση του Drupal είναι αρκετά γυμνή. Οι προγραμματιστές καλούνται να δημιουργήσουν τις δικές τους λύσεις . Αν και αυτό δεν το κάνει πολύ φιλικό για τους απλούς χρήστες, υπόσχεται μια σειρά από δυνατότητες για τους προγραμματιστές .

- Ισχυρές δυνατότητες SEO: Το Drupal έχει σχεδιαστεί από το έδαφος προς τα πάνω για να είναι φιλικό προς τις μηχανές αναζήτησης.
- Φιλικό προς τις Επιχειρίσεις: Η ισχυρή διαχειριστική έκδοση με ACL δυνατότητες φέρνουν το Drupal CMS πρώτο για τους πελάτες των επιχειρήσεων. Το λογισμικό μπορεί επίσης να χειριστεί εκατοντάδες χιλιάδες σελίδες του περιεχομένου με ευκολία .
- Σταθερότητα: Στο Drupal οι κλίμακες είναι σταθερές, ακόμα και όταν εξυπηρετούν χιλιάδες χρήστες ταυτόχρονα .

Μειονεκτήματα του Drupal

- Απότομα Learning Curve: Ας πούμε ότι έχετε μια ιστοσελίδα σε WordPress και την ξανακατασκευάζετε για Drupal. Η διαφορά θα είναι ότι μέχρι σήμερα οδηγάγατε ένα αυτοκίνητο και ξάφνικά το αυτοκίνητό σας μετατράπηκε σε cockpit Boeing 747 με εσάς στην θέση του οδηγού. Όπως καταλαβαίνετε τα πράγματα γίνονται περίπλοκα. Εάν δεν έχετε μεγάλες δυνατότητες κωδικοποίησης και δεν σας αρέσει να διαβάζετε τους τόνους των τεχνικών εγγράφων, θα βρείτε Drupal εξαιρετικά δύσκολο για να το χρησιμοποιήσετε για κανονική χρήση .
- Έλλειψη Δωρεάν Plugins: Τα Plugins στο Drupal ονομάζονται « modules » . Λόγω του ότι θεωρείτε επιχειρισιακό προϊόν, τα περισσότερα καλά module δεν είναι δωρεάν.
- Έλλειψη Templates: Η γυμνή εγκατάσταση του Drupal μοιάζει με μια έρημο μετά από μια ξηρασία και η έλλειψη των θεμάτων δεν κάνει τα πράγματα καλύτερα. Θα πρέπει να βρείτε ένα καλό σχεδιαστή - διαχειριστή, εάν δεν θέλετε να δείτε τον την ιστοσελίδα σας σαν ένα θλιβερό λείψανο από το 2002.

Συμπέρασμα

Το Drupal είναι ένα ολοκληρωμένο επιχειρησιακό CMS. Συνιστάται για μεγάλα έργα όπου η σταθερότητα , επεκτασιμότητα και η δύναμη που έχουν είναι προτεραιότητα έναντι στην ευκολία χρήσης και την αισθητική .

2.3.2 Joomla

Το Joomla είναι ένα λογισμικό διαχείρισης περιεχομένου ανοικτού κώδικα με αρκετές διακλαδώσεις από το Mambo . Είναι μία από τις πιο δημοφιλείς λύσεις CMS στον κόσμο και μπορεί να υπερηφανεύεται για τα πάνω από 30 εκατ. downloads μέχρι σήμερα. Θα πρέπει να σημειωθεί κάτι το οποίο δεν έιναι τυχαίο το Cloud.com και το Linux.com όπως και άλλα πολύ μεγάλα site ειναι κατασκευασμένα σε Joomla.

Πλεονεκτήματα του Joomla

- Φιλικό προς το χρήστη: Το Joomla δεν είναι WordPress , αλλά είναι ακόμα σχετικά εύκολο στη χρήση.
- Ισχυρή Κοινότητα προγραμματιστών: Όπως το WordPress , έτσι και το Joomla έχει μια ισχυρή κοινότητα προγραμματιστών. Η βιβλιοθήκη plugin (που ονομάζεται « extensions ») είναι πολή μεγάλη, με έναν τόνο από δωρεάν προς χρήση, plugins.
- Επέκταση και Μεταβλητότητα: Οι επεκτάσεις Joomla χωρίζονται σε πέντε κατηγορίες - components, plugins, templates, ενότητες και γλώσσες. Κάθε ένα από αυτά διαφέρει σε λειτουργία, τη δύναμη και την ικανότητα. Τα Components, για παράδειγμα, λειτουργούν ως « μίνι - εφαρμογές », που μπορούν να αλλάξουν την εγκατάσταση του Joomla συνολικά. Οι Ενότητες, από την άλλη πλευρά, προσθέτουν δευτερεύουσες δυνατότητες όπως δυναμικό περιεχόμενο, RSS feeds, και λειτουργία αναζήτησης στην ιστοσελίδα σας.
- Ισχυρές δυνατότητες διαχείρισης περιεχομένου: Σε αντίθεση με WordPress, το Joomla αρχικά είχε σχεδιαστεί ως ένα επιχειρησιακό CMS όπως το drupal. Αυτό το καθιστά πολύ πιο ικανό στο χειρισμό μεγάλου όγκου των αντικειμένων από WordPress.

Μειονεκτήματα του Joomla

- Χρειάζεται πολύ εκμάθηση: Δεν μπορείτε να μεταβείτε σε μια εγκατάσταση Joomla και να αρχίσετε να στείνετε από μόνος σας την ιστοσελίδα σας, εφόσον δεν είστε εξοικειωμένοι με το λογισμικό. Η καμπύλη εκμάθησης δεν είναι

απότομη , αλλά αυτό μπορεί να είναι αρκετό για να εκφοβίσει τους περιστασιακούς χρήστες.

- Στερείται SEO Δυνατότητες: Κάνοντας το WordPress SEO friendly είναι τόσο εύκολο όσο και η εγκατάσταση ενός δωρεάν plugin. Με το Joomla , θα χρειαστείτε έναν τόνο δουλειά για να φτάσετε στο ίδιο επίπεδο φιλικότητας στις μηχανές αναζήτησης. Αν δεν έχετε τον προϋπολογισμό για να προσλάβετε έναν έμπειρο SEO expert, ίσως να θέλετε να εξετάσετε εναλλακτικές λύσεις.
- Υποστήριξη Περιορισμένη ACL: Ο ACL (κατάλογος Access Control) αναφέρεται σε μια λίστα δικαιωμάτων που μπορεί να χορηγηθεί σε συγκεκριμένους χρήστες για συγκεκριμένες σελίδες. Ο ACL είναι ένα ζωτικής σημασίας συστατικό της κάθε επιχείρησης. Το Joomla άρχισε να υποστηρίζει ACL μόνο μετά από την έκδοση 1.6. Η Υποστήριξη ACL είναι ακόμη περιορισμένη στη σταθερή έκδοση v2.5.1 , καθιστώντας το ακατάλληλο για εταιρικούς πελάτες. Βέβαια στην σταθερή έκδοση 3.2. πλέον τα πραγματα είναι πολύ καλά.

Συμπέρασμα

Το Joomla σας δίνει τη δυνατότητα να κατασκευάσετε μια ιστοσελίδα με περισσότερη δομική σταθερότητα του περιεχόμενου από το WordPress και έχει ένα αρκετά διαισθητικό interface. Αν θέλετε ένα πρότυπο δικτυακό τόπο με πρότυπες ικανότητες - ένα blog , μια στατική / δυναμική front-end ιστοσελίδα, ένα φόρουμ , κλπ. τότε επιβάλλετε να χρεισημοποιήσετε το Joomla . Το Joomla είναι επίσης μια καλή επιλογή για μικρά και μεσαίου μεγέθους καταστήματα ηλεκτρονικού εμπορίου. Αν θέλετε κάτι πιο ισχυρό για τη χρήση της επιχειρήσής σας, να εξετάσετε το Drupal αλλά θα σας φανεί πάρα πολύ δύσκολό και θα χρειαστείτε support από κάποιον expert.

2.3.3 WordPress

Η New York Times, το CNN, το Forbes και το Reuters είναι κατασκευασμένα σε Wordpress. Περισσότερα από 68 εκατομμύρια ιστοσελίδες χρησιμοποιούν WordPress , καθιστώντας το, το αγαπημένο λογισμικό blogging στον κόσμο το οποίο είναι αρκετά ευέλικτο.Παρακάτω, θα ρίξουμε μια ματιά στα πλεονεκτήματα και μειονεκτήματα από τη χρήση WordPress :

Πλεονεκτήματα του WordPress

- Πολλαπλές Συγγραφείς: Το WordPress χτίστηκε από το έδαφος προς τα πάνω για να φιλοξενήσει πολλούς συγγραφείς - ένα κρίσιμο χαρακτηριστικό για κάθε σοβαρή δημοσίευση.
- Τεράστια Plugin Library: Το WordPress είναι η αφίσα του παιδιού της κοινότητας προγραμματιστών ανοιχτού κώδικα, η οποία έχει αναπτύξει εκατοντάδες χιλιάδες plugins για αυτό. Βέβαια υπάρχουν μερικά πράγματα που το WordPress δεν μπορεί να κάνει, ακόμα και με την εκτεταμένη βιβλιοθήκη των plugins που διαθέτει.
- Φιλικό προς το χρήστη: Το WordPress είναι εύκολο στη χρήση και διαθέτει πού καλή αισθητική, για bloggers για που ξεκινάνε τώρα για πρώτη φορά. Μπορείτε να ανοιξετε ένα θέμα, να προσθέστε μερικά plugins, και να αρχίσετε το blogging μέσα σε λίγα λεπτά.
- Ισχυρές δυνατότητες SEO: Με plugins όπως All in One SEO και όχι μόνο του σαν cms, όπως το joomla, μπορείτε να ξεκινήσετε το blogging αμέσως χωρίς να ανησυχείτε για on-site SEO.
- Εύκολη Παραμετροποίηση: Το σύστημα WordPress έχει σχεδιαστεί για εύκολη προσαρμογή. Καθένας με μια μικρή κατανόηση της HTML και CSS μπορεί να προσαρμόσει θέματα για να ταιριάξει στις ανάγκες του.
- Ευελιξία: Το WordPress μπορεί να κάνει σχεδόν τα πάντα, να τρέξει ένα κατάστημα e-commerce, να φιλοξενήσει μια ιστοσελίδα βίντεο, να χρησιμεύσει ως ένα χαρτοφυλάκιο, ένα blog της εταιρείας σας χάρη στα plugins και τα προσαρμοσμένα θέματα.

Μειονεκτήματα του WordPress

- Ασφάλεια: Όπως στην κατηγορία κορυφαίο λογισμικό με τα εκατομμύρια των εγκαταστάσεων, Το WordPress είναι συχνά ο στόχος των χάκερ. Το ίδιο το λογισμικό δεν είναι πολύ ασφαλές έξω από το κουτί και θα πρέπει να εγκαταστήσετε plugins τρίτων για να ενισχύσετε την ασφάλεια της εγκατάστασης WordPress.

- Ασυμβατότητα με μεγάλα Plugins: Η ομάδα του WordPress αποδεσμεύει συνεχώς νέες ενημερώσεις για να διορθώσετε τα κενά ασφαλείας και τα προβλήματα patch. Αυτές οι ενημερώσεις είναι συχνά ασύμβατες με μεγάλα plugins . Αν το site σας βασίζεται σε παλαιότερα πρόσθετα, ίσως χρειαστεί να κρατήσετε μακριά την ενημέρωση (η οποία κάνει το site σας ακόμα πιο επιρρεπή σε επιθέσεις hacking).
- Περιορισμένες Επιλογές Design: Ακόμα κι αν το WordPress είναι απείρως προσαρμόσιμο, οι περισσότερες εγκαταστάσεις WordPress εξακολουθούν να μοιάζουν με WordPress Default Template. Παρά το γεγονός ότι οι πρόσφατες ενημερώσεις και βελτιώσεις σε plugins - θέματα έχουν διορθώσει αυτό το πρόβλημα κάπως, το WordPress εξακολουθεί να παρεμποδίζεται από τις περιορισμένες επιλογές σχεδιασμού.
- Περιορισμένες δυνατότητες διαχείρισης περιεχομένου: Το WordPress σχεδιάστηκε αρχικά ως μια πλατφόρμα blogging. Αυτό έχει επηρεάσει την ικανότητά του να χειριστεί μεγάλες ποσότητες περιεχομένου. Αν σκοπεύετε να δημοσιεύσετε εκατοντάδες θέσεις blog ανά εβδομάδα (δεν είναι ασυνήθιστο για τους μεγάλους εκδοτικούς οίκους), θα απογοητευτείτε και καλό θα ήταν να στραφείτε προς το joomla.

Συμπεράσματα

Το WordPress συχνά αποκαλείται «μίνι CMS » . Δεν είναι τόσο ισχυρό ή ικανές όσο το Drupal ή το Joomla, αλλά είναι αρκετά εύκολο για κάθε επαγγελματία χρήστη. Χρησιμοποιήστε το WordPress, αν θέλετε μια απλή - εύκολη στη χρήση blogging λύση που μπορεί να φιλοξενήσει πολλούς συγγραφείς εύκολα.

3 ΤΕΧΝΟΛΟΓΙΕΣ ΚΑΙ ΕΡΓΑΛΕΙΑ ΑΝΑΠΤΥΞΗΣ ΙΣΤΟΤΟΠΟΥ

Σε αυτό το κεφάλαιο θα αναφερθούμε στις τεχνολογίες αλλά και τα εργαλεία ανάπτυξης που χρησιμοποιήθηκαν για την ανάπτυξη του ιστότοπου μας.

3.1 Apache HTTP Server

Ο Apache HTTP Server, είναι ένα λογισμικό διακομιστή Web και μπορεί να εκτελεστεί τόσο σε Unix λειτουργικό σύστημα όσο και σε windows.

Όταν ένας χρήστης επισκέπτεται μέσω του φυλλομετρητή του έναν ιστότοπο, τότε ο φυλλομετρητής επικοινωνεί μέσω του πρωτοκόλλου HTTP με τον διακομιστή ο οποίος παράγει τις ιστοσελίδες και τις στέλνει ως απάντηση στον χρήστη. Ο Apache βρίσκεται ακριβώς επάνω σε αυτήν την διαδικασία μετάδοσης των δεδομένων που προέρχονται από τον διακομιστή και μεταφέρονται προς τον χρήστη, χρησιμοποιώντας αντίστοιχα το πρωτόκολλο HTTP.

Το λογισμικό Apache είναι ανοιχτού κώδικα το οποίο επιτηρεί το Ίδρυμα Λογισμικού Apache. Η πρώτη του έκδοση δημιουργήθηκε από τον Robert McCool και κυκλοφόρησε το 1993 με όνομα NCSA HTTPd. Έπαιξε πολύ σημαντικό ρόλο στην επέκταση του παγκόσμιου ιστού, διότι ήταν η πρώτη βιώσιμη εναλλακτική επιλογή απέναντι στον εξυπηρετητή http της εταιρείας Netscape. Από το 1996, ο Apache έχει γίνει το πιο δημοφιλές λογισμικό διακομιστή HTTP που χρησιμοποιείται παγκοσμίως. Το 2011 υπολογίστηκε ότι ο Apache εξυπηρετεί το 63% όλων των δικτυακών τόπων, και το 66% των πιο πολυσύχναστων.

3.2 PHP

Εικόνα 3.1 Λογότυπο php

Τα αρχικά PHP προέρχονται από τις λέξεις Personal Home Page (σε ελεύθερη μετάφραση Προσωπική Ιστοσελίδα) η οποία είναι μια γλώσσα προγραμματισμού που σχεδιάστηκε για την δημιουργία δυναμικών ιστοσελίδων και είναι γνωστή ως Hypertext PreProcessor (προεπεξεργαστής υπερκειμένου).

Σε αντίθεση με άλλες γλώσσες scripting του διαδικτύου όπως η html, η css , κ.α. η γλώσσα php είναι μια server-side scripting (εκτελείται στον εξυπηρετητή) γλώσσα που συνήθως γράφεται πλαισιωμένη από HTML για την εμφάνιση των αποτελεσμάτων. Έτσι η php δεν στέλνεται άμεσα σε έναν πελάτη (client), αν τ' αυτού πρώτα αναλύεται και μετά αποστέλλεται το παραγόμενο αποτέλεσμα (εξού και η ευρέως ονομασία της σε HyperText preprocessor). Η php είναι ευρέως διαδεδομένη για την πληθώρα δυνατοτήτων που μπορεί να προσφέρει σε διαδικτυακές εφαρμογές, όπως να θέσει ερωτήματα σε βάσεις δεδομένων, να δημιουργήσει εικόνες, να διαβάσει και να γράψει αρχεία, να συνδεθεί σε απομακρυσμένους υπολογιστές, κ.α.

Η php δημιουργήθηκε από τον φοιτητή Rasmus Lerdorf ως μια συλλογή από scripts γραμμένα στην γλώσσα προγραμματισμού perl που τα χρησιμοποιούσε στην προσωπική του ιστοσελίδα. Η αρχική χρήση της php από τον Rasmus ήταν η παρακολούθηση στατιστικών στοιχείων που αφορούσαν την επισκεψιμότητα στο προσωπικό του βιογραφικό. Αργότερα έγραψε ξανά τα scripts σε γλώσσα C για λόγους καλύτερης απόδοσης, επεκτείνοντας ταυτόχρονα τις δυνατότητες της, υποστηρίζοντας έτσι την χρήση διαδικτυακών forms και σύνδεση με βάσεις δεδομένων. Το πρώτο επίσημο όνομα της php ήταν PHP/FI από τα «Personal Home Page/Forms Interpreter» (Προσωπική Ιστοσελίδα / διερμηνέας φορμών). Μετά από αυτή την δημιουργία ο Rasmus διέθεσε τον κώδικα στην ιστοσελίδα του ώστε να επωφεληθούν κι άλλοι από αυτόν.

Σήμερα πλέον η php είναι η πιο διαδεδομένη γλώσσα προγραμματισμού για ιστοσελίδες παγκοσμίως, μετρά την 5η έκδοση της και πλήθος σχεδιαστικών εργαλείων και έτοιμων scripts έχουν στηριχθεί στην γλώσσα αυτή.

3.3 PhpMyAdmin

Εικόνα 3.2 Λογότυπο phpMyAdmin

Γραμμένο σε κώδικα PHP το phpMyAdmin είναι ένα εργαλείο ανοικτού κώδικα με σκοπό την διαχείριση της MySQL που βρίσκεται εγκατεστημένη σε κάποιον web server. Είναι ένα σύστημα διαχείρισης βάσεων δεδομένων σε μορφή ιστοσελίδας που καθιστά πιο εύκολη την διαχείριση αυτή εξ αποστάσεως. Προσφέρει δυνατότητες όπως εκτέλεση κώδικα SQL, δημιουργία, τροποποίηση ή διαγραφή βάσεων δεδομένων , πινάκων, πεδίων, ή και γραμμών επίσης διαθέτει την δυνατότητα εκτέλεσης κώδικα SQL καθώς και διαχείρισης χρηστών που έχουν πρόσβαση σε αυτό.

Δημιουργός του phpMyAdmin αποτέλεσε ο τότε σύμβουλος και αργότερα ιδρυτής της εταιρείας λογισμικού Maguma, Tobias Ratschillerm όπου το 1998 άρχισε να εργάζεται πάνω στην δημιουργία διαδικτυακής πλατφόρμας για την MySQL, βασιζόμενο σε PHP, και εμπνευσμένο από το MySQL-Webadmin. Το 2000 κι ενώ είχε δημιουργήσει τότε το phpAdsNew παράτησε το έργο λόγω έλλειψης χρόνου.

Έκτοτε το phpMyAdmin έγινε η πιο δημοφιλής εφαρμογή διαχείρισης MYSQL γραμμένο σε PHP με μεγάλη κοινότητα χρηστών και συνεργάτες. Τελικά το 2001 μια ομάδα τριών προγραμματιστών έγραψαν ξανά το phpMyAdmin στο SourceForge.net και ανέλαβαν την ανάπτυξη.

Μερικές από τις δύνατότητες που απρέχονται από το phpmyadmin είναι:

- Εισαγωγή δεδομένων από CSV και SQL,
- Εξαγωγή δεδομένων σε διάφορες μορφές: CSV, SQL, XML, PDF, ISO/IEC 26300 – Word, Excel, LaTeX και άλλα,
- Διαχείρηση πολλαπλών διακομιστών,
- Δημιουργία γραφικών PDF της βάσης δεδομένων,
- Δημιουργία σύνθετων ερωτημάτων χρησιμοποιώντας το Query-by-Example (QBE),
- Αναζήτηση σε παγκόσμιο επίπεδο σε μια βάση δεδομένων ή ένα υποσύνολο αυτής.

3.4 MySQL

Εικόνα 3.3 Λογότυπο MySQL

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων ανοιχτού λογισμικού βασισμένο στην γλώσσα SQL. Πήρε το όνομα της από την κόρη του δημιουργού της Michael Widenius ονόματι My.

Χρησιμοποιείται σε μια πληθώρα εφαρμογών, αλλά συνήθως σε διακομιστές Web. Μια δυναμική ιστοσελίδα γραμμένη πχ σε php στηρίζει τις πληροφορίες που ανταλλάσσει μεταξύ των χρηστών της μέσω μιας βάση δεδομένων. Η MySQL αποτελεί αυτό ακριβώς το αναπόσπαστο κομμάτι μιας δυναμικής ιστοσελίδας προσθέτοντας, αφαιρώντας , και τροποποιώντας πληροφορίες που υπάρχουν αποθηκευμένες σε μία βάση δεδομένων και προέρχονται από την αλληλεπίδραση των χρηστών με τον ιστότοπο.

Πλεονεκτήματα της MySQL

- Απόδοση
  Η MySQL είναι χωρίς αμφιβολία γρήγορη. Πολλές από αυτές τις δοκιμές δείχνουν ότι η MySQL είναι αρκετά πιο γρήγορη από τον ανταγωνισμό.
- Χαμηλό κόστος
  Η MySQL είναι διαθέσιμη δωρεάν, με άδεια ανοικτού κώδικα ή με χαμηλό κόστος, αν πάρετε εμπορική άδεια, αν απαιτείται από την εφαρμογή σας.
- Ευκολία χρήσης
  Οι περισσότερες μοντέρνες βάσεις δεδομένων χρησιμοποιούν SQL. Αν έχετε

χρησιμοποιήσει ένα άλλο σύστημα διαχείρισης βάσεων δεδομενων δεν θα έχετε πρόβλημα να προσαρμοστείτε σε αυτό.

- Μεταφερσιμότητα
  Η MySQL μπορεί να χρησιμοποιηθεί σε πολλά διαφορετικά συστήματα Unix όπως επίσης και στα Microsoft Windows.
- Κώδικας Προέλευσης
  Όπως και με την PHP, μπορείτε να πάρετε και να τροποποιήσετε τον κώδικα προέλευσης της MySQL.

3.5 HTML

Εικόνα 3.4 Λογότυπο HTML

Τα αρχικά της HTML προέρχονται από τις λέξεις HyperText Markup Language που σε ελεύθερη μετάφραση σημαίνουν «Γλώσσα Σήμανσης Υπερκειμένου». Η HTML δεν είναι γλώσσα προγραμματισμού όπως η java ή η C που μεταγλωττίζονται πρώτα σε εκτελέσιμα αρχεία προκειμένου να εκτελεστούν, αλλά είναι μια γλώσσα σήμανσης, δηλαδή ένας ειδικός τρόπος γραφής κειμένου, αυτό την καθιστά πολύ έυκολα εκτελέσιμη χωρίς καμίας μορφής μεταγλώττιση παρά μόνο έναν φυλλομετρητή που να μπορεί να την ερμηνεύσει Οποιοσδήποτε μπορεί να δημιουργήσει ένα αρχείο HTML χρησιμοποιώντας έναν απλό επεξεργαστή κειμένου. Ιστορικά προήλθε από την γλώσσα SGML (Standard Generalized Markup Language) που επινοήθηκε από την IBM προκειμένου να λυθεί το πρόβλημα της μη τυποποιημένης εμφάνισης κειμένων στα διάφορα υπολογιστικά συστήματα. Η HTML είναι η πρώτη και πιο διαδεδομένη γλώσσα περιγραφής της δομής μιας ιστοσελίδας, στην οποία και πλέον όλο το διαδίκτυο χρησιμοποιεί.

Βασικό χαρακτηριστικό της HTML είναι οι ειδικές ετικέτες της ή αλλιώς tags, όπου με βάση αυτές τις ετικέτες καθορίζεται η μορφοποίηση του κειμένου που γράφεται μέσα σε ένα αρχείο html. Οι ετικέτες αυτές ορίζουν συνήθως την αρχή και το τέλος μιας λειτουργίας και συνήθως τα ονόματα των ετικετών αυτών βρίσκονται πάντα μεταξύ των συμβόλων του μικρότερου «<» και του μεγαλύτερου «>», πχ το «<B>» που σημαίνει έντονα γράμματα (προέρχεται από το Bold) χρησιμοποιείται με μια ετικέτα ανοίγματος, το περιεχόμενο της και ύστερα την ετικέτα κλεισίματος που περιλαμβάνει ότι και η ανοίγματος μόνο που διαθέτει και έναν επιπλέον χαρακτήρα, μια μεσοκάθετο «/». Πχ για να τονίσουμε έντονα την λέξη «καλημέρα» θα γράφαμε «<B>Καλημέρα</B>». Η γλώσσα html είναι case insensitive, δηλαδή δεν υπάρχει διαφορά από τα tags που γράφονται με κεφαλαία ή μικρά (πεζά) γράμματα. Συνήθως ένα αρχείο html έχει κατάληξη htm ή html.

Για να μπορέσει ενας φυλλομετρητής να ερμηνεύσει σωστά την html έχουν θεσπιστεί κάποιοι κανόνες, όπου οι κανόνες αυτοί για να υπάρχει απόλυτη συμβατότητα μεταξύ των συστημάτων, καθιερώθηκαν ως προδιαγραφές. Έτσι κάθε έιδος υπολογιστή μπορεί να ανοίξει και να δείξει μια ιστοσελίδα. Οι πρώτες προδιαγραφές ήταν η html 2.0, διάφορες όμως εταιρείες λογισμικού οπως η microsoft και η netscape μετάλλασαν συνεχώς την html προσθέτοντας νέες δυνατότητες, δημιουργόντας έτσι πρόβλημα συμβατότητας όπου ορισμένες ιστοσελίδες ήταν μόνο με τους φυλλομετρητές των εν λόγω εταιρειών. Ακόμη και σήμερα υπάρχουν διαφορες στην απεικόνηση κάποιας ιστοσελίδας απο διαφορετικούς φυλλομετρητές αλλα όχι λόγω HTML. Τα σημερινά συνήθως προβλήματα προκείπτουν απο την χρήση της Javascript μέσα σε αυτήν.

Η πληθώρα επιλογών που σήμερα έχει δοθεί στους προγραμματιστές και στους απλούς χρήστες για να αναπτύξουν μια ιστοσελίδα περιλαμβάνουν ένα σωρό από προγράμματα και εφαρμογές που μπορεί κάποιος είτε να δημιουργήσει μια ιστοσελίδα με απλά κλικ του ποντικιού του είτε να συγγράψει σε επίπεδο κώδικα HTML. Σαφώς την πρώτη περίπτωση ο δημιουργός δεν έχει τον απόλυτο έλεγχο του κώδικα με αποτέλεσμα πολλές φορές να υπάρχει οπτικό χάος στην προσπάθεια των φυλλομετρητών να εμφανίσουν την ιστοσελίδα. Γι αυτούς ακριβός τους λόγους και επειδή θα έπρεπε να καλυφθεί με κάποιο τρόπο το χάσμα μεταξύ του στησίματος

μερικών κλικ και της συγγραφής κώδικα html , υπάρχουν σχεδιαστικά προγράμματα όπως το Dreamweaver της Adobe ή το Aptana που είναι open source στο διαδίκτυο, που συνδυάζουν και τους δύο τρόπους ανάπτυξης, κάνοντας έτσι την ανάπτυξη ιστοσελίδας, μια διαδικασία πιο πλήρης.

3.6 CSS

Εικόνα 3.5 Λογότυπο CSS

Τα αρχικά CSS προέρχονται από τα Cascading Style Sheet (σε ελεύθερη μετάφραση Επικαλυπτόμενα Φύλλα Στύλ). Το CSS αποτελεί γλώσσα μορφοποίησης όπως και η HTML μόνο που απευθύνεται προς την HTML. Η κύρια χρησιμότητα της είναι η δημιουργία στυλ που αφορούν το κείμενο, τα μεγέθη αντικειμένων μέσα σε μια ιστοσελίδα και οποιαδήποτε άλλη ιδιότητα μορφοποίησης μπορεί να έχει ένα αντικείμενο HTML.

Το CSS αποτελεί το βασικό εργαλείο για μια ομοιόμορφη εμφάνιση ενός ιστότοπου που αποτελείται από πολλές ιστοσελίδες Ο τρόπος με τον οποίο λειτουργεί είναι απλούστατος, κάθε tag της html είναι και ένα αντικείμενο για την CSS όπου μπορεί να του ορίσει ιδιότητες εφαρμογής μορφοποίησης Τα στυλ αυτά μπορούν να καθοριστούν μόνο μια φορά σε ένα αρχείο css και να εφαρμόζονται σε κάθε ιστοσελίδα html που το αναφέρει μέσα στον κώδικα της. Έτσι ένας προγραμματιστής μπορεί να αλλάξει στυλ σε πολλές ιστοσελίδες ταυτόχρονα, αλλάζοντας αυτό το αυτόνομο αρχείο CSS.

Το CSS διαθέτει πληθώρα επιλογών όσο αφορά την μορφοποίηση, και δεν αναφέρεται μόνο στο στυλ κειμένου, αλλά επίσης και σε οποιαδήποτε άλλο αντικείμενο της HTML, όπως για παράδειγμα την μορφοποίηση ενός πίνακα, μιας λίστας κουκίδων ή αριθμητική κ.α. καθορίζοντας το πάχος και το χρώμα το συνόρων, το κενό ανάμεσα

τους κ.τ.λ. Το CSS χρησιμοποιεί μια ιστοσελίδα HTML ως βάση προκειμένου να εφαρμόσει τα στυλ του, δίνοντας έτσι έναν πιο ακριβή έλεγχο για το πως θα εμφανίζονται οι ιστοσελίδες άσχετα με την μορφοποίηση που ήδη διαθέτει από την αρχή η HTML. Για τον λόγο αυτό σήμερα όλες οι νεες ιστοσελίδες στηρίζονται ως βασικό συστατικό δημιουργία τους το CSS.

3.7 Javascript

Εικόνα 3.6 Λογότυπο Javascript

Η Javascript προήλθε από την γλώσσα προγραμματισμού C και σε βάθος χρόνου εξελίχθηκε ενσωματώνοντας χαρακτηριστικά από νεότερες γλώσσες. Αρχικός σκοπός της δημιουργίας της ήταν η εκτέλεση απομακρυσμένου κώδικα στην πλευρά του πελάτη από τον φυλλομετρητή του, και έτσι χαρακτηρίστηκε ως client-side γλώσσα προγραμματισμού Η εκτέλεση της στην πλευρά του πελάτη, παράγει εκείνη την στιγμή το αποτέλεσμα για το οποίο έχει προγραμματιστεί να κάνει, συνήθως HTML.

Εκτός από την εκτέλεση της από την πλευρά του πελάτη αρχικά χρησιμοποιήθηκε και για την συγγραφή κώδικα από την πλευρά του διακομιστή, από την ίδια τη Netescape στο προϊόν LiveWire, με μικρή επιτυχία. Η χρήση της Javascript σε διακομιστές εμφανίζεται μέχρι και σήμερα, με τη διάδοση του Node.js, ενός μοντέλο προγραμματισμού βασισμένο στα γεγονότα.

Σήμερα η κύρια χρήση της Javascript εστιάζει κυρίως στο δυναμικό περιεχόμενο των ιστοσελίδων τόσο στο περιβάλλον εμφάνισης μιας ιστοσελίδας όσο και στο περιβάλλον λειτουργίας. Χάρις την Javascript οι ιστοσελίδες μπορούν να αποκτήσουν κίνηση, να εμφανίσουν και να εξαφανίσουν κομμάτια τους, να αλλάξουν εμφάνιση με ένα κλικ καθώς ακόμη και να αποτελέσει βασικό κρίκο της αλυσίδας μιας λειτουργίας, όπως η αποστολή αιτήματος προς ένα διακομιστή και η αναμονή για

απάντηση των αποτελεσμάτων και άλλα πολλά. Για να λειτουργεί αρμονικά η javascript με την μετάδοση τέτοιων πληροφοριών, έχει αναπτυχθεί το πρότυπο JSON (τα αρχικά προέρχονται από τα JavaScript Object Notation δηλ Αντικειμενοστραφής Συμβολισμός Javascript) όπου με έναν συγκεκριμένο τρόπο οι πληροφορίες μεταδίδονται με μια δομή που γίνεται εύκολα προσπελάσιμη μέσα από τις αντίστοιχες εντολές της Javascript. Η συγκεκριμένη δομή είναι εύκολη τόσο στο να παραχθεί όσο και στο να διαβαστεί από διαφορετικές γλώσσες προγραμματισμού και όχι μόνο από την Javascript.

Για να εισαχθεί κώδικας javascript σε μια ιστοσελίδα, χρησιμοποιειται η ετικέτα HTML «script», για παράδειγμα «<script type='text/javascript'> /* Κώδικας Javascript */ </script>». Η Javascript ακολουθεί ένα δικό της συντακτικό στο οποίο τα κεφαλαία διαφέρουν από τα μικρά γράμματα, και κάθε εντολή χωρίζεται με το ελληνικό ερωτηματικό από μια άλλη εντολή.

Τόσο το πρότυπο JSON όσο και η Javascript χρησιμοποιούνται κατά κόρων στις σημερινές ιστοσελίδες, προσφέροντας έτσι μια πληθώρα δυνατοτήτων και μια ευρεία γκάμα συμβατότητας για τους διαφορετικές πλατφόρμες, όπως πχ η εφαρμογή facebook για android στηρίζεται στα παραγόμενα JSON της κανονικής ιστοσελίδες του facebook.

3.8 XAMPP

Εικόνα 3.7 Λογότυπο XAMPP

Το XAMPP είναι ένα πακέτο προγραμμάτων ελεύθερου λογισμικού, και λογισμικού ανοιχτού κώδικα και ανεξαρτήτου πλατφόρμας το οποίο περιέχει εξυπηρετητή ιστοσελίδων http Apache, βάση δεδομένων MySQL και έναν διερμηνέα για κώδικα γραμμένο σε γλώσσες προγραμματισμού PHP και Perl.

Τα αρχικά ειναι ακρωνύμιο και αναφέρεται στα αρχικά των X που σημαίνει cross-platform δηλαδή ανεξαρτήτου πλατφόρμας, Apache , MySQL, Php και Perl. Το λογισμικό XAMPP μπορεί να τρέξει σε Windows, Linux, και Mac OS και χρησιμοποιείται ευρέως για την σχεδίαση ιστοσελίδων με τεχνολογίες όπως PHP, JSP και Servlets.

3.9 Zend Eclipse Luna

Εικόνα 3.8 Λογότυπο eclipse

Το Eclipse ειναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated development environment, IDE), μια σουίτα δηλαδή λογισμικού που βοηθά στην ανάπτυξη προγραμμάτων υπολογιστή. Κυκλοφορεί ελεύθερο, είναι ανοικτού κώδικα και αποτελείται από έναν επεξεργαστή πηγαίου κώδικα, έναν μεταγλωττιστή, εργαλεία αυτόματης παραγωγής κώδικα, αποσφαλματωτή, συνθέτη, σύστημα ελέγχου εκδόσεων και εργαλεία κατασκευής γραφικών διασυνδέσεων χρήστη.

Το eclipse μπορεί να χρησιμοποιηθεί για την ανάπτυξη εφαρμογών σχεδόν σε οποιαδήποτε γλώσσα όπως Java, Ada, ABAP, C, C++, COBOL, Fortran, Haskell, Javascript, Lasso, Lua, Natural, Perl, PHP, Prolog, Python, R, Ruby, Scala, Clojure, Groovy, Scheme και Erlang. Επίσης μπορεί να χρησιμοποιηθεί για την ανάπτυξη πακέτων για το λογισμικό Mathematica. Το eclipse έχει μεγάλες δυνατότητες επεκτασιμότητας και γι αυτό διακρίνεται σε πολλές παραλλαγές ανάλογα με την γλώσσα. Κάθε παραλλαγή συνοδεύεται και από τα αντίστοιχα πρόσθετα (plugins) ανάπτυξης προσαρμοσμένα στην αντίστοιχη γλώσσα όπως για παράδειγμα το Java development tools (JDT) το οποίο χρησιμοποιείται στην Java και στην Scala, το Eclipse CDT που χρησιμοποιείται στην C/C++ και το Eclipse PDT στην PHP.

Η πρώτη έκδοση προήλθε από την IBM VisualAge. Το πακέτο λογισμικού ανάπτυξης (Software Development Kit - SDK), το οποίο περιλαμβάνει και τα εργαλεία ανάπτυξης Java, δημιουργήθηκε για Java Developers, οι χρήστες όμως μπορούν να

επεκτείνουν τις δυνατότητες εγκαθιστώντας πρόσθετα γραμμένα για την πλατφόρμα του Eclipse, όπως πακέτα ανάπτυξης για άλλες γλώσσες, καθώς επίσης ενα πρόσθετο αποκλειστικά στα μέτρα του εκάστοτε χρήστη που το έγραψε.

Το Zend Eclipse Luna είναι ενα IDE που περιλαμβάνει το PDT για την ανάπτυξη ιστοσελίδων υποστηριζόμενες σε γλώσσες HTML, CSS, JS, PHP, SQL στο περιβάλλον του eclipse. Καθιστά ευκολότερη την ανάπτυξη web εφαρμογών και την επεκτασιμότητα τους.

Μερικά από τα χαρακτηριστικά που προσφέρει είναι:

- Μορφοποίηση κώδικα για ευκολότερη ανάγνωση,
- Περιήγηση ανάμεσα στον κώδικα,
- Έτοιμα πρότυπα κώδικα,
- Αποσφαλμάτωση php,
- Έλεγχος σύνταξης, και
- Υποβοήθηση συγγραφής με αυτόματη συμπλήρωση και εμφάνιση πληροφοριών σχετικά με εντολές.

3.10 Gimp

Εικόνα 3.9 Λογότυπο Gimp

Το Gimp (GNU Image Manipulation Program) είναι ένα δωρεάν λογισμικό επεξεργασίας γραφικών τύπου raster. Είναι εργαλείο που ασχολείται κυρίως με την διαμόρφωση και την επεξεργασία εικόνας και είναι ελεύθερα διαθέσιμο σε εκδόσεις προσαρμοσμένες για τα πιο δημοφιλή λειτουργικά συστήματα όπως τα Microsoft Windows, το Mac OS X της Apple, και το GNU/Linux.

Εκτός από την λεπτομερή επιδιόρθωση της εικόνας και τη σχεδίαση ελεύθερης μορφής, το GIMP μπορεί να φέρει εις πέρας βασικές εργασίες επεξεργασίας εικόνας, όπως η αλλαγή μεγέθους, η επεξεργασία και η «καλλιέργεια» φωτογραφιών, το φωτομοντάζ συνδυάζοντας πολλαπλές εικόνες και η μετατροπή μεταξύ διαφορετικών μορφών εικόνας. Επίσης μπορεί να χρησιμοποιηθεί για να δημιουργήσει κινούμενες εικόνες σε πολλές μορφές, όπως GIF και MPEG μέσω του Animation plug-in.

Το όραμα του προϊόντος είναι ότι το GIMP είναι ένα δωρεάν λογισμικό για απαιτητικούς χρήστες που βασίζεται στην εφαρμογή γραφικών, στην επεξεργασία και δημιουργία πρωτότυπων εικόνων, στα γραφικά στοιχεία των ιστοσελίδων και στην τέχνη μετατροπής των στοιχείων μιας διεπαφής που έχει ως επίκεντρο τον χρήστη.

3.11 Mozilla Firefox

Εικόνα 3.10 Λογότυπο Firefox

Ο Mozilla Firefox είναι ελεύθερος και ανοικτού κώδικα φυλλομετρητής (browser) του παγκόσμιου ιστού. Προήλθε από το Application Suite της Mozilla και η ανάπτυξή του εξακολουθεί να γίνεται κατά μεγάλο ποσοστό από την Mozilla Corporation, ενώ συνεισφέρουν και μεμονωμένοι χρήστες σε μικρότερο βαθμό. Ο Firefox κατείχε το 25% της καταγεγραμμένης χρήσης φυλλομετρητών Ιστού για τον Νοέμβριο του 2009, κατατάσσοντας τον στην δεύτερη θέση των πιο δημοφιλών φυλλομετρητών παγκοσμίως, μετά τον Internet Explorer.

Στις λειτουργίες του Firefox περιλαμβάνονται φραγή αυτόκλητων αναδυόμενων παραθύρων, περιήγηση με καρτέλες, ορθογραφικός έλεγχος, επιμέρους εύρεση, ενεργοί σελιδοδείκτες, διαχείριση των μεταφορτώσεων, ιδιωτική περιήγηση και ένα ενσωματωμένο πεδίο αναζήτησης με δυνατότητα επιλογής της επιθυμητής μηχανής αναζήτησης. Περαιτέρω λειτουργίες ενεργοποιούνται μέσω πρόσθετων που

αναπτύχθηκαν από τρίτους. Τα πιο δημοφιλή από τα πρόσθετα είναι το NoScript που απενεργοποιεί τα σενάρια JavaScript, ο ενσωματωμένος στην γραμμή κατάστασης αναπαραγωγέας πολυμέσων FoxyTunes, το Adblock Plus που κάνει φραγή διαφημίσεων, το StumbleUpon, το DownThemAll! και η γραμμή εργαλείων Web Developer.

Για την απεικόνιση των ιστοσελίδων, ο Firefox χρησιμοποιεί την μηχανή διάταξης Gecko, η οποία εφαρμόζει τα περισσότερα από τα σημερινά πρότυπα του Παγκόσμιου Ιστού αλλά και επιπλέον πρότυπα που θα ισχύουν στον μέλλον. Ο Firefox λειτουργεί σε αρκετές εκδόσεις των Microsoft Windows, στο Mac OS X, στο GNU/Linux, Android και σε πολλά λειτουργικά συστήματα που προήλθαν από το Unix.

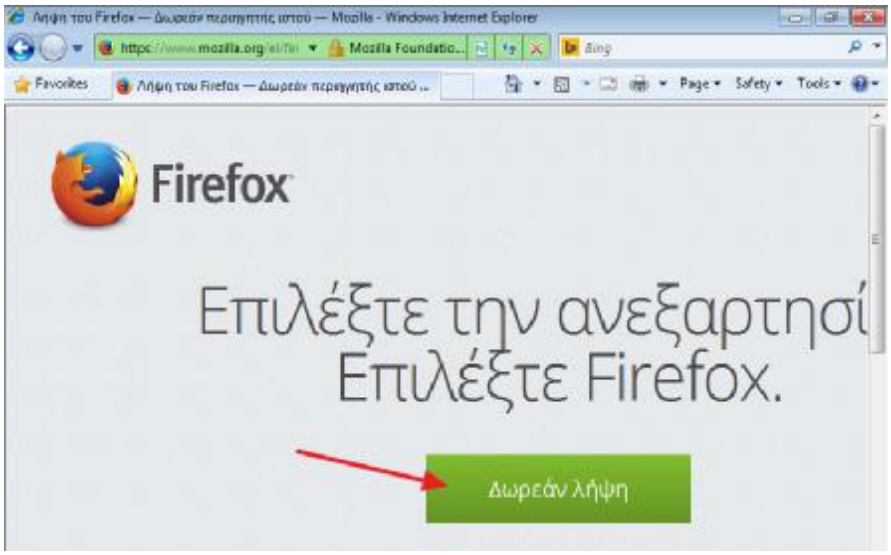
4 ΠΡΟΕΤΟΙΜΑΣΙΑ ΚΑΙ ΕΓΚΑΤΑΣΤΑΣΗ ΛΟΓΙΣΜΙΚΟΥ

Πριν ξεκινήσουμε την διαδικασία της ανάπτυξης του ιστότοπου μας, θα εγκαταστήσουμε τα κατάλληλα εργαλεία μέσω των οποίων η ανάπτυξη θα γίνει εφικτή. Σε αυτό το κεφάλαιο θα δούμε βήμα-βήμα την εγκατάσταση αυτών των εργαλείων.

4.1 Εγκατάσταση Firefox

Σε όλη την εκπόνηση της πτυχιακής εργασίας χρησιμοποιούμε τον firefox ως φυλλομετρητή, διότι μας δίνει μεγαλύτερο έλεγχο πάνω στην επίβλεψη του κώδικα της ιστοσελίδας μας, παρέχοντας μας τα κατάλληλα εργαλεία για να παρατηρήσουμε την εμφάνιση και την λειτουργικότητα του.

Για να εγκαταστήσουμε τον firefox θα πρέπει πρώτα να τον κατεβάσουμε από κάποιο άλλο πρόγραμμα περιήγησης, εμείς χρησιμοποιήσαμε τον Internet Explorer. Ξεκινώντας λοιπόν τον IE γράφουμε στο πεδίο διευθύνσεων την διεύθυνση «www.mozilla.org», όπου μας εμφανίζεται η παρακάτω αρχική οθόνη της ιστοσελίδας του firefox (Εικόνα 4.1).

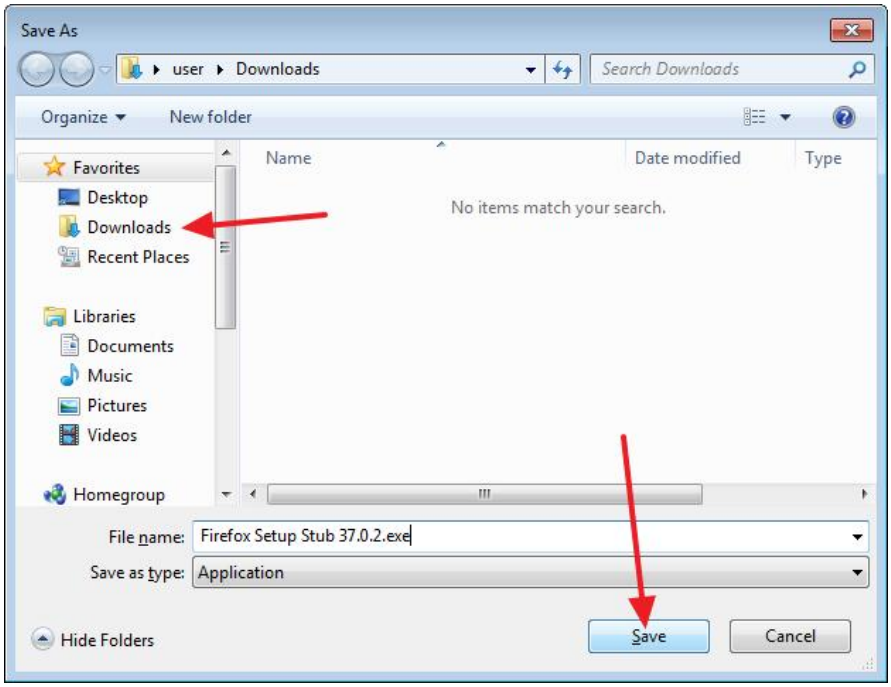


Εικόνα 4.1 Αρχική σελίδα Mozilla.org

Κάνοντας κλικ στο κουμπί «Δωρεάν λήψη» ερωτούμαστε αν θέλουμε να τρέξουμε ή να αποθηκεύσουμε αυτό το αρχείο. Εμείς επιλέξαμε την αποθήκευση (Εικόνα 4.2).

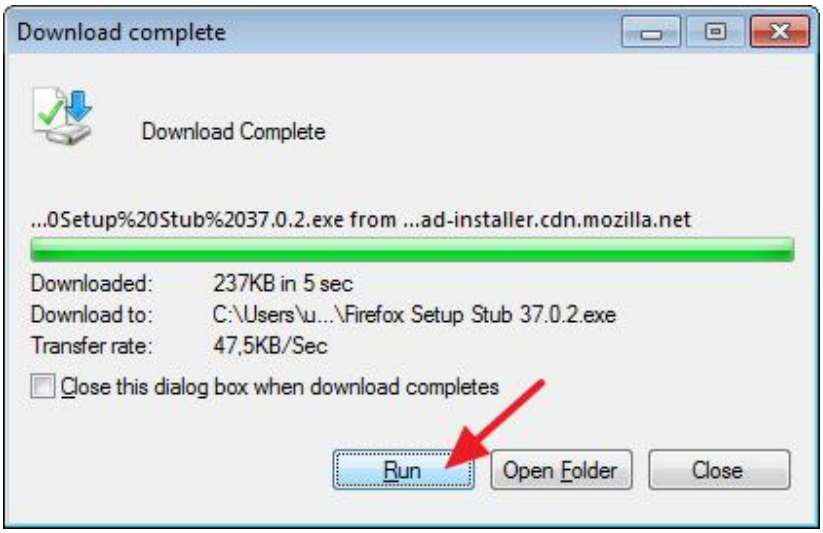
Εικόνα 4.2 Αποθήκευση αρχείου εγκατάστασης firefox

Ύστερα μας γίνεται ερώτηση για την τοποθεσία της αποθήκευσης του αρχείου εγκατάστασης. Επιλέγοντας τον κατάλογο «downloads» κάνουμε κλικ στο αποθήκευση (save) (Εικόνα 4.3).



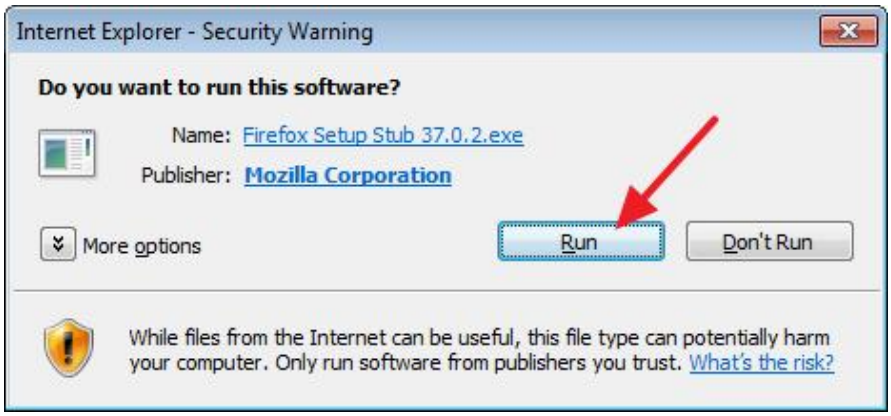
Εικόνα 4.3 Ορισμός σημείου αποθήκευσης αρχείου εγκατάστασης firefox

Ένα παράθυρο προόδου μεταφόρτωσης εμφανίζεται μετά την αποθήκευση όπου μας ενημερώνει για το πόσο χρονικό διάστημα χρειάζεται μέχρι να μεταφορτωθεί πλήρως. Όταν ολοκληρωθεί η διαδικασία τα κουμπιά «Run», «Open Folder» και «Close» γίνονται διαθέσιμα. Εμείς επιλέξαμε «εκτέλεση» (run) (Εικόνα 4.4).



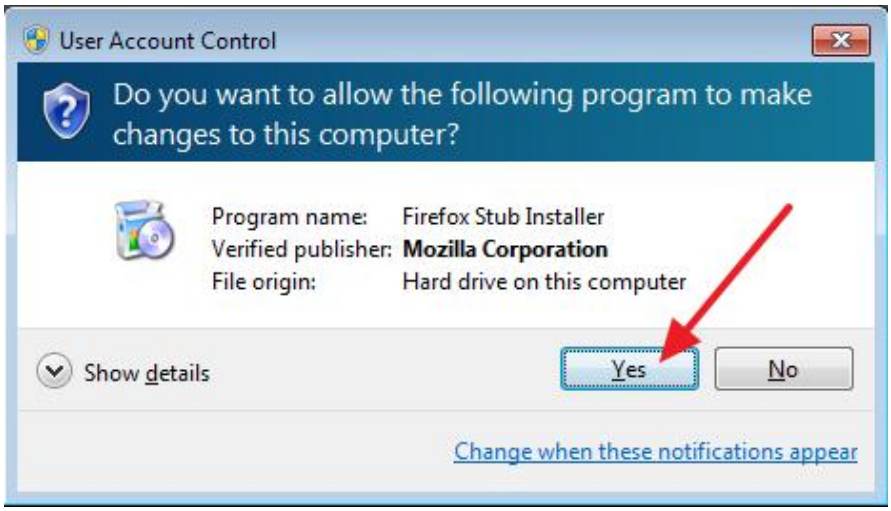
Εικόνα 4.4 Εκτέλεση αρχείου εγκατάστασης firefox

Ένα παράθυρο ασφαλείας μας ενημερώνει για τυχόν κινδύνους που μπορεί να φέρει ένα εκτελέσιμο αρχείο κατεβασμένο από το Διαδίκτυο. Εφόσον μεταφορτώσαμε το αρχείο εγκατάστασης από τον επίσημο ιστότοπο της Mozzila δεν έχουμε κάτι να φοβηθούμε, επομένως κάνουμε κλικ στο «εκτέλεση» (run) (Εικόνα 4.5).



Εικόνα 4.5 Αποδοχή ασφαλείας για την εκτέλεση του αρχείου εγκατάστασης του firefox

Αποδεχόμαστε με «ναι» (yes) τον έλεγχο UAC των windows (Εικόνα 4.6).

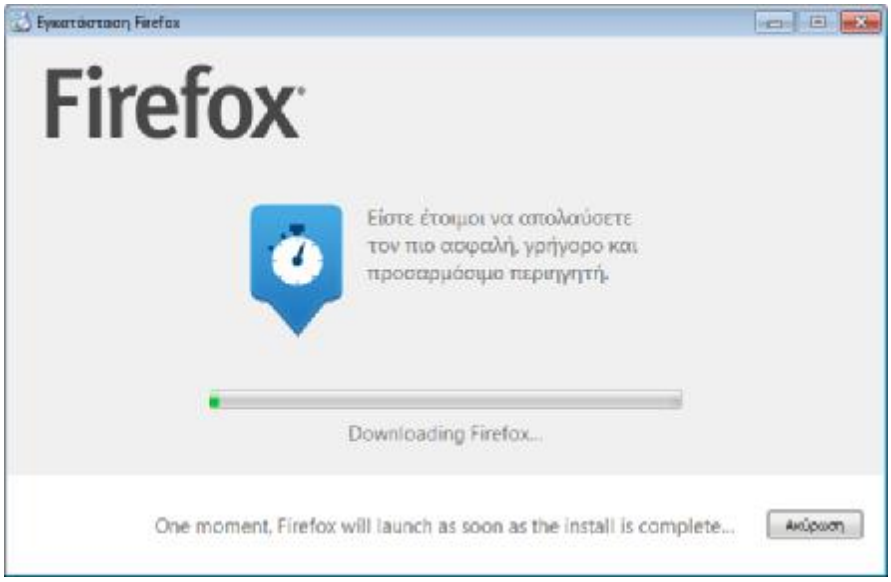


Εικόνα 4.6 Αποδοχή UAC για το αρχείο εγκατάστασης του firefox

Εμφανίζεται το παράθυρο εγκατάστασης του firefox. Μπορούμε αν θέλουμε να ελέγξουμε τι θα εγκατασταθεί και που κάνοντας κλικ στο κουμπί «Επιλογές». Εμείς δεν προχωρήσαμε σε περεταίρω αλλαγές, κάναμε επιτόπου εγκατάσταση κάνοντας κλικ στο ανάλογο κουμπί «Εγκατάσταση» (Εικόνα 4.7).

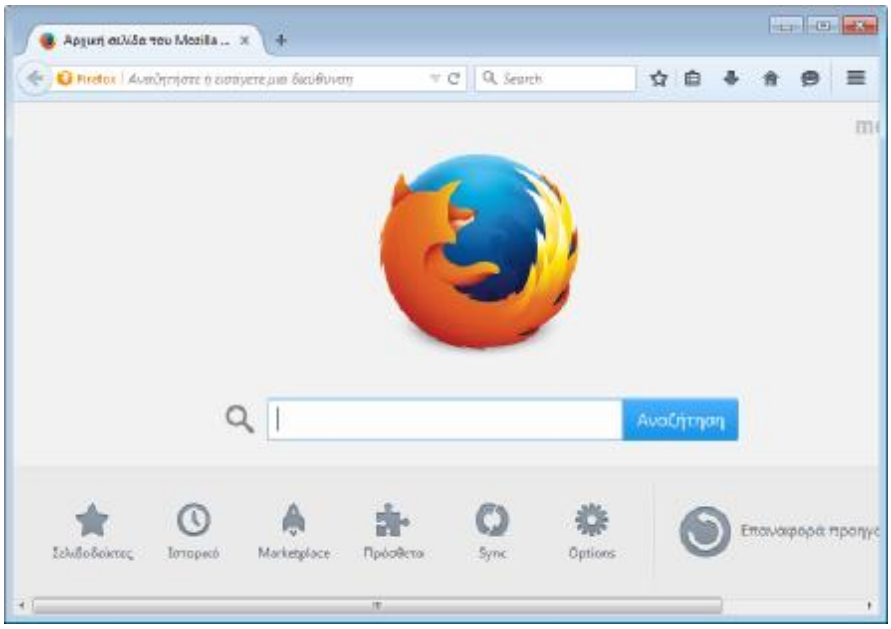
Εικόνα 4.7 Αρχική οθόνη οδηγού εγκατάστασης firefox

Ένα νέο παράθυρο μας ενημερώνει για την πρόοδο εγκατάστασης του Mozilla firefox (Εικόνα 4.8).



Εικόνα 4.8 Πρόοδος εγκατάστασης του firefox

Όταν η διαδικασία ολοκληρωθεί ο φυλλομετρητής firefox είναι πλέον εγκαταστημένος στο σύστημα μας και έτοιμος για χρήση! (Εικόνα 4.9).

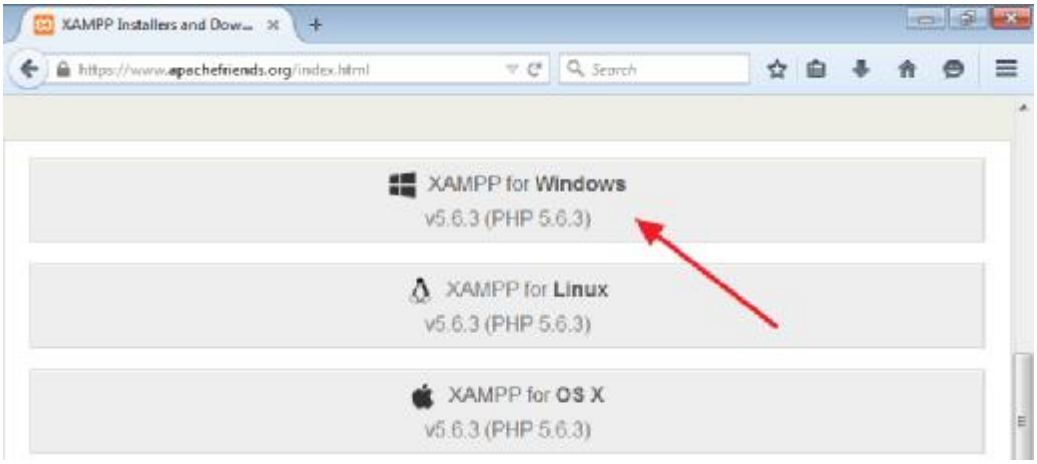


Εικόνα 4.9 Ολοκλήρωση εγκατάστασης του firefox

4.2 Εγκατάσταση Xampp

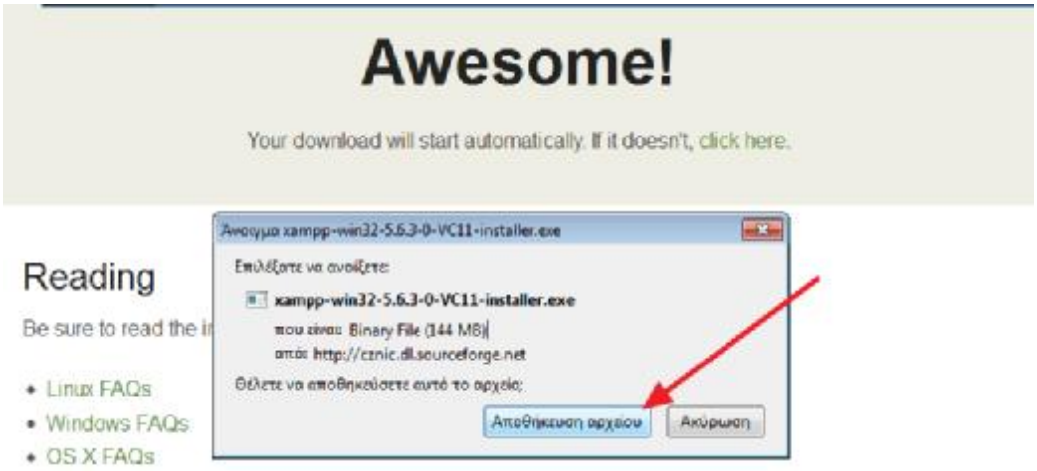
Για να έχουμε πρόσβαση στην ιστοσελίδα μας τοπικά έτσι ώστε να βλέπουμε άμεσα τις αλλαγές που κάνουμε, θα εγκαταστήσουμε έναν τοπικό εξυπηρετητή στον υπολογιστή μας. Γι αυτό τον λόγο επιλέξαμε το XAMPP, το οποίο παρέχεται δωρεάν στην διεύθυνση: https://www.apachefriends.org/index.html.

Αφού εισάγουμε την διεύθυνση στον Mozilla Firefox μεταφερόμαστε στην παρακάτω ιστοσελίδα. Επιλέγουμε την έκδοση που ταιριάζει στο λειτουργικό μας σύστημα. Στην δικιά μας περίπτωση επιλέξαμε XAMPP for Windows (Εικόνα 4.10).



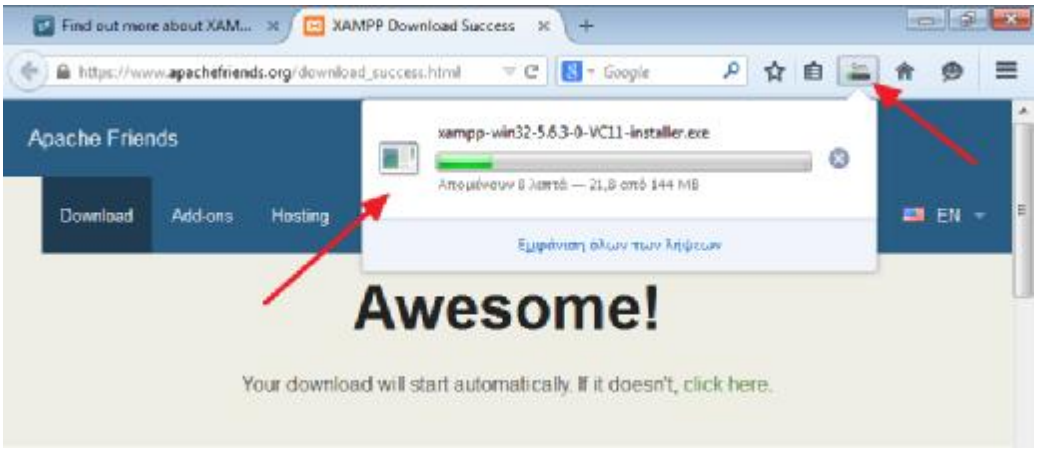
Εικόνα 4.10 Επιλογή XAMPP for Windows

Αφού κάνουμε κλικ στην ανάλογη έκδοση XAMPP που μας ενδιαφέρει, μεταφερόμαστε σε μια νέα ιστοσελίδα στη οποία αναγράφεται «Your download will start automatically. If it doesn't, click here» δηλαδή «Το αρχείο σας θα ξεκινήσει να κατεβαίνει αυτόματα. Αν αυτό δεν γίνει κάντε κλικ εδώ» (Εικόνα 4.11).



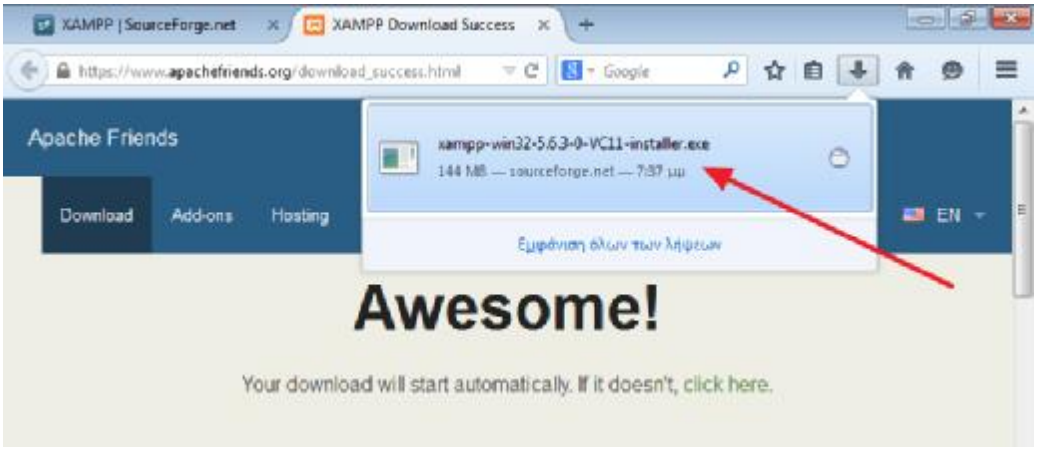
Εικόνα 4.11 Αποθήκευση αρχείου εγκατάστασης xampp

Στην περίπτωση μας όλα πήγαν κατ, ευχή και μετά από λίγα δευτερόλεπτα ξεκίνησε το κατέβασμα του αντίστοιχου αρχείου (Εικόνα 4.12).



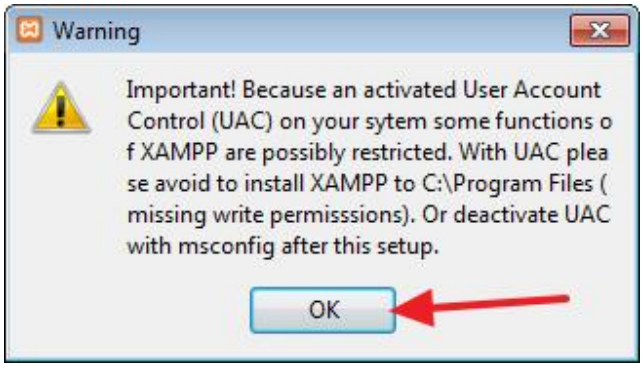
Εικόνα 4.12 Διαδικασία μεταφόρτωσης

Έχοντας πλέον στην κατοχή μας το αρχείο εγκατάστασης, κάνουμε κλικ πάνω του από την μπάρα ολοκλήρωσης μεταφορτώσεων του Firefox (Εικόνα 4.13).



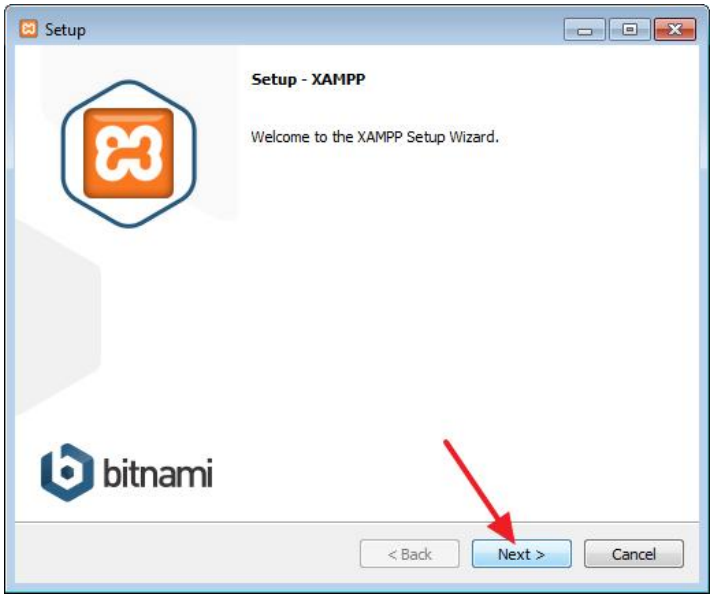
Εικόνα 4.13 Εκτέλεση οδηγού εγκατάστασης

Ένα αναδυόμενο παράθυρο μας ενημερώνει πως άμα έχουμε ενεργοποιημένο το UAC στα Windows, είναι πιθανό να επιδράσουν περιορισμοί στις λειτουργίες του XAMPP (Εικόνα 4.14).



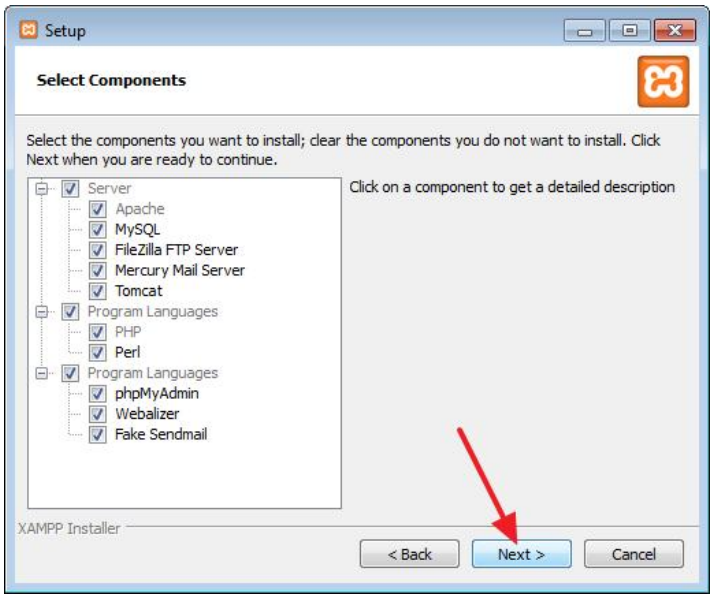
Εικόνα 4.14 Αποδοχή Προειδοποίησης UAC

Ανοίγει ο αντίστοιχος οδηγός εγκατάστασης του XAMPP, οποίος και μας καλωσορίζει στην διαδικασία. Συνεχίζουμε κάνοντας κλικ στο κουμπί «Next» (Εικόνα 4.15).



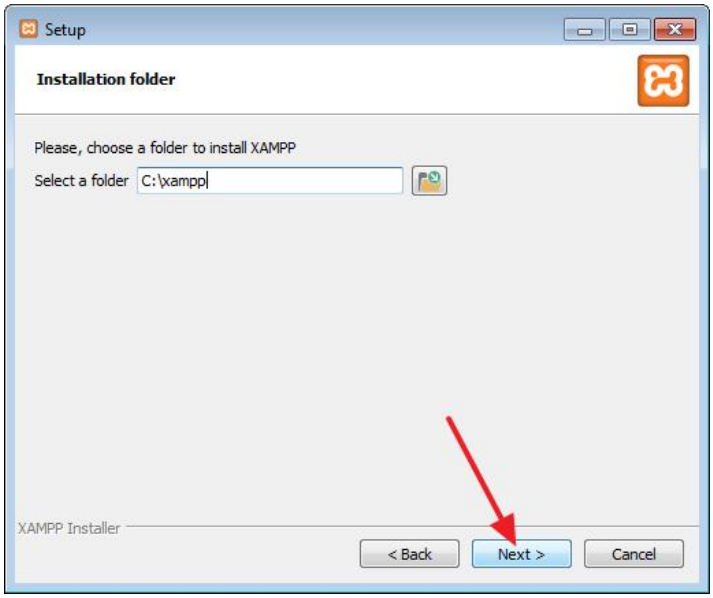
Εικόνα 4.15 Οδηγός εγκατάστασης

Η επόμενη οθόνη του οδηγού μας δείχνει μια λίστα με όλα τα χαρακτηριστικά που θέλουμε το XAMPP να εγκαταστήσει στον υπολογιστή μας. Στην περίπτωση μας ενδιαφέρουν όλα τα χαρακτηριστικά επομένως τα αφήνουμε όλα επιλεγμένα και κάνουμε κλικ στο κουμπί «Next» (Εικόνα 4.16).



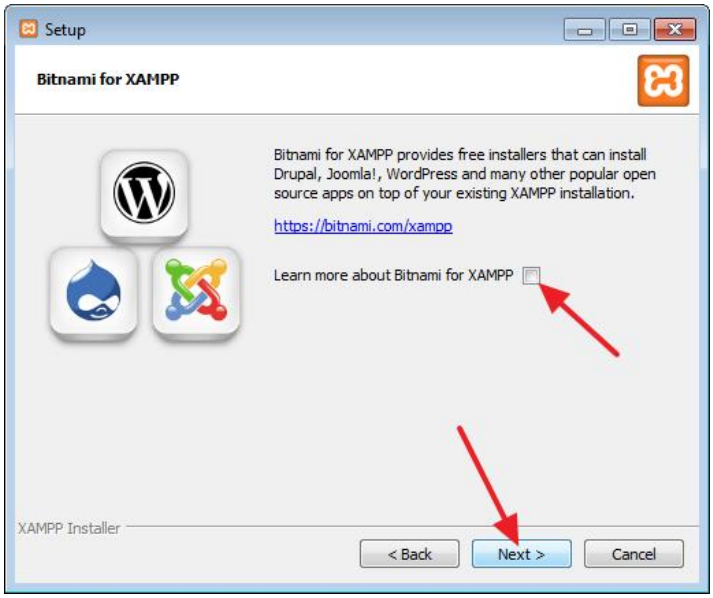
Εικόνα 4.16 Επιλογή χαρακτηριστικών

Στο επόμενο παράθυρο μας εμφανίζεται η δυνατότητα επιλογής διαδρομής εγκατάστασης για το XAMPP. Στην περίπτωση μας, αφήνουμε τα προεπιλεγμένα. Κάνουμε κλικ στο κουμπί «Next» (Εικόνα 4.17).



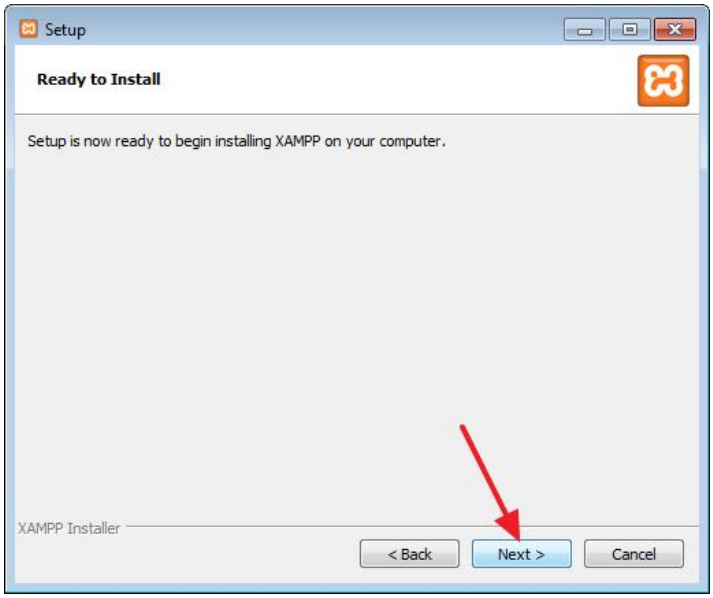
Εικόνα 4.17 Επιλογή διαδρομής εγκατάστασης

Αμέσως μετά εμφανίζεται μια ενημερωτική οθόνη σχετικά με την εταιρεία BitNami. Στην περίπτωση μας αφαιρέσαμε την επιλογή στο κουτάκι της ενημέρωσης και κάναμε κλικ στο κουμπί «Next» (Εικόνα 4.18).



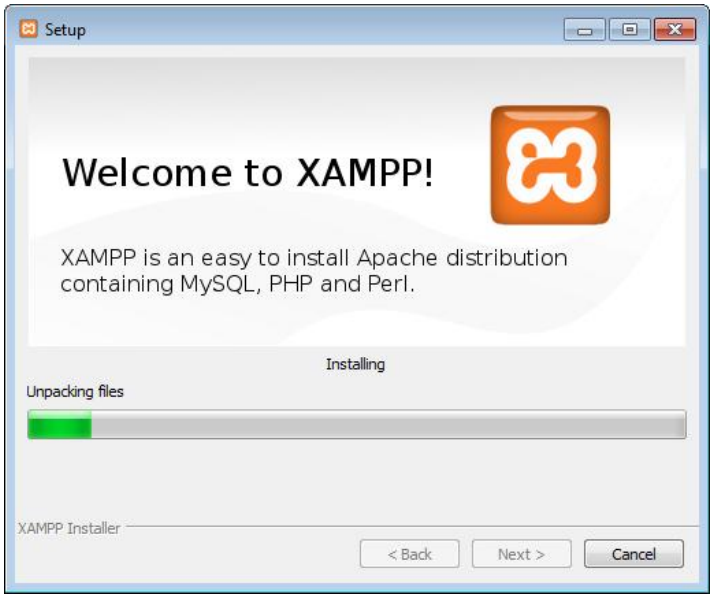
Εικόνα 4.18 Πληροφορίες για την Bitnami

Ο οδηγός πλέον είναι ρυθμισμένος και έτοιμος για εγκατάσταση, κάνουμε κλικ στο κουμπί «Next» (Εικόνα 4.19).



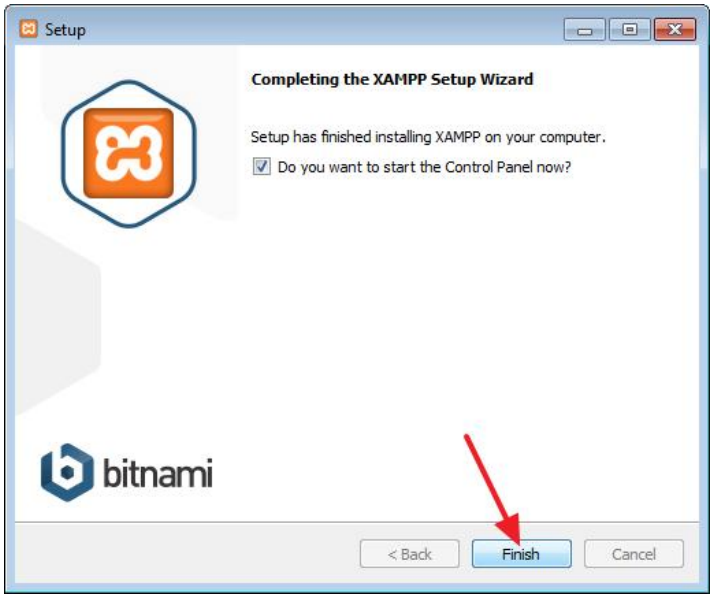
Εικόνα 4.19 Έναρξη της εγκατάστασης

Η επόμενη οθόνη μας ενημερώνει για την διαδικασία της εγκατάστασης με την αντίστοιχη μπάρα προόδου. Εδώ αναμένουμε να ολοκληρωθεί η διαδικασία (Εικόνα 4.20).



Εικόνα 4.20 Πρόοδος εγκατάστασης

Μόλις ολοκληρωθεί εμφανίζεται η τελική οθόνη επιβεβαίωσης της εγκατάστασης του XAMPP , με μια επιλογή (τικ) για την έναρξη του πίνακα ελέγχου του XAMPP αμέσως μετά το κλικ στο κουμπί «Finish». Κάνουμε κλικ στο κουμπί «Finish» (Εικόνα 4.21).



Εικόνα 4.21 Ολοκλήρωση εγκατάστασης

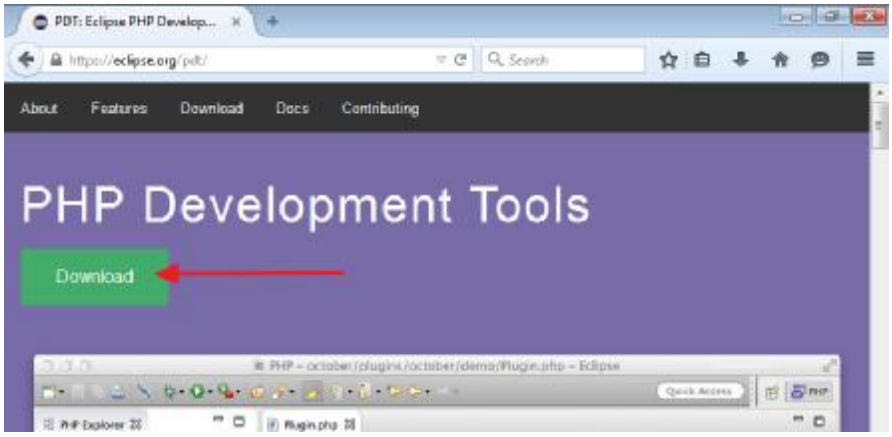
Αμέσως μετά εμφανίζεται ο σχετικός πίνακας ελέγχου του XAMPP, για τον οποίο θα δούμε αναλυτικότερα σε άλλη ενότητα της λειτουργίες του (Εικόνα 4.22).

Εικόνα 4.22 Διαχειριστικό πάνελ XAMPP

4.3 Εγκατάσταση eclipse

Για να μπορέσουμε να αναπτύξουμε μια ιστοσελίδα θα χρειαστούμε και το αντίστοιχο εργαλείο ανάπτυξης. Στην περίπτωση μας χρησιμοποιούμε το Eclipse Luna το οποίο

διατίθεται δωρεάν προς χρήση στην διεύθυνση «https://eclipse.org/pdt/». Αφού εισάγουμε την διεύθυνση στον Mozilla Firefox, μεταφερόμαστε στην παρακάτω ιστοσελίδα (Εικόνα 4.23):



Εικόνα 4.23 Αρχική σελίδα του eclipse pdt

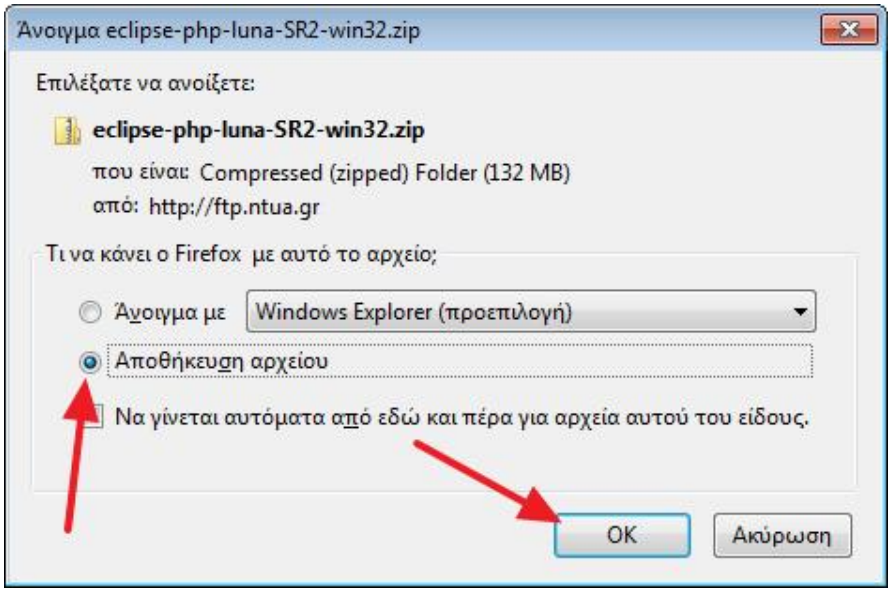
Κάνοντας κλικ στο κουμπί «Download» μεταφερόμαστε στις διαθέσιμες εκδόσεις του Eclipse Luna. Διαλέγουμε αυτή που ταιριάζει με το λειτουργικό μας, στην περίπτωση μας Windows 32bit (Εικόνα 4.24).

Εικόνα 4.24 Επιλογή έκδοσης έκδοσης eclipse που ταιριάζει στο σύστημα μας

Ύστερα μεταφερόμαστε στην σελίδα μεταφόρτωσης όπου με ένα σύνδεσμο από το κοντινότερο εξυπηρετητή μπορούμε να κατεβάσουμε το σχετικό πρόγραμμα, κάνουμε κλικ στον σύνδεσμο (Εικόνα 4.25).

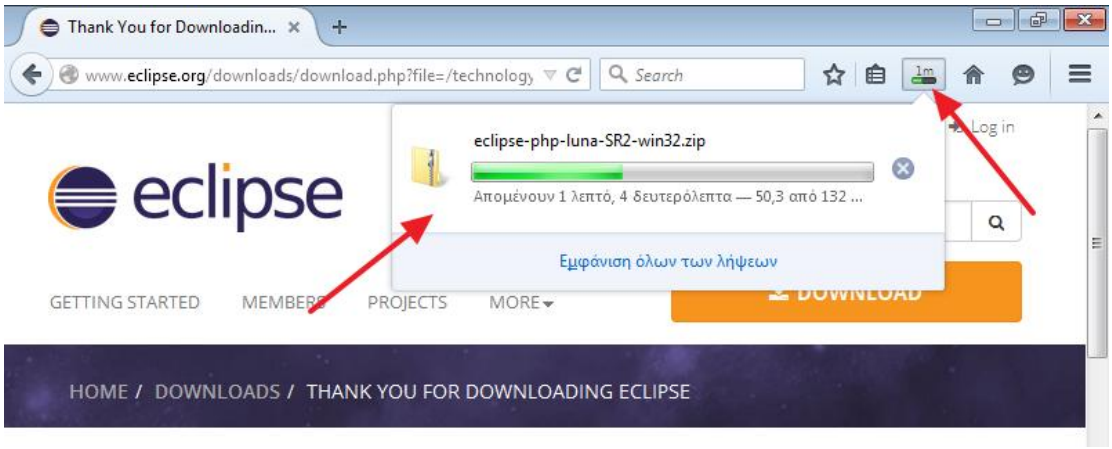
Εικόνα 4.25 Επιλογή εξυπηρετητή για την μεταφόρτωση του αρχείου του eclipse

Ύστερα εμφανίζεται αναδυόμενο παράθυρο όπου επιλέγουμε «Αποθήκευση Αρχείου» και «OK» (Εικόνα 4.26).



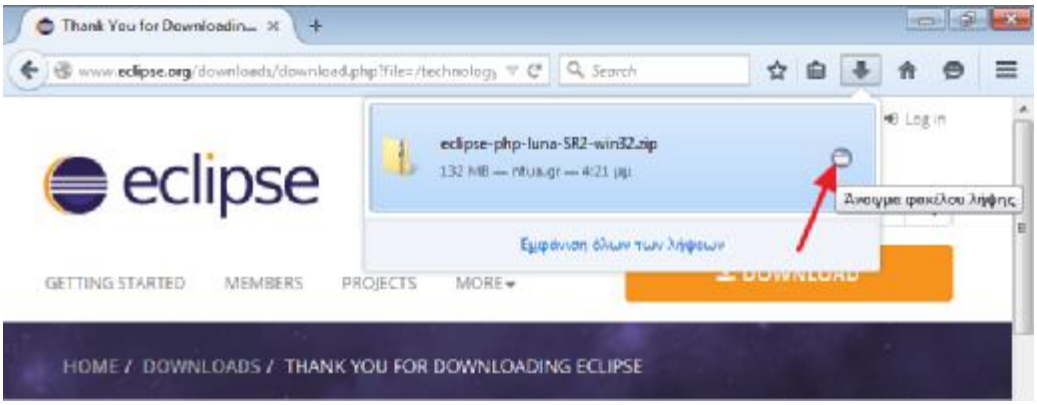
Εικόνα 4.26 Ερώτηση αποθήκευσης του αρχείου του eclipse

Αφού αποδεχθήκαμε την μεταφόρτωση του αρχείου μπορούμε πλέον πάνω δεξιά στον Firefox να δούμε την πρόοδο μεταφόρτωσης (1m = 1minute) (Εικόνα 4.27).



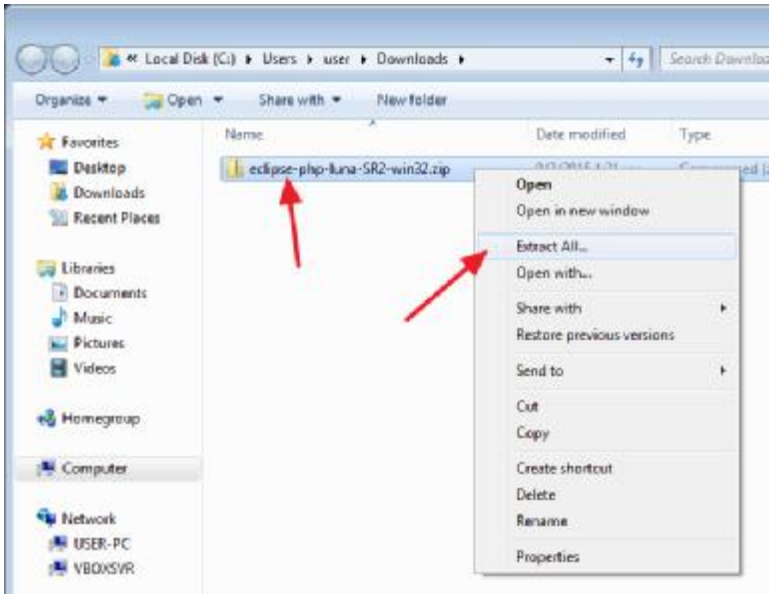
Εικόνα 4.27 Πρόοδος μεταφόρτωσης του firefox για το αρχείο του eclipse

Αφού ολοκληρωθεί η μεταφόρτωση, επιλέγουμε να ανοίξουμε τον φάκελο στον οποίο έχει κατέβει το αρχείο μας (Εικόνα 4.28).



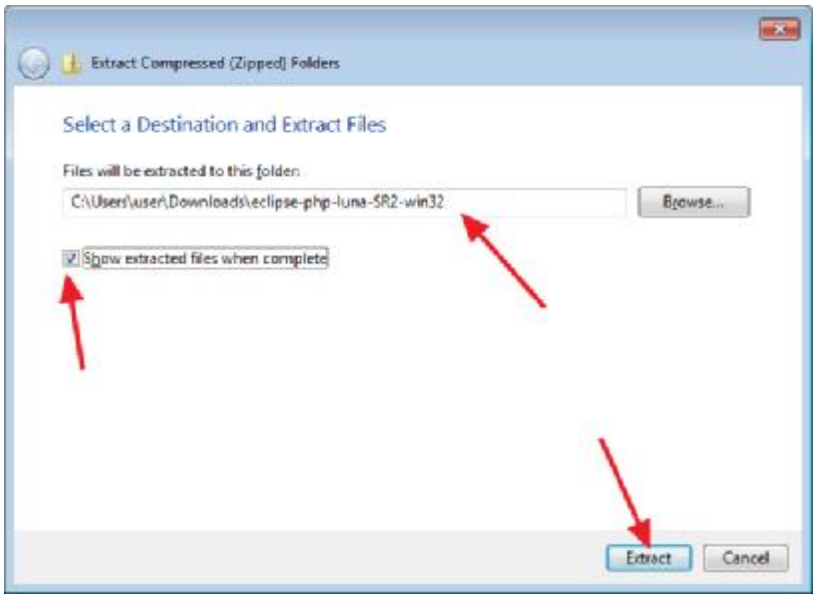
Εικόνα 4.28 Άνοιγμα φακέλου που περιέχει το αρχείο του eclipse

Μεταφερόμαστε στον φάκελο από όπου κάνουμε δεξί κλικ πάνω στο αρχείο που μεταφορτώσαμε, και ύστερα επιλέγουμε «Extract All» (Εξαγωγή όλων) (Εικόνα 4.29).



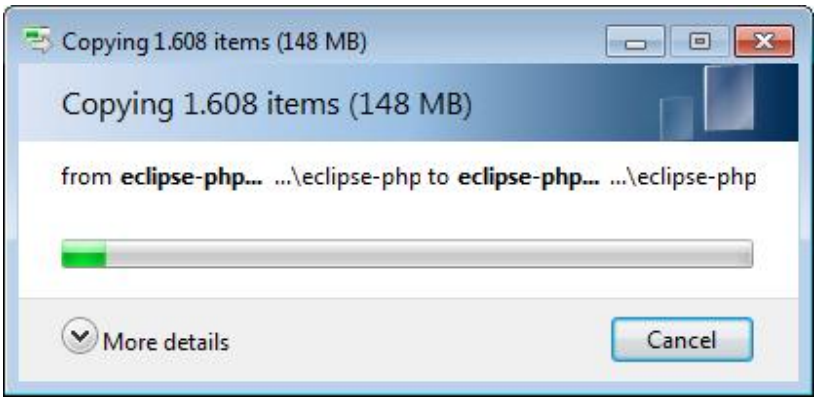
Εικόνα 4.29 Διαδικασία αποσυμπίεσης του ληφθέντος αρχείου του eclipse

Ανοίγει σχετικός οδηγός εξαγωγής αρχείων από συμπιεσμένα αρχεία. Εκεί διαλέγουμε που θέλουμε να εξαχθούν τα αρχεία, εμείς το αφήσαμε ως έχει, και ύστερα «Extract» (Εικόνα 4.30).



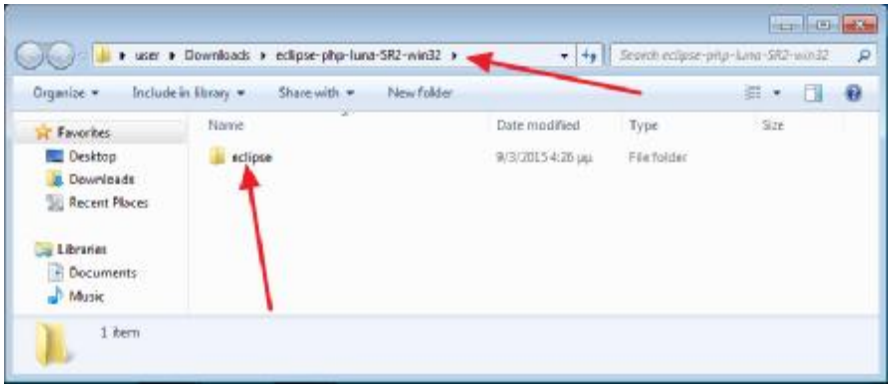
Εικόνα 4.30 Ερώτηση τοποθεσίας εξαγωγής των αρχείων του eclipse

Μέσω αναδυόμενου παραθύρου πληροφορούμαστε για την πρόοδο της εξαγωγής (Εικόνα 4.31).



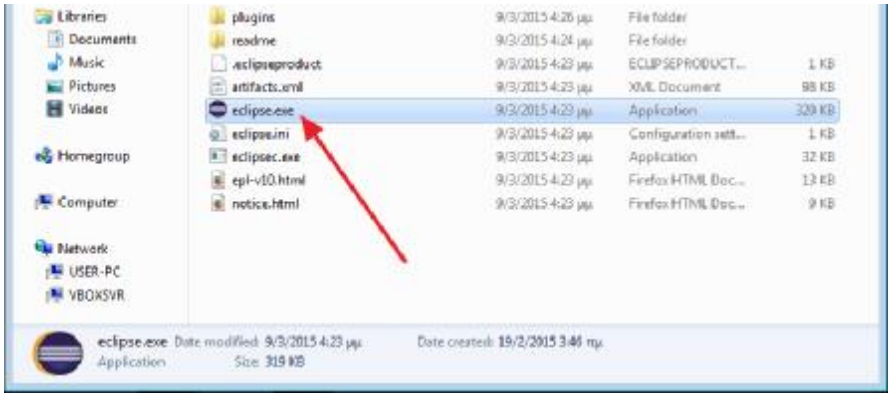
Εικόνα 4.31 Διαδικασία εξαγωγής των αρχείων του eclipse

Όταν ολοκληρωθεί η εξαγωγή, ανοίγει νέο παράθυρο με περιεχόμενα τα αρχεία που προέκυψαν. Κάνουμε διπλό αριστερό κλικ στον φάκελο με όνομα «Eclipse» (Εικόνα 4.32).



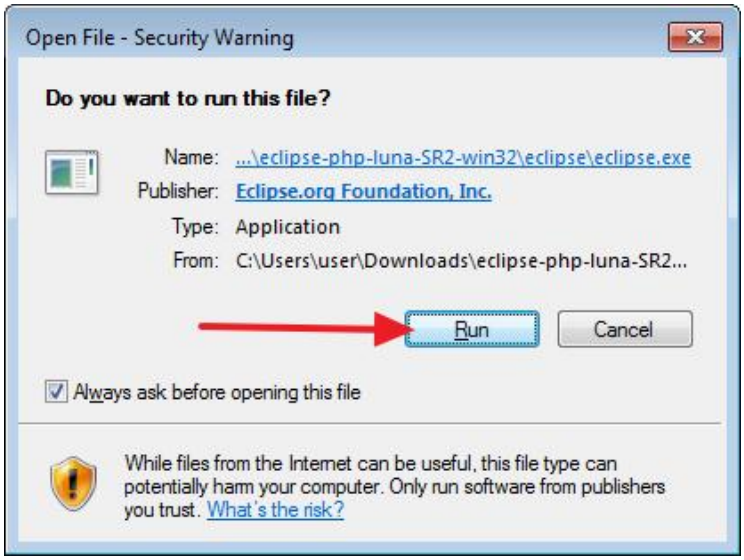
Εικόνα 4.32 Το παραγόμενο αποτέλεσμα της εξαγωγής των αρχείων του eclipse

Μέσα σε αυτόν τον φάκελο βλέπουμε τα αρχεία του προγράμματος. Κάνουμε διπλό αριστερό κλικ στο εκτελέσιμο αρχείο «Eclipse.exe» (Εικόνα 4.33).



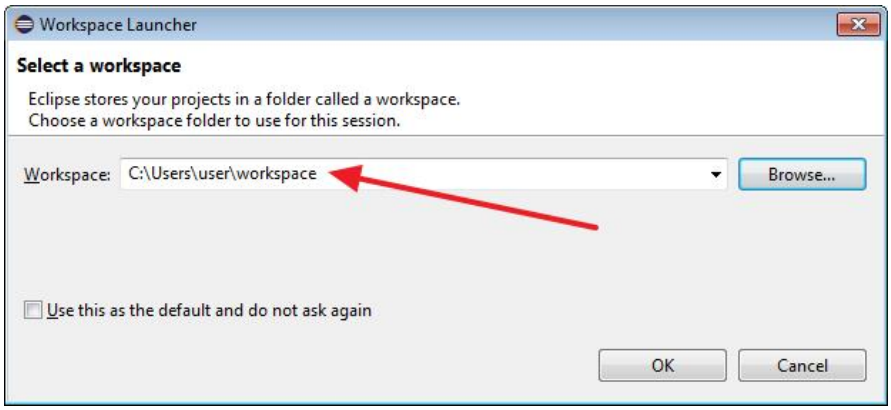
Εικόνα 4.33 Το εκτελέσιμο αρχείο του eclipse ανάμεσα σε όλα τα αλλα αρχεία που διαθέτει

Ένα αναδυόμενο παράθυρο ασφάλειας θα μας ρωτήσει εάν θέλουμε να τρέξουμε το αρχείο δεδομένης της διαδικτυακής του προέλευσης. Αποδεχόμαστε κάνοντας κλικ στο «Run» (Εικόνα 4.34).



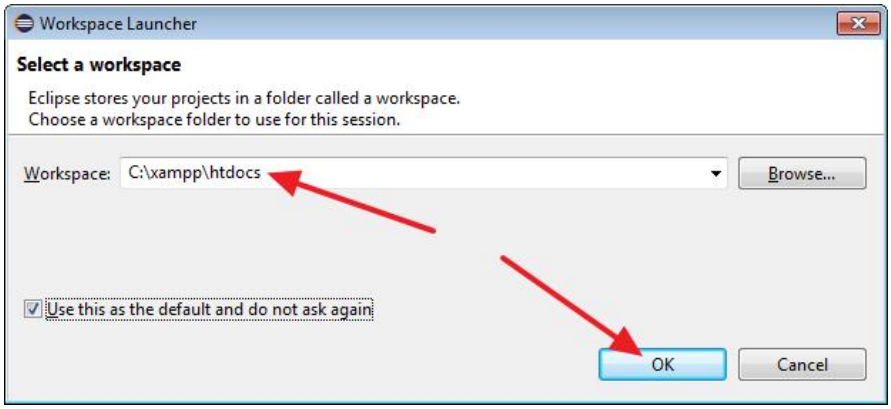
Εικόνα 4.34 Ερώτηση ασφαλείας για την εκτέλεση του eclipse

Το eclipse ξεκινά, με δικό του αναδυόμενο παράθυρο μας ρωτά για την τοποθεσία του workspace (χώρου εργασίας), όπου τα παραγόμενα projects θα αποθηκεύονται (Εικόνα 4.35).



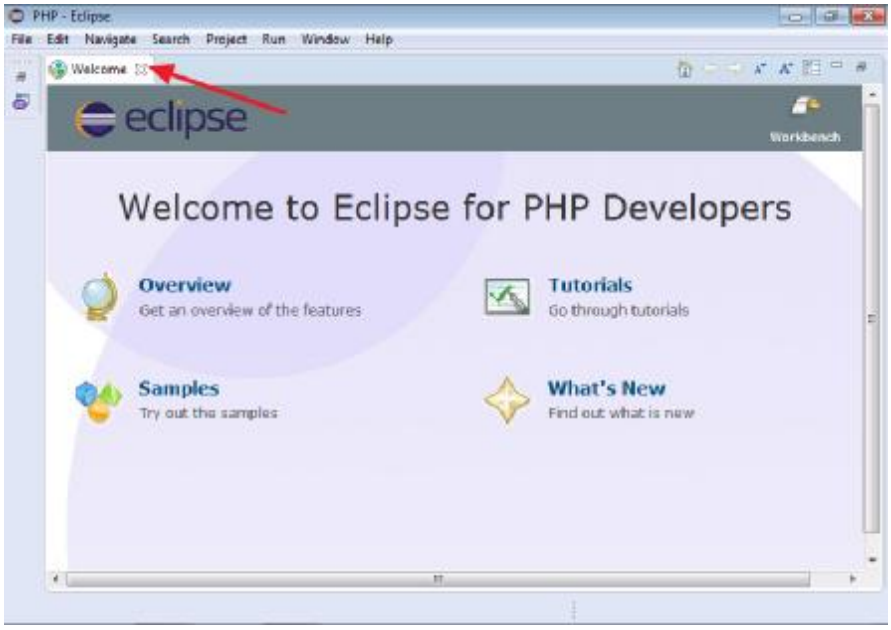
Εικόνα 4.35 Ερώτηση τοποθεσίας του χώρου εργασίας του eclipse

Μέσα στο σχετικό πλαίσιο αλλάζουμε την επιλογή από την προεπιλεγμένη «C:\xampp\htdocs». Επιλέγουμε το κουτάκι «Use this as the default and do not ask again», και ύστερα «OK» (Εικόνα 4.36).



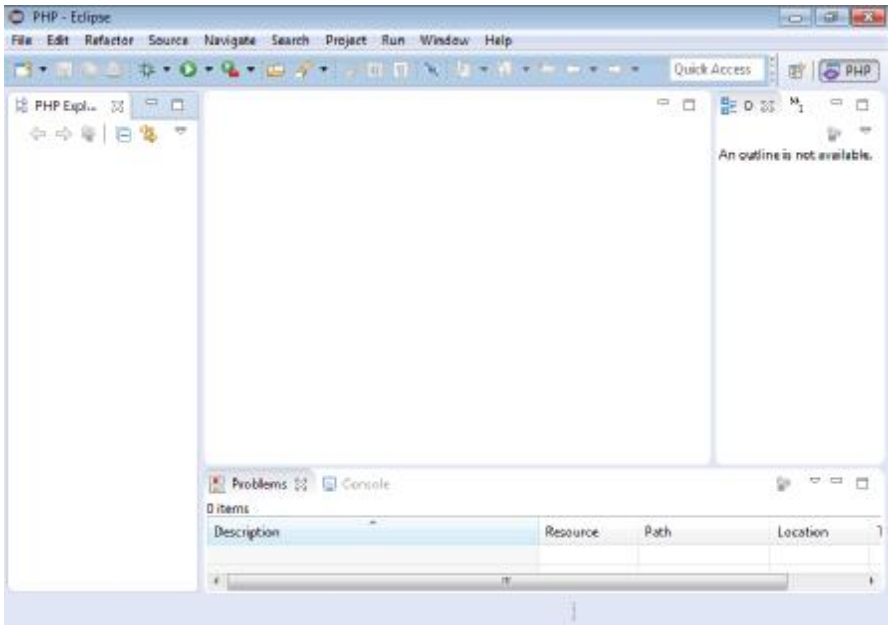
Εικόνα 4.36 Αλλαγή τοποθεσίας του χώρου εργασίας του eclipse

Μας εμφανίζεται η αρχική οθόνη καλωσορίσματος του eclipse. Την κλείνουμε κάνοντας κλικ στο «X» (Εικόνα 4.37).



Εικόνα 4.37 Αρχική οθόνη καλοσορίσματος του eclipse

Η τελική μορφή και χώρος εργασίας του eclipse μας παρουσιάζεται (Εικόνα 4.38).



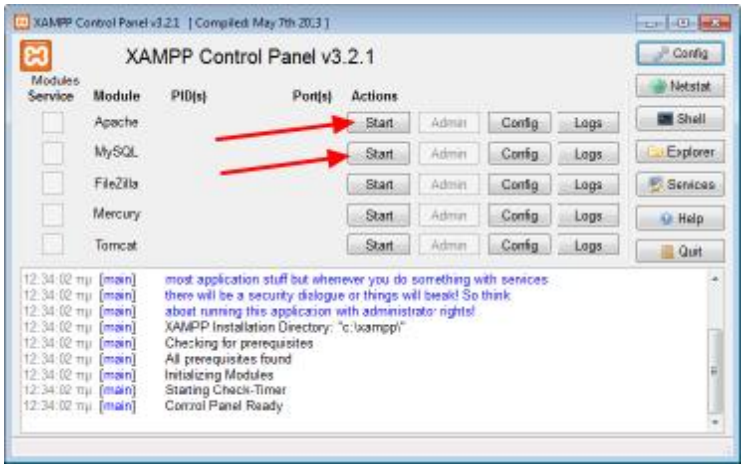
Εικόνα 4.38 Χώρος εργασίας του eclipse

4.4 Ρύθμιση xampp

Σε αυτήν την ενότητα θα δούμε πως θα ρυθμίσουμε κατάλληλα το XAMPP έτσι ώστε να αποτελέσει ένα λειτουργικό εξυπηρετητή Apache και MySQL, όπου θα

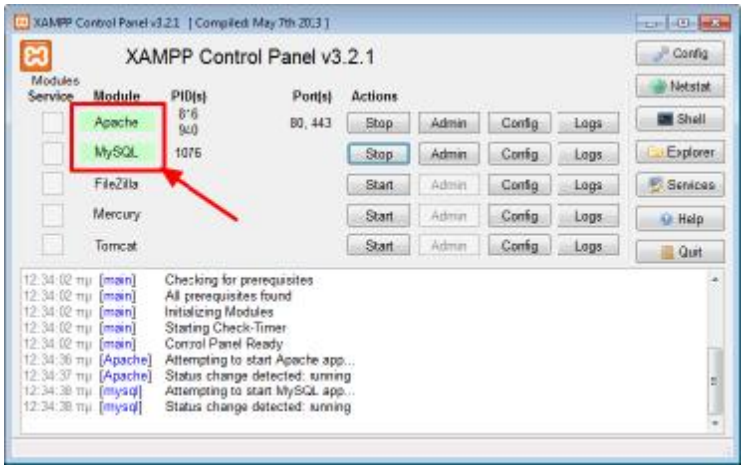
μπορέσουμε να τον χρησιμοποιήσουμε για να αναπτύξουμε την ιστοσελίδα μας τοπικά στον υπολογιστή μας.

Ανοίγοντας το control panel του xampp από την συντόμευση στην επιφάνεια εργασίας βλέπουμε τα στοιχεία που περιλαμβάνει. Αριστερά βρίσκεται μια λίστα με όλες τις υπηρεσίες που εμπεριέχονται, δεξιά βρίσκουμε κουμπιά «Start» και «Config» για κάθε υπηρεσία.



Εικόνα 4.39 Εκκίνηση υπηρεσιών apache και MySQL

Για να μπορέσουμε να δούμε και να διαχειριστούμε την ιστοσελίδα μας θα πρέπει να οι δυο υπηρεσίες «Apache» και «MySQL» να τρέχουν κάνοντας κλικ στο «Start» και επιβεβαιώνοντας πως οι ετικέτα της κάθε υπηρεσίας έχει πράσινο φόντο.



Εικόνα 4.40 Κατάσταση εκκίνησης των υπηρεσιών apache και MySQL

Υπάρχουν πολλές ρυθμίσεις που μπορούμε να κάνουμε μέσα από το διαχειριστικό πάνελ του XAMPP αλλά εμείς θα επικεντρωθούμε μόνο στις απαραίτητες για την ιστοσελίδα μας. Έχοντας ξεκινήσει και τις δυο υπηρεσίες εισάγουμε ως διεύθυνση στον firefox την «http://localhost/» και μας φορτώνεται η αρχική σελίδα του εξυπηρετητή μας. Από εκεί στην αριστερή πλευρά της σελίδας κάνουμε κλικ στην επιλογή «Security».

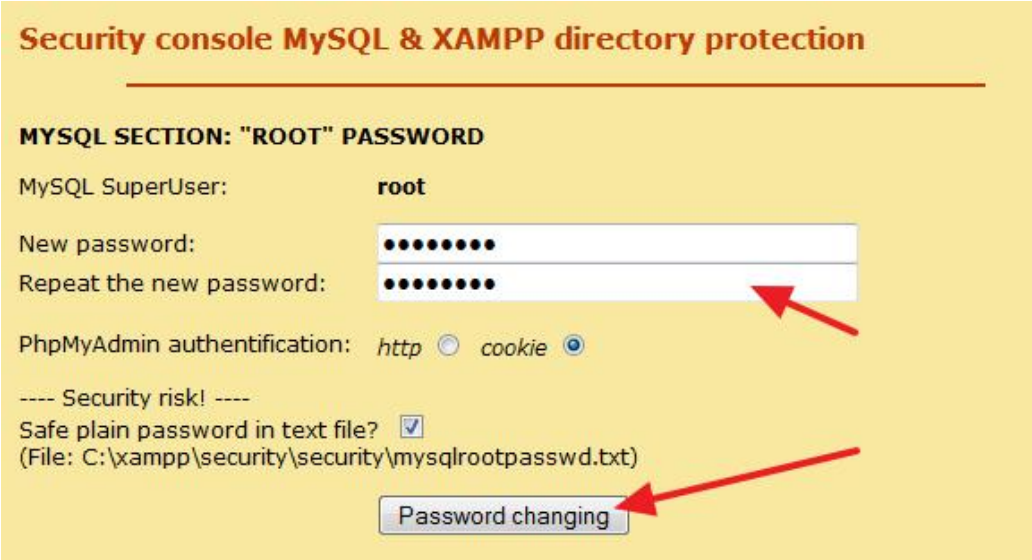
Εικόνα 4.41 Αρχική οθόνη εξυπηρετητή XAMPP

Από την νέα οθόνη που μας φορτώνεται βλέπουμε την κατάσταση ασφαλείας του XAMPP σε διάφορα θέματα που αφορούν τους εξυπηρετητές που έχουμε ενεργούς αυτή την στιγμή (apache, MySQL). Κάτω από τον πίνακα κατάστασης υπάρχει ένας σύνδεσμος για αλλαγή ρυθμίσεων ασφαλείας, κάνουμε κλικ επάνω του.

Εικόνα 4.42 Κατάσταση ασφαλείας εξυπηρετητών XAMPP

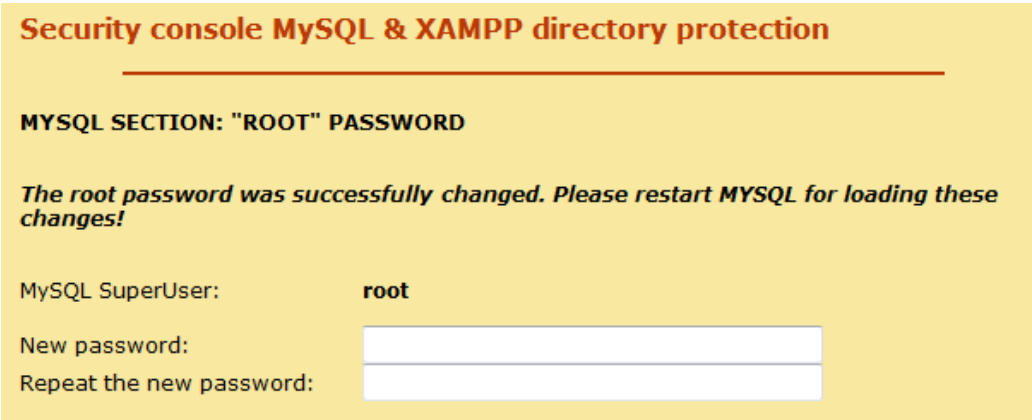
Υπάρχουν αρκετές επιλογές στην νέα σελίδα που εμφανίζεται. Εμείς αλλάξαμε μόνο τον κωδικό του χρήστη «root» για την MySQL. Εισάγαμε κωδικό στα πεδία με ετικέτα

«New password» και «Repeat the new password» και ύστερα κάναμε κλικ στο κουμπί «Password changing».



Εικόνα 4.43 Αλλαγή κωδικού root για την MySQL

Αμέσως λαμβάνουμε το μήνυμα επιβεβαίωσης για την αλλαγή του κωδικού, στην ίδια σελίδα στην οποία βρισκόμαστε. Οι ρυθμίσεις ασφαλείας μας έχουν τελειώσει.



Εικόνα 4.44 Επιβεβαίωση αλλαγής κωδικού για τον χρήστη root

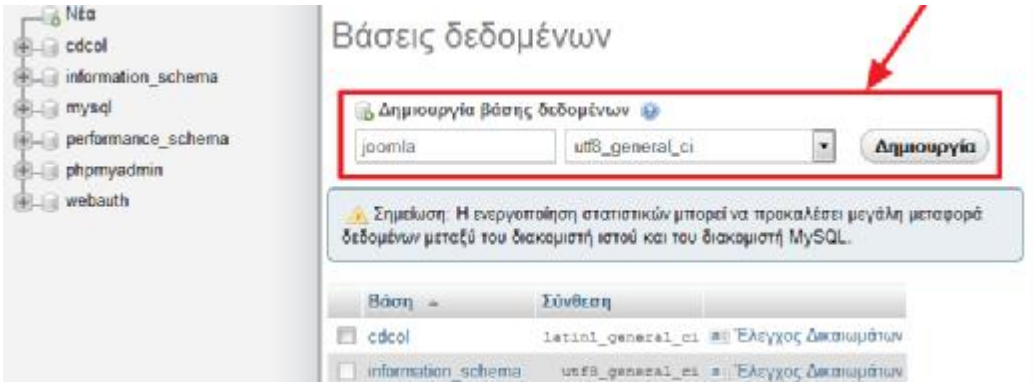
4.5 Δημιουργία βάσης δεδομένων

Πριν εγκαταστήσουμε το joomla πρέπει πρώτα να έχουμε δημιουργήσει μια βάση δεδομένων στην οποία θα εισάγονται τα νέα άρθρα, οι κατηγορίες και γενικά τα δεδομένα του ιστότοπου.

Για να δημιουργήσουμε μια βάση δεδομένων στην mysql του XAMPP θα πρέπει να μεταβούμε στο σύστημα διαχείρισης βάσεων δεδομένων που διαθέτει το XAMPP, το phpmyadmin. Αυτό γίνεται εφικτό αν επισκεφθούμε την διεύθυνση «http://localhost/phpmyadmin». Εισάγουμε αυτή την διεύθυνση στον firefox και μας εμφανίζεται η παρακάτω οθόνη (Εικόνα 4.45).

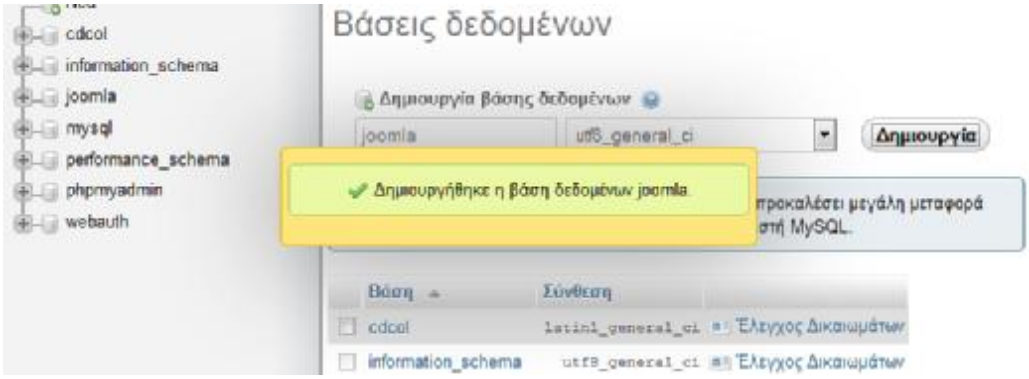
Εικόνα 4.45 Αρχική οθόνη phpmyadmin

Πιέζοντας το κουμπί «Βάσεις δεδομένων» στην κορυφή της οθόνης, μας εμφανίζεται η οθόνη δημιουργίας νέας βάσης δεδομένων και προβολής των υπαρχόντων βάσεων δεδομένων. Εισάγουμε στο πεδίο «Όνομα βάσης δεδομένων» το όνομα που επιθυμούμε να έχει η βάση δεδομένων που θα δημιουργήσουμε, στην δικιά μας περίπτωση «joomla» και από το διπλανό πεδίο ορίζουμε την κωδικοποίηση της βάσης σε «utf8_general_ci» για μεγαλύτερη συμβατότητα με όλες τις συμβολοσειρές και τις διαθέσιμες γλώσσες του πλανήτη (Εικόνα 4.46).



Εικόνα 4.46 Δημιουργία νέας βάσης δεδομένων

Ύστερα κάνοντας κλικ στο κουμπί «Δημιουργία» μας εμφανίζεται και το αντίστοιχο μήνυμα επιβεβαίωσης της δημιουργίας τις βάσης που επιθυμούμε (Εικόνα 4.47):



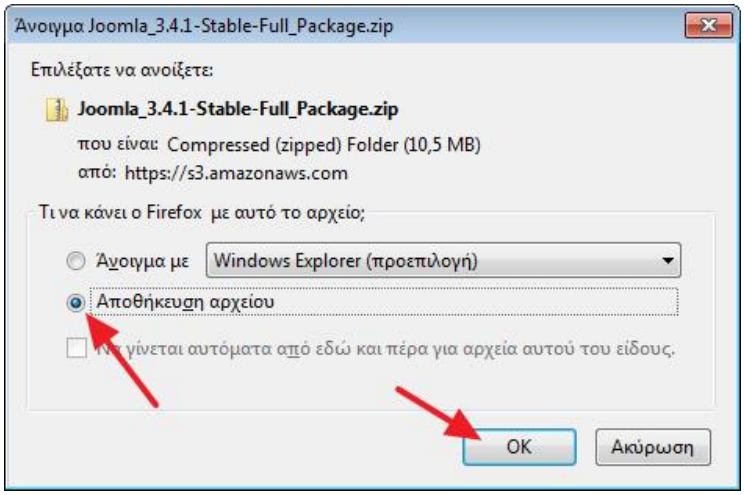
Εικόνα 4.47 Επιβεβαίωση δημιουργίας βάσης δεδομένων

4.6 Εγκατάσταση Joomla

Η εγκατάσταση του joomla δεν είναι δύσκολη υπόθεση, είναι όμως απαραίτητη για την ολοκλήρωση της πτυχιακής εργασίας. Προκειμένου να λειτουργήσει η παρακάτω διαδικασία θα πρέπει ο Apache server και ο Mysql να είναι ενεργοποιημένοι και να έχουμε δημιουργήσει ήδη μια βάση δεδομένων στην οποία θα εγκατασταθεί το Joomla. Για να εγκαταστήσουμε το Joomla θα πρέπει να εισάγουμε στον φυλλομετρητή μας την διεύθυνση «http://www.joomla.gr/download.html» από όπου μπορούμε να κατεβάσουμε τα αρχεία εγκατάστασης του. Στην σελίδα που μας φορτώνεται κάνουμε κλικ στο πράσινο κουμπάκι με τίτλο «Download Joomla! 3.4.1» (Εικόνα 4.48).

Εικόνα 4.48 Σελίδα μεταφόρτωσης joomla

Αμέσως μας εμφανίζεται ερώτηση αποθήκευσης του μεταφορτώμενου αρχείου, επιλέγουμε «Αποθήκευση αρχείου» και «ΟΚ» (Εικόνα 4.49).

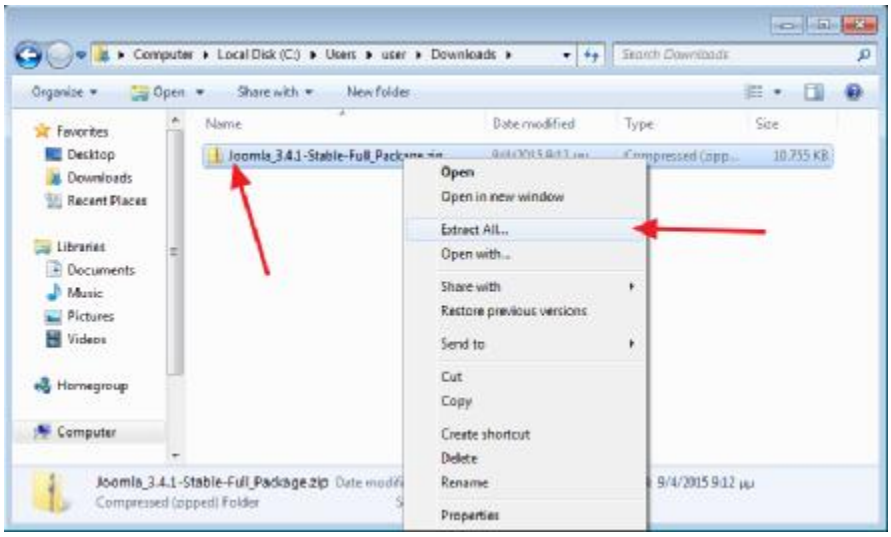


Εικόνα 4.49 Ερώτηση αποθήκευσης αρχείου joomla

Παρατηρούμε την πρόοδο στην διαδικασία της μεταφόρτωσης πάνω δεξιά στον Mozilla Firefox, μόλις ολοκληρωθεί η διαδικασία κάνουμε κλικ στο στρογγυλό εικονίδιο σε σχήμα φακέλου δίπλα από το όνομα αρχείου (Εικόνα 4.50).

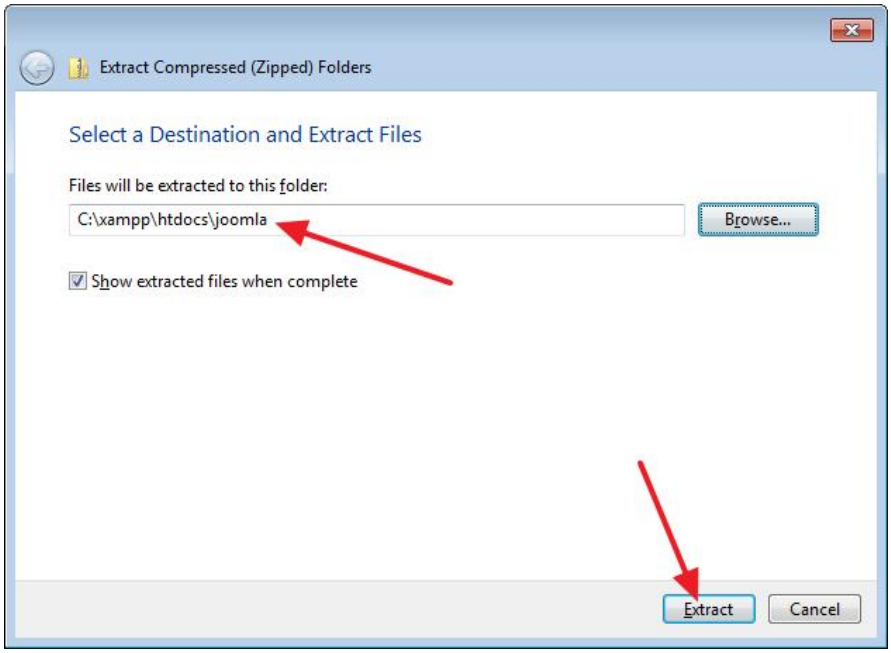
Εικόνα 4.50 Άνοιγμα προβολής των μεταφορτωμένω αρχείων του joomla

Ανοίγεται ένα νέο παράθυρο όπου βλέπουμε το συμπιεσμένο αρχείο το οποίο κατεβάσαμε. Κάνοντας δεξί κλικ επάνω του επιλέγουμε «Extract All…» (ή Αποσυμπίεση όλων) (Εικόνα 4.51).



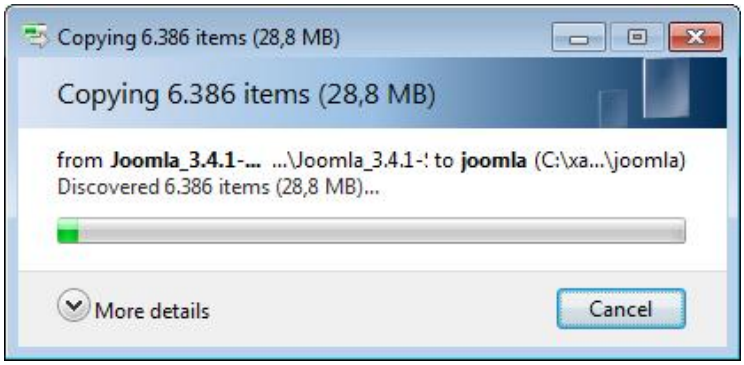
Εικόνα 4.51 Αποσυμπίεση αρχείων joomla

Μας δύναται η δυνατότητα να αποφασίσουμε εμείς που θέλουμε να αποσυμπιέσουμε τα αρχεία μας, επομένως διαλέγουμε κάποιον κατάλογο μέσα στους καταλόγους του XAMPP, και πιο συγκεκριμένα κάποιος υποκατάλογος στον κατάλογο «C:\xampp\htdocs\» εμείς επιλέξαμε να ονομάσουμε τον νέο κατάλογο «joomla». Ο υποκατάλογος αυτός όπου εξάγαμε τα αρχεία μας είναι πολύ σημαντικός για το XAMPP, διότι σε αυτόν κατάλογο γίνεται η κατεύθυνση όλων των αιτημάτων HTTP που θα αναλάβει ο server μας να κατευθύνει (Εικόνα 4.52).



Εικόνα 4.52 Επιλογή σημείου αποσυμπίεσης των αρχείων του joomla

Μόλις κάνουμε κλικ στο «OK» μας εμφανίζεται η πρόοδος της αποσυμπίεσης. Περιμένουμε ώστε να ολοκληρωθεί (Εικόνα 4.53).



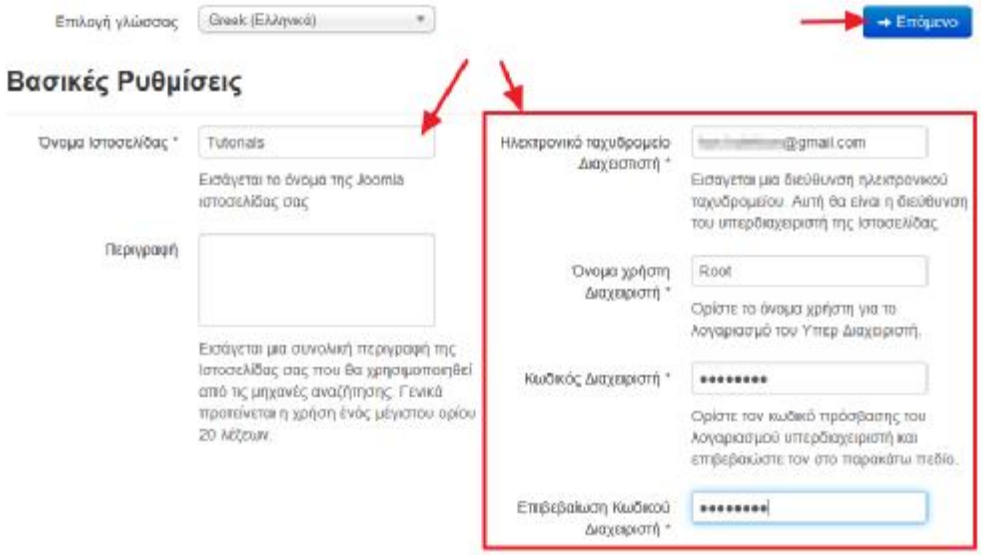
Εικόνα 4.53 Πρόοδος αποσυμπίεσης

Αφού ολοκληρωθεί, δηλαδή το παράθυρο της διαδικασίας αποσυμπίεσης κλείσει αυτόματα, τότε εισάγουμε την διεύθυνση «http://localhost/joomla/» στον φυλλομετρητή μας, και μας εμφανίζεται ο οδηγός εγκατάστασης του joomla. Εισάγουμε στα πεδία με αστερίσκο τα απαραίτητα στοιχεία όπως:

- Το όνομα της ιστοσελίδας,
- Το ηλεκτρονικό ταχυδρομείο του διαχειριστή,

- Το όνομα χρήστη του διαχειριστή,
- Τον κωδικό του διαχειριστή,
- Και την επιβεβαίωση του κωδικού του διαχειριστή.

Ύστερα προαιρετικά αν θέλουμε μπορούμε να γράψουμε μια σύντομη περιγραφή του ιστότοπου συμπεριλαμβάνοντας τις κατάλληλες λέξεις-κλειδιά που θα χρησιμοποιηθούν στις μηχανές αναζήτησης. Ύστερα κάνουμε κλικ στο κουμπί «Επόμενο» (Εικόνα 4.54).



Εικόνα 4.54 Εισαγωγή βασικών στοιχείων οδηγού εγκατάστασης joomla

Η επόμενη οθόνη μας ζητά τα στοιχεία για την ρύθμιση της βάσης δεδομένων μας.

1. Εμείς επειδή χρησιμοποιούμε XAMPP η βάση δεδομένων μας είναι MySQL , οπότε επιλέγουμε ως είδος βάσης δεδομένων MySQLi αφού υποστηρίζεται από τον εξυπηρετητή μας (συνήθως αυτή η επιλογή γίνεται αυτόματα).
2. Ύστερα ως όνομα διακομιστή χρησιμοποιούμε «localhost» αφού ο εξυπηρετητής μας είναι τοπικά στον υπολογιστή μας,
3. Όνομα χρήστη της βάσης δεδομένων «root» ο οποίος είναι και ο προεπιλεγμένος, διαφορετικά εισάγετε ως όνομα χρήστη αυτό που εισάγετε και στο phpmyadmin κατά την είσοδο σας. Ύστερα κωδικός του χρήστη της βάσης

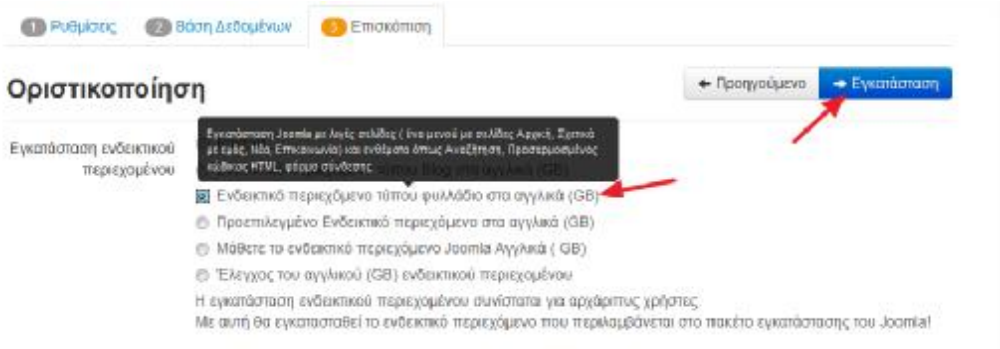
δεδομένων. Ο προεπιλεγμένος είναι κενός, εκτός αν τον αλλάξαμε στην ρύθμιση του XAMPP σε προηγούμενη ενότητα, τότε εισάγουμε αυτόν..

4. Τέλος το όνομα της βάσης δεδομένων στην οποία θα εγκατασταθεί το joomla. Σε προηγούμενη ενότητα δημιουργήσαμε μια βάση δεδομένων με ένα συγκεκριμένο όνομα το όνομα αυτό μπαίνει εδώ. Στην περίπτωση μας «joomla».

Ύστερα κάνουμε κλικ στο κουμπί «Επόμενο» (Εικόνα 4.55).

Εικόνα 4.55 Εισαγωγή στοιχείων βάσης δεδομένων οδηγού εγκατάστασης joomla

Στην επόμενη οθόνη ενημερωνόμαστε για τις ήδη υπάρχουσες επιλογές που κάναμε κατά την διάρκεια της εγκατάστασης και μας γίνεται ερώτηση σχετικά με το τι είδους περιεχόμενο να εισαχθεί στον ιστότοπο προκειμένου να μην είναι κενός στην πρώτη επίσκεψη του. Εμείς επιλέξαμε «ενδεικτικό περιεχόμενο τύπου φυλλάδιο στα αγγλικά (GB)» και ύστερα κάναμε κλικ στο κουμπί «Εγκατάσταση» (Εικόνα 4.56).



Εικόνα 4.56 Επιλογή εισαγωγής ενδεικτικού περιεχομένου του οδηγού εγκατάστασης joomla

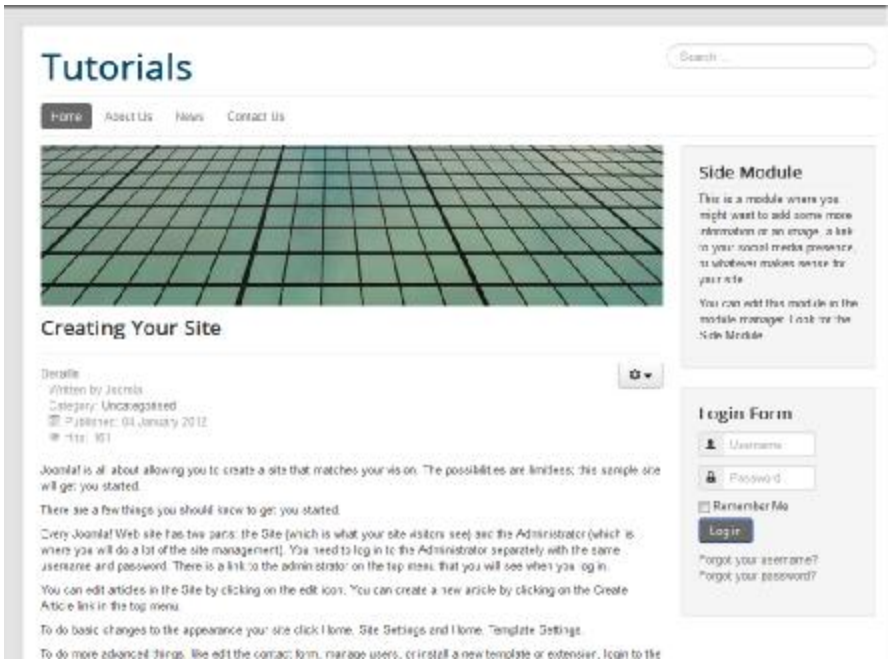
Στην επόμενη οθόνη βλέπουμε την πρόοδο εγκατάστασης το joomla. Μόλις ολοκληρωθεί μας εμφανίζεται το σχετικό μήνυμα επιβεβαίωσης με την ένδειξη «ΠΑΡΑΚΑΛΩ ΘΥΜΗΘΕΙΤΕ ΝΑ ΔΙΑΓΡΑΨΕΤΕ ΤΟΝ ΦΑΚΕΛΟ INSTALLATION», κάνοντας κλικ στο κουμπί «Αφαίρεση φακέλου installation» επιλύουμε αυτό το πρόβλημα (Εικόνα 4.57).

Εικόνα 4.57 Διαγραφή φακέλου installation του οδηγού εγκατάστασης

Στην επώμενη οθόνη βλέπουμε πως το κουμπί έχει αλλάξει ονομα σε «Ο φάκελος installation έχει αφαιρεθεί επιτυχώς» (Εικόνα 4.58).

Εικόνα 4.58 Επιβεβαιώση διαγραφής

Ύστερα κάνουμε κλίκ στο κουμπί «Ιστοσελίδα» ώστε να δούμε τον νέο μας ιστότοπο δημιουργημένο από joomla! (Εικόνα 4.59).



Εικόνα 4.59 Η έτοιμη ιστοσελίδα

5 ΔΗΜΙΟΥΡΓΙΑ ΚΑΙ ΑΝΑΠΤΥΞΗ ΙΣΤΟΤΟΠΟΥ

Σε αυτό το κεφάλαιο θα ασχοληθούμε με την δημιουργία και την ανάπτυξη του ιστότοπου. Θα προχωρήσουμε σε ειδικές παραμετροποιήσεις και θα παρουσιάσουμε

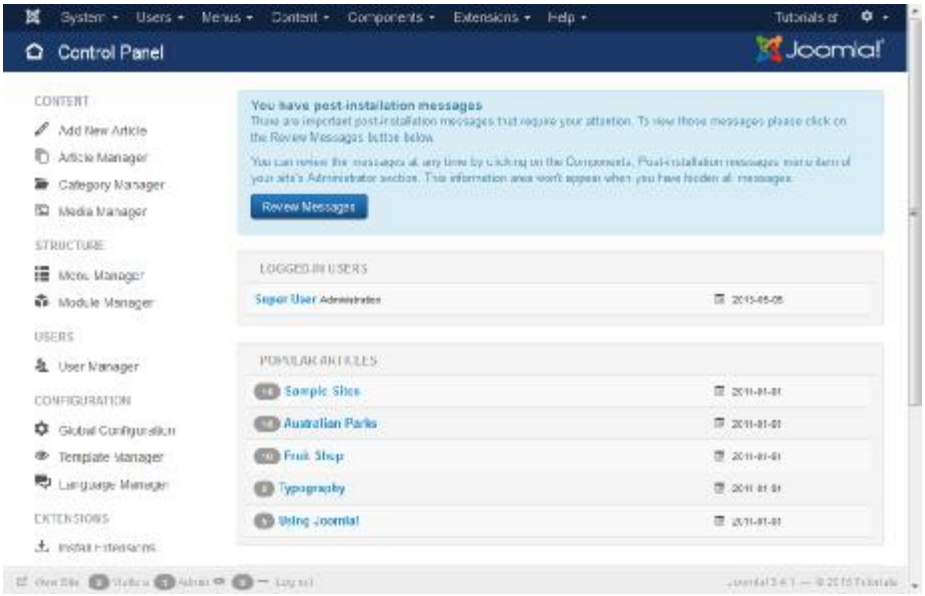
τον τρόπο με τον οποίο δημιουργήθηκαν όλα τα μενού, τα άρθρα και άλλα πράγματα που αποτελούν τα βασικά χαρακτηριστικά του.

5.1 Διαχείρηση του ιστότοπου

Έχοντας πλέον ολοκληρώσει την εγκατάσταση του Joomla μπορούμε να προχωρήσουμε στην δημιουργία της ιστοσελίδας μας. Μέσω του φυλλομετρητή μας επισκεπτόμαστε την διεύθυνση διαχείρισης της τοπικής μας ιστοσελίδας «http://localhost/joomla/administrator», αμέσως μας εμφανίζεται οθόνη όπου ζητούνται τα στοιχεία εισόδου, όνομα χρήστη και κωδικός πρόσβασης προκειμένου να εισαχθούμε στην διαχείριση (Εικόνα 5.1).

Εικόνα 5.1 Οθόνη εισαγωγής στο σύστημα διαχείρισης

Αφού εισάγουμε σωστά τα στοιχεία, μας εμφανίζεται η αρχική οθόνη διαχείρισης της ιστοσελίδας. Στην αρχική οθόνη βρίσκονται τα πιο πρόσφατα άρθρα τα οποία έχουν δημιουργηθεί, ένα βασικό μενού επιλογών με εικονίδια και ένα μενού στην κορυφή της σελίδας με όλες τις διαχειριστικές επιλογές τους συστήματος (Εικόνα 5.2).



Εικόνα 5.2 Αρχική οθόνη συστήματος διαχείρισης

5.2 Επεκτάσεις

Το Joomla αποτελείται από πολλά διαφορετικά κομμάτια, τα οποία συνεργάζονται μεταξύ τους προκειμένου να προσφέρουν το αποτέλεσμα της ιστοσελίδας. Γι' αυτό τον λόγο αυτά τα κομμάτια είναι όσο το δυνατόν πιο ευέλικτα, έτσι ώστε οι διάφορες επεκτάσεις και ενσωματώσεις να γίνονται εύκολα.

Μερικά παραδείγματα επεκτάσεων είναι τα πρόσθετα (γνωστά και ως «Mambots»). Τα πρόσθετα είναι επεκτάσεις του υπόβαθρου του joomla με τα οποία επεκτείνονται οι λειτουργικές δυνατότητες του συστήματος. Το HS Highlighter για παράδειγμα όπου επιτρέπει την εισαγωγή παραθέσεων κώδικα σε οποιαδήποτε γλώσσα προγραμματισμού, όταν φορτώνεται η ιστοσελίδα που περιέχει αυτή την παράθεση μορφοποιεί τον κώδικα δυναμικά έτσι ώστε να είναι πιο εύκολα αναγνώσιμος, άλλη μια επέκταση σε back-end μορφή είναι ο Ark Editor, όπου αντικαθίστανται ο ήδη υπάρχων επεξεργαστής άρθρων του joomla προσθέτοντας πληθώρα νέων δυνατοτήτων, όπως άμεση μορφοποίηση με βάση το πρότυπο που χρησιμοποιείται κι άλλα πολλά.

Πέρα από τα πρόσθετα υπάρχουν και πιο πολύπλοκες επεκτάσεις τα λεγόμενα «components». Οι επεκτάσεις επιτρέπουν την εκτέλεση εργασιών όπως δημιουργία

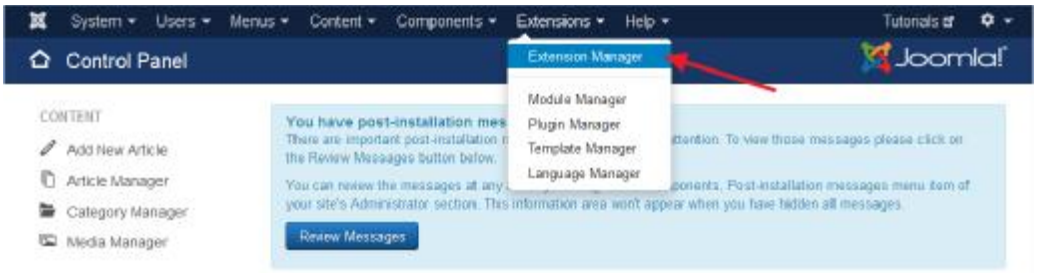
κοινοτήτων, δημιουργία αντιγράφων ασφαλείας του ιστοχώρου ή μετάφραση περιεχομένου και δημιουργία διευθύνσεων URL που είναι πιο φιλικές τόσο στον χρήστη όσο και στις μηχανές αναζήτησης.

Τέλος υπάρχουν και τα ενθέματα τα οποία εκτελούν συνήθως εργασίες εμφάνισης ενός πρόσθετου στην ιστοσελίδα, όπως πχ η εμφάνιση ενός ημερολογίου ή η εισαγωγή κώδικα html μέσα στην ιστοσελίδα. Τα ενθέματα διαθέτουν αποκλειστική θέση μέσα στον κώδικα μιας ιστοσελίδας, τον οποίο αποφασίζουμε εμείς να χρησιμοποιήσουμε μέσα από το διαχειριστικό σύστημα που μας παρέχεται.

5.2.1 Γλώσσες

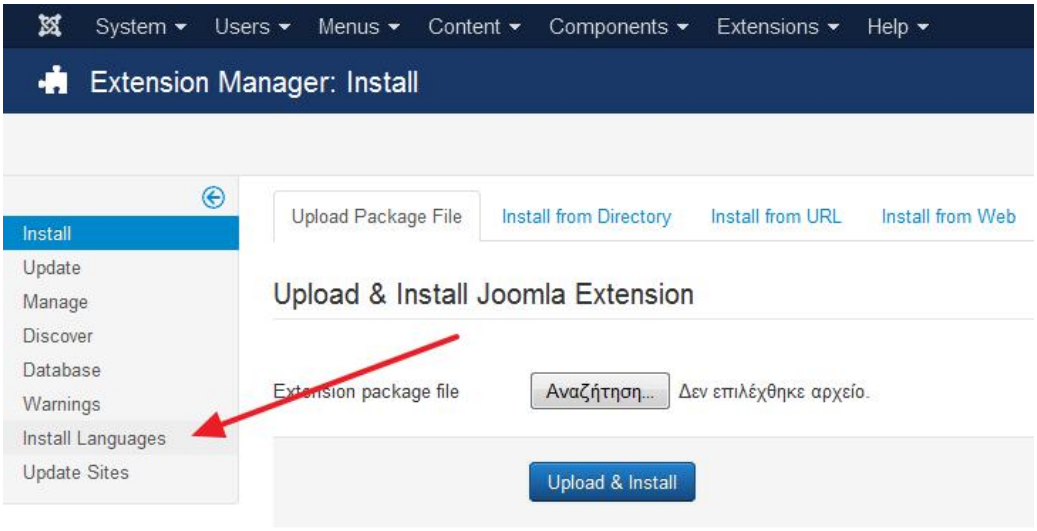
Μέσα από το διαχειριστικό σύστημα μας δύναται η δυνατότητα της εγκατάστασης νέων γλωσσών, την επιλογή προεπιλεγμένης μέσα από τις ήδη υπάρχουσες, καθώς και την διαχείριση αυτών.

Κάνοντας είσοδο στην διαχείριση της ιστοσελίδας μας διαλέγουμε από το μενού «Extensions» την επιλογή «Extension Manager» (Εικόνα 5.3).



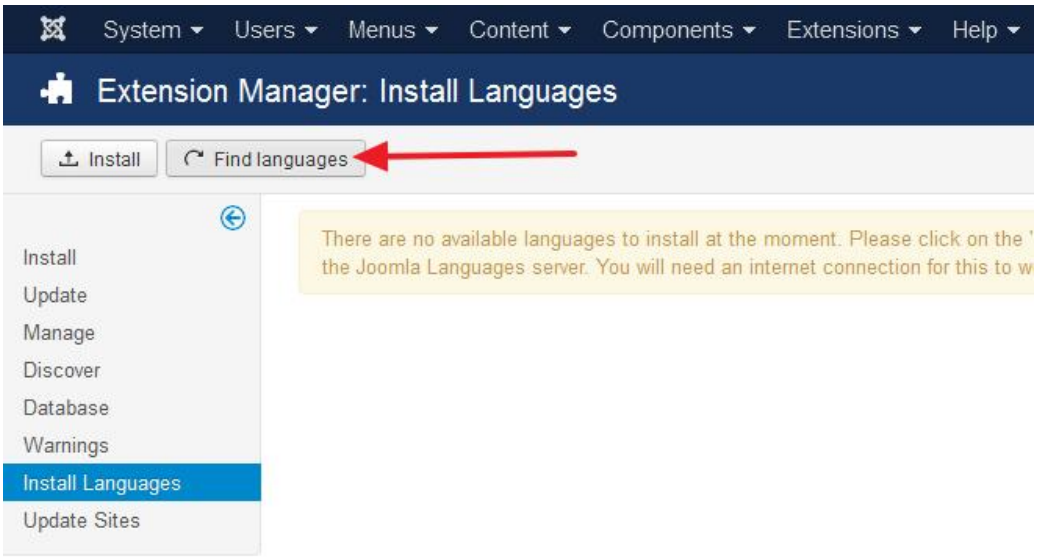
Εικόνα 5.3 Επιλογή μενού επεκτάσεων

Μας φορτώνεται η βασική οθόνη των επεκτάσεων του συστήματος μας, όπου μπορούμε να εγκαταστήσουμε μια νέα γλώσσα. Από το αριστερό μενού διαλέγουμε την επιλογή «Install Languages» (Εικόνα 5.4).



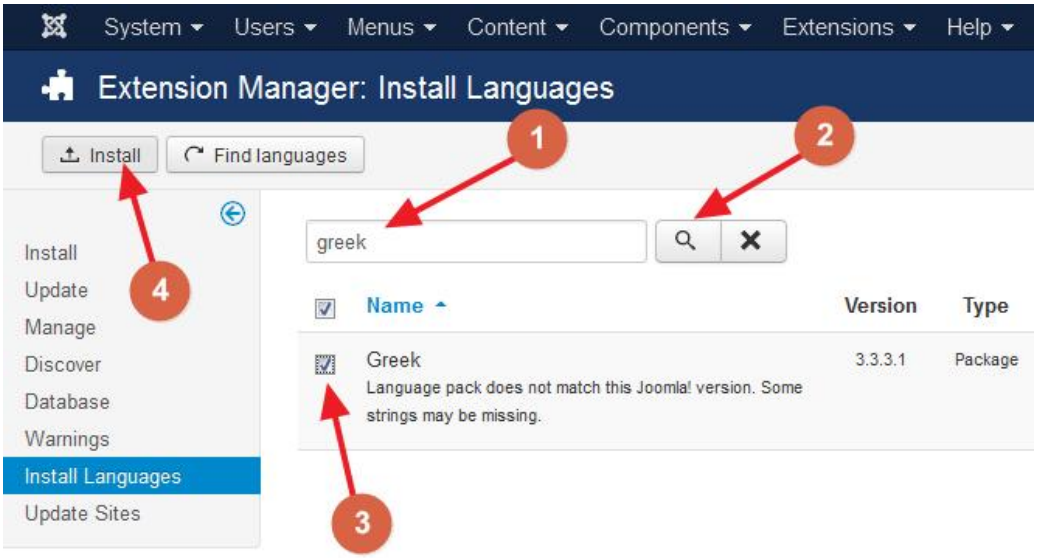
Εικόνα 5.4 Επιλογή μενού εγκατάσταση γλώσσας

Στην νέα οθόνη που προκύπτει υπάρχει ενημερωτικό μήνυμα σχετικά με τις γλώσσες που δεν υπάρχουν τοπικά στο joomla μας. Κάνουμε κλικ στο κουμπί «Find Languages» (Εικόνα 5.5).



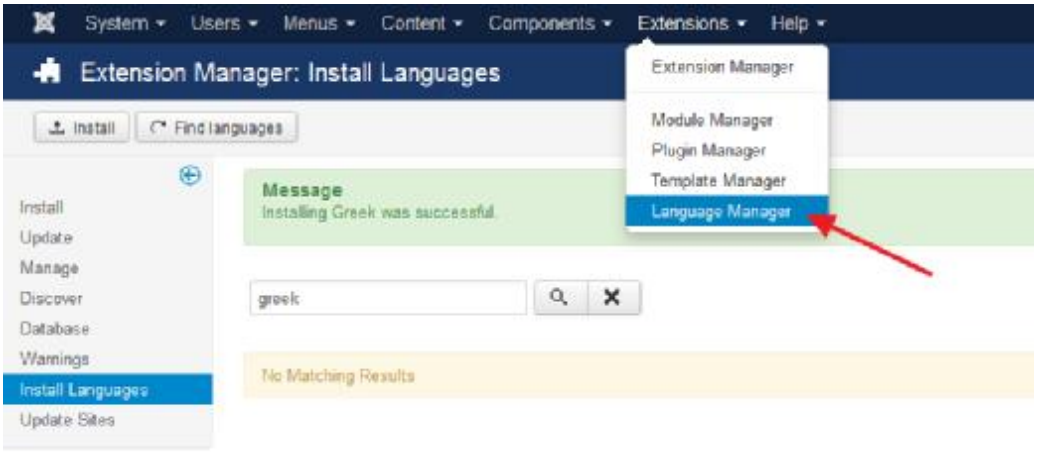
Εικόνα 5.5 Αναζήτηση γλώσσας για εγκατάσταση

Εισάγουμε στο πεδίο κειμένου το αλφαριθμητικό «greek» και κάνουμε κλικ στον μεγεθυντικό φακό για αναζήτηση. Από την λίστα που θα προκύψει από κάτω, κάνουμε τικ στο «Greek» και ύστερα κλικ στο κουμπί «Install» (Εικόνα 5.6).



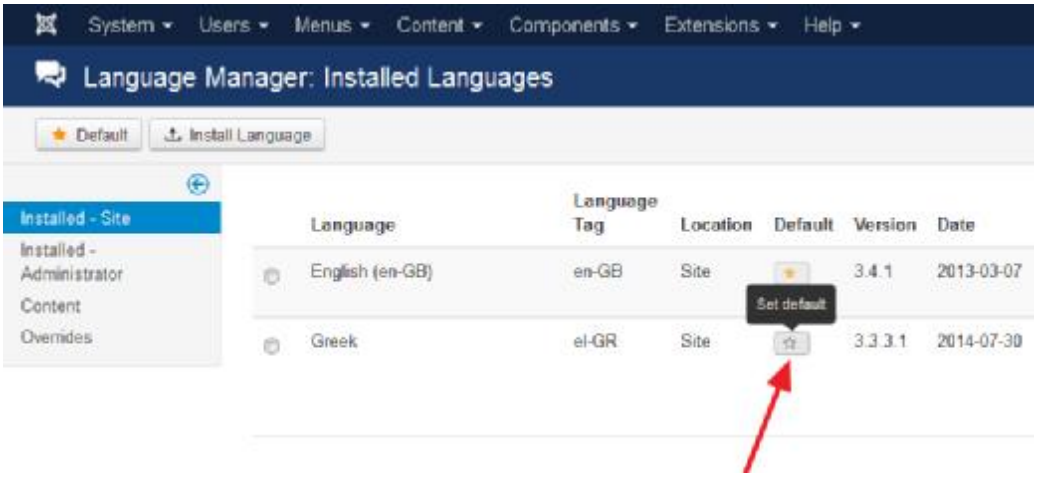
Εικόνα 5.6 Βήματα για την εγκατάσταση Ελληνικών στο joomla

Ενημερωνόμαστε για την επιτυχή εγκατάσταση της ελληνικής γλώσσας και ύστερα μεταφερόμαστε στην επιλογή «Language Manager» από το μενού «Extensions» (Εικόνα 5.7).



Εικόνα 5.7 Μήνυμα επιβεβαίωσης της επιτυχής εγκατάστασης των Ελληνικών

Από την οθόνη που μας εμφανίζεται, στο αριστερό μενού υπάρχει επιλεγμένο το «Installed - Site», βρίσκουμε το «Greek», και κάνουμε κλικ στο αστέρι που υπάρχει στην στήλη «Default» (Εικόνα 5.8).



Εικόνα 5.8 Ορισμός ως προεπιλεγμένης γλώσσας των Ελληνικών στο front-end

Ενημερωνόμαστε για την επιτυχή αλλαγή της γλώσσας και πως αυτό επηρεάζει κάθε χρήστη χωριστά. Ύστερα κάνουμε κλικ στο αριστερό μενού στην επιλογή «Installed - Administrator» για να αλλάξουμε και την γλώσσα του διαχειριστικού (Εικόνα 5.9).

Εικόνα 5.9 Επιλογή διαχείρης γλώσσας για το back-end

Παρομοίως με την προηγούμενη διαδικασία, κάνουμε κλικ στο αστέρι που βρίσκεται στην ίδια γραμμή με το «Greek» στην στήλη «Default» (Εικόνα 5.10).

Εικόνα 5.10 Επιλογή προεπιλεγμένης γλώσσας Ελληνικών για το back-end

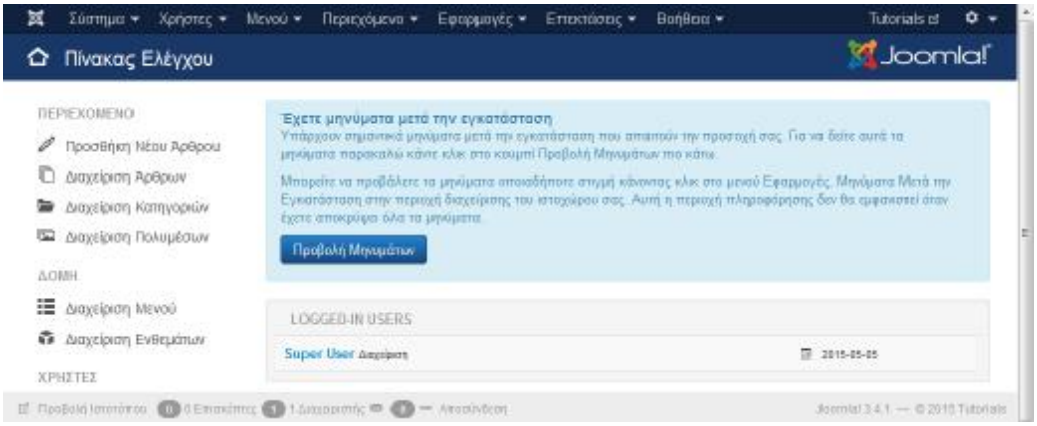
Επιβεβαιώνεται με παρόμοιο τρόπο όπως και προηγουμένως η αλλαγή της γλώσσας. Για να επιβεβαιώσουμε την αλλαγή θα κάνουμε αποσύνδεση και σύνδεση στο διαχειριστικό. Επομένως στο εικονίδιο με το γρανάζι πάνω δεξιά στην γωνία της οθόνης επιλέγουμε από το μενού την επιλογή «Logout» (Εικόνα 5.11).

Εικόνα 5.11 Αποσύνδεση από το διαχειριστικό σύστημα

Αμέσως μεταφερόμαστε στην οθόνη εισόδου χρήστη, από όπου παρατηρούμε την χρήση της ελληνικής γλώσσας (Εικόνα 5.12).

Εικόνα 5.12 Αποτέλεσμα αλλαγής γλώσσας στην οθόνη εισόδου

Κάνοντας είσοδο στο διαχειριστικό, παρατηρούμε την ίδια αλλαγή και στα μενού της αρχικής σελίδας της διαχείρισης. Η ελληνική γλώσσα εγκαταστάθηκε επιτυχώς! (Εικόνα 5.13).

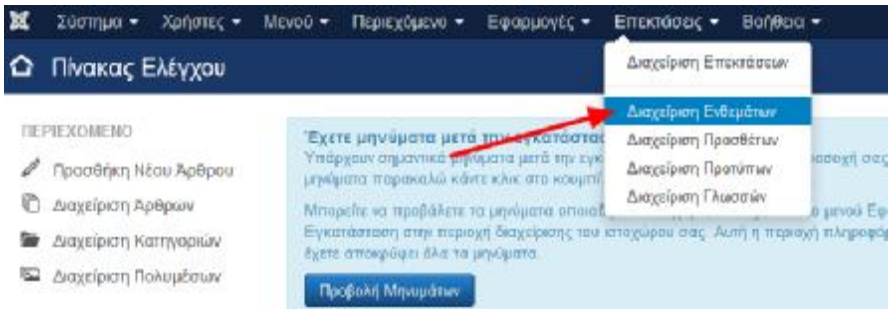


Εικόνα 5.13 Αποτέλεσμα αλλαγής γλώσσας στην αρχική οθόνη διαχειριστκού συστήματος

5.2.2 Ενθέματα

Στην διαχείριση ενθεμάτων μπορούμε να δημιουργήσουμε ενθέματα, να επεξεργαστούμε τα ήδη υπάρχοντα ή να τα διαγράψουμε, καθώς και να τους αλλάξουμε ονομασία ή τοποθεσία κ.α.

Για να δημιουργήσουμε ένα νέο ένθεμα στον ιστότοπο μας θα πρέπει να κάνουμε κλικ στην επιλογή «Διαχείριση Ενθεμάτων» στο μενού «Επεκτάσεις» του συστήματος διαχείρισης (Εικόνα 5.14).

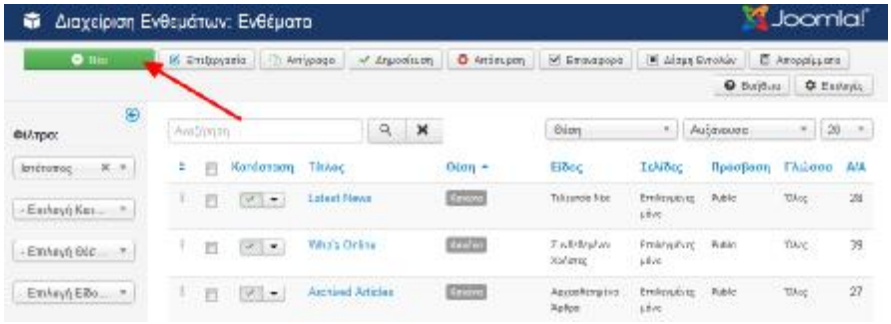


Εικόνα 5.14 Μετάβαση στο μενού ενθεμάτων

Μας φορτώνεται η λίστα με τα ήδη υπάρχοντα ενθέματα καθώς επίσης και πληροφορίες σχετικά με αυτά, όπως:

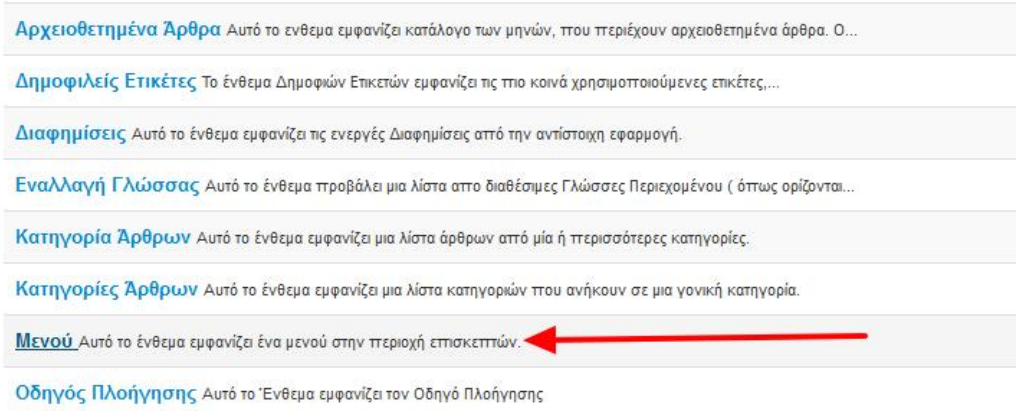
- Η κατάσταση: Αν εμφανίζεται στην ιστοσελίδα μας ή όχι,
- Ο τίτλος: Ο τίτλος που θα εμφανίζεται πάνω από το ένθεμα τόσο στην προβολή της ιστοσελίδας μας αλλά και στο διαχειριστικό σύστημα έτσι ώστε να μπορούμε να το αναγνωρίσουμε εύκολα.
- Θέση: Το όνομα της θέσης στο οποίο βρίσκεται αυτή την στιγμή το ένθεμα,
- Είδος: Το είδος το ενθέματος,
- Σελίδες: Σελίδες στον ιστότοπο μας όπου θα εμφανίζεται το ένθεμα αυτό,
- Πρόσβαση: Καθορίζει σε τι επίπεδο εμφανίζεται το ένθεμα, για παράδειγμα αν είναι public σημαίνει ότι εμφανίζεται σε όλους τους επισκέπτες, αν είναι users μόνο στους εγγεγραμμένους χρήστες κτλ.
- Γλώσσα: Σε ποίες γλώσσες έχει μεταφραστεί και θα εμφανίζεται στον ιστότοπο.
- Α/Α : Ο αύξοντας αριθμός της δημιουργίας τους.

Κάνουμε κλικ στο πράσινο κουμπί με ετικέτα «Νέο» πάνω αριστερά στην οθόνη (Εικόνα 5.15).



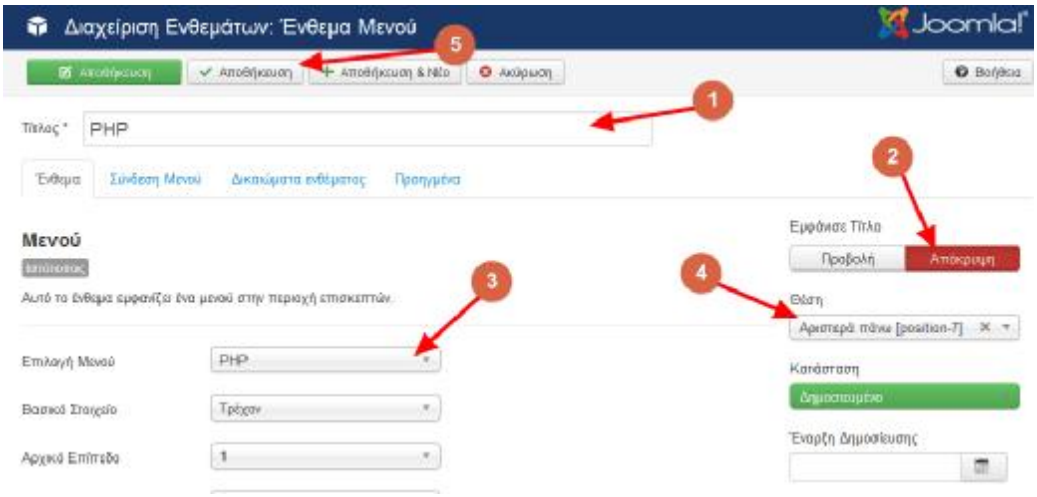
Εικόνα 5.15 Δημιουργία νέου ενθέματος

Από την οθόνη που μας εμφανίζεται επιλέγουμε το είδος του ενθέματος που θέλουμε να δημιουργήσουμε. Εμείς διαλέξαμε «Μενού» (Εικόνα 5.16).



Εικόνα 5.16 Επιλογή τύπου ενθέματος

Ύστερα μας εμφανίζεται η οθόνη δημιουργίας νέου ενθέματος, εκεί εισάγουμε ένα τίτλο, επιλέγουμε αν θέλουμε να εμφανίζεται η όχι ο τίτλος στους επισκέπτες (εμείς επιλέξαμε απόκρυψη) και ύστερα διαλέγουμε πιο μενού από αυτά που έχουμε δημιουργήσει θα χρησιμοποιηθεί στο ένθεμα από την επιλογή «Επιλογή Μενού». Τέλος διαλέγουμε την θέση στην οποία θα εμφανίζεται το ένθεμα (εμείς διαλέξαμε την θέση «Αριστερά πάνω position-7». Ύστερα κάνουμε κλικ στο αποθήκευση (Εικόνα 5.17).

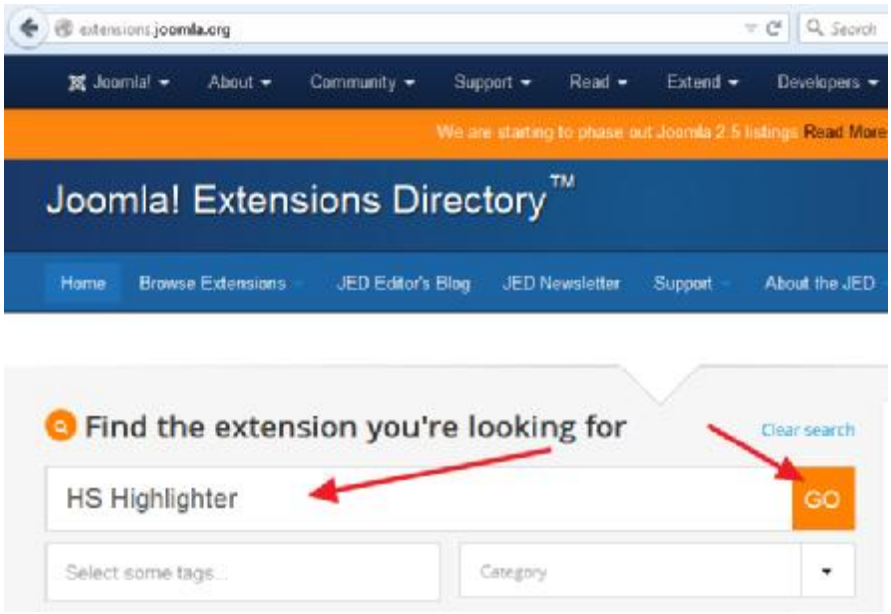


Εικόνα 5.17 Εισαγωγή απαραίτητων στοιχείων για την δημιουργία νέου ενθέματος

5.2.3 Πρόσθετα

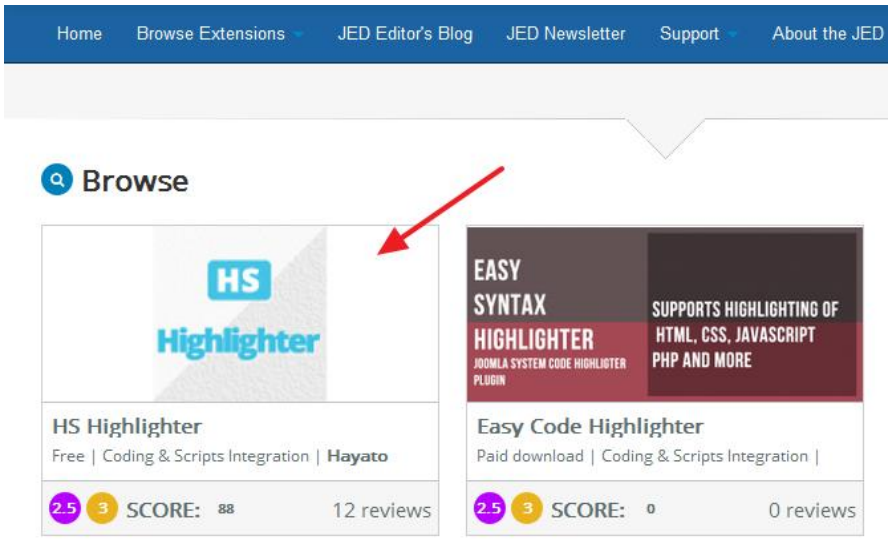
Στην διαχείριση πρόσθετων μπορούμε να εγκαταστήσουμε νέα ή να διαγράψουμε παλιά, να επεξεργαστούμε τις ρυθμίσεις των ήδη υπαρχόντων .κ.α. Προκειμένου να εγκαταστήσουμε ένα νέο πρόσθετο στην ιστοσελίδα μας θα πρέπει πρώτα να έχουμε εντοπίσει αυτό το πρόσθετο στην ιστοσελίδα πρόσθετων του joomla.org, να κατεβάσουμε το αρχείο εγκατάστασης του και ύστερα να το εγκαταστήσουμε στην δικιά μας ιστοσελίδα.

Ξεκινάμε επισκέπτοντας την σελίδα πρόσθετων του joomla.org στην διεύθυνση «http://extensions.joomla.org», από την οθόνη που μας εμφανίζεται γράφουμε τις αντίστοιχες λέξεις – κλειδιά στο πεδίο «Search» έτσι ώστε να βρούμε αυτό που μας ενδιαφέρει. Εμείς ψάξαμε το «HS Highlighter» (Εικόνα 5.18).



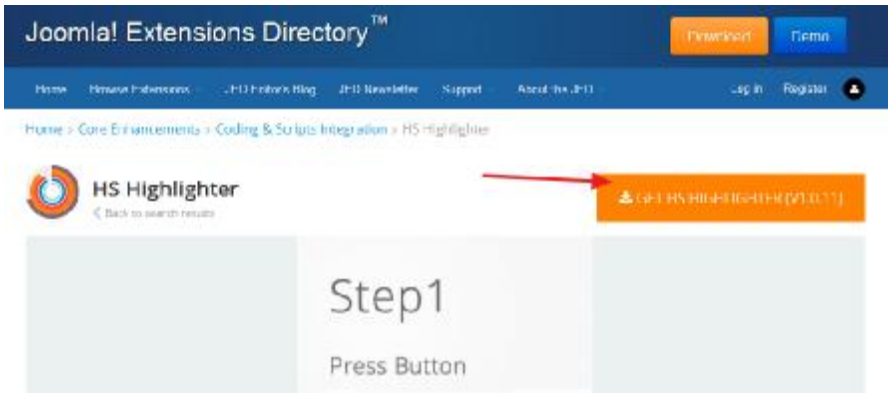
Εικόνα 5.18 Αναζήτηση πρόσθετων του joomla

Κάνοντας κλικ στο κουμπί «GO» μεταφερόμαστε στα αποτελέσματα της αναζήτησης. Πρώτη επιλογή για εμάς είναι αυτό το οποίο αναζητούμε. Κλικ σε αυτό (Εικόνα 5.19).



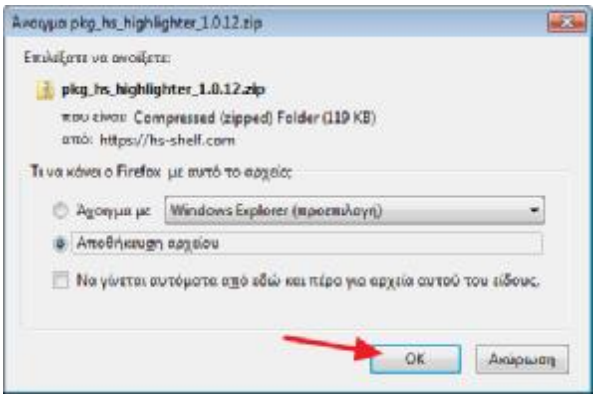
Εικόνα 5.19 Επιλογή πρόσθετου προς επισκόπηση

Μεταφερόμαστε σε οθόνη προβολής στοιχείων για το πρόσθετο. Σε αυτή την οθόνη υπάρχει ένα κουμπί με όνομα «GET HS HIGHLIGHTER» κάνουμε κλικ εκεί (Εικόνα 5.20).



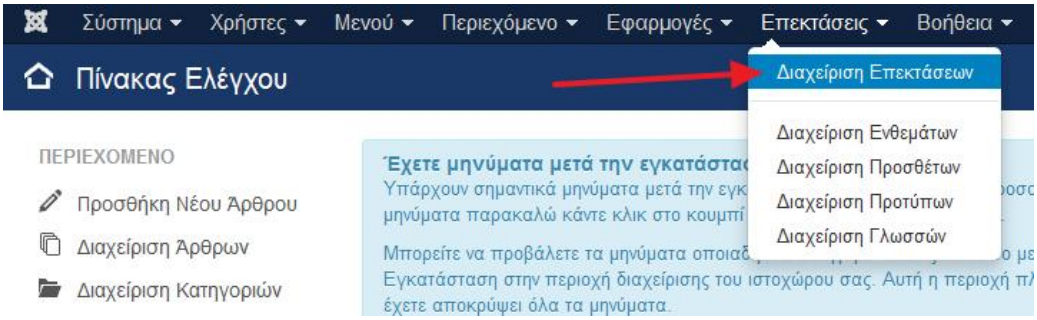
Εικόνα 5.20 Μεταφόρτωση πρόσθετου

Μας εμφανίζεται αναδυόμενο παράθυρο ερώτησης αποθήκευσης αρχείου. Κάνουμε κλικ στο «OK» (Εικόνα 5.21).



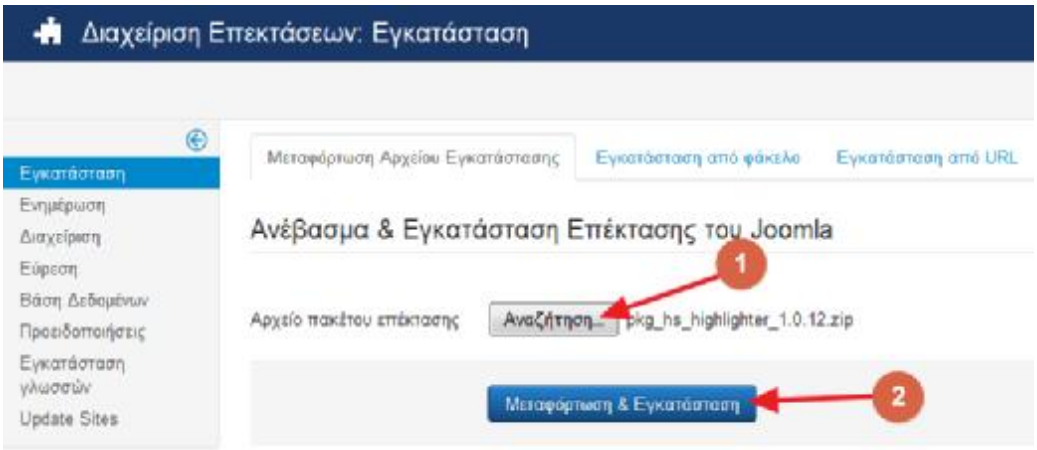
Εικόνα 5.21 Αποθήκευση αρχείου τοπικά στον υπολογιστή μας

Ύστερα από την επιτυχή μεταφόρτωση του πρόσθετου αρχείου, μεταφερόμαστε γράφοντας στην γραμμή διευθύνσεων του firefox στην διεύθυνση «http://localhost/joomla/administrator» όπου βρίσκεται το διαχειριστικό σύστημα της τοπικής μας ιστοσελίδας. Αν έχουμε κάνει είσοδο στο σύστημα από πριν μας εμφανίζεται η αρχική οθόνη του διαχειριστικού όπου θα μεταφερθούμε στην επιλογή «Διαχείριση Επεκτάσεων» από το μενού «Επεκτάσεις» (Εικόνα 5.22).



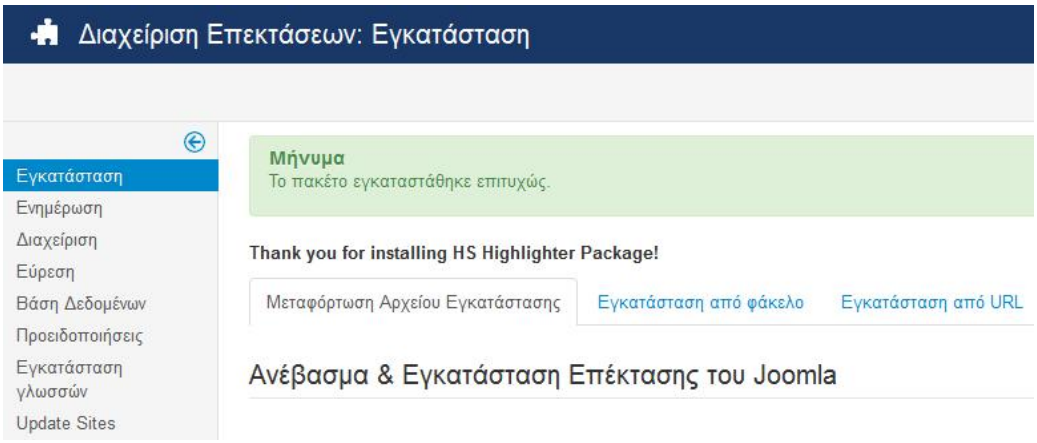
Εικόνα 5.22 Μετάβαση στην διαχείρηση επεκτάσεων

Κάνουμε κλικ στο κουμπί «Αναζήτηση» και βρίσκουμε το αρχείο που κατεβάσαμε προηγουμένως. Πιθανώς να βρίσκεται στον κατάλογο «downloads» του υπολογιστή μας αν στο αναδυόμενο παράθυρο μεταφόρτωσης προηγουμένως δεν αλλάξαμε τοποθεσία μεταφόρτωσης. Αφού το επιλέξουμε κάνουμε κλικ στο κουμπί «Μεταφόρτωση & Εγκατάσταση» (Εικόνα 5.23).



Εικόνα 5.23 Διαδικασία για την εγκατάσταση νέας επέκτασης

Αμέσως ενημερωνόμαστε για την επιτυχή διαδικασία εγκατάστασης. Το πρόσθετο μας εγκαταστάθηκε! (Εικόνα 5.24).



Εικόνα 5.24 Ενημερωτικό μήνυμα επιτυχούς εγκατάστασης πρόσθετου/επέκτασης

5.3 Πρότυπα εμφάνισης

Τα πρότυπα (templates) καθορίζουν την εμφάνιση και το στυλ της ιστοσελίδας και διατηρούνται ξεχωριστά από το υπόλοιπο περιεχόμενο της. Τα πρότυπα αυτά διαθέτουν τόσο πληροφορίες που αποθηκεύονται στην βάση δεδομένων της MySQL όσο και αρχεία κώδικα και εικόνων που βρίσκονται σε ένα ξεχωριστό υποκατάλογο των αρχείων του joomla.

Υπάρχουν δυο τύποι προτύπων, τα site Templates και τα Administrator Templates. Τα site Templates αναφέρονται στην εμφάνιση της ιστοσελίδας όπως θα την βλέπουν οι επισκέπτες που θα την επισκέπτονται, είναι δηλαδή front-end πρότυπα για την ιστοσελίδα μας, ενώ τα Administrator Templates αναφέρονται στην εμφάνιση της ιστοσελίδας στο σύστημα διαχείρισης της που έχουν πρόσβαση μόνο οι διαχειριστές και αυτοί που συντηρούν και την κατέχουν, είναι δηλαδή back-end πρότυπα.

Το joomla διαθέτει διαχειριστικό περιβάλλον για την διαχείριση των προτύπων που διαθέτει και στους δύο τύπους προτύπων. Μπορούμε δηλαδή μέσα από το joomla να αλλάξουμε πρότυπο εμφάνισης μέσω μιας λίστας των ήδη εγκαταστημένων προτύπων με ένα απλό κλικ, να εγκαταστήσουμε ένα καινούργιο πρότυπο, ή να διαχειριστούμε και να ρυθμίσουμε πληθώρα επιλογών μέσα από το ίδιο το πρότυπο.

Δεν είναι απόλυτη η χρήση ενός μοναδικού προτύπου για έναν ιστότοπο. Πολλά πρότυπα μπορούν να χρησιμοποιηθούν ταυτόχρονα σε διαφορετικά τμήματα του ιστότοπου. Αυτό καθιστά τον ιστότοπο μας εξαιρετικά ευέλικτο και προσαρμοστικό

σε άμεσες αλλαγές. Έτσι θα μπορούσαμε να χρησιμοποιούμε διαφορετικά πρότυπα σε διαφορετικές υπό-σελίδες της ιστοσελίδας μας.

5.3.1 Εγκατάσταση πρότυπου

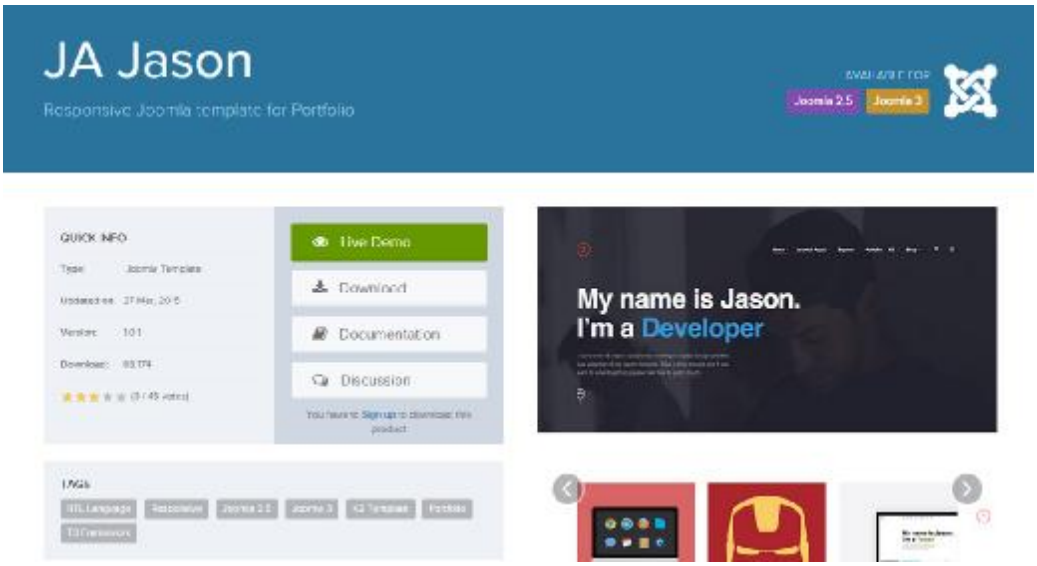
Ο τρόπος με τον οποίο εγκαθιστούμε ένα πρότυπο είναι σχεδόν κοινός με τον τρόπο με τον οποίο εγκαθιστάμε ένα πρόσθετο στο joomla. Η διαφορά εδώ βρίσκεται στο ότι τα πρότυπα μπορούν να βρεθούν οπουδήποτε στο internet, είτε επί πληρωμή είτε και ελεύθερα. Η ανάπτυξη τέτοιων προτύπων πλέον έχει γίνει επάγγελμα και πολλές εταιρείες προσφέρουν μοναδικά και εξελιγμένα πρότυπα επί πληρωμή.

Πριν εγκαταστήσουμε κάποιο θα πρέπει να το δούμε, να το δοκιμάσουμε και να επιβεβαιώσουμε ότι ταιριάζει στην έκδοση joomla που διαθέτουμε αλλά και στις προτιμήσεις μας.

Μερικές από τις ιστοσελίδες που εμείς ελέγξαμε είναι:

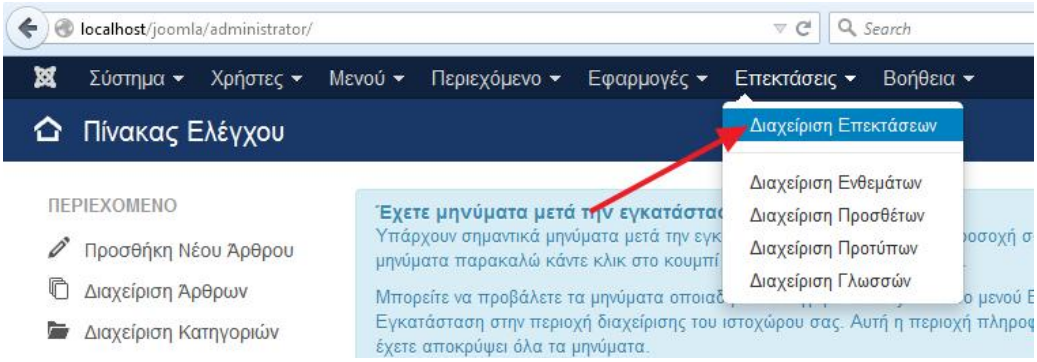
1. http://www.joomlart.com
2. http://www.rockettheme.com
3. http://www.yootheme.com

Καταλήξαμε στο επί πληρωμή πρότυπο «JA Jason» από την διεύθυνση: https://www.joomlart.com/joomla/templates/ja-jason για την έκδοση 3 του joomla που διαθέτουμε. Αφού έχουμε κάνει τις απαραίτητες διαδικασίες εγγραφής στην ιστοσελίδα αγοράς του προτύπου, φορτώνοντας την παραπάνω διεύθυνση μας εμφανίζεται η παρακάτω σελίδα (Εικόνα 5.25).



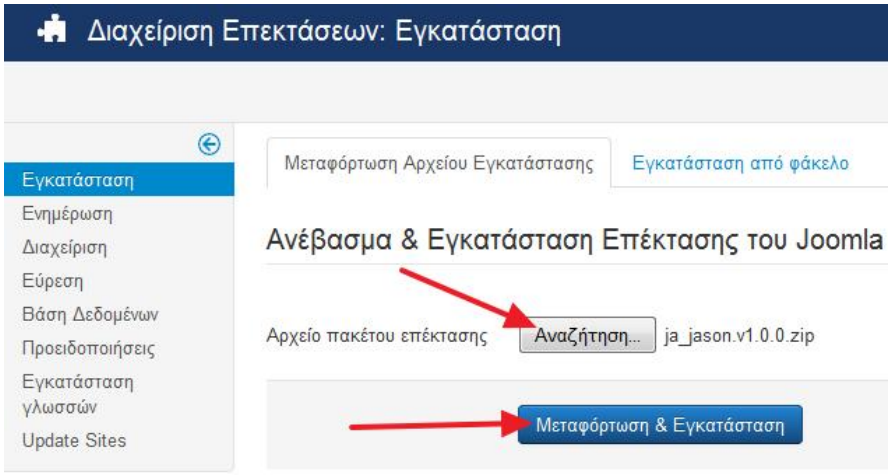
Εικόνα 5.25 Σελίδα προτύπου JA Jason

Κάνοντας κλικ στο «Live Demo» μπορούμε να δούμε μια δοκιμαστική έκδοση του προτύπου online. Για κατεβάσουμε το πρότυπο κάνουμε κλικ στο κουμπί «Download». Αφού κατεβάσουμε το αρχείο στον κατάλογο «downloads» του υπολογιστή μας, επισκεπτόμαστε το διαχειριστικό της ιστοσελίδας μας στο «http://localhost/joomla/administrator» και μεταφερόμαστε στην επιλογή «Διαχείριση Επεκτάσεων» στο μενού «Επεκτάσεις» (Εικόνα 5.26).



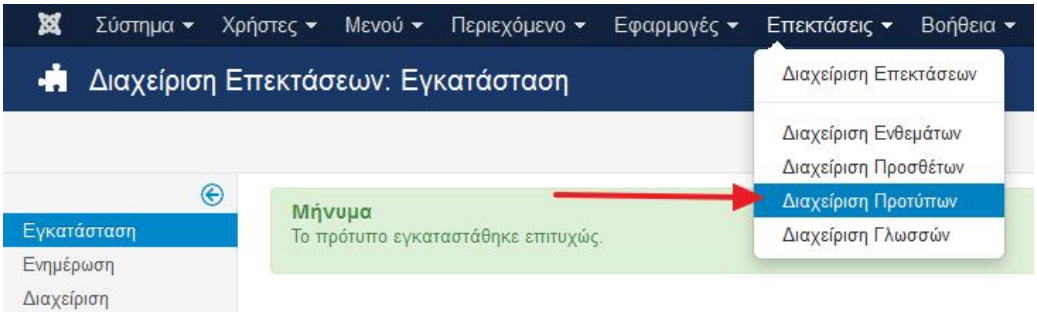
Εικόνα 5.26 Μετάβαση στην διαχείρηση επεκτάσεων

Από την οθόνη που μας εμφανίζεται κάνουμε κλικ στο κουμπί «αναζήτηση» και επιλέγουμε το αρχείο που κατεβάσαμε προηγουμένως. Θα πρέπει να βρίσκεται σε κατάληξη «.zip». Αφού το διαλέξουμε κάνουμε κλικ στο κουμπί «Μεταφόρτωση & Εγκατάσταση» (Εικόνα 5.27).



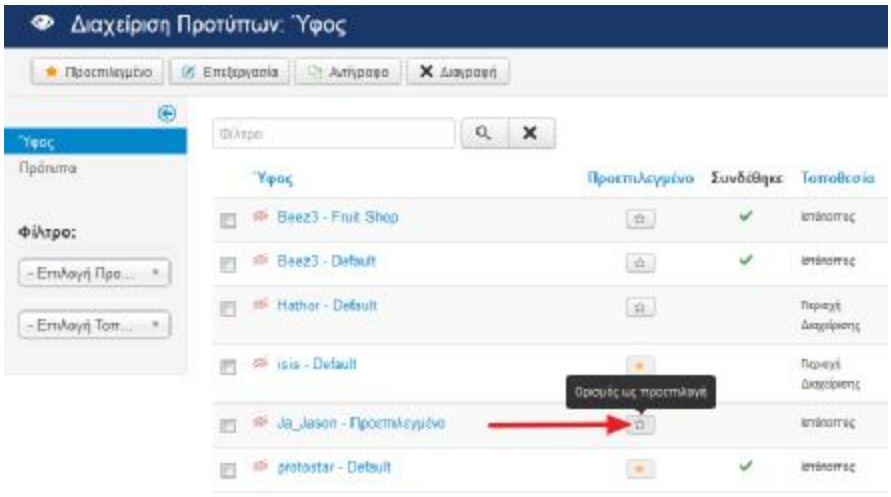
Εικόνα 5.27 Επιλογή τοπικού αρχείου για εγκατάσταση επέκτασης

Ενημερωνόμαστε για την επιτυχή εγκατάσταση του πρότυπου και μεταφερόμαστε στην επιλογή «Διαχείριση Προτύπων» από το μενού «Επεκτάσεις» (Εικόνα 5.28).



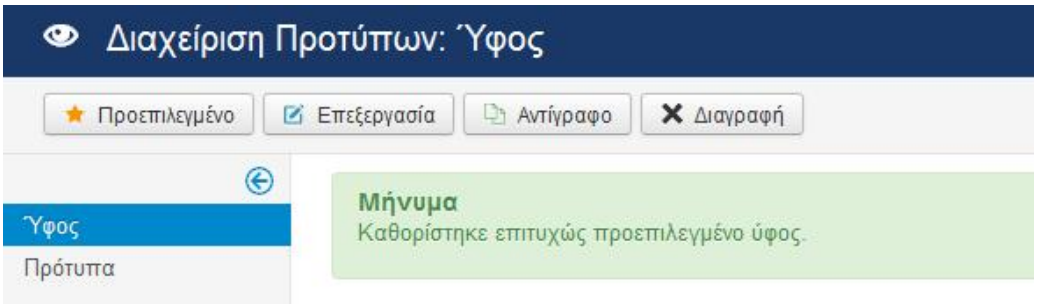
Εικόνα 5.28 Μετάβαση στην διαχείρηση προτύπων

Από την λίστα που μας φορτώνεται κάνουμε κλικ στο αστεράκι που βρίσκεται στην στήλη «Προεπιλεγμένο» στην ίδια ευθεία οριζοντίως με το όνομα του πρότυπου που εγκαταστήσαμε (Εικόνα 5.29).



Εικόνα 5.29 Επιλογή προεπιλεγμένου προτύπου

Η διαδικασία επιλογής του προτύπου ως βασικό ολοκληρώθηκε επιτυχώς! (Εικόνα 5.30).



Εικόνα 5.30 Μήνυμα επιβεβαίωσης αλλαγής προεπιλεγμένου προτύπου

Για επιβεβαίωση επισκεπτόμαστε την ιστοσελίδα μας στην διεύθυνση «http://localhost/joomla» (Εικόνα 5.31).

Εικόνα 5.31 Αποτέλεσμα εφαρμογής προτύπου στην σελίδα

5.4 Περιεχόμενο

Το περιεχόμενο είναι ένα από τα σημαντικότερα μέρη ενός ιστότοπου. Ένας ιστότοπος χωρίς περιεχόμενο και χωρίς να έχει «κάτι να δείξει» στους επισκέπτες του είναι απλά ένας αποτυχημένος ιστότοπος. Στην περίπτωση μας το περιεχόμενο μας περιλαμβάνει τους οδηγούς εκμάθησης της κάθε γλώσσας. Η php και Javascript της οποίας το περιεχόμενο θα πρέπει να κατανεμηθεί με βάση την εκάστοτε διάθρωση. Έτσι αφού εξετάστηκαν καλά οι παραπάνω γλώσσες και αποφασίστηκε η ύλη που θα καλυφθεί, δημιουργήθηκαν και τα ανάλογα κεφάλαια και ενότητες του οδηγού εκμάθησης.

5.4.1 Κατηγορίες ιστότοπου

Οι κατηγορίες στο joomla είναι κάτι το πολύ σημαντικό κυρίως για την οργάνωση και την διαδικασία της διαχείρισης. Αποτελεί κάτι τον ακρογωνιαίο λίθο της σωστής διαχείρισης ενός ιστότοπου κάνοντας ευκολότερη την εύρεση, την επεξεργασία και την διαγραφή υλικού που εμπεριέχεται στην εκάστοτε κατηγορία. Οι κατηγορίες συνήθως δεν «φαίνονται» στον επισκέπτη παρά μόνο αν το επιτρέψουμε εμείς μέσω των ανάλογων ρυθμίσεων ή ενθεμάτων. Εφόσον έχουμε σκοπό να χρησιμοποιήσουμε τόσο back-end όσο και front-end τις κατηγορίες θα πρέπει να εξασφαλίσουμε ότι δεν θα είναι κουραστικό για κάποιον να μπορέσει να περιηγηθεί μέσα σε αυτές τις κατηγορίες, και πως θα του παρέχεται η κατάλληλη θεματική συνοχή στην σειρά με

την οποία παρουσιάζονται. Έτσι καταλήξαμε στην δημιουργία κατηγοριών τριών επιπέδων για την ανάλογη γλώσσα. Παρακάτω παρουσιάζονται αναλυτικά οι κατηγορίες που προέκυψαν:

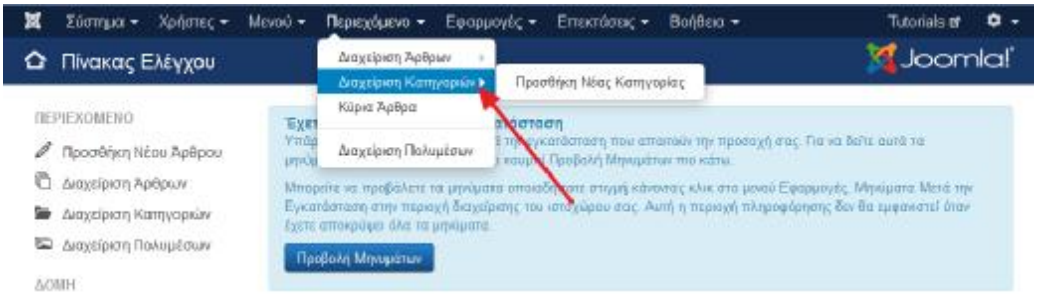
PHP

1. Εισαγωγή
2. HTML & PHP
3. Μεταβλητές
4. Βασικές Εντολές
5. Μέθοδοι
6. Αλφαριθμητικά
7. Πίνακες
8. Αντικείμενα
9. MySQL & PHP

Javascript

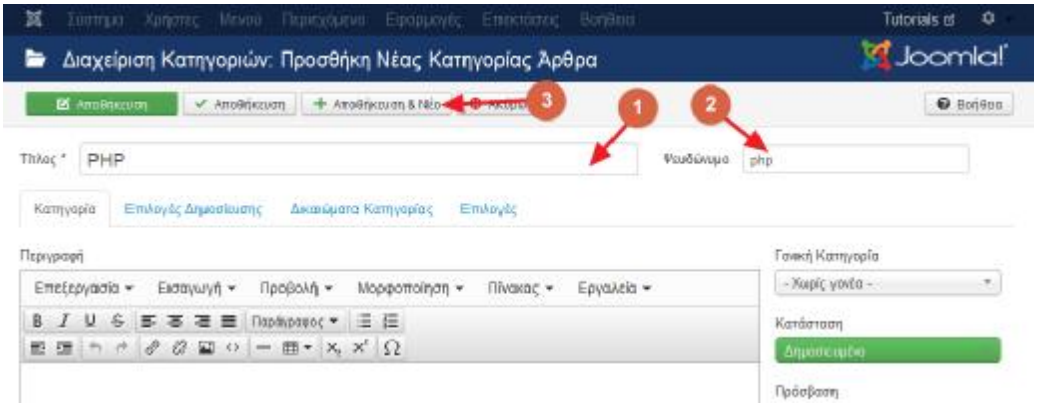
1. Εισαγωγή
2. HTML & Javascript
3. Μεταβλητές και πράξεις
4. Μέθοδοι
5. Αποφάσεις
6. Βρόγχοι
7. Πίνακες

Για να δημιουργήσουμε μια νέα κατηγορία άρθρων θα πρέπει να μεταβούμε κάνοντας κλικ στην επιλογή «Προσθήκη Νέας Κατηγορίας» η οποία βρίσκεται στο μενού «Περιεχόμενο» και ύστερα «Διαχείριση Κατηγοριών» (Εικόνα 5.32).



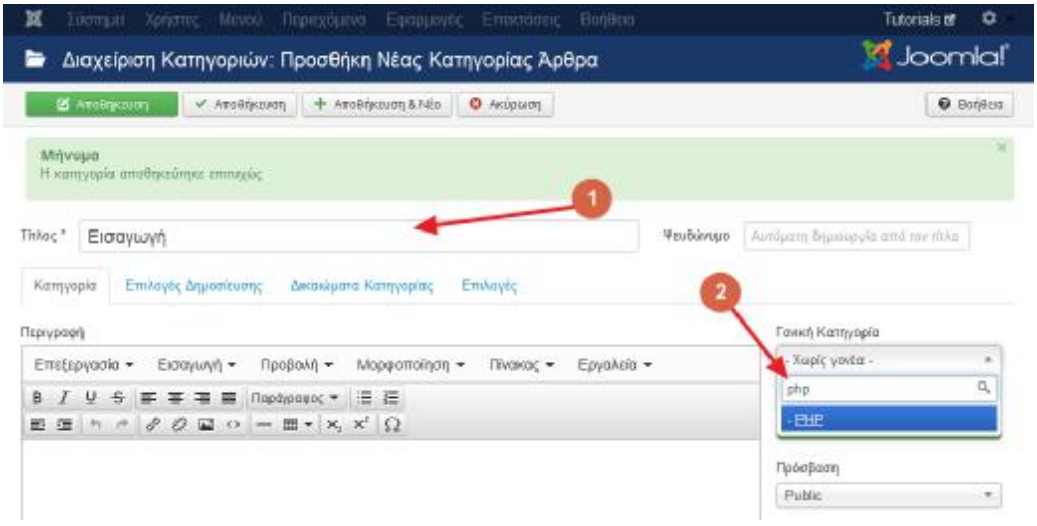
Εικόνα 5.32 Μετάβαση στην δημιουργία νέας κατηγορίας

Μας εμφανίζεται η αντίστοιχη σελίδα δημιουργίας μιας κατηγορίας άρθρων όπου θα πρέπει, Να εισάγουμε ένα όνομα για την κατηγορία μας, και ενα ψευδώνυμο (Εικόνα 5.33).



Εικόνα 5.33 Εισαγωγή απαραίτητων στοιχείων για την δημιουργία νέας κατηγορίας

Αν έχουμε να εισάγουμε υπο-κατηγορίες στην κατηγορία που δημιουργήσαμε τότε στην Εικόνα 5.33 κάνουμε κλίκ στο «Αποθήκευση & Νέο» αλλιώς στο «Αποθήκευση». Για να δημιουργήσουμε μια υπο-κατηγορία αρκεί μονο στην εισαγωγή των στοιχείων (στην νέα οθόνη δημιουργίας) να ορίσουμε γονική κατηγορία ακολουθόντας κατά τα άλλα ότι θα κάναμε δημιουργόντας μια οποιαδήποτε νεα κατηγορία (Εικόνα 5.34).



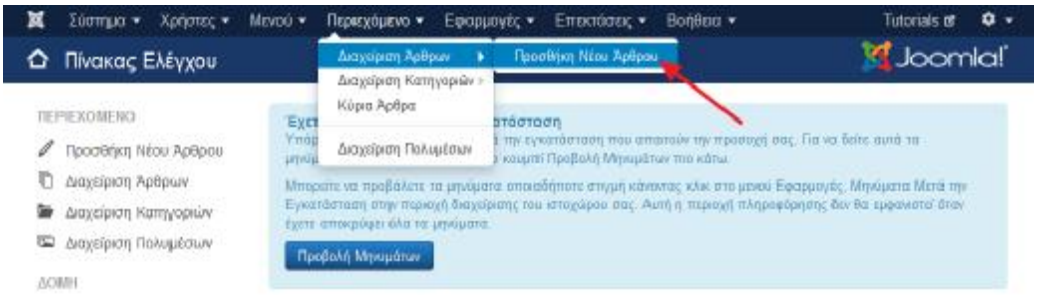
Εικόνα 5.34 Δημιουργία υποκατηγορίας

5.4.2 Άρθρα

Τα άρθρα της ιστοσελίδας μας είναι το περιεχόμενο της. Ότι περιλαμβάνει η ιστοσελίδα μας, από κείμενο, εικόνες, μέχρι και βίντεο , όλα αυτά παρουσιάζονται αποσπασματικά μέσω άρθρων. Θα πρέπει να προσέχουμε το είδος της πληροφορίας που παρουσιάζουμε έτσι ώστε ούτε να είναι περισσότερη αλλά ούτε λιγότερη από όση ο επισκέπτης μπορεί να ανταπεξέλθει διαβάζοντας την.

Η φύση της ιστοσελίδας μας καθιστά δυσκολότερη την επισκεψιμότητα της λόγω των δυσνόητων όρων και του τρόπου υλοποίησης της κάθε γλώσσας. Έτσι εστιάζουμε να παρουσιάσουμε μόνο τα απαραίτητα στην κάθε περίπτωση, με όσο πιο απλό και κατανοητό τρόπο. Τα άρθρα μας αποτελούνται από επεξηγηματικό κείμενο παρουσίασης της κάθε λειτουργίας, το εκάστοτε δοκιμαστικό κώδικα για την κάθε επεξήγηση, και εικόνες που υποδεικνύουν το αποτέλεσμα στην εκτέλεση του κώδικα.

Για να δημιουργήσουμε ένα άρθρο μέσα στο διαχειριστικό του Joomla θα πρέπει να κάνουμε κλικ στην επιλογή «Προσθήκη Νέου Άρθρου» που βρίσκεται στο μενού «Περιεχόμενο» και ύστερα «Διαχείριση Άρθρων» (Εικόνα 5.35).



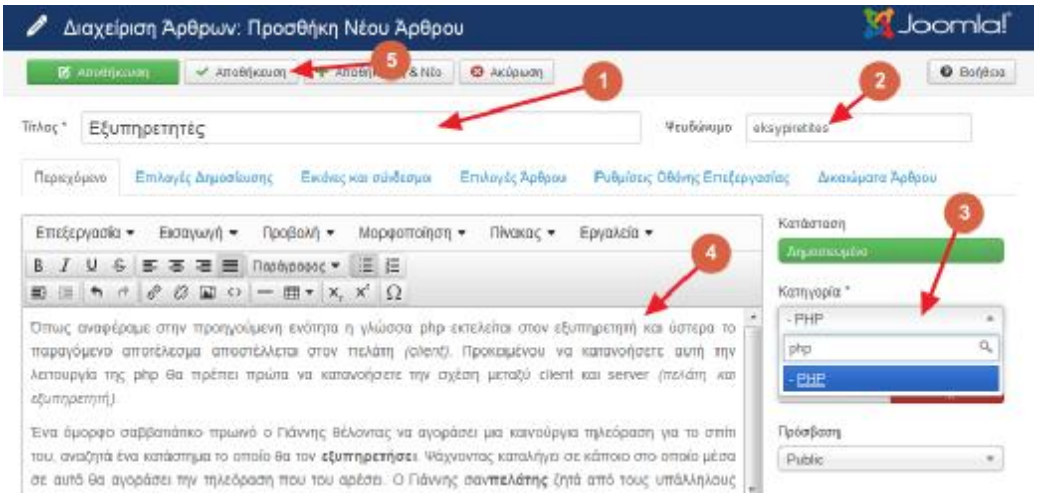
Εικόνα 5.35 Μετάβαση στην προσθήκη νέου άρθρου

Η οθόνη που μας φορτώνεται περιλαμβάνει όλα τα απαραίτητα στοιχεία τα οποία χρειάζονται για να δημιουργηθεί ένα άρθρο (Εικόνα 5.36).

1. Αρχικά εισάγουμε έναν τίτλο για το άρθρο μας, ο τίτλος είναι σημαντικός και θα πρέπει να είναι αντιπροσωπευτικός του τι πρόκειται να ακολουθήσει όταν κάποιος κάνει κλικ στο τίτλο του για να το διαβάσει.
2. Προαιρετικά αν θέλουμε μπορούμε να εισάγουμε ένα ψευδώνυμο για το άρθρο μας με λατινικούς συνήθως χαρακτήρες όπου θα φαίνεται στο url στο οποίο θα

βρίσκεται το άρθρο μας. Αν δεν εισάγουμε εμείς ψευδώνυμο το joomla δημιουργεί από μόνο του με βάση τον τίτλο που εισάγαμε.

3. Θα πρέπει να διαλέξουμε κατηγορία στην οποία ανήκει το συγκεκριμένο άρθρο που πρόκειται να γράψουμε. Η σωστή αρχειοθέτηση βοηθά τους επισκέπτες να βρίσκουν παρόμοια άρθρα σε αντίστοιχες κατηγορίες.
4. Εισάγουμε το περιεχόμενο του άρθρου μέσα από τον αντίστοιχο επεξεργαστή κειμένου που διαθέτουμε. Εδώ μπορούμε να προσθέσουμε μορφοποιημένο κείμενο, συνδέσμους, εικόνες κ.α.
5. Τέλος για να ολοκληρωθεί η διαδικασία κάνουμε κλικ στο αποθήκευση με το τικ δίπλα του, αν θέλουμε να αποθηκεύσουμε και να το κλείσουμε ή κάνουμε κλικ στο πράσινο αποθήκευση αν θέλουμε να αποθηκευτεί η πρόοδος και να συνεχίσουμε την επεξεργασία του ίδιου άρθρου.



Εικόνα 5.36 Προσθήκη όλων των απαραίτητων στοιχείων για την δημιουργία νέας κατηγορίας

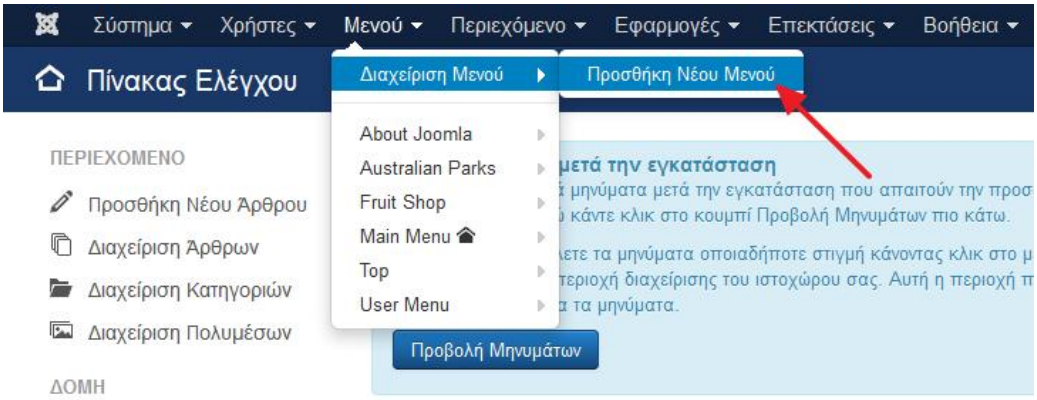
5.4.3 Μενού

Στο Joomla κάθε περιεχόμενο που διαθέτει (κείμενο, φωτογραφίες) είναι ανεξάρτητο από τον τρόπο με τον οποίο εμφανίζεται αυτό το περιεχόμενο στην ιστοσελίδα. Δημιουργώντας ένα άρθρο για παράδειγμα αυτό δεν σημαίνει κάλλιστα πως το άρθρο θα συμπεριλαμβάνεται στην αρχική σελίδα του ιστότοπου μας ή σε κάποιο μενού αυτόματα. Συνήθως συμπεριλαμβάνεται μόνο στην αντίστοιχη κατηγορία. Για να εμφανιστεί το περιεχόμενο αυτό στην ιστοσελίδα συνήθως θα πρέπει να δημιουργηθεί ένας σύνδεσμος ο οποίος θα οδηγεί σε αυτό.

Με την χρήση των μενού επιτυγχάνεται η πλοήγηση και η πρόσβαση στις διάφορες περιοχές της ιστοσελίδας. Πρόκειται στην ουσία για συνδέσμους (links) σε τομείς, κατηγορίες, συστατικά, η εξωτερικές σελίδες. Αυτοί οι σύνδεσμοι ονομάζονται Αντικείμενα Μενού. Κάθε μενού θα πρέπει να έχει ένα αναγνωριστικό όνομα το οποίο θα εμφανίζεται εσωτερικά στο joomla αλλά θα περιλαμβάνεται αντίστοιχα και στο url που θα «δείχνει» στο περιεχόμενο που θέλουμε.

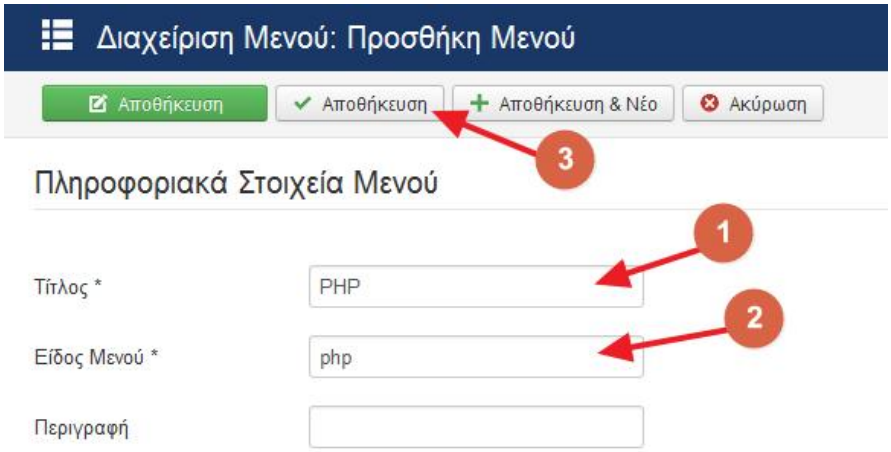
Τα μενού της ιστοσελίδας μας αποτελούν το front-end στοιχείο ραχοκοκαλιάς περιεχομένου που οι επισκέπτες θα έχουν άμεση επαφή. Τα μενού θα πρέπει να έχουν σύντομες ονομασίες, εύκολα διακριτές, και χωρίς πολλά υπομενού. Στην περίπτωση μας χρησιμοποιούμε τον νόμο των τριών κλικ, όπου ένας επισκέπτης με βάση τα μενού και την περιήγηση στην ιστοσελίδα μας θα μπορεί να βρει αυτό που ψάχνει μόνο με την χρήση τριών κλικ.

Για να δημιουργήσουμε ένα μενού κάνουμε κλικ στην επιλογή «Προσθήκη Νέου Μενού» που βρίσκεται στο μενού «Περιεχόμενο» και ύστερα «Διαχείριση μενού» (Εικόνα 5.37).



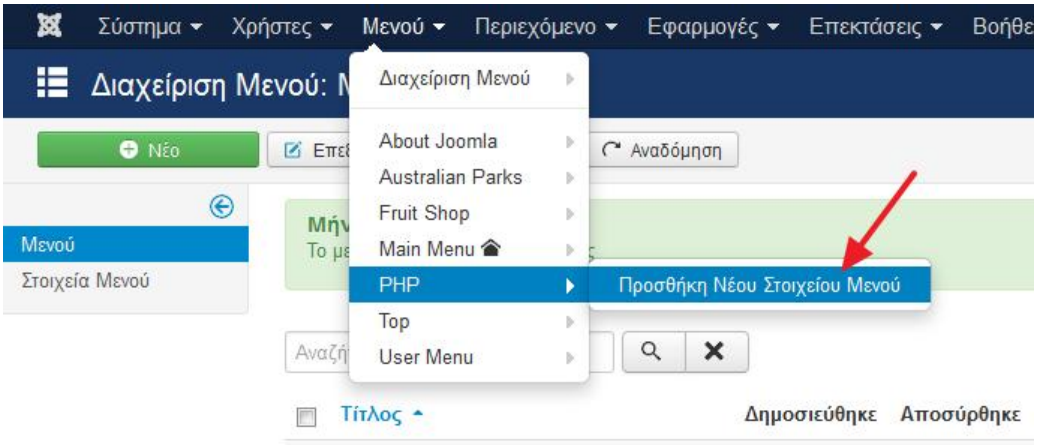
Εικόνα 5.37 Μετάβαση στην προσθήκη νέου μενού

Ύστερα μας εμφανίζεται η οθόνη δημιουργίας νέου μενού. Εισάγουμε τίτλο μενού και είδος μενού και κάνουμε κλικ στο κουμπί «Αποθήκευση» (Εικόνα 5.38).



Εικόνα 5.38 Συμπλήρωση φόρμας δημιουργίας μενού

Αφού δημιουργήσουμε το νέο μας μενού, το επόμενο βήμα είναι η δημιουργία των στοιχείων μενού (δηλ των συνδέσμων). Κάνουμε κλικ στην επιλογή «Προσθήκη Νέου Στοιχείου Μενού» που βρίσκεται στο μενού με όνομα «Μενού» και ύστερα στο όνομα του μενού που δημιουργήσαμε προ λίγου, στην περίπτωση μας «PHP» (Εικόνα 5.39).

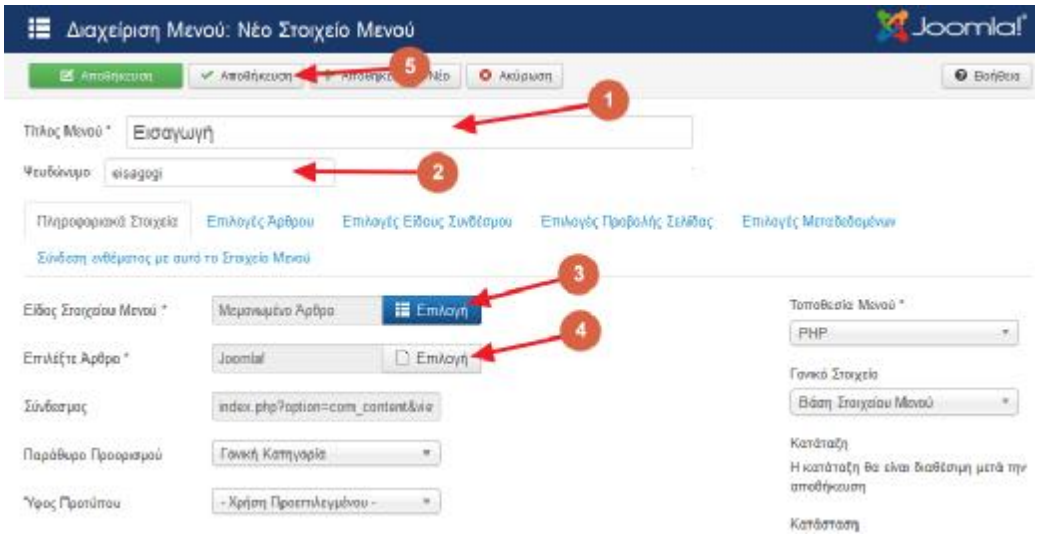


Εικόνα 5.39 Μετάβαση στην προσθήκη νέου στοιχείου μενού

Αμέσως εμφανίζεται η οθόνη δημιουργίας νέου στοιχείου μενού όπου (Εικόνα 5.40):

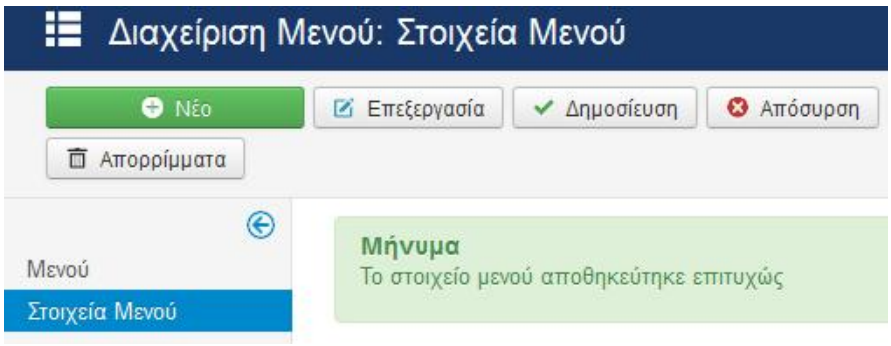
1. Εισάγουμε τίτλο για το μενού μας. Ο τίτλος είναι αυτός ο οποίος θα φαίνεται και στους χρήστες,
2. Προαιρετικά ψευδώνυμο, το οποίο θα χρησιμοποιηθεί στο url,

3. Διαλέγουμε στο «Είδος Στοιχείου Μενού» το είδος που επιθυμούμε, στην περίπτωση μας «Μεμονωμένο Άρθρο»,
4. Επιλέγουμε το άρθρο το οποίο θέλουμε να χρησιμοποιηθεί γι αυτό το μενού κάνοντας κλικ στο κουμπί «Επιλογή».



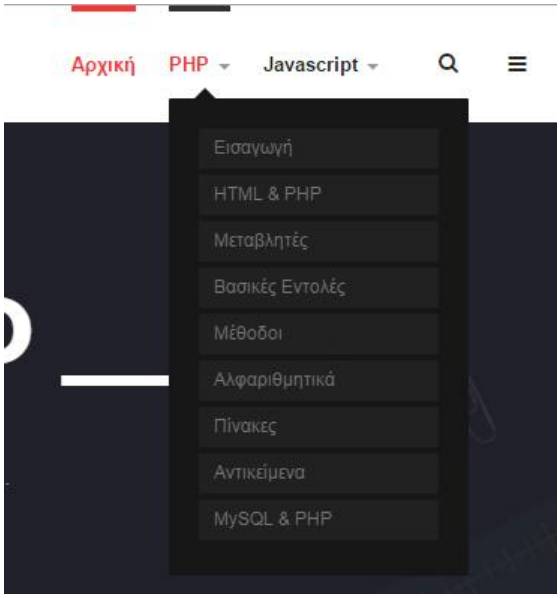
Εικόνα 5.40 Συμπλήρωση φόρμας δημιουργίας νέου στοιχείου μενού

Αφού τελειώσουμε με όλα τα παραπάνω κάνουμε κλικ στο κουμπί «Αποθήκευση» και η διαχείριση μας ενημερώνει ότι το στοιχείο μενού αποθηκεύτηκε επιτυχώς (Εικόνα 5.41).



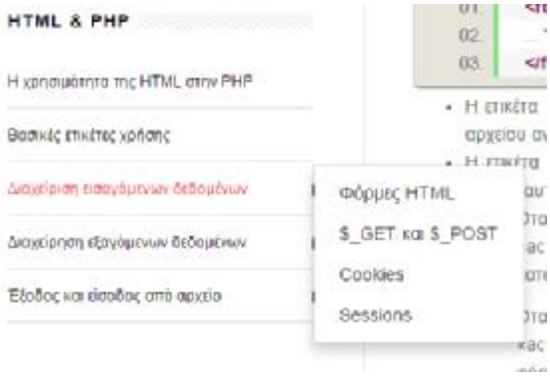
Εικόνα 5.41 Μήνυμα επιβεβαίωσης δημιουργίας στοιχείου μενού

Για να εμφανιστεί το κάθε μενού στον ιστότοπο μας θα πρέπει να χρησιμοποιήσουμε ένα ένθεμα τοποθετημένο σε κάποιο σημείο της ιστοσελίδας μας θα εμφανίζεται αντίστοιχα το μενού μας εκεί. Το αποτέλεσμα της δημιουργίας του μενού php που προκύπτει έχει ως εξής (Εικόνα 5.42):



Εικόνα 5.42 Βασικό μενού περιήγησης ιστοσελίδας

Ενώ αντίστοιχα κι άλλα ενθέματα που δημιουργήσαμε για την περιήγηση μέσα στους οδηγούς προέκυψαν όπως στην παρακάτω εικόνα (Εικόνα 5.43):



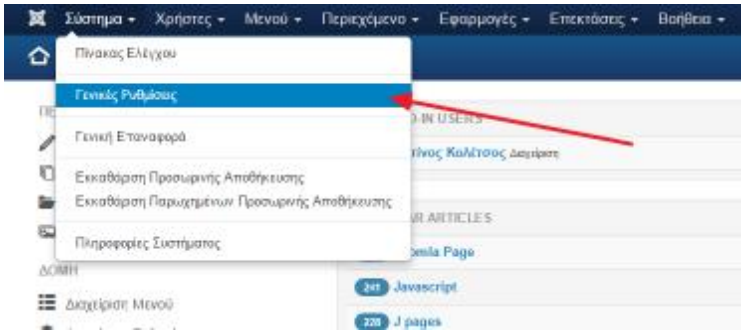
Εικόνα 5.43 Μενού περιήγησης μέσα σε οδηγούς

5.5 Γενικές ρυθμίσεις

Το joomla μας παρέχει ένα σύνολο ολοκληρωμένων ρυθμίσεων για τον ιστότοπο μας δίνοντας μας έτσι του πλήρη ελέγχου για το τι συμβαίνει σε αυτόν. Οι ρυθμίσεις χωρίζονται σε κατηγορίες ανάλογα με το θέμα το οποίο απασχολούν. Έτσι έχουμε τις ρυθμίσεις ιστότοπου, τις ρυθμίσεις συστήματος, τις ρυθμίσεις διακομιστή, τις ρυθμίσεις δικαιωμάτων, και τις ρυθμίσεις φίλτρων κειμένου.

- Στις ρυθμίσεις ιστότοπου μπορούμε να αλλάξουμε τις βασικές ρυθμίσεις του ιστότοπου αλλα και τις ρυθμίσεις βελτιστοποίησης των μηχανών αναζήτησης καθώς επίσης μπορούμε να κάνουμε ρυθμίσεις αρχείων ή να αλλάξουμε τα δεδομένα περιγραφής της ιστοσελίδας μας.
- Στις ρυθμίσεις συστήματος μπορούμε να έχουμε πρόσβαση σε ρυθμίσεις αποσφαλμάτωσης, ρυθμίσεις προσωρινής αποθήκευσης καθώς επίσης και ρυθμίσεις συνεδρίας.
- Στις ρυθμίσεις διακομιστή μπορούμε να ρυθμίσουμε την τοποθεσία των αρχείων στον διακομιστή, να κάνουμε ρυθμίσεις ftp όπως επίσης και ρυθμίσεις σχετικά με τις βάση δεδομένων οπου ο ιστότοπος μας είναι συνδεδεμένος, καθώς επίσης και ρυθμίσεις σχετικά με το ηλεκτρονικό ταυχδρομείο.
- Στις ρυθμίσεις δικαιωμάτων μπορούμε να διαχειριστούμε τα επίπεδα πρόσβασης για τις ομάδες χρηστών που έχουμε ήδη.
- Στις ρυθμίσεις φίλτρων κειμένου μπορούμε να φιλτράρουμε λέξεις και φράσεις που διάφοροι εγγεγραμένοι χρήστες μπορεί ενδεχομένως να χρησιμοποιήσουν στους κειμενογράφους,

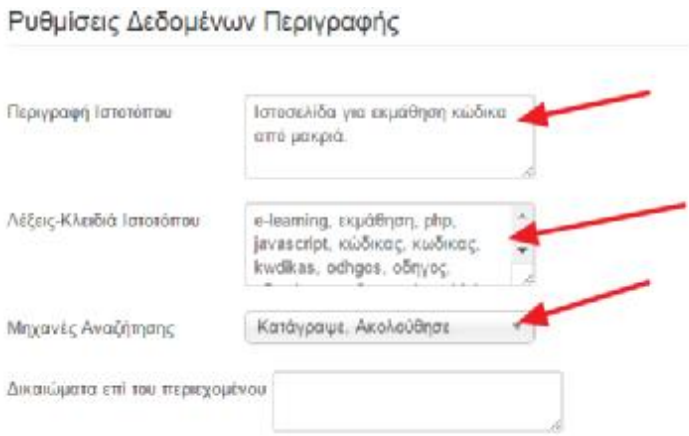
Για να αποκτήσουμε πρόσβαση στις γενικές ρυθμίσεις, κάνουμε είσοδο στο σύστημα διαχείρησης του ιστότοπου μας και επιλέγουμε την επιλογή «Γενικές ρυθμίσεις» από το μενού «Σύστημα»(Εικόνα 5.44).



Εικόνα 5.44 Μετάβαση στις γενικές ρυθμίσεις του ιστότοπου

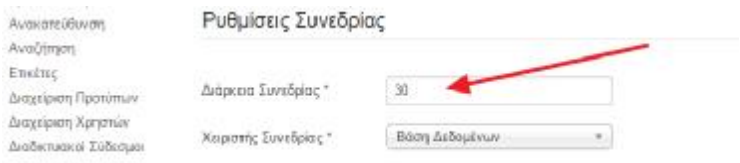
Από την οθόνη που μας φορτώνεται έχουμε πρόσβαση τόσο σε γενικές ρυθμίσεις όσο και σε ρυθμίσεις εφαρμογών που έχουμε εγκαταστημένες στον ιστότοπο μας. Οι γενικές ρυθμίσεις βρίσκονται ως καρτέλες στην κορυφή της οθόνης όπου επιλεγμένο

είναι το «Ιστότοπος». Σε αυτή την οθόνη υπάρχουν πολλες ρυθμίσεις που μπορούν να γίνουν εμείς θα επικεντρωθούμε στις «Ρυθμίσεις Δεδομένων Περιγραφής» οι οποίες βοηθούν τις μηχανές αναζήτησης να βρουν και να ανιχνεύσουν καλύτερα τον ιστότοπο μας. Εισάγουμε την περιγραφή ιστότοπου, λέξεια κλειδιά. Φροντίζουμε έτσι ώστε στην επιλογή «Μηχανές Αναζήτησης» να υπάρχει επιλεγμένο το «Κατέγραψε, Ακολούθησε», και ύστερα κάνουμε κλίκ στην καρτέλα «Σύστημα» (Εικόνα 4.45).



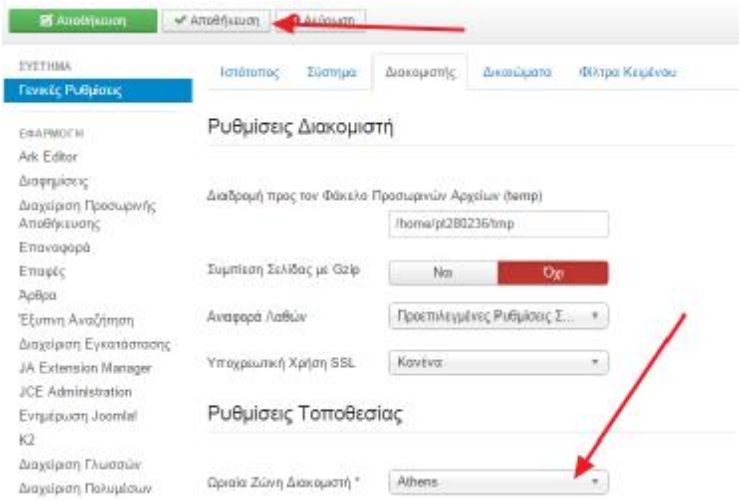
Εικόνα 5.45 Ρυθμίσεις SEO του ιστότοπου

Στην καρτέλα σύστημα θα επικεντρωθούμε στην διάρκεια της συνεδριάς, εισάγουμε ως τιμή στην επιλογή «Διάρκειας Συνεδρίας» το πλήθος των λεπτών της ώρας που θέλουμε να διαρκεία μια συνεδρία όταν ένας χρήστης εισάγεται στο σύστημα, αλλα είναι ενενεργός. Η προεπιλεγμένη τιμή είναι 15 λεπτά, εμείς διπλασιάσαμε αυτόν τον χρόνο για να γινει εφικτή η εισάγωγή όλων των οδηγών χωρίς να έχουμε συχνές εισόδους και εξόδους από το σύστημα. Ύστερα κάνουμε κλίκ στην καρτέλα «Διακομιστής» (Εικόνα 5.46).



Εικόνα 5.46 Ρυθμίσεις συνεδρίας ιστότοπου

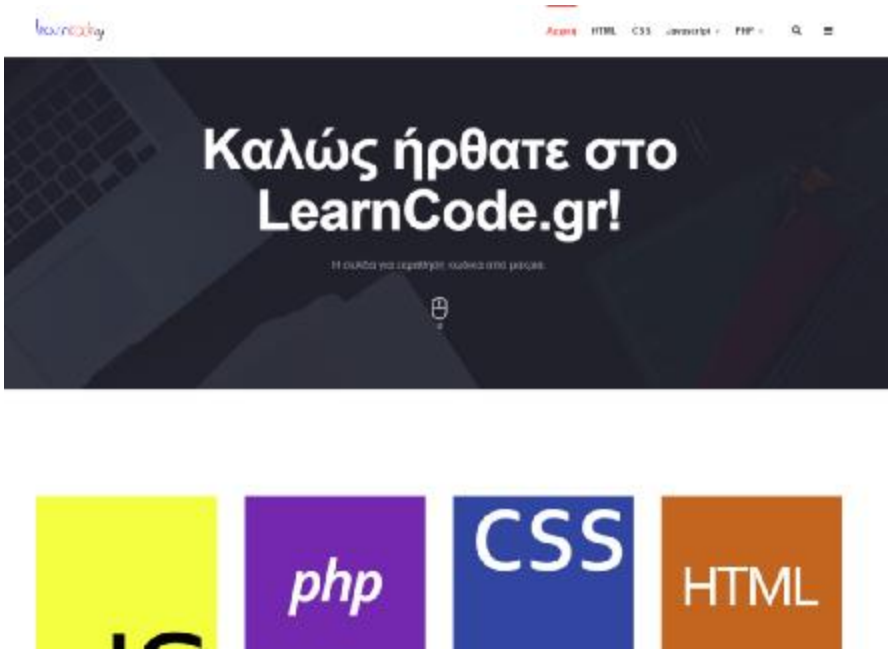
Από τις ρυθμίσεις διακομιστή μας ενδιαφέρει να αλλάξουμε την «Ωριαία Ζώνη Διακομιστή» στην περιοχή «Ρυθμίσεις Τοποθεσίας. Ύστερα κάνουμε κλίκ στο κουμπί «Αποθήκευση» (Εικόνα 5.47).



Εικόνα 5.47 Αλλαγή ωριαίας ζώνης διακομιστή

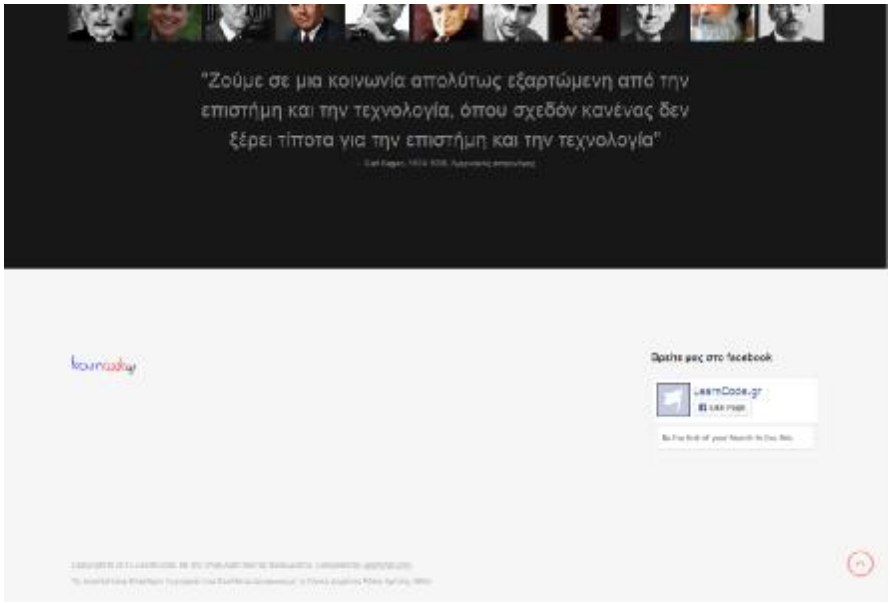
5.6 Απεικόνιση

Ο ιστότοπος μας πλέον έχει ολοκληρωθεί και μπορούμε να δούμε πως τον βλέπουν όσοι τον επισκέπτονται.



Εικόνα 5.48 Αρχική σελίδα (1)

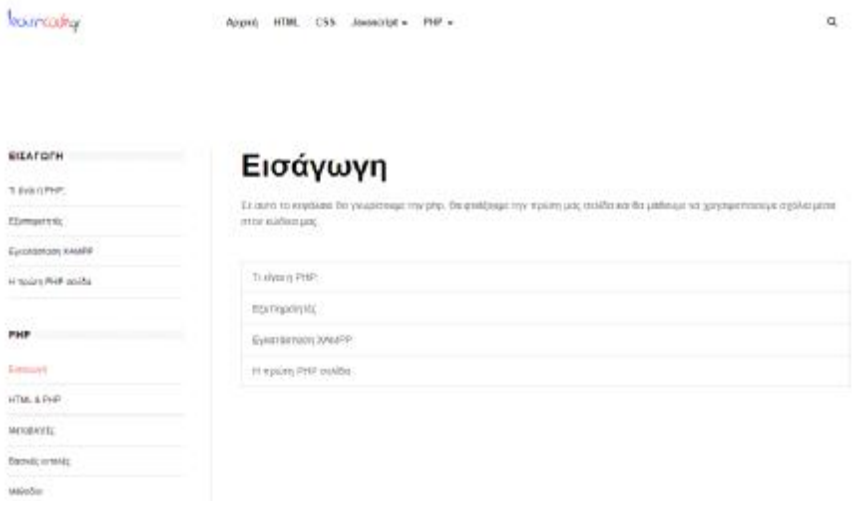
Εικόνα 5.49 Αρχική σελίδα (2)



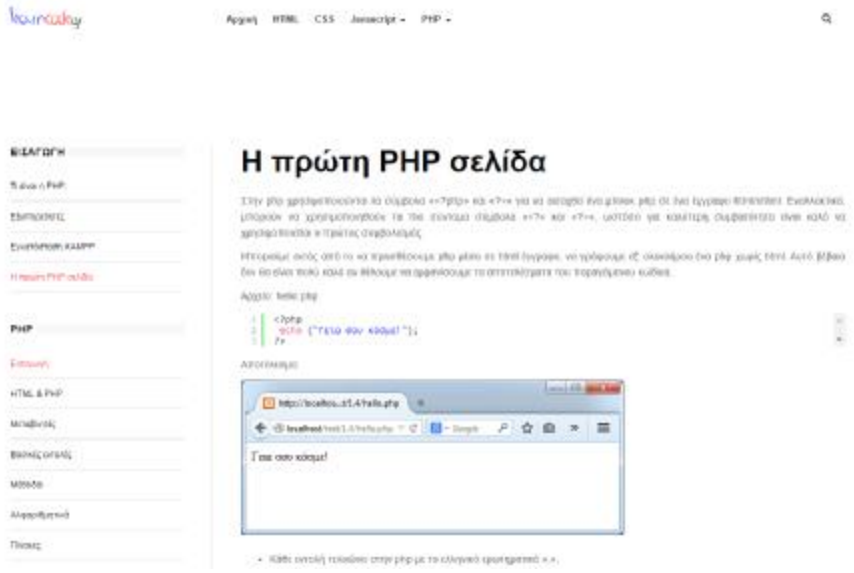
Εικόνα 5.50 Αρχική σελίδα (3)

Εικόνα 5.51 Βασικό μενού περιήγησης

Εικόνα 5.52 Σελίδα εισαγωγής της php



Εικόνα 5.53 Προβολή κεφαλαίου «Εισαγωγή» της php



Εικόνα 5.54 Προβολή σελίδας οδηγού εκμάθησης

Εικόνα 5.55 Σελίδα εισαγωγής για την Javascript

6 ΣΥΜΠΕΡΑΣΜΑΤΑ

Η ιστοσελίδα που προέκυψε από την εκπόνηση της πτυχιακής εργασίας, απευθύνεται σε άτομα που ενδιαφέρονται για αυτοδίδακτη γνώση πάνω στον προγραμματισμό. Οι επισκέπτες έχουν την δυνατότητα να παρακολουθούν οδηγούς βήμα προς βήμα, μέσω τον διαθέσιμων άρθρων που βρίσκουν στην ιστοσελίδα για κάθε διαφορετική θεματική ενότητα που τους ενδιαφέρει.

Η ενασχόληση μου με το Joomla ήταν κάτι πρωτόγνωρο. Αφιέρωσα κάποιο χρόνο ώστε να μάθω τις βασικότερες έννοιες ώστε να μπορέσω να ξεκινήσω την δημιουργία της ιστοσελίδας. Έτσι παράλληλα με την δημιουργία της ιστοσελίδας εξοικειώθηκα και στην χρήση του Joomla εμπλουτίζοντας τις γνώσεις μου.

Κατάληξη μου είναι ότι με τα συστήματα διαχείρισης περιεχομένου, μας παρέχονται αρκετές δυνατότητες ώστε να μπορέσουμε να δημιουργήσουμε και να διαμορφώσουμε εύχρηστες και όμορφες ιστοσελίδες. Ιδιαίτερα με κάποιες επιπλέον γνώσεις προγραμματισμού, μπορούμε να κάνουμε τις δικές μας επεμβάσεις και να δώσουμε κάποια χαρακτηριστικά που μπορούν να βελτιώσουν σε διάφορους τομείς την ιστοσελίδα μας. Ωστόσο υπάρχουν πληθώρα επιλογών για το πώς μπορούμε να παρουσιάσουμε το υλικό που διαθέτουμε. Στο μέλλον θα μπορούσαν όλοι οι οδηγοί

που γράφτηκαν σε αυτή την εργασία να εξελιχθούν μεταβάλλοντας το περιεχόμενο και δέχοντας το αντίστοιχο feedback από τους επισκέπτες της ιστοσελίδας. Επίσης θα μπορούσαν να παρέχονται μαζί με το άρθρο και κάποιο βίντεο παρουσίασης της διαδικασίας βήμα – βήμα για την ολοκλήρωση της κάθε ενότητας, παρέχοντας έτσι όχι μόνο εξοικείωση στην αντίστοιχη γλώσσα αλλά και στην χρήση των κατάλληλων εργαλείων σχετικά με αυτήν.

7 ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Διαδίκτυο:
   http://el.wikipedia.org/wiki/Διαδίκτυο

2. Ιστοσελίδα:
   http://el.wikipedia.org/wiki/Ιστοσελίδα

3. Στατική ιστοσελίδα:
   http://el.wikipedia.org/wiki/Στατική_ιστοσελίδα

4. Δυναμική ιστοσελίδα:
   http://el.wikipedia.org/wiki/Δυναμική_ιστοσελίδα

5. Όνομα τομέα:
   http://el.wikipedia.org/wiki/Όνομα_τομέα

6. Φιλοξενία ιστοσελίδων:
   http://el.wikipedia.org/wiki/Φιλοξενία_ιστοσελίδων

7. Εξυπηρετητής ιστού:
   http://el.wikipedia.org/wiki/Εξυπηρετητής

8. Φυλλομετρητής:
   http://el.wikipedia.org/wiki/Web_browser

9. Συστήματα διαχείρησης περιεχομένου:
   http://www.kepa.gov.cy/diktiothite/Portal/PortalDocuments.aspx?DocumentId=a5d27b95-5b46-4a56-a535-0c2324141d42

10. Είδη συστημάτων διαχείρησης περιεχομένου:
    http://pacific.jour.auth.gr/content_management_systems/eidi.htm

11. Δημοφιλέστερα CMS:
http://www.cnctech.gr/blog/joomla-vs-wordpress-vs-drupal

12. Apache HTTP Server:
http://priwac.com/apache-web-server/

13. PHP:
https://el.wikipedia.org/wiki/PHP

14. PhpMyAdmin:
https://en.wikipedia.org/wiki/PhpMyAdmin

15. MySQL:
https://el.wikipedia.org/wiki/MySQL

16. HTML:
https://el.wikipedia.org/wiki/HTML

17. CSS:
https://el.wikipedia.org/wiki/CSS

18. Javascript:
https://el.wikipedia.org/wiki/JavaScript

19. XAMPP:
https://el.wikipedia.org/wiki/XAMPP

20. Zend Eclipse Luna:
https://www.zend.com/en/community/pdt

21. Gimp:
https://el.wikipedia.org/wiki/GIMP

22. Mozilla Firefox:
https://el.wikipedia.org/wiki/Mozilla_Firefox

23. Εγκατάσταση Firefox:
https://support.mozilla.org/en-US/kb/how-download-and-install-firefox-windows

24. Εγκατάσταση xampp:
http://www.wikihow.com/Install-XAMPP-for-Windows

25. Εγκατάσταση eclipse:
http://www.wikihow.com/Download,-Install,-and-Run-JDK-and-Eclipse

26. Ρύθμιση xampp:
https://www.youtube.com/watch?v=YlidpnKF5KE

27. Δημιουργία βάσης δεδομένων:
https://www.siteground.com/tutorials/phpmyadmin/phpmyadmin_create_database.htm

28. Εγκατάσταση joomla:
https://docs.joomla.org/J3.x:Installing_Joomla

29. Οδηγός εκμάθησης Javascript:
http://www.w3schools.com/js/
David Flanagan, 2011, JavaScript The Definitive Guide Sixth Edition, Εκδόσεις O'Reilly Media Inc , ΗΠΑ.

30. Οδηγός εκμάθησης php:
http://www.w3schools.com/php/
David sklar και Adam Trachtenberg, 2014, PHP Cookbook Third Edition, Εκδόσεις O'Reilly Media Inc, ΗΠΑ.

Παράρτημα Α:PHP

PHP

Οδηγός εκμάθησης

ΣΠΟΥΔΑΣΤΗΣ:Κωνσταντίνος Κολέτσος

Επιβλέπων καθηγητής: Κωνσταντίνος Γιωτόπουλος

Πάτρα – Μάιος 2015

Α.1 ΕΙΣΑΓΩΓΗ

Σε αυτό το κεφάλαιο θα γνωρίσουμε την php, θα φτιάξουμε την πρώτη μας σελίδα και θα μάθουμε να χρησιμοποιούμε σχόλια μέσα στον κώδικα μας.

Α.1.1 Τι είναι η php

Τα αρχικά PHP προέρχονται από τις λέξεις «Personal Home Page». Η php είναι μια γλώσσα προγραμματισμού που σχεδιάστηκε για την δημιουργία δυναμικών ιστοσελίδων και είναι γνωστή ως Hypertext PreProcessor.

Σε αντίθεση με άλλες γλώσσες scripting του διαδικτύου όπως η html η css , κ.α. η γλώσσα php είναι μια server-side (εκτελείται στον εξυπηρετητή) scripting γλώσσα που συνήθως γράφεται πλαισιωμένη από HTML για την εμφάνιση των αποτελεσμάτων. Έτσι η php δεν στέλνεται άμεσα σε έναν πελάτη (client), αντ' αυτού πρώτα αναλύεται και μετά αποστέλλεται το παραγόμενο αποτέλεσμα (έτσι προέκυψε και η ευρέως γνωστή ονομασία της σε HyperText preprocessor). Η php είναι διαδεδομένη για την πληθώρα δυνατοτήτων που μπορεί να προσφέρει σε διαδικτυακές εφαρμογές, όπως να θέσει ερωτήματα σε βάσεις δεδομένων, να δημιουργήσει εικόνες, να διαβάσει και να γράψει αρχεία, να συνδεθεί σε απομακρυσμένους υπολογιστές, κ.α.

Η php δημιουργήθηκε από τον φοιτητή Rasmus Lerdorf ως μια συλλογή από scripts γραμμένα στην γλώσσα προγραμματισμού perl που τα χρησιμοποιούσε στην προσωπική του ιστοσελίδα. Η αρχική χρήση της php από τον Rasmus ήταν η παρακολούθηση στατιστικών στοιχείων που αφορούσαν την επισκεψιμότητα στο προσωπικό του βιογραφικό. Αργότερα έγραψε ξανά τα scripts σε γλώσσα C, για λόγους καλύτερης απόδοσης, επεκτείνοντας ταυτόχρονα τις δυνατότητες της υποστηρίζοντας έτσι την χρήση διαδικτυακών forms και σύνδεση με βάσεις δεδομένων. Το πρώτο επίσημο όνομα της php ήταν PHP/FI από τα «Personal Home Page/Forms Interpreter» (Προσωπική Ιστοσελίδα / Διερμηνέας Φορμών). Μετά από αυτή την δημιουργία ο Rasmus διέθεσε τον κώδικα στην ιστοσελίδα του ώστε να επωφεληθούν κι άλλοι από αυτόν.

Α.1.2 Εξυπηρετητές

Η γλώσσα php εκτελείται στον εξυπηρετητή και ύστερα το παραγόμενο αποτέλεσμα αποστέλλεται στον πελάτη (client). Προκειμένου να γίνει κατανοητή αυτή την λειτουργία της php θα πρέπει πρώτα να γνωρίζουμε την σχέση μεταξύ client και server (πελάτη και εξυπηρετητή).

Ένα όμορφο σαββατιάτικο πρωινό ο Γιάννης θέλοντας να αγοράσει μια καινούργια τηλεόραση για το σπίτι του, αναζητά ένα κατάστημα το οποίο θα τον εξυπηρετήσει. Ψάχνοντας, καταλήγει σε κάποιο στο οποίο θα αγοράσει την τηλεόραση που του αρέσει. Έτσι ζητά από τους υπάλληλους του καταστήματος να τον εξυπηρετήσουν καθώς ξέρει τι ψάχνει και σαφώς θέλει να ξοδέψει λιγότερο χρόνο. Λέει σε έναν υπάλληλο να του δείξει όλες τις τηλεοράσεις με είσοδο HDMI, με ψηφιακό δέκτη και με θύρα USB. Ως πελάτης ο Γιάννης κάνει ένα αίτημα προς τον εξυπηρετητή (τον υπάλληλο δηλαδή) να του δείξει όλες τις τηλεοράσεις με τα χαρακτηριστικά που επιθυμεί.

Έτσι λειτουργεί το μοντέλο πελάτης - εξυπηρετητής στο διαδίκτυο. Τόσο ο πελάτης όσο και ο εξυπηρετητής διαθέτουν λογισμικά τα οποία μπορούν να αποστείλουν και να λάβουν δεδομένα. Ένας πελάτης μπορεί να ζητήσει από έναν εξυπηρετητή έναν πόρο, τα αποτέλεσμα ενός υπολογισμού κ.ο.κ. Ο εξυπηρετητής αναλαμβάνει να φέρει εις πέρας αυτό αίτημα κάνοντας τις απαραίτητες ενέργειες και δίνοντας το αποτέλεσμα στον πελάτη. Προκειμένου να λειτουργήσει αυτό το μοντέλο θα πρέπει ο πελάτης και ο εξυπηρετητής να διαθέτουν επικοινωνία μεταξύ τους (είτε ενσύρματη είτε ασύρματη).

Στην γλώσσα php ένας πελάτης συνήθως χρησιμοποιεί έναν φυλλομετρήτη (η πλοηγό , browser) διαδικτύου. Γράφοντας μια διεύθυνση στην γραμμή διευθύνσεων του εκάστοτε φυλλομετρητή, ο εξυπηρετητής που αντιστοιχείται σε αυτή την διεύθυνση λαμβάνει το αίτημα της εμφάνισης της σελίδας από τον πελάτη, και στέλνει τα δεδομένα σε αυτόν.

Ο εξυπηρετητής για να το καταφέρει αυτό διαθέτει κάποιο ειδικό λογισμικό, συνήθως έναν apache server, τον διερμηνέα της php, και μια βάση δεδομένων (συνήθως MYSQL).

- Ο apache αναλαμβάνει την διαδικασία δημοσιοποίησης στο διαδίκτυο λαμβάνοντας και στέλνοντας τα αιτήματα - δεδομένα, υποστηρίζοντας δηλαδή το πρωτόκολλο http.
- Ο διερμηνέας php αναλαμβάνει να «τρέξει» την στιγμή της αίτησης τα script που αφορούν την σελίδα που διαθέτει ο εξυπηρετητής, και να δημιουργήσει το παραγόμενο αποτέλεσμα, βασιζόμενος πάντα στα εισαγόμενα στοιχεία από την apache που έλαβε ως αίτημα.
- Η βάση δεδομένων (MYSQL) αναλαμβάνει την αποθήκευση δεδομένων που αφορούν συνήθως τους πελάτες, ή αλληλεπιδραστικών οντοτήτων που έχουν άμεση σχέση με αυτούς, πχ λογαριασμοί χρηστών, προϊόντα, προμηθευτές κ.ο.κ. Λαμβάνοντας ένα αίτημα από την php, η mysql επιστρέφει τα αντίστοιχα δεδομένα ξανά στην php και ύστερα παράγεται το αποτέλεσμα σε HTML το οποίο θα σταλεί μέσω του apache στον πελάτη.

A.1.3 Εγκατάσταση και χρήση XAMPP

Στην παρούσα ενότητα θα εγκαταστήσουμε από το διαδίκτυο και θα χρησιμοποιήσουμε το XAMPP. Τα αρχικά προέρχονται από τα Cross(X), Apache, MySQL, PHP, Perl. Το «X» προέρχεται από το cross platform το οποίο σημαίνει ότι μπορεί να τρέξει σε οποιαδήποτε πλατφόρμα.

Βήμα πρώτο

Αρχικά επισκεπτόμαστε την σελίδα «https://www.apachefriends.org» η οποία περιλαμβάνει συνδέσμους για να μπορέσουμε να κατεβάσουμε την νεότερη έκδοση του XAMPP. Κάνουμε κλικ όπου αναφέρεται το XAMPP για το λειτουργικό σύστημα που διαθέτουμε. Εμείς κάναμε κλικ στο «XAMPP for Windows».

Εικόνα Α.1 Επιλογή έκδοσης XAMPP

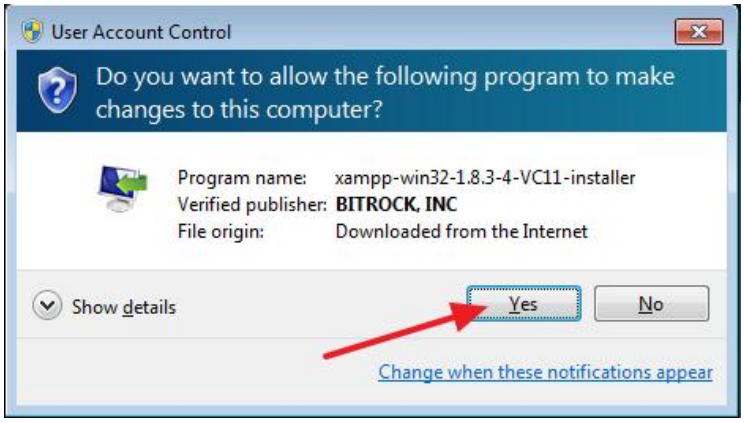
Βήμα δεύτερο

Μας εμφανίζεται η οθόνη επιβεβαίωσης της λήψης του αρχείου εγκατάστασης. Περιμένουμε να ολοκληρωθεί η λήψη.

Εικόνα Α.2 Μήνυμα επιβεβαίωσης λήψης

Βήμα τρίτο

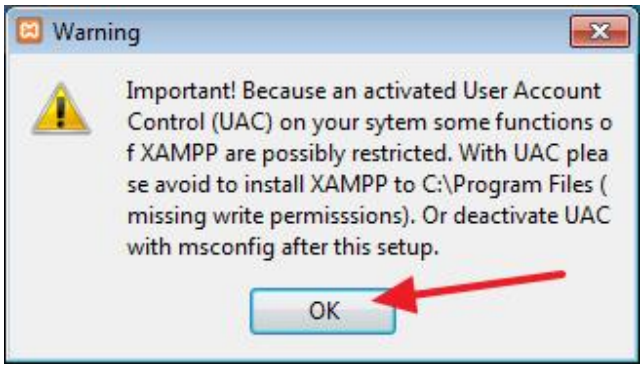
Αφού η μεταφόρτωση του αρχείου εγκατάστασης ολοκληρωθεί, κάνουμε διπλό κλικ επάνω του έτσι ώστε να το εκτελέσουμε. Μας εμφανίζεται το παρακάτω παράθυρο ασφαλείας σχετικά για την εκτέλεση του αρχείου. Κάνουμε κλικ στο «Yes».



Εικόνα Α.3 Αποδοχή ασφάλειας UAC

Βήμα τέταρτο

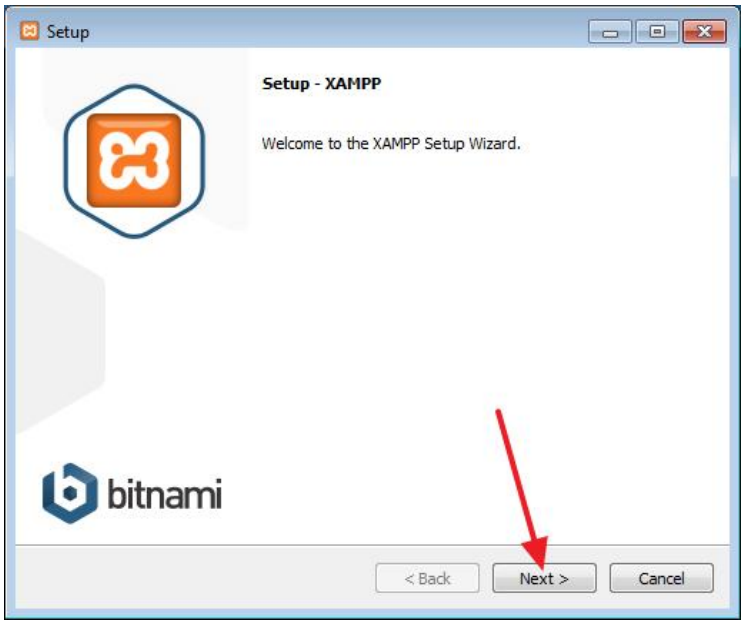
Εμφανίζεται επίσης ένα μικρό ενημερωτικό παράθυρο σχετικά με την εμπλοκή του UAC σε μερικές από τις λειτουργίες του XAMPP, έτσι μερικές από αυτές μπορεί να λειτουργούν περιορισμένα. Κάνουμε κλικ στο «OK».



Εικόνα Α.4 Ενημέρωση σχετικά με την λειτουργικότητα του XAMPP

Βήμα πέμπτο

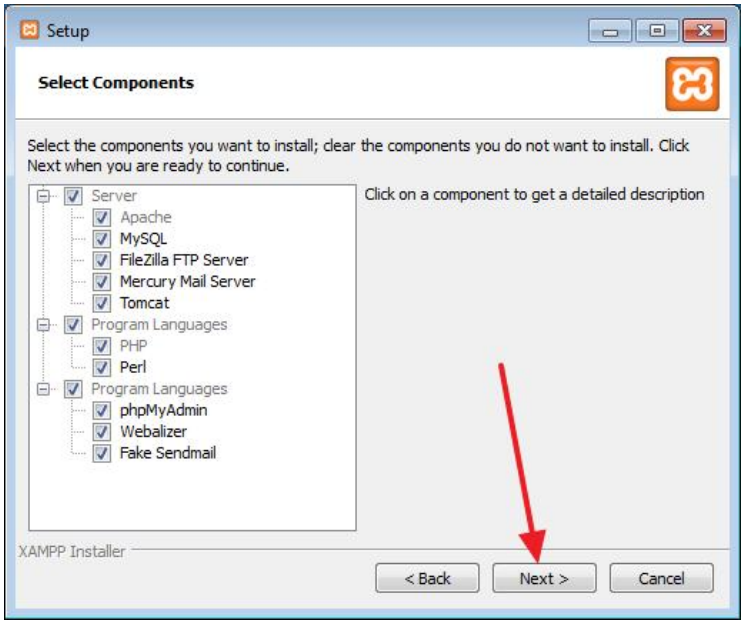
Η πρώτη οθόνη του οδηγού εγκατάστασης εμφανίζεται, κάνουμε κλικ στο «Next».



Εικόνα Α.5 Αρχική οθόνη οδηγού εγκατάστασης XAMPP

Βήμα έκτο

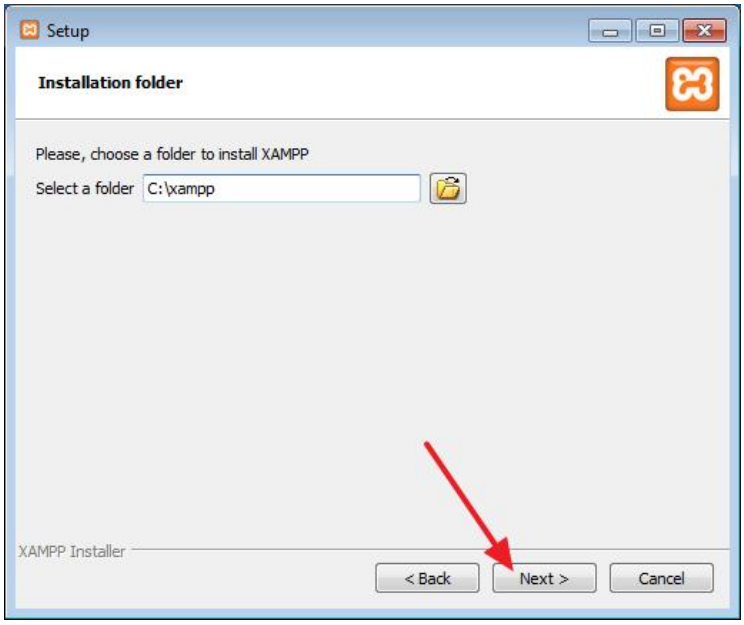
Στην παρούσα οθόνη μας εμφανίζονται όλες οι υπηρεσίες που θα εγκατασταθούν. Αφήνουμε τα προεπιλεγμένα επιλεγμένα και ύστερα κάνουμε κλικ στο «Next».



Εικόνα Α.6 Υπηρεσίες XAMPP προς εγκατάσταση

Βήμα έβδομο

Σε αυτή την οθόνη επιλέγουμε την διαδρομή εγκατάστασης του xampp στον σκληρό μας δίσκο. Το αφήνουμε όπως έχει και κάνουμε κλικ στο «Next»



Εικόνα Α.7 Κατάλογος εγκατάστασης XAMPP

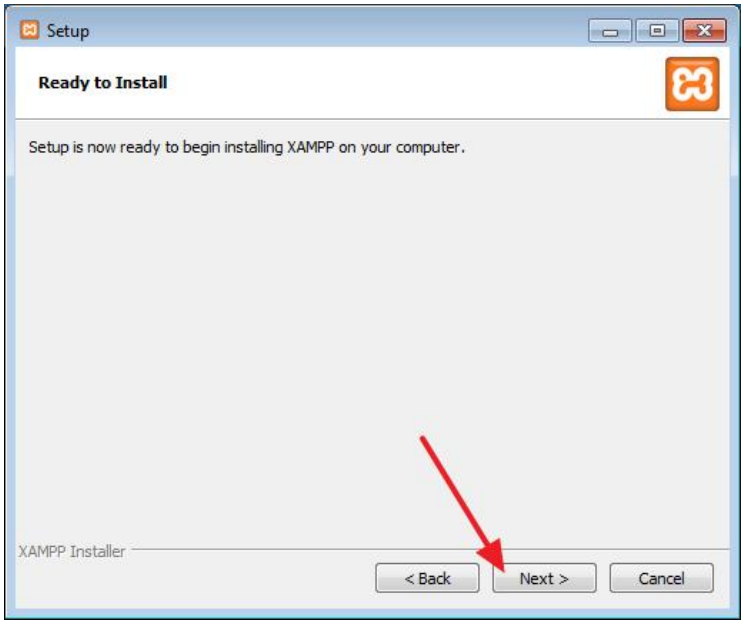
Βήμα όγδοο

Μια ενημερωτική οθόνη μας ενημερώνει για το Bitnami, αφαιρούμε το τικ και κάνουμε κλικ στο κουμπί «Next».

Εικόνα Α.8 Οθόνη ενημέρωσης για το Bitnami

Βήμα ένατο

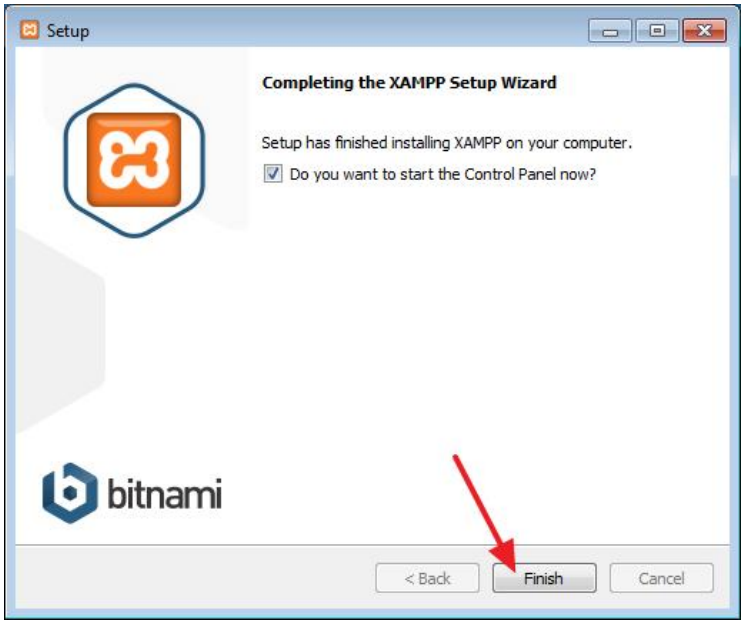
Ενημερωνόμαστε για την ετοιμότητα του οδηγού να εγκαταστήσει το XAMPP στο σύστημα μας. Κάνουμε κλικ στο «Next».



Εικόνα Α.9 Οδηγός έτοιμος για εγκατάσταση

Βήμα δέκατο

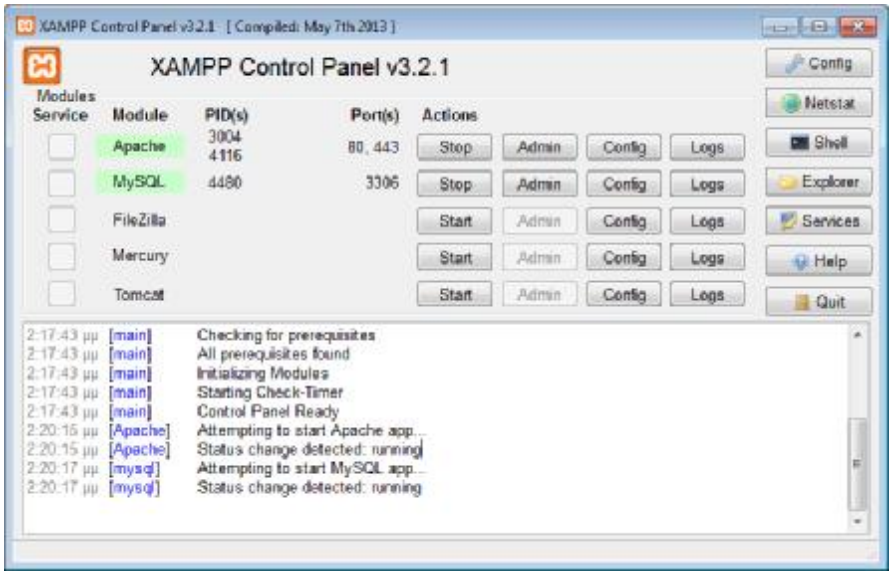
Περιμένουμε να ολοκληρωθεί η διαδικασία εγκατάστασης. Μόλις εμφανιστεί το παράθυρο ολοκλήρωσης κάνουμε κλικ στο «Finish».



Εικόνα Α.10 Ολοκλήρωση διαδικασίας εγκατάστασης

Βήμα ενδέκατο

Μας ανοίγει ο πίνακας ελέγχου του XAMPP. Παρατηρούμε πως για κάθε υπηρεσία υπάρχουν αντίστοιχα κουμπιά «Start», «Config», «Logs». Τα κουμπιά «Start» χρησιμοποιούνται έτσι ώστε να ξεκινήσουμε τις αντίστοιχες υπηρεσίες. Αυτές στις οποίες εστιάζουμε σε αυτούς τους οδηγούς είναι η Apache και η MySQL. Αν κάνουμε κλικ στα κουμπιά Start που αφορούν μόνο τις συγκεκριμένες υπηρεσίες τότε το όνομα τους παίρνει ένα πράσινο φόντο το οποίο σημαίνει ότι οι υπηρεσίες ξεκίνησαν επιτυχώς.



Εικόνα Α.11 Πίνακας ελέγχου XAMPP

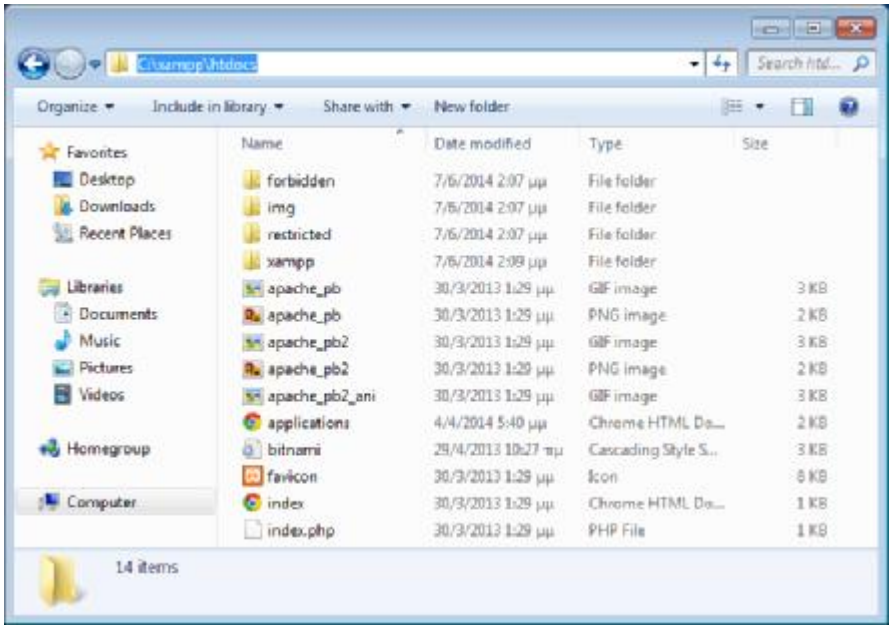
Βήμα δωδέκατο

Για επιβεβαίωση επισκεπτόμαστε την διεύθυνση «http://localhost/», και μας φορτώνεται η αρχική σελίδα του εξυπηρετητή μας με το σήμα του XAMPP.

Εικόνα Α.12 Αρχική οθόνη τοπικού εξυπηρετητή XAMPP

Βήμα δέκατο τρίτο

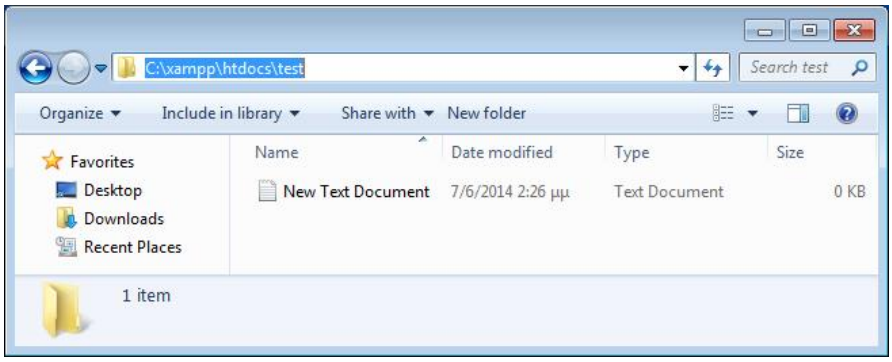
Για να δημιουργήσουμε αρχεία php θα πρέπει να τα τοποθετήσουμε στον κατάλληλο κατάλογο στο σημείο όπου εγκαταστάθηκε το XAMPP. Ο προεπιλεγμένος κατάλογος για να τοποθετήσουμε τα αρχεία php μας είναι ο «C:\xampp\htdocs». Αν εδώ τοποθετήσουμε το οτιδήποτε μέσα σε αυτόν τον κατάλογο θα είναι προσπελάσιμο από τον φυλλομετρητή μας.



Εικόνα Α.13 Δημόσιο κατάλογος του XAMPP

Βήμα δέκατο τέταρτο

Δημιουργούμε για επιβεβαίωση έναν φάκελο test με περιεχόμενο του ένα αρχείο κειμένου.



Εικόνα Α.14 Δημιουργία δοκιμαστικού αρχείου

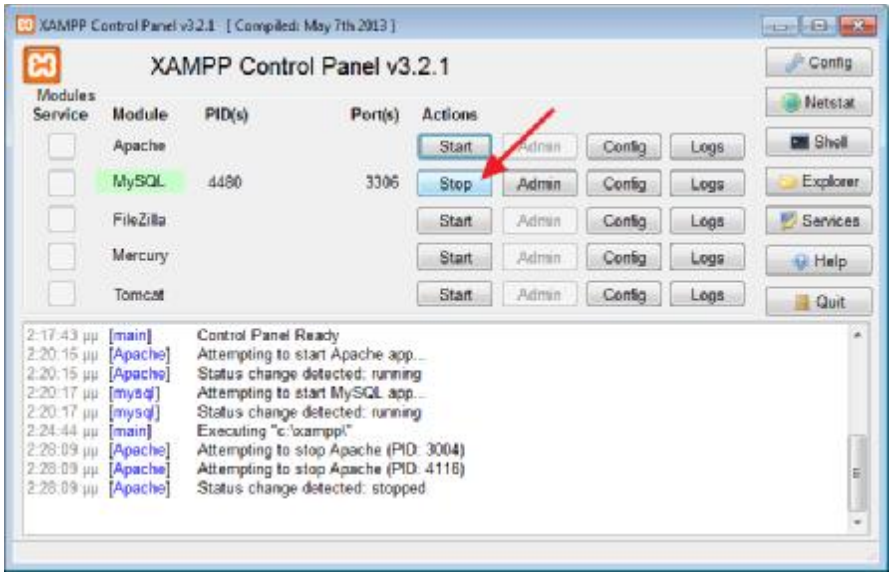
Βήμα δέκατο πέμπτο

Επισκεπτόμαστε την τοποθεσία «http://localhost/test/» από τον φυλλομετρητή μας και βλέπουμε τα περιεχόμενα του φακέλου. Ο εξυπηρετητής μας λειτουργεί κανονικά!

Εικόνα Α.15 Επίσκεψη καταλόγου μέσω φυλλομετρητή

Βήμα δέκατο έκτο

Κάθε φορά που δεν χρειαζόμαστε τις υπηρεσίες του XAMPP κάνουμε κλικ στο «Stop» που υπάρχει δίπλα από το όνομα τους ώστε να τερματίζονται. Να έχετε υπόψη σας πως ο apache χρησιμοποιεί την πόρτα 80 για να επικοινωνεί με τους φυλλομετρητές. Υπάρχουν πολλές εφαρμογές, όπως το skype, οι οποίες επίσης χρησιμοποιούν την πόρτα 80 για τον ίδιο λόγο. Για να μπορέσουν να ξεκινήσουν οι υπηρεσίες θα πρέπει τέτοιες εφαρμογές να μην τρέχουν στον υπολογιστή μας την στιγμή που θέλουμε να τις εκκινήσουμε, διαφορετικά θα εμφανιστεί σφάλμα εκκίνησης της υπηρεσίας , καθώς οι πόρτες δεσμεύονται από τις εφαρμογές που εκτελούνται στο σύστημα μας.



Εικόνα Α.16 Παύση λειτουργίας υπηρεσιών που δεν χρειάζονται

A.1.4 Η πρώτη PHP σελίδα

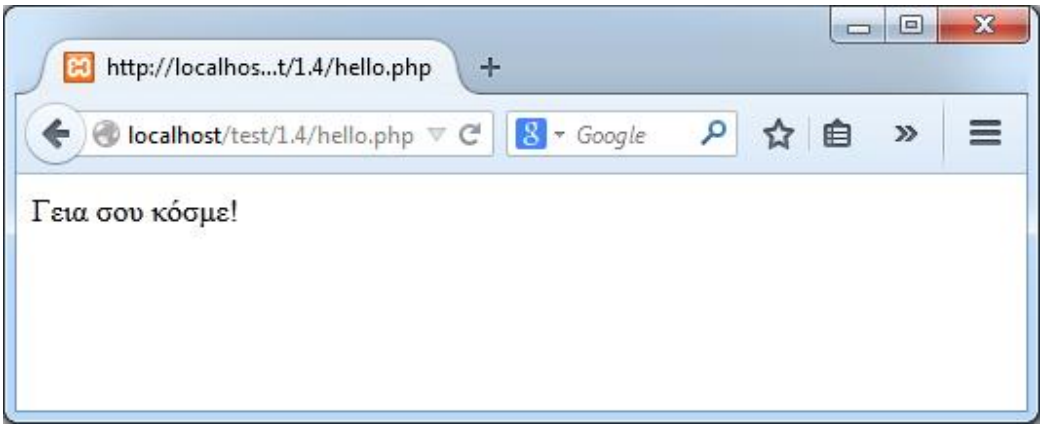
Στην php χρησιμοποιούνται τα σύμβολα «<?php» και «?>» για να εισαχθεί ένα μπλοκ php σε ένα έγγραφο html/xhtml. Εναλλακτικά, μπορούν να χρησιμοποιηθούν τα πιο σύντομα σύμβολα «<?» και «?>», ωστόσο για καλύτερη συμβατότητα είναι καλό να χρησιμοποιείται ο πρώτος συμβολισμός.

Μπορούμε εκτός από το να προσθέσουμε php μέσα σε html έγγραφο, να γράψουμε εξ' ολοκλήρου ένα php χωρίς html. Αυτό βέβαια δεν θα είναι πολύ καλό αν θέλουμε να εμφανίσουμε τα αποτελέσματα του παραγόμενου κώδικα.

Αρχείο: hello.php

```
<?php
 echo ("Γεια σου κόσμε!");
?>
```

Αποτέλεσμα:



Εικόνα Α.17 Η πρώτη php σελίδα!

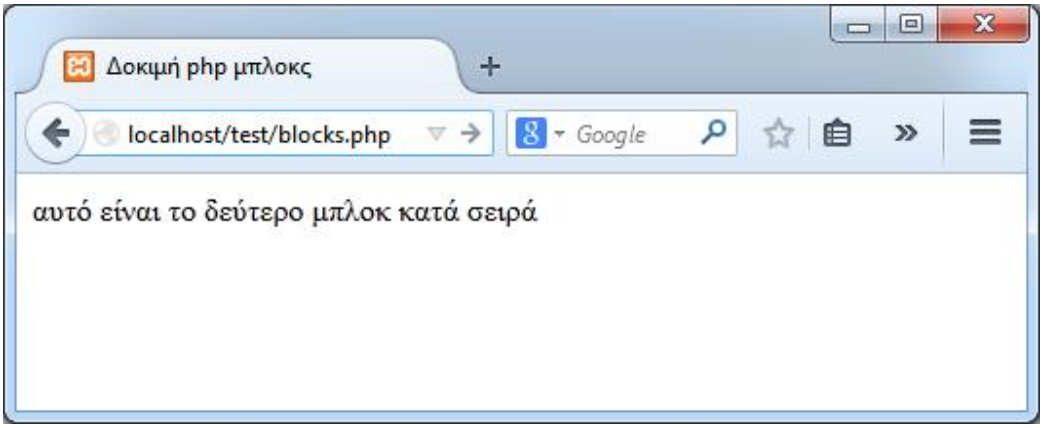
- Κάθε εντολή τελειώνει στην php με το ελληνικό ερωτηματικό «;»,
- Έχει μεγάλη σημασία για το αν τα γράμματα που χρησιμοποιούνται στον κώδικα (όχι στα αλφαριθμητικά) είναι κεφαλαία ή μικρά, επομένως η php είναι case sensitive. Άρα το «echo» δεν είναι ίδιο με το «Echo».

Ένα έγγραφο είτε είναι php είτε είναι html μπορεί αν ενσωματώνει πολλά περισσότερα από ένα μπλοκ php. Για παράδειγμα, σε ένα έγγραφο html θα μπορούσαμε να χρησιμοποιήσουμε το μπλοκ php για όσες φορές θέλουμε:

Αρχείο: blocks.php

```
<html>
 <head>
  <title>
   <?php echo "Δοκιμή php μπλοκς"; ?>
  </title>
 </head>
 <body>
  <?php
   echo "αυτό είναι το δεύτερο μπλοκ κατά σειρά";
  ?>
 </body>
</html>
```

Αποτέλεσμα:



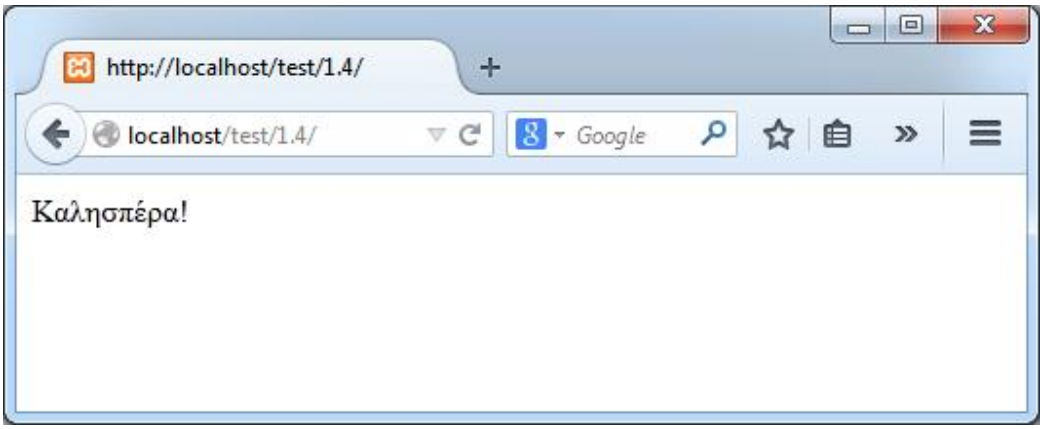
Εικόνα Α.18 Πολλαπλά blocks σε ένα έγγραφο php

Στην php χρησιμοποιούνται σχόλια προκειμένου κατά την διάρκεια ανάπτυξης του κώδικα να μπορεί να αποδοθεί και η πρέπουσα σημασία σε αυτόν. Όταν κατά την διάρκεια της εκτέλεσης του script ο διερμηνέας φτάσει σε ένα σχόλιο, δεν θα το διερμηνεύσει αλλά θα το προσπεράσει και θα συνεχίσει στην επόμενη γραμμή script που δεν είναι σχόλιο. Υπάρχουν δυο είδη σχολίων στην php, της μίας γραμμής και των πολλών γραμμών. Για να εισάγουμε ένα σχόλιο μιας γραμμής χρησιμοποιούμε δυο καθέτους «//», για να εισάγουμε ένα σχόλιο πολλών γραμμών χρησιμοποιούμε ως άνοιγμα «/*» και ως κλείσιμο «*/», πχ

Αρχείο: index.php

```php
<?php
 echo "Καλησπέρα!";
 // Αυτό είναι ένα σχόλιο μίας γραμμής

 /* Αυτό είναι ένα
  σχόλιο πολλών γραμμών
 */
?>
```

Αποτέλεσμα:



Εικόνα Α.19 Χρήση σχόλιων μέσα σε κώδικα php

Α.2 HTML & PHP

Σε αυτό το κεφάλαιο μας δίνεται η δυνατότητα να κατανοήσουμε την χρησιμότητα της html στις σελίδες της php, θα μάθουμε να χρησιμοποιούμε μερικές από τις πιο βασικές ετικέτες της, και θα κάνουμε εισαγωγή αλλά και εξαγωγή δεδομένων.

Α.2.1 Η χρησιμότητα της HTML σε συνδυασμό με την php

Η html είναι πολύ σημαντική στην php, διότι χρησιμοποιείται τόσο στην «είσοδο» των δεδομένων προς αυτή, καθώς και στην «έξοδο» από αυτήν. Βρίσκεται σε κάθε περίπτωση, ένα βήμα πριν και ένα βήμα μπροστά από την εκτέλεση της php.

Η html είναι όλα αυτά που ο χρήστης μπορεί να αλληλεπιδράσει με αυτά. Αυτά που βλέπει και αυτά που χρησιμοποιεί (Πχ ένα πεδίο κειμένου, ένα κουμπί, μια ετικέτα). Τα αποτελέσματα μιας εκτέλεσης php που δεν διαθέτει html μέσα της θα ήταν πολύ απλοποιημένα και ίσως δύσκολα στην ανάγνωση. Η html αναλαμβάνει να «μορφοποιήσει» σε επίπεδο δομής τα προβαλλόμενα στοιχεία, έτσι ώστε κάτι να είναι

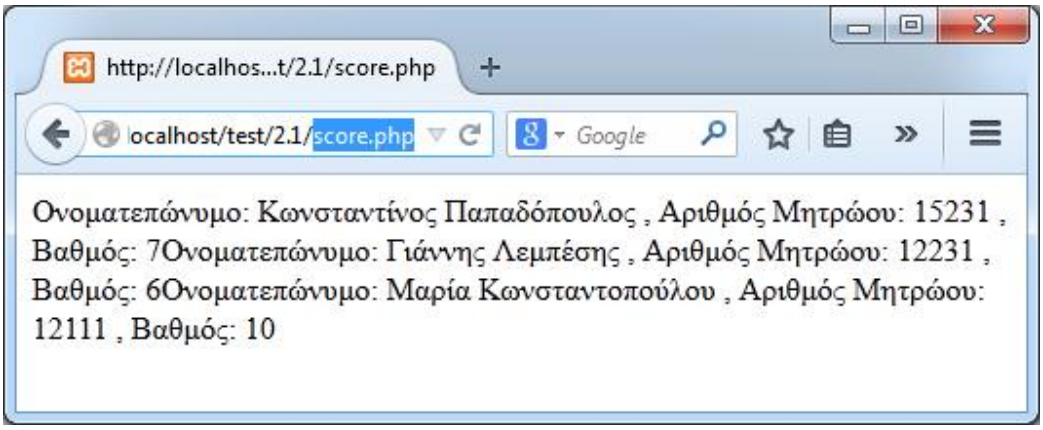
πολύ πιο προσιτό και πολύ πιο ευανάγνωστο στον απλό χρήστη που θα επισκεφθεί την σελίδα μας.

Φανταστείτε την εμφάνιση στοιχείων που έχουν να κάνουν με την βαθμολόγηση φοιτητών σε ένα πανεπιστήμιο:

Αρχείο: score.php

```
<?php
    echo "Ονοματεπώνυμο: Κωνσταντίνος Παπαδόπουλος , Αριθμός Μητρώου:
15231 , Βαθμός: 7";
    echo "Ονοματεπώνυμο: Γιάννης Λεμπέσης , Αριθμός Μητρώου: 12231 ,
Βαθμός: 6";
    echo "Ονοματεπώνυμο: Μαρία Κωνσταντοπούλου , Αριθμός Μητρώου:
12111 , Βαθμός: 10";
?>
```

Αποτέλεσμα:



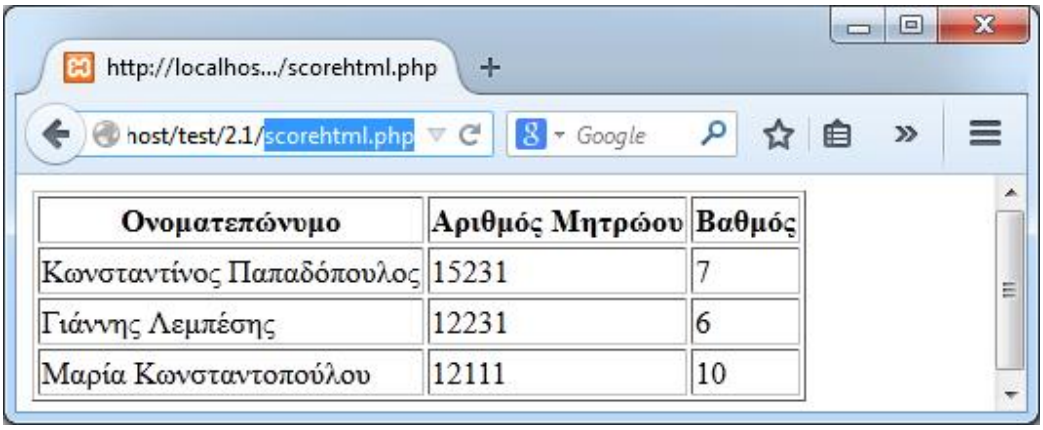
Εικόνα Α.20 Τύπωση αποτελεσμάτων χωρίς χρήση html

Τα αποτελέσματα της εμφάνισης, αυτών που έχουμε γράψει, χάνονται μέσα στους πολλούς χαρακτήρες, χωρίς διαχωριστικά μέρη και χωρίς κάτι διακριτό. Είναι δύσκολο σε κάποιον να βγάλει νόημα, γι' αυτό χρησιμοποιούμε την html (πολλές φορές σε συνδυασμό με την css). Ας δούμε το προηγούμενο παράδειγμα χρησιμοποιώντας html και συγκεκριμένα την δυνατότητα των πινάκων της:

Αρχείο: scorehtml.php

```php
<?php
    echo "<table border='1'>";
    echo "<tr><th>Ονοματεπώνυμο</th><th>Αριθμός
Μητρώου</th><th>Βαθμός</th></tr>";
    echo "<tr><td>Κωνσταντίνος
Παπαδόπουλος</td><td>15231</td><td>7</td></tr>";
    echo "<tr><td>Γιάννης
Λεμπέσης</td><td>12231</td><td>6</td></tr>";
    echo "<tr><td>Μαρία
Κωνσταντοπούλου</td><td>12111</td><td>10</td></tr>";
    echo "</table>";
?>
```

Αποτέλεσμα:



Εικόνα Α.21 Τύπωση αποτελεσμάτων με χρήση html

Είναι ευκολότερη η περιήγηση και η ανάγνωση με την χρήση html. Στην συγκεκριμένη περίπτωση δεν θα συναντούσαμε απλά 3 στοιχεία φοιτητών αλλά θα βρίσκαμε μια τεράστια λίστα με πάνω από 100 ή 200 ονόματα. Συγκρίνετε πως θα εμφανίζονταν χωρίς html και πως με html. Δεν είναι σαφώς καλύτερη η δεύτερη περίπτωση ;

Όπως προείπαμε με την html δεν εμφανίζουμε μόνο αποτελέσματα αλλά μπορούμε να εισάγουμε και στοιχεία μέσω αυτής. Ένα πεδίο κειμένου, ένα κουτί επιλογής, ένα checkbox, ένα κουμπί αποστολής. Όλα αυτά είναι στοιχεία της html όπου μας βοηθούν να αλληλεπιδράμε με μια ιστοσελίδα.

Α.2.2 Βασικές ετικέτες χρήσης

Μια ιστοσελίδα για να μπορέσει ένας φυλλομετρητής να την ερμηνεύσει θα πρέπει να διαθέτει και τις ανάλογες ετικέτες δήλωσης της ως ιστοσελίδα html, μέσα στον κώδικα

της. Για να ορίσουμε ότι το έγγραφο που γράφουμε εκείνη την στιγμή είναι αρχείο το οποίο θα ερμηνευτεί ως ιστοσελίδα από τους φυλλομετρητές χρησιμοποιούμε το tag <!DOCTYPE html>, με αυτή την ετικέτα έχουμε ορίσει ότι παρακάτω ο φυλλομετρητής βρίσκοντας αντίστοιχες ετικέτες θα αρχίσει να τις ερμηνεύει.

Η σωστή δομή μιας ιστοσελίδας βασιζόμενοι στα υπάρχοντα πρότυπα έχει ως εξής:

```
<!DOCTYPE html>
<html>
 <head>
 <meta charset='UTF-8'/>
  <title> </title>
 </head>
 <body>
 </body>
</html>
```

Μεταξύ του «head» εισάγουμε κομμάτια κώδικα τα οποία δεν «φαίνονται» στο φόρτωμα της σελίδας, όπως meta tags, scripts κ.α., στο μεταξύ του «<body>» και του «</body>» εισάγουμε όλες τις ετικέτες εκείνες που εμφανίζονται στο φόρτωμα της σελίδας μας, συμπεριλαμβάνοντας εκεί όλα αυτά που επεξηγούνται παρακάτω και όπου θα χρησιμοποιήσουμε κατά κόρον σε όλες τις επόμενες ενότητες μας.

A.2.2.1 Βασικές ετικέτες

Ο Πίνακας Α. περιγράφει μερικές από τις βασικότερες ετικέτες που χρησιμοποιούνται στην HTML.

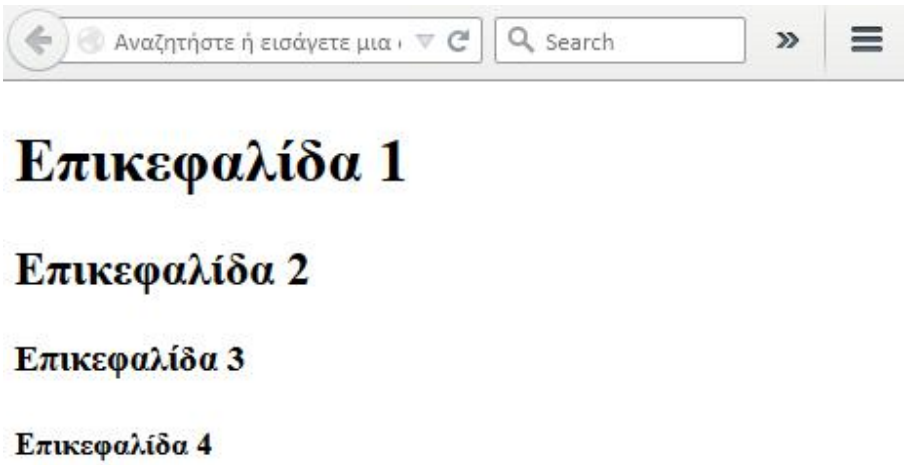
Ετικέτα	Παράδειγμα	Περιγραφή
hx	`<h1>Τίτλος</h1>`	Τίτλοι κειμένων, από 1 έως 6
p	`<p>Παράγραφος</p>`	Περιοχή κειμένου
a	`<a href="συνδεσμος">τίτλος</a>`	Χρήση συνδέσμου και τίτλου αυτού. Το url μπαίνει στο href και το εμφανιζόμενο κείμενο μέσα στο tag.

Πίνακας Α.1 Βασικές ετικέτες html

Παράδειγμα:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset='UTF-8'/>
  <title> Δοκιμή βασικών ετικετών HTML</title>
 </head>
 <body>
  <h1>Επικεφαλίδα 1 </h1>
  <h2>Επικεφαλίδα 2 </h2>
  <h3>Επικεφαλίδα 3 </h3>
  <h4>Επικεφαλίδα 4 </h4>
  <h5>Επικεφαλίδα 5 </h5>
  <h6>Επικεδαλίδα 6 </h6>
  <p>Παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου
παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου
παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου
παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου
παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου</p>
  <a href='http://www.google.gr/'> Σύνδεσμος κειμένου </a>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Α.22 Παράδειγμα χρήσης βασικών ετικετών στην html

Α.2.2.2 Μορφοποίηση κειμένου

Στην HTML υπάρχουν επίσης κάποιες ετικέτες για την μορφοποίηση κειμένου, μερικές από αυτές περιγράφει ο Πίνακας Α.2.

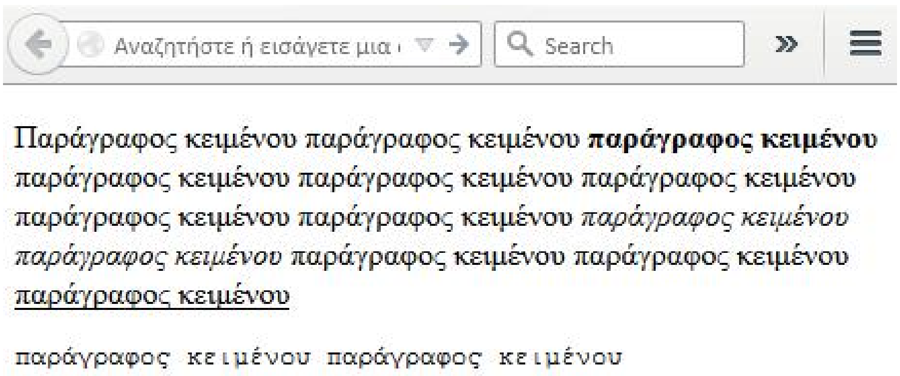
Ετικέτα	Περιγραφή	Παράδειγμα
b	Έντονο κείμενο	`<b>Κώστας</b>`
i	Πλάγιο κείμενο	`<i>Γιάννης</i>`
u	Υπογραμμισμένο κείμενο	`<u>Ελένη</u>`
pre	Παράγραφος σταθερού πλάτους	`<pre>…κείμενο… </pre>`

Πίνακας Α.2 Ετικέτες μορφοποίησης κειμένου html

Παράδειγμα:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset='UTF-8'/>
  <title> Δοκιμή μορφοποίησης κειμένου HTML</title>
 </head>
 <body>
  <p>Παράγραφος κειμένου παράγραφος κειμένου <b>παράγραφος
κειμένου</b> παράγραφος κειμένου παράγραφος κειμένου παράγραφος
κειμένου παράγραφος κειμένου παράγραφος κειμένου <i> παράγραφος
κειμένου παράγραφος κειμένου</i> παράγραφος κειμένου παράγραφος
κειμένου <u>παράγραφος κειμένου</u> <pre>παράγραφος κειμένου
παράγραφος κειμένου</pre></p>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα A.23 Παράδειγμα χρήσης μορφοποίησης κειμένου στην html

A.2.2.3 Περιοχές αντικειμένων

Οι περιοχές αντικειμένων καθορίζουν τις περιοχές στις οποίες αντικείμενα μπορούν να εισαχθούν. Στα κείμενα καθορίζουν σε ποια σημεία αλλάζουν γραμμές ή που εισάγονται οριζόντιες γραμμές που χωρίζουν το περιεχόμενο.

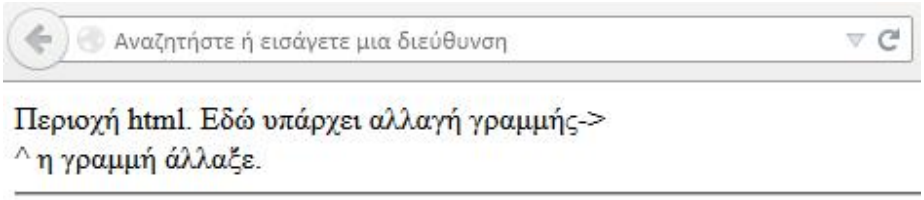
Ετικέτα	Περιγραφή	Παράδειγμα
div	Κοντέινερ που περιέχει στοιχεία html	`<div><b>ένα</b></div>`
br	Κενή γραμμή	` `
hr	Οριζόντια γραμμή	`<hr />`

Πίνακας A.3 Ετικέτες περιοχών αντικειμένων html

Παράδειγμα:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset='UTF-8'/>
  <title> Δοκιμή περιοχών αντικειμένων HTML</title>
 </head>
 <body>
  <div>
   Περιοχή html. Εδώ υπάρχει αλλαγή γραμμής-> </br> ^ η γραμμή
άλλαξε. <hr/>
  </div>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Α.24 Χρήση περιοχών αντικειμένων στην html

Α.2.2.4 Εικόνα

Η εικόνα χρησιμοποιείται με την ετικέτα «img» παίρνοντας αντίστοιχα κάποιες παραμέτρους:

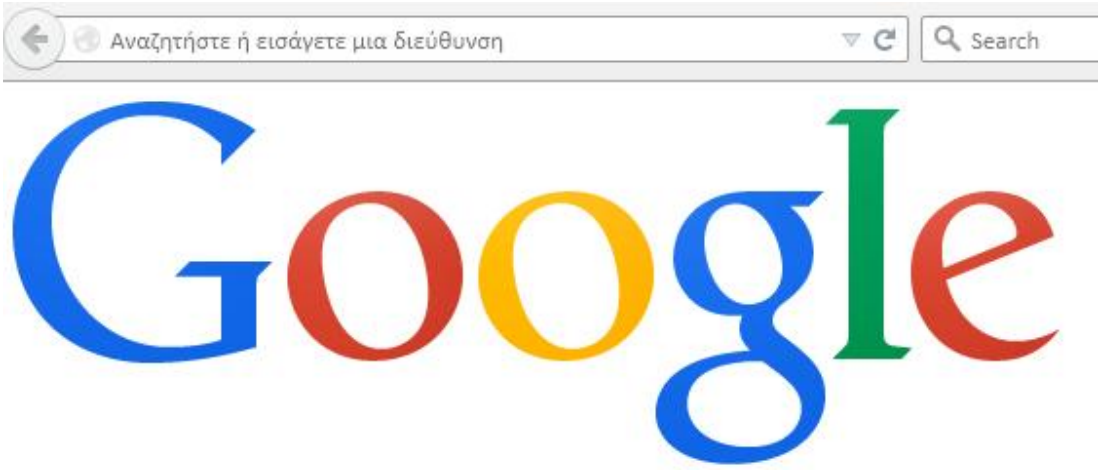
```
<img src="https://www.google.com/images/srpr/logo11w.png"
alt='google'/>
```

Στην παράμετρο «src» εισάγουμε το link όπου βρίσκεται η εικόνα, ενώ στην παράμετρο «alt» εισάγουμε εναλλακτικό κείμενο το οποίο θα εμφανίζεται αν δεν μπορεί για οποιοδήποτε λόγο να φορτωθεί η εικόνα.

Παράδειγμα:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="UTF-8"/>
  <title>Δοκιμή img</title>
 </head>
 <body>
  <img src="https://www.google.com/images/srpr/logo11w.png"
alt="google"/>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Α.25 Αποτέλεσμα χρήσης ετικέτας img

Α.2.2.5 Πίνακες

Οι πίνακες στην html είναι ένας τρόπο χωρικής ταξινόμησης προβαλλόμενων δεδομένων. Έτσι ότι προβάλλουμε είναι πιο εύκολα κατανοητό και ευανάγνωστο.

Ετικέτα	Περιγραφή	Παράδειγμα
table	Ετικέτα που δείχνει ότι ξεκινά και τελειώνει πίνακας	<table> <tr> <th>Τίτλος 1</th> <td>Κελί 2</td> </tr> </table>
tr	Table row = Γραμμή πίνακα	
th	Table heading = Επικεφαλίδα πίνακα	
td	Table division = Κοντέινερ πίνακα (Κελί πίνακα)	

Πίνακας Α.4 Ετικέτες πίνακα στην html

Αρχείο: table.html

```
<html>
 <body>
  <table border='1'>
   <tr><th>Όνομα</th><th>Επώνυμο</th></tr>
   <tr><td>Χρήστος</td><td>Νικολακόπουλος</td></tr>
   <tr><td>Ιωάννα</td><td>Χριστοπούλου</td></tr>
  </table>
 </body>
</html>
```

Αποτέλεσμα:

Εικόνα Α.26 Χρήση πινάκων της html

Η παράμετρος «border» καθορίζει πόσο θα είναι το μέγεθος των διαχωριστικών γραμμών που διαχωρίζουν τα κελιά μεταξύ τους και μετριέται σε pixels.

Α.2.2.6 Φόρμες

Οι φόρμες είναι ένας τρόπος για την html έτσι ώστε ένας χρήστης να μπορεί να εισάγει δικά του δεδομένα και αυτά τα δεδομένα να μπορούν να χρησιμοποιηθούν κάπως μέσα στα scripts μας. Παρακάτω θα δούμε μερικές από τις πιο σημαντικές ετικέτες στις φόρμες html.

Ετικέτα: form

Η ετικέτα form χρησιμοποιείται για να δηλώσει ότι τα περιεχόμενα αντικείμενα της θα χρησιμοποιηθούν έτσι ώστε τα στοιχεία που θα εισαχθούν σε αυτά από κάποιον χρήστη, θα σταλθούν κάπου συγκεκριμένα και με συγκεκριμένο τρόπο.

Παράδειγμα:

```
<form method='post' action='file.php'>
 …Υπόλοιπα στοιχεία φόρμας…
</form>
```

- Το χαρακτηριστικό «action» καθορίζει που θα σταλούν τα δεδομένα μετά την υποβολή της φόρμας. Μπορεί να είναι ένα απλό όνομα αρχείου αν βρίσκεται στον ίδιο κατάλογο με την φόρμα ή μπορεί να είναι και ολόκληρο url.

- Το χαρακτηριστικό «method» καθορίζει τον τρόπο με τον οποίο μπορούν να σταλθούν τα στοιχεία και υπάρχουν δυο τρόποι για να συμβεί αυτό post και get.
    - Όταν χρησιμοποιείται η μέθοδος «post» τότε τα στοιχεία μεταφέρονται από την φόρμα στο αρχείο της παραμέτρου «action» χωρίς να αποθηκεύονται στο ιστορικό του φυλλομετρητή και χωρίς να αφήνονται «ίχνη» πίσω τους. Είναι κατάλληλη έτσι για μεταφορά ευαίσθητων δεδομένων, όπως κωδικούς, προσωπικά στοιχεία κ.α.
    - Όταν χρησιμοποιείται η μέθοδος «get» τότε τα στοιχεία μεταφέρονται από την φόρμα στο αρχείο της παραμέτρου «action» μέσω του url του φυλλομετρητή. Έτσι τα στοιχεία αυτά είναι εμφανή ακόμη και μετά την ολοκλήρωση της φόρμας και την αποστολή των στοιχείων, μέσω του url με την μορφή «ονομα_πεδίου=τιμή». Αυτή η μέθοδος χρησιμοποιείται για απλή μεταφορά μη ευαίσθητων δεδομένων, διότι είναι δυνατή η αποθήκευση των στοιχείων τόσο στο ιστορικό του φυλλομετρητή καθώς και ευάλωτη σε πρόσβαση από τρίτους.

Ετικέτα: input

Η ετικέτα input είναι μια ετικέτα που πρέπει να συμπεριληφθεί μέσα στα περιεχόμενα μιας ετικέτας form. Διαθέτει κάποιες παραμέτρους που καθορίζουν τον τρόπο με τον οποίο συμπεριφέρεται. Ο τύπος της ετικέτας είναι αυτός που καθορίζει τον τρόπο με τον οποίο θα αλληλεπιδρά ένας χρήστης με αυτήν.

Παράμετρος	Τύπος	Παράδειγμα χρήσης
text	Μικρό κείμενο	Όνομα: Κείμενο
password	Κωδικός	Κωδικός: •••••••••
checkbox	Τικ επιλογής	Διατήρηση σύνδεσης:
radio	Επιλογή μόνο μίας απάντησης	Φύλλο: Άνδρας Γυναίκα
button submit	Κουμπί Κουμπί υποβολής	Αποστολή

Πίνακας Α.5 Παράμετροι χαρακτηριστικού type για την ετικέτα input

Χαρακτηριστικό	Περιγραφή
name	όνομα μεταβλητής όταν υποβληθεί
value	προεπιλεγμένη τιμή

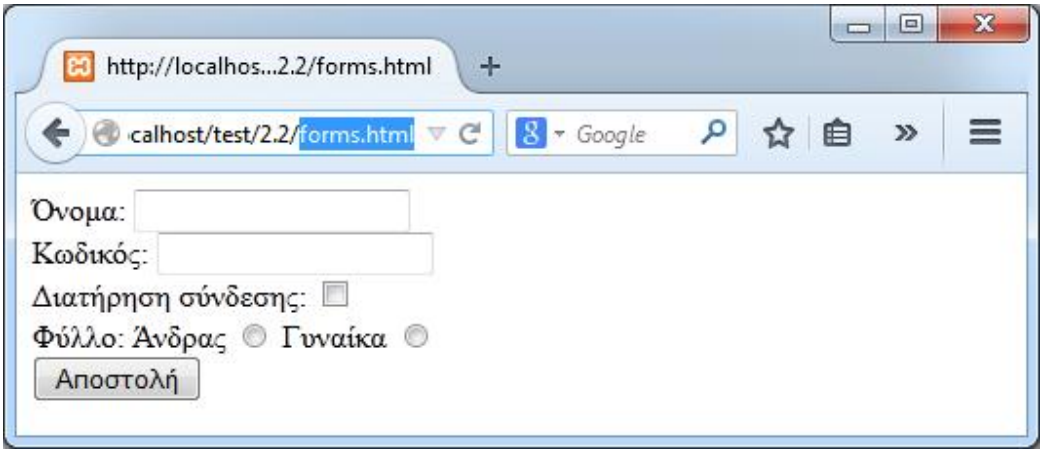
size	προβαλλόμενο μέγεθος πεδίου.
maxlength	μέγιστος αριθμό χαρακτήρων
checked	Σε radio ή checkbox μαρκάρει κάποιο συγκεκριμένο.

Πίνακας Α.6 Χαρακτηριστικά της ετικέτας input

Αρχείο: form.html

```
<html>
 <body>
  <form method='get' action='data.php'>
   Όνομα: <input type='text' name='name'/>
   <br />
   Κωδικός: <input type='password' name='password'/>
   <br />
   Διατήρηση σύνδεσης: <input type='checkbox' name='staylogged'/>
   <br />
   Φύλλο:
   Άνδρας <input type='radio' name='sex' value='male'/>
   Γυναίκα <input type='radio' name='sex' value='female'/>
   <br />
   <input type='submit' value='Αποστολή'/>
  </form>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Α.27 Αποτέλεσμα εκτέλεσης αρχείου forms.html

Α.2.3 Διαχείριση εισαγόμενων δεδομένων

Τα εισαγόμενα δεδομένα μπορούν να έχουν διάφορους τρόπους εισόδου στο script μας. Σε αυτή την ενότητα θα δούμε αναλυτικά όλους τους πιθανούς τρόπους από τους οποίους μπορούμε να αντλήσουμε δεδομένα στην php, καθώς επίσης και πως μπορούμε να τα εκμεταλλευτούμε.

A.2.3.1 Φόρμες HTML

Οι φόρμες χρησιμεύουν στην php κυρίως για την εισαγωγή δεδομένων και στοιχείων που μπορεί ένας χρήστης να πληκτρολογήσει ή να επιλέξει. Τα στοιχεία αυτά θα αποσταλούν ύστερα σε ένα αρχείο php το οποίο θα τα διαχειριστεί ανάλογα.

Για να δημιουργηθεί μια φόρμα θα πρέπει να χρησιμοποιηθεί η ετικέτα «form» η οποία θα πρέπει να έχει τις ιδιότητες «action» και «method».

Η «action» λειτουργεί ως δείκτης και μας λέει σε ποιο αρχείο php στον εξυπηρετητή θα αποσταλούν τα στοιχεία . πχ

```
action="useregister.php"
```

Η «method» καθορίζει τον τρόπο με τον οποίο θα αποστέλλονται τα δεδομένα, πχ:

```
method="post"
method="get"
```

Για να αποσταλούν τα στοιχεία θα πρέπει ο χρήστης να κάνει κλικ στο κουμπί submit.

Αρχείο: form.php

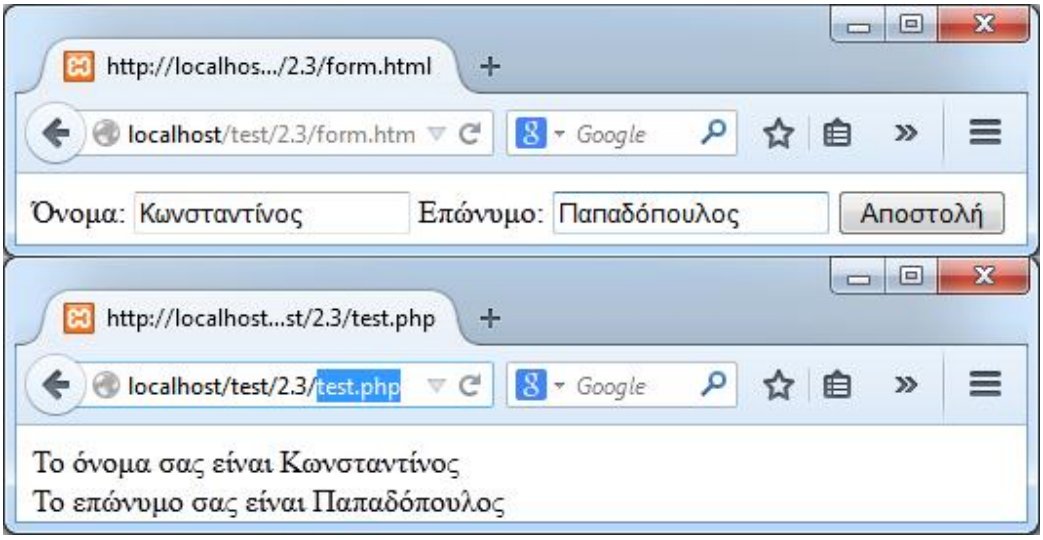
```
<form action="formtest.php" method="post">
 Όνομα: <input type="text" name="userfname" />
 Επώνυμο: <input type="text" name="userlname" />
 <input type="submit" value="Αποστολή" />
</form>
```

Αρχείο: formtest.php

```
Το όνομα σας είναι <?php echo $_POST['userfname']; ?>
<br />
Το επώνυμο σας είναι <?php echo $_POST['userlname']; ?>
```

Τόσο το αρχείο form.html όσο και το αρχείο test.php τα τοποθετούμε στο ίδιο directory στον server μας.

Αν πληκτρολογήσουμε «Κωνσταντίνος» στο πεδίο του ονόματος και «Παπαδόπουλος» στο πεδίο του επωνύμου, και ύστερα πιέσουμε το κουμπί «Αποστολή» θα εμφανιστεί το εξής αποτέλεσμα:



Εικόνα Α.28 Παράδειγμα μεταφοράς δεδομένων μέσω φόρμας html

A.2.3.2 $_GET και $_POST

Όπως προσέξατε και στο προηγούμενο παράδειγμα για να μπορέσουμε να χρησιμοποιήσουμε τα στοιχεία που εισάγαμε στην φόρμα έπρεπε να κάνουμε χρήση των μεταβλητών «$_POST» στο αρχείο php όπου επεξεργαζόμασταν τα δεδομένα αυτά.

Η «$_GET» χρησιμοποιείται όταν τα στοιχεία αποστέλλονται με την μέθοδο «get» και

- Τα στοιχεία παραμένουν στο ιστορικό του φυλλομετρητή (browser) γιατί είναι μέρος του url,
- Μπορούν να αποθηκευτούν ως σελιδοδείκτης,
- Δεν είναι ασφαλές για την μεταφορά κωδικών ή ευαίσθητων δεδομένων,
- Υπάρχει όριο 7607 μέγιστο πλήθος χαρακτήρων,
- Παράδειγμα: test.php?userfname=κώστας.

Η «$_POST» χρησιμοποιείται όταν τα στοιχεία αποστέλλονται με την μέθοδο «post», και

- Τα στοιχεία δεν αποθηκεύονται στο ιστορικό του φυλλομετρητή browser,
- Δεν μπορούν να αποθηκευτούν ως σελιδοδείκτες,
- Χρησιμοποιείται για την αποστολή κωδικών και ευαίσθητων δεδομένων,
- Υπάρχει όριο 8mb μέγιστο μέγεθος για το κάθε POST αίτημα,

- Παράδειγμα: test.php.

Τέλος υπάρχει και η μεταβλητή «$_REQUEST» η οποία χρησιμοποιείται και για τις δύο περιπτώσεις ανεξάρτητα ποια μέθοδος έχει χρησιμοποιηθεί, αντικαθιστώντας τις $_GET, $_POST και $_COOKIE.

A.2.3.3 Cookies

Τα cookies είναι μικρά αρχεία τα οποία αποθηκεύονται τοπικά στον υπολογιστή του χρήστη και περιέχει συνήθως δεδομένα σχετικά με την ταυτότητα του και το προφίλ του.

Έχοντας αποθηκευμένα αυτά τα αρχεία είναι δυνατό ένας χρήστης να αναγνωρίζεται από μια διαδικτυακή εφαρμογή χωρίς να χρειάζεται ο ίδιος να εισέλθει σε αυτήν με το όνομα του και τον κωδικό του.

Τα cookies μπορούν επίσης να χρησιμοποιηθούν για να καταγράψουν την δραστηριότητα ενός χρήστη, όπως για παράδειγμα αν έχει πάρει μέρος σε διαδικτυακές ψηφοφορίες, προσωπικές ρυθμίσεις π.χ. χρωματικά θέματα… και άλλα.

A.2.3.3.1 Συνάρτηση setcookie()

Μας επιτρέπει να δημιουργήσουμε ένα cookie στο μηχάνημα του χρήστη. Παίρνει 6 παραμέτρους:

setcookie(όνομα, τιμή, λήξη, διαδρομή, domain, ασφάλεια);

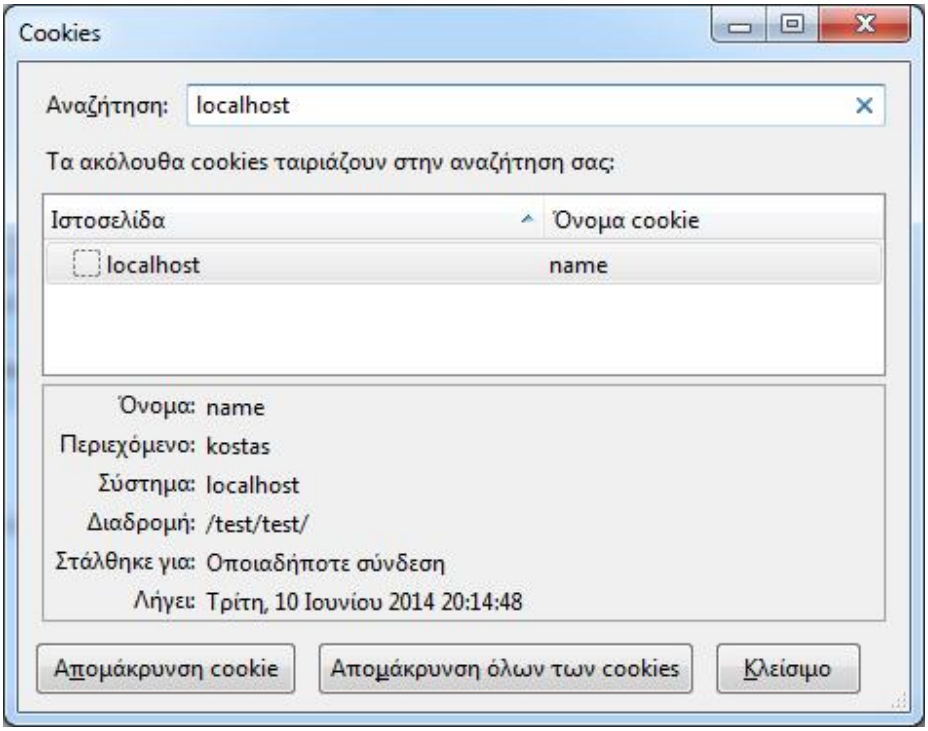
1. **Όνομα**: Το όνομα του cookie.
2. **Τιμή**: Τα δεδομένα του.
3. **Λήξη**: Ημερομηνία λήξης του.
4. **Διαδρομή**: Η διαδρομή για το οποίο το cookie είναι έγκυρο. Συνήθως αφήνεται κενό και η προκαθορισμένη τιμή είναι το path του εγγράφου που δημιούργησε το cookie. Πχ «/login».
5. **Domain**: Το domain name για το οποίο το cookie είναι έγκυρο. Συνήθως αφήνεται κενό και προεπιλεγμένα διαλέγεται το domain name του εγγράφου που το δημιούργησε. Πχ «www.google.com».

6. **Ασφάλεια**: True ή False για να υποδείξει ότι το cookie χρησιμοποιείται σε ασφαλείς συνδέσεις όπως SSL. Το προκαθορισμένο είναι false.

Αρχείο: cookie.php

```
<?php
 setcookie("name","kostas",time()+3600);
 echo "ok";
?>
```

Αποτέλεσμα:



Εικόνα Α.29 Προβολή cookie που δημιουργήθηκε

Στις ιδιότητες του ιστότοπου από τις ρυθμίσεις του firefox, μπορούμε να δούμε πως, δημιουργήθηκε ένα cookie με όνομα «name» και τιμή «kostas» το οποίο θα έληγε 3600 δευτερόλεπτα από την στιγμή της δημιουργίας του.

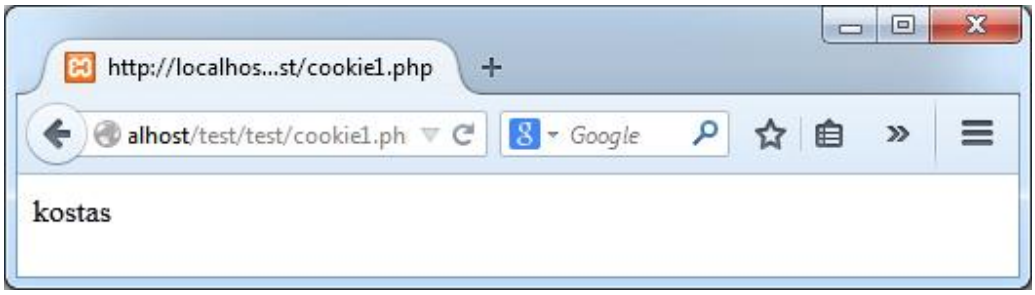
Α.2.3.3.2 Μεταβλητή $_COOKIE

Όταν ένα cookie υπάρχει, τότε ο μόνος λόγος που υπάρχει είναι έτσι ώστε να μπορούμε αργότερα να έχουμε πρόσβαση στα δεδομένα του. Ο τρόπος με τον οποίο γίνεται είναι χρησιμοποιώντας τις μεταβλητές «$_COOKIE» όπως τις «$_GET» και «$_POST» που είδαμε σε προηγούμενη ενότητα.

Αρχείο: cookie1.php

```
<?php
 echo $_COOKIE["name"];
?>
```

Αποτέλεσμα:



Εικόνα Α.30 Πρόσβαση σε αποθηκευμένο cookie με την php

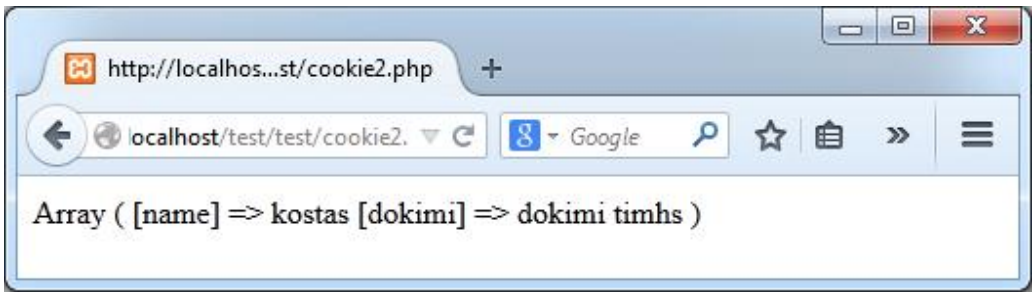
Αν το cookie «name» υπάρχει τοπικά στο μηχάνημα του χρήστη τότε επιστρέφεται η τιμή του που στην προκειμένη είναι «kostas».

Αν θέλουμε να διαβάσουμε ΟΛΑ τα cookies που υπάρχουν στο εν λόγο μηχάνημα τότε χρησιμοποιούμε σκέτη την «$_COOKIE» η οποία επιστρέφει έναν πίνακα με όλα τα αποτελέσματα. Πχ:

Αρχείο: cookie2.php

```
<?php
 print_r($_COOKIE);
?>
```

Αποτέλεσμα:



Εικόνα Α.31 Επιστροφή όλων των διαθέσιμων cookies.

Α.2.3.3.3 Συνάρτηση isset()

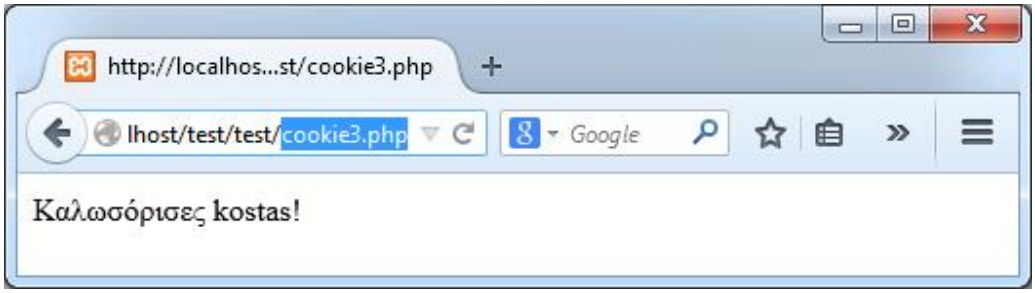
Καλό είναι για προγραμματιστικούς λόγους να ελέγχεται πριν την χρήση του ένα cookie για την ύπαρξη του με την συνάρτηση isset(). Η συγκεκριμένη συνάρτηση

ελέγχει αν υπάρχει η μεταβλητή και επιστρέφει true, σε διαφορετική περίπτωση επιστρέφει false. Πχ:

Αρχείο: cookie3.php

```
<?php
 if(isset($_COOKIE['name'])){
  echo "Καλωσόρισες ".$_COOKIE['name']."!";
 }else{
  echo "Καλοσόρισες επισκέπτη!";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.32 Έλεγχος ύπαρξης μεταβλητής

A.2.3.4 Sessions

Οι απλές μεταβλητές γενικώς στην php αν κάποιος μεταφερθεί από την σελίδα Α στην σελίδα Β, οι μεταβλητές της Α δεν θα είναι προσπελάσιμες από την Β. Γι' αυτό τον λόγο τα sessions χρησιμοποιούνται ώστε να υπάρχουν global μεταβλητές στις οποίες θα έχουν πρόσβαση όλοι.

Για να χρησιμοποιηθούν τα sessions θα πρέπει ο υπολογιστής του χρήστη να δέχεται cookies καθώς επίσης και όλα τα αρχεία php που χρησιμοποιούν session μεταβλητές θα πρέπει να αρχίζουν με την εντολή session start().

A.2.3.4.1 session_start() και $_SESSION

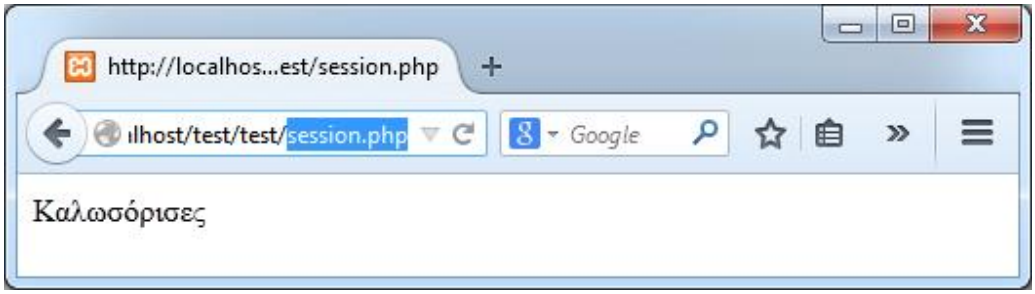
Τα sessions είναι παρόμοια με τα cookies. Ένα session ξεκινά με την εντολή session_start() η οποία σηματοδοτεί και την έναρξη του. Η συγκεκριμένη εντολή δημιουργεί ένα cookie στο μηχάνημα του επισκέπτη με όνομα PHPSESSID και μια τυχαία αλφαριθμητικά τιμή της μορφής: mqqee34fjsdfhhe342d.

Για να αποκτήσουμε πρόσβαση σε αυτή την τιμή χρησιμοποιούμε την εντολή session_id(). Με την global μεταβλητή «$_SESSION» μπορούμε να δημιουργήσουμε δικό μας session με όνομα και τιμή ή να χρησιμοποιήσουμε το ίδιο το id που μας προσφέρει η php.

Αρχείο: session.php

```
<?php
 session_start();
 $_SESSION['visitorid']=session_id();

 if(isset($_SESSION['visitorid']) &&
$_SESSION['visitorid']===session_id()){
  echo "Καλωσόρισες";
 }else{
  echo "Δεν υπάρχει session";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.33 Χρήση sessions στην php

Α.2.3.4.2 unset και session_destroy()

Η unset() αναλαμβάνει να διαγράψει ένα μόνο στοιχείο από τον υπολογιστή του χρήστη.

```
<?php
 session_start();
 unset($_SESSION['visitorid']);
?>
```

Η session_destroy() διαγράφει όλα τα αποθηκευμένα δεδομένα από τον υπολογιστή του χρήστη.

```
<?php
 session_start();
 session_destroy();
?>
```

A.2.4 Διαχείριση εξαγόμενων αποτελεσμάτων

Τι θα ήταν η php αν δεν μπορούσαμε ότι υπολογίζουμε μέσα σε αυτήν να το τυπώνουμε σε κάποιο αρχείο ή στην οθόνη;

Η php προσφέρει αρκετούς και διαφορετικούς τρόπους έτσι ώστε να κάνουμε τελικά αυτό που χρειάζεται πάνω από όλα σε μια γλώσσα, να εξάγουμε τα αποτελέσματά μας. Στην παρούσα ενότητα θα δούμε πώς να χρησιμοποιήσουμε το echo , το print, το printf, το sprintf και άλλα.

A.2.4.1 Echo

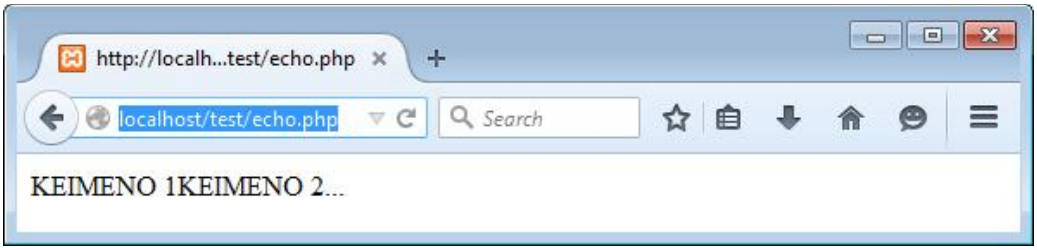
Για να μπορέσουμε να εξάγουμε ένα οποιοδήποτε είδος κειμένου σε μορφή αλφαριθμητικού από την εκτέλεση script, τότε μπορούμε να χρησιμοποιήσουμε την μέθοδο-εντολή echo. Αν και μοιάζει με μια κοινή μέθοδο τελικά πρόκειται για κατασκευαστή που βρίσκεται εμφωλευμένος μέσα στην ίδια την php. Η διαφορά με τις κοινές μεθόδους έγκειται στο ότι μπορούμε προαιρετικά να χρησιμοποιήσουμε παρενθέσεις. Αν πάλι δεν χρησιμοποιήσουμε η σύνταξή της και πάλι είναι σωστή.

Μπορούμε επίσης να τυπώσουμε πολλαπλά αντικείμενα χωρισμένα με κόμμα. Το οποίο αποτέλεσμα που θα προκύψει από αυτήν την χρήση είναι ένα μοναδικό αλφαριθμητικό όπου κάθε στοιχείο αποτελεί συνέχεια του άλλου. Δεν μπορούμε να τυπώσουμε πολλές τιμές μέσα σε παρενθέσεις όπως κάναμε παραπάνω χωρίς αυτές.

Αρχείο: echo.php

```
<?php
 echo "KEIMENO 1","KEIMENO 2","...";
?>
```

Αποτέλεσμα:



Εικόνα Α.34 Αποτέλεσμα εκτέλεσης εντολής echo

Επειδή το echo δεν είναι κανονική μέθοδος δεν μπορεί να χρησιμοποιηθεί σαν μέρος μεγαλύτερων σύνθετων εντολών.

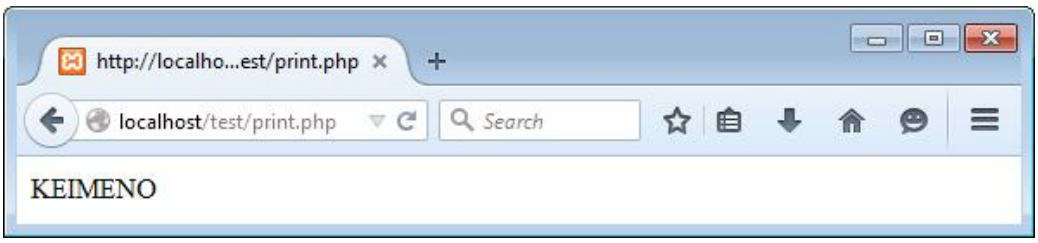
A.2.4.2 Print

Σε αντίθεση με το echo το print δεν είναι εμφωλευμένο στον κατασκευαστή της php επομένως μπορεί να χρησιμοποιηθεί σύνθετα, αλλά απαραίτητα και υποχρεωτικά με παρενθέσεις, διότι πλέον έχουμε να κάνουμε με μια κοινή μέθοδο.

Αρχείο: print.php

```
<?php
 print ("KEIMENO");
?>
```

Αποτέλεσμα:



Εικόνα Α.35 Αποτέλεσμα εκτέλεσης εντολής print

A.2.4.3 Printf

Η printf λειτουργεί με παρενθέσεις, μέσα στις οποίες παίρνει κάποια ορίσματα. Το πρώτο της όρισμα υποχρεωτικά είναι ένα αλφαριθμητικό στοιχείο, μέσα στο οποίο αναγράφουμε πάντα αυτό που θέλουμε να τυπώσουμε, συμπεριλαμβάνοντας όμως κάποιους ειδικούς χαρακτήρες ανάμεσα σε αυτά. Οι ειδικοί χαρακτήρες χρησιμοποιούνται ως αντικατάσταση για τα σημεία στα οποία θα πρέπει να μπουν τιμές μεταβλητών, και ονομάζονται προσδιοριστές τιμών.

Οι προσδιοριστές ξεκινούν με έναν χαρακτήρα του ποσοστού τοις εκατό και συνεχίζουν με έναν λατινικό χαρακτήρα ή και αριθμούς ανάμεσα στο ποσοστό τοις εκατό και τον λατινικό χαρακτήρα. Συγκεκριμένα:

1. Αναγράφεται αρχικά το ειδικό σύμβολο ποσοστού τοις εκατό (%). Αυτό το σύμβολο σημαίνει ότι ξεκινά ένας προσδιοριστής, αν δεν θέλετε να χρησιμοποιήσετε έναν τότε θα πρέπει να γράψετε δυο φορές το ποσοστό τοις

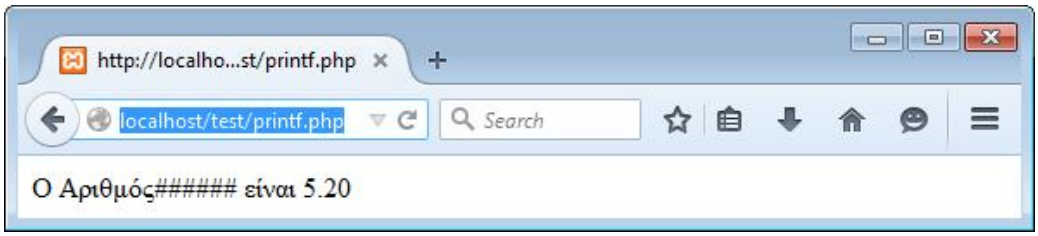
εκατό «%%» έτσι ώστε να χρησιμοποιηθεί το ποσοστό ως χαρακτήρας και όχι ως προσδιοριστής.

2. Ύστερα προαιρετικά αναγράφεται ένας αριθμός, ένας χαρακτήρας ή ένα κενό με πρόθεμα ένα μονό εισαγωγικό «'», ο οποίος αριθμός, χαρακτήρας ή κενό χρησιμοποιείται σε περίπτωση που το αποτέλεσμα είναι μικρότερο από το ελάχιστο μήκος χαρακτήρων που θα ορίσουμε (στο βήμα 4), έτσι θα πρέπει με κάποιο τρόπο «να γεμίσουν οι κενές θέσεις» με τον χαρακτήρα/αριθμό/κενό που επιλέξαμε να γράψουμε δίπλα από το μονό εισαγωγικό.
3. Ένα πρόσημο θετικό ή αρνητικό. Τα πρόσημα λειτουργούν διαφορετικά σε κάθε τύπο δεδομένων. Σε ένα αλφαριθμητικό αν χρησιμοποιήσουμε αρνητικό πρόσημο τότε το κείμενο θα στοιχηθεί στα αριστερά από ότι προεπιλεγμένα θα ήταν στοιχισμένο στα δεξιά. Σε έναν ακέραιο αριθμό το θετικό πρόσημο αναγκάζει το αποτέλεσμα να είναι αυστηρά θετικό προσθέτοντας αντίστοιχα και το πρόσημο μπροστά από τον αριθμό (πχ το -6 θα τυπωθεί +6).
4. Το ελάχιστο αριθμό χαρακτήρων που το στοιχείο θα τυπωθεί. Εάν το περιεχόμενο της μεταβλητής είναι μικρότερο από τον αριθμό που έχουμε ορίσει τότε θα τυπωθούν μαζί με τον αριθμό ή το αλφαριθμητικό και μερικά κενά ή ότι χαρακτήρα ορίσαμε στο βήμα 2 ώστε να πληρούνται οι ελάχιστες θέσεις που έχουμε ορίσει.
5. Αν έχουμε να κάνουμε με δεκαδικούς αριθμούς τότε συνεχίζουμε γράφοντας μια τελεία «.» και ύστερα ακολουθεί ένας αριθμός ακόμη όπου καθορίζει το πόσα ψηφία θα εμφανίζονται μετά της υποδιαστολής. Ο αριθμός αυτός λειτούργει σε συνάρτηση με τον γενικό αριθμό ψηφίων που έχουμε γράψει στο βήμα 4, αυτό σημαίνει πως αν έχουμε γενικό αριθμό ψηφίων 5 και μετά από την τελεία γράψουμε το 2, τότε τα τρία ψηφία θα δεσμευτούν για την αναπαράσταση του ακέραιου μέρους του αριθμού, και δύο ψηφία θα δεσμευτούν για το δεκαδικό μέρος πχ 333.33.

Αρχείο: printf.php

```php
<?php
 $keimeno = "Αριθμός";
 $arithmos = 5.20;

 printf("Ο %'#-20s είναι %'04.2f",$keimeno,$arithmos);
?>
```

Αποτέλεσμα:



Εικόνα Α.36 Αποτέλεσμα εκτέλεσης printf στην php

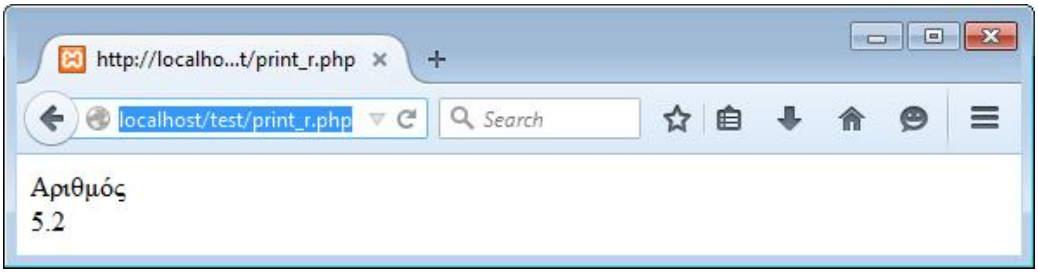
A.2.4.4 Print_r

Η print_r όταν δεχθεί κάποιο αντικείμενο ή κάποιον πίνακα ως είσοδο , τυπώνει όλη την δομή του αντικειμένου/πίνακα και μορφοποιεί το αποτέλεσμα εξόδου έτσι ώστε να εμφανίζεται ως λίστα κατά την τύπωση των αποτελεσμάτων διευκολύνοντας έτσι την ανάγνωση του. Όταν εισαχθεί αντικείμενο η τύπωση ξεκινά με την λέξη «object» ενώ αν εισαχθεί πίνακας με την λέξη «array».

Αρχείο: print_r.php

```php
<?php
 $keimeno = "Αριθμός";
 $arithmos = 5.20;

 print_r($keimeno);
 echo "<br/>";
 print_r($arithmos);
 echo "<br/>";
 print_r(false);
?>
```

Αποτέλεσμα:



Εικόνα Α.37 Αποτέλεσμα εκτέλεσης εντολής print_r για διάφορες απλές μεταβλητές

Χρησιμοποιώντας την εντολή αυτή για να τυπώσουμε έναν πίνακα ο δείκτης του πίνακα μετακινείται στο τέλος του. Είναι ανώφελο να προσπαθήσουμε να τυπώσουμε τιμές boolean και null. Σε επόμενες ενότητες θα δούμε την χρησιμότητα του στους πίνακες της php.

A.2.4.5 Var_dump

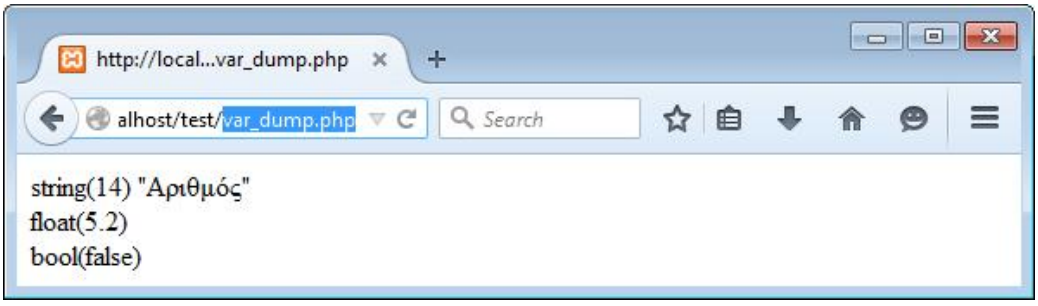
Το var_dump() το οποίο λειτουργεί ως εντολή αποσφαλμάτωσης, τυπώνει τα πάντα, ακόμη και boolean, null τιμές ή οποιαδήποτε άλλη λεπτομέρεια δεν τυπώνει το print_r.

Εδώ θα πρέπει να σημειώσουμε ότι πρέπει να αποφεύγεται η χρήση τόσο του print_r όσο και του var_dump σε αναδρομικές δομές όπως τα $GLOBALS (το οποίο για παράδειγμα διαθέτει μια εγγραφή που δείχνει πίσω στον εαυτό του). Σε μια τέτοια περίπτωση το print_r() συνεχίζει επ' αόριστο να τυπώνει ότι βρίσκει, έτσι είναι πιθανό να πέσουμε μέσα σε ατέρμονο βρόγχο. Ενώ η var_dump() μόλις εντοπίσει ότι έχει επισκεφθεί το ίδιο στοιχείο τρεις συνεχόμενες φορές σταματάει την επανάληψη διάσχισης.

Αρχείο: var_dump.php

```
<?php
 $keimeno = "Αριθμός";
 $arithmos = 5.20;

 var_dump($keimeno);
 echo "<br/>";
 var_dump($arithmos);
 echo "<br/>";
 var_dump(false);
?>
```

Αποτέλεσμα:



Εικόνα Α.38 Αποτέλεσμα εκτέλεσης var_dump για είσοδο απλών μεταβλητών

A.2.5 Έξοδος και είσοδος από αρχείο

Ένας από τους πολλούς τρόπους που διαθέτει η php για να αποθηκεύει δεδομένα είναι η έξοδος σε αρχείο κειμένου. Αντίστοιχα για να μπορεί να αποθηκεύει εκεί θα πρέπει να μπορεί να αντλεί κι από εκεί δεδομένα. Σε αυτό το κεφάλαιο θα δούμε πως με την php μπορούμε να αποθηκεύσουμε δεδομένα σε αρχεία κειμένου και να τα αντλήσουμε αντίστοιχα ξανά.

A.2.5.1 Η έξοδος

Η έξοδος δεδομένων είναι μια χρήσιμη διαδικασία, ειδικά όταν πρόκειται για έξοδο σε κάποιο αρχείο το οποίο μπορεί να χρησιμοποιηθεί ως αρχείο ανάκτησης δεδομένων. Μέσω της εξόδου γράφουμε κάποια δεδομένα σε κάποιο «εξωτερικό» αρχείο, έστω αρχείο κειμένου, και ύστερα αυτό μπορούμε να το εκμεταλλευτούμε όπως εμείς θέλουμε πχ ως αρχείο ανάκτησης δεδομένων. Βέβαια δεν είναι δεσμευτικό το ότι το αρχείο που θα εξάγουμε θα είναι μόνο για ανάκτηση δεδομένων, αλλά μπορεί να είναι ένα απλό αρχείο κειμένου το οποίο απλά να γράφουμε κάτι για να το διαβάσουμε αργότερα.

A.2.5.1.1 fwrite

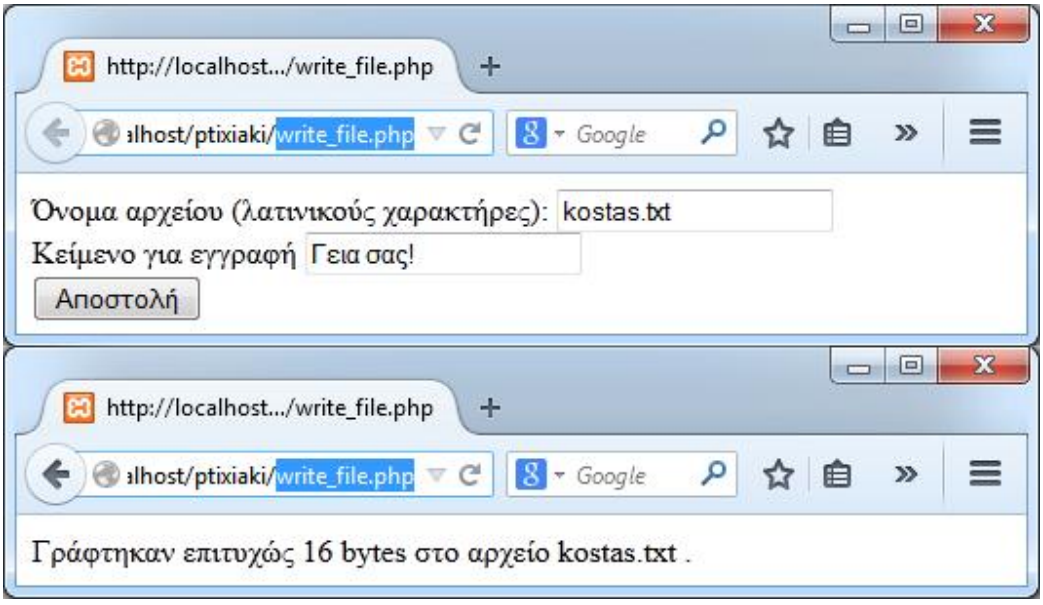
Η εντολή fwrite είναι μια εντολή εξόδου δεδομένων σε αρχείο. Ας δούμε πως χρησιμοποιείται:

- Χρησιμοποιούμε μια μεταβλητή και την εντολή «fopen» προκειμένου να δημιουργήσουμε ένα στιγμιότυπο αντικειμένου που αναφέρεται στο συγκεκριμένο αρχείο. Η συνάρτηση παίρνει δυο παραμέτρους το όνομα του αρχείου, και έναν χαρακτήρα διαχείρισης του συγκεκριμένου αρχείου. Όλοι οι χαρακτήρες επεξηγούνται σε επόμενη ενότητα αναλυτικά.
- Ύστερα χρησιμοποιούμε την εντολή «fwrite» σε συνδυασμό με την μεταβλητή του αρχείου που μόλις φτιάξαμε. Η συνάρτηση παίρνει δυο παραμέτρους, η μια είναι η μεταβλητή που αντιπροσωπεύει το αρχείο, και η άλλη είναι το κείμενο που θέλουμε να γράψουμε μέσα σε αυτό.
- Μόλις τελειώσουμε τις εγγραφές. Χρησιμοποιούμε την εντολή «fclose» για να κλείσουμε την εγγραφή του αρχείου. Η συνάρτηση παίρνει μια μόνο παράμετρο, την μεταβλητή που αντιπροσωπεύει το αρχείο.

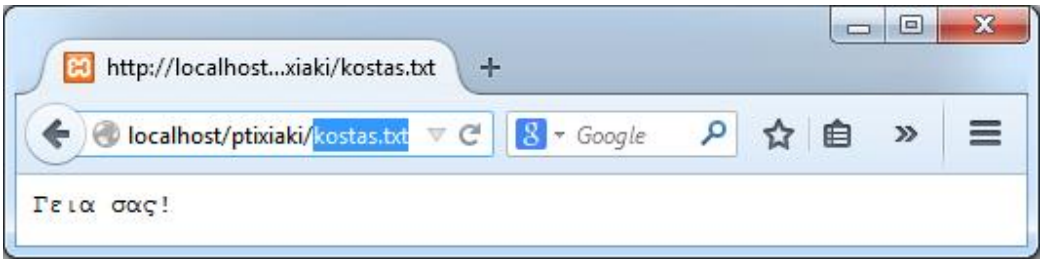
Αρχείο: write_file.php

```
<html>
 <head>
  <meta charset="UTF-8">
  <title></title>
 </head>
 <body>
  <?php
   if (isset($_POST['text']) && isset($_POST['file'])) {
    $file = $_POST['file'];
    $f = fopen($file,'w');
    $n = fwrite($f,$_POST['text']);
    echo "Γράφτηκαν επιτυχώς $n bytes στο αρχείο $file .";
   } else {
    ?>
    <form action='write_file.php' method='post'>
     Όνομα αρχείου (λατινικούς χαρακτήρες): <input type="text"
name="file"/><br/>
     Κείμενο για εγγραφή <input type='text' name="text"/><br/>
     <input type='submit' value='Αποστολή'/>
    </form>
    <?php
     }
    ?>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Α.39 Εγγραφή κειμένου μέσω φόρμας html σε αρχείο



Εικόνα Α.40 Προβολή του αρχείου kostas.txt για επαλήθευση

Α.2.5.1.2 fputs

Η συγκεκριμένη συνάρτηση λειτουργεί όπως η fwrite μόνο που γράφει μόνο μια γραμμή κειμένου.

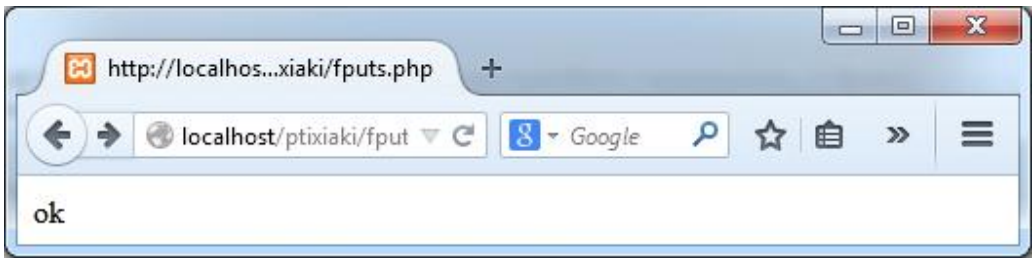
Αρχείο: names.txt

```
Kostas
Γιάννης
Έλενα
Δημήτρης
Χρήστος
```

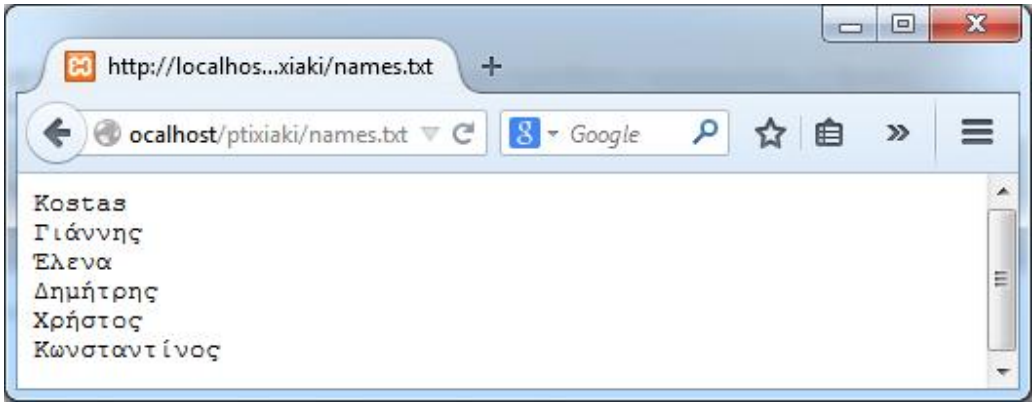
Αρχείο: fputs.php

```
<?php
 $f = fopen("names.txt", "a");
 fputs($f, "Κωνσταντίνος\n");
 fclose($f);
 echo "ok";
?>
```

Αποτέλεσμα:



Εικόνα Α.41 Αποτέλεσμα εκτέλεσης αρχείου fputs.php



Εικόνα Α.42 Επαλήθευση εισαγωγής κειμένου στο αρχείο names.txt

Α.2.5.2 Η είσοδος

Όπως μπορούμε να «βγάλουμε» δεδομένα από μια διαδικτυακή εφαρμογή σε ένα αρχείο έτσι μπορούμε να εισάγουμε κιόλας.

Α.2.5.2.1 fread

Όπως «ανοίξαμε» ένα αρχείο για να το γράψουμε έτσι μπορούμε να το ανοίξουμε για να το διαβάσουμε, αρκεί να αλλάξουμε δυο πράγματα:

- Αντί για «w» ως δεύτερη παράμετρο στο fopen θα βάλουμε «r».
- Αντί για fwrite χρησιμοποιούμε πλέον fread το οποίο παίρνει δυο παραμέτρους ως είσοδο, η πρώτη είναι η μεταβλητή που αντιπροσωπεύει το αρχείο και η δεύτερη είναι μια ακέραια τιμή που ορίζει τον μέγιστο αριθμό bytes που θα διαβαστούν.

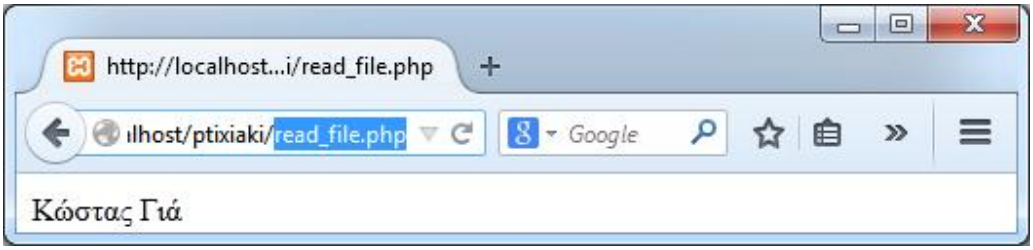
Αρχείο: names.txt

```
Κώστας
Γιάννης
Έλενα
Δημήτρης
Χρήστος
```

Αρχείο: read_file.php

```
<?php
 $f = fopen('names.txt','r');
 $s = fread($f,20);
 fclose($f);
 echo $s;
?>
```

Αποτέλεσμα:



Εικόνα Α.43 Αποτέλεσμα εκτέλεσης αρχείου read_file.php

A.2.5.2.2 fgets

Η fgets διαβάζει και επιστρέφει μια ολόκληρη γραμμή από ένα αρχείο κειμένου μαζί με τον χαρακτήρα αλλαγής γραμμής στο τέλος της. Σε περίπτωση αποτυχίας επιστρέφεται false. Το μόνο όρισμα που παίρνει ως παράμετρο είναι την μεταβλητή που αναφέρεται στο αρχείο.

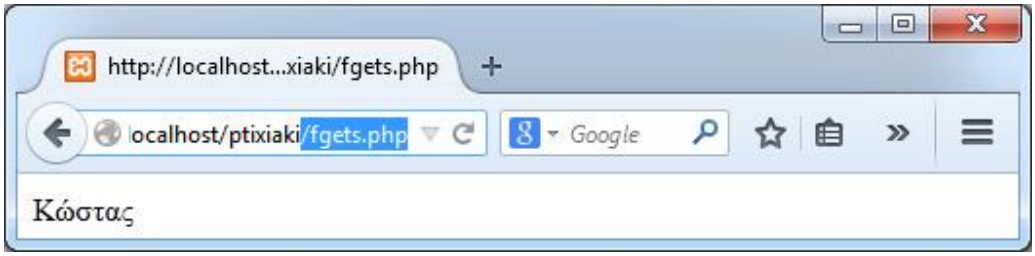
Αρχείο: names.txt

```
Κώστας
Γιάννης
Έλενα
Δημήτρης
Χρήστος
```

Αρχείο: fgets.php

```
<?php
 $f = fopen('names.txt','r');
 $s = fgets($f);
 fclose($f);
 echo $s;
?>
```

Αποτέλεσμα:



Εικόνα Α.44 Αποτέλεσμα εκτέλεσης fgets.php

A.2.5.2.3 fgetc

Διαβάζει ένα αρχείο και επιστρέφει έναν χαρακτήρα από αυτό. Εάν διαβάσει το τέλος του αρχείου τότε επιστρέφει false. Παίρνει ένα μόνο όρισμα, την μεταβλητή του αρχείου που αντιστοιχεί στο fopen. Θυμηθείτε πως οι ελληνικοί χαρακτήρες δεν αποτελούνται από έναν μόνο χαρακτήρα κειμένου, αλλά κάθε ελληνικό γράμμα είναι τρία ή και τέσσερα λατινικά γράμματα.

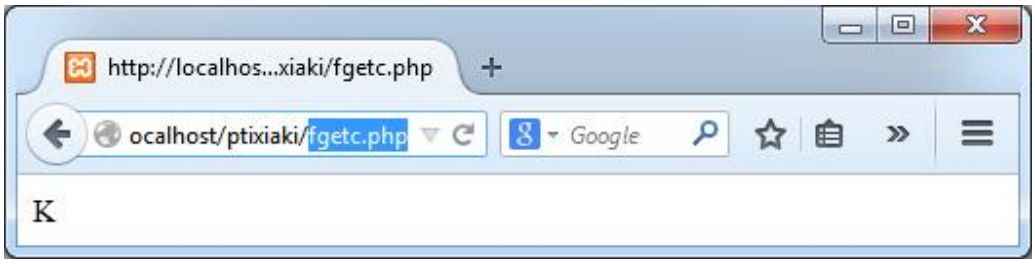
Αρχείο: names.txt

```
Kostas
Γιάννης
Έλενα
Δημήτρης
Χρήστος
```

Αρχείο: fgetc.php

```
<?php
 $f = fopen('names.txt','r');
 $s = fgetsc($f);
 fclose($f);
 echo $s;
?>
```

Αποτέλεσμα:



Εικόνα Α.45 Αποτέλεσμα εκτέλεσης αρχείου fgetc.php

A.2.5.2.4 fgetcsv

Διαβάζει μια γραμμή και επιστρέφει έναν πίνακα μόνο αν αυτό που διαβάσει βρίσκεται σε μορφή csv. Αν αποτύχει επιστρέφει false. Ως είσοδο παίρνει την μεταβλητή που αντιπροσωπεύει το αρχείο.

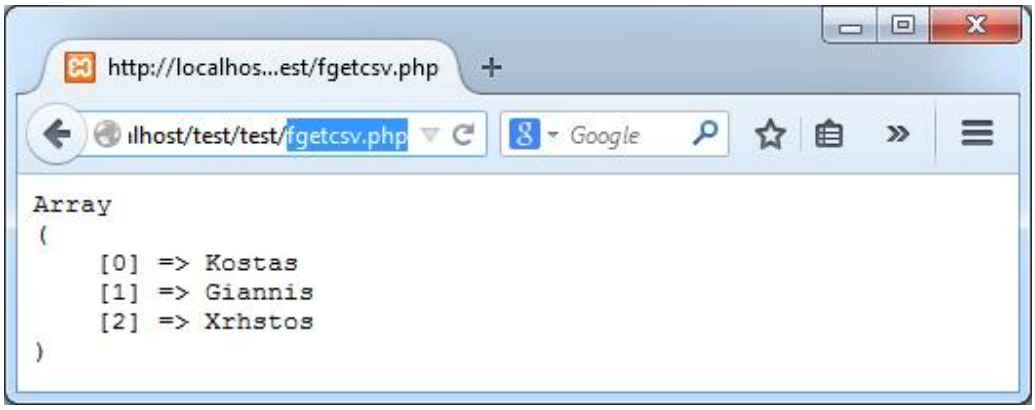
Αρχείο: names.txt

```
Kostas,Giannis,Xrhstos
```

Αρχείο: fgetcsv.php

```php
<?php
 $f=fopen("names.txt",'r');
 $array = fgetcsv($f);
 echo "<pre>";
 print_r($array);
 echo "</pre>";
 fclose($f);
?>
```

Αποτέλεσμα:



Εικόνα Α.46 Αποτέλεσμα εκτέλεσης αρχείου fgetcsv.php

Α.2.5.2.5 feof

Ελέγχει αν ο δείκτης έχει φτάσει στο τέλος του αρχείου, αν ναι επιστρέφει true αλλιώς false. Ως είσοδο παίρνει την μεταβλητή που αντιπροσωπεύει το αρχείο.

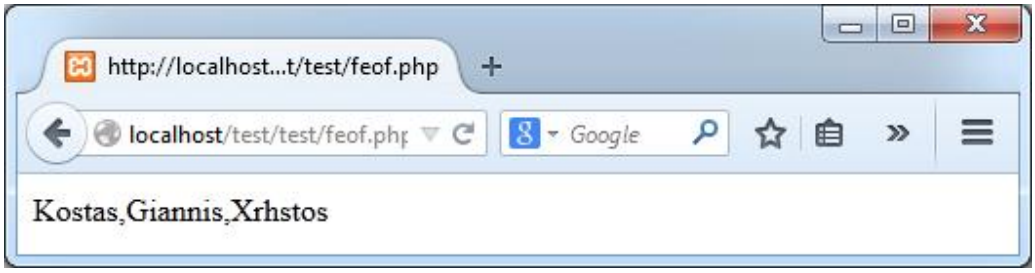
Αρχείο: names.txt

```
Kostas,Giannis,Xrhstos
```

Αρχείο: feof.php

```php
<?php
 $file=fopen("names.txt",'r');
 while(!feof($file)){
  echo fgets($file);
  echo "<br/>";
 }
 fclose($file);
?>
```

Αποτέλεσμα:



Εικόνα Α.47 Αποτέλεσμα χρήσης feof στην τύπωση δεδομένων από αρχείο

A.2.5.2.6 filesize

Επιστρέφει το μέγεθος του αρχείου σε bytes σε μια ακέραιη τιμή. Αν συμβεί κάποιο λάθος επιστρέφει false. Ως είσοδο παίρνει το όνομα του αρχείου.

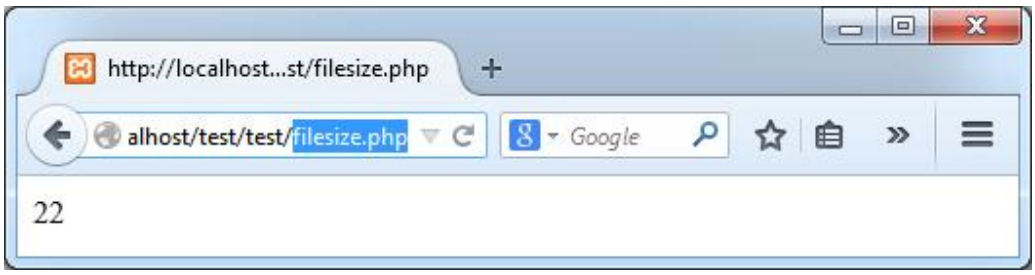
Αρχείο: names.txt

```
Kostas,Giannis,Xrhstos
```

Αρχείο: filesize.php

```
<?php
 echo filesize("names.txt");
?>
```

Αποτέλεσμα:



Εικόνα Α.48 Χρήση εντολής filesize για την γνωστοποίηση του μεγέθους ενός αρχείου

A.2.5.3 Οι χαρακτήρες διαχείρισης

Κάθε χαρακτήρας καθορίζει το πώς θα συμπεριφερθεί η fopen στο αρχείο.

Χαρακτήρας	Επεξήγηση
r	Μόνο ανάγνωση.
w	Μόνο γραφή. Αν το αρχείο δεν υπάρχει, δημιουργείται.
r+	Γραφή και ανάγνωση.
w+	Γραφή και ανάγνωση. Αν το αρχείο δεν υπάρχει, δημιουργείται.
a	Πρόσθεση σε ήδη υπάρχων αρχείο. Αν δεν υπάρχει δημιουργείται
a+	Γραφή και ανάγνωση σε ήδη υπάρχων αρχείο. Αν δεν υπάρχει, δημιουργείται.

Πίνακας Α.7 Χαρακτήρες διαχείρησης αρχείων με την fopen

A.2.5.4 Δείκτης

Ο δείκτης μας βοηθά να γνωρίζουμε που βρισκόμαστε κατά την διάρκεια της εγγραφής και ανάγνωσης ενός αρχείου. Όταν αλληλεπιδρούμε με ένα αρχείο … πχ γράφουμε σε αυτό μια λέξη, τότε η επόμενη λέξη θα γραφτεί αμέσως μετά την προηγούμενη, και όχι στην αρχή του αρχείου. Επομένως υπάρχει «κάτι» το οποίο κρατάει το σημείο στο οποίο «έφτασε» η εγγραφή ή η ανάγνωση.

Γι αυτό τον λόγο χρησιμοποιούνται οι παρακάτω συναρτήσεις.

A.2.5.4.1 fseek

Τοποθετεί τον δείκτη αρχείου σε συγκεκριμένη θέση μέσα στο αρχείο. Αν αποτύχει επιστρέφει -1 αλλιώς 0. Παίρνει δυο παραμέτρους μία μεταβλητή που αναφέρεται στο συγκεκριμένο αρχείο και έναν αριθμό που αναφέρεται σε ποιο σημείο να μεταφερθεί ο δείκτης.

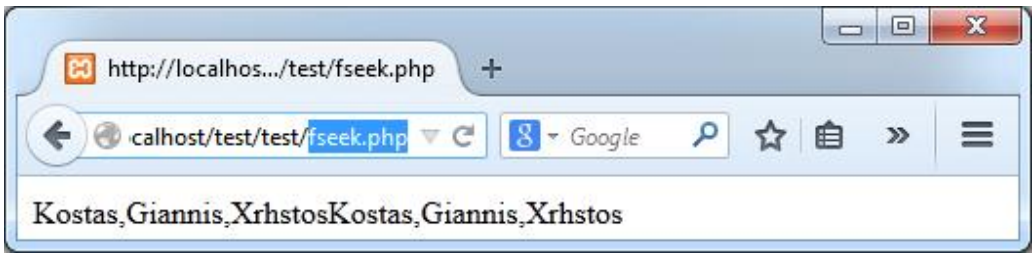
Αρχείο: names.txt

```
Kostas,Giannis,Xrhstos
```

Αρχείο: fseek.php

```
<?php
 $file = fopen("names.txt","r");
 echo fgets($file);  // διαβάζει την πρώτη γραμμή
 fseek($file,0);// μεταφέρει τον δείκτη στην αρχή
 echo fgets($file); //ξανά διαβάζει την πρώτη γραμμή
 fclose($file);
?>
```

Αποτέλεσμα:



Εικόνα Α.49 Αποτέλεσμα εκτέλεσης fseek για ανάγνωση με την χρήση του δείκτη

A.2.5.4.2 rewind

Τοποθετεί τον δείκτη στην αρχή του αρχείου. Αν αποτύχει επιστρέφει false αλλιώς true.

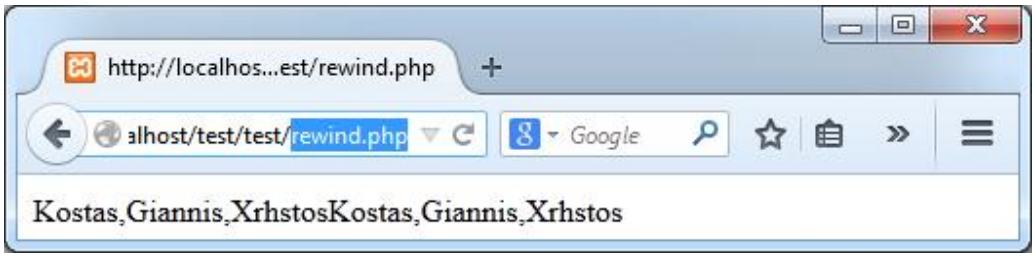
Αρχείο: names.txt

```
Kostas,Giannis,Xrhstos
```

Αρχείο: rewind.php

```
<?php
 $file = fopen("names.txt","r");
 echo fgets($file);  // διαβάζει την πρώτη γραμμή
 rewind($file);// μεταφέρει τον δείκτη στην αρχή.
 echo fgets($file); //ξανά διαβάζει την πρώτη γραμμή
 fclose($file);
?>
```

Αποτέλεσμα:



Εικόνα Α.50 Αποτέλεσμα εκτέλεσης rewind για ανάγνωση με την χρήση του δείκτη

Α.3 ΜΕΤΑΒΛΗΤΕΣ ΚΑΙ ΤΕΛΕΣΤΕΣ

Σε αυτό το κεφάλαιο θα μελετήσουμε τους τύπους δεδομένων που υπάρχουν στην php, θα γνωρίσουμε τι είναι οι μεταβλητές και θα μάθουμε να κάνουμε πράξεις με αυτές.

Α.3.1 Τύποι Δεδομένων

Στην php υπάρχουν 8 διαφορετικοί τύποι δεδομένων, από τους οποίους οι τέσσερις είναι διακριτοί (όπως ακέραιοι, πραγματικοί, αλφαριθμητικά και booleans), δύο είναι σύνθετοι τύποι (πίνακες και αντικείμενα), και οι δυο τελευταίοι είναι ειδικοί τύποι (κενό και αναφορές). Σε αυτή την ενότητα θα επεξηγήσουμε μόνο τους διακριτούς τύπους δεδομένων. Για τους πίνακες τα αντικείμενα και τους ειδικούς τύπους θα ανατρέξετε σε επόμενες ενότητες.

Α.3.1.1 Ακέραιοι

Οι ακέραιοι αριθμοί είναι όλοι οι αριθμοί χωρίς υποδιαστολή όπως το 0, το 23, και το 333. Το αποδεκτό εύρος συνήθως εξαρτάται από την εκάστοτε πλατφόρμα που

χρησιμοποιούμε αλλά συνήθως είναι μεταξύ -2.147.483.648 έως +2.147.483.647. Οι αριθμοί αυτοί μπορούν να αναπαρασταθούν με διάφορους τρόπους, όπως με το δεκαδικό σύστημα, το δυαδικό, το οκταδικό ή το δεκαεξαδικό. Αν χρησιμοποιήσουμε δεκαδική αναπαράσταση δεν χρειάζεται να συμπεριλάβουμε τα αρχικά μηδενικά. Πριν τον αριθμό μπορούμε να συμπεριλάβουμε έναν προσδιοριστή πρόσθεσης (+) ή μείωσης (-) με τον οποίο καθορίζουμε αν ο αριθμός είναι θετικός ή αρνητικός. Αν δεν χρησιμοποιηθεί ένας τέτοιος προσδιοριστής τότε αυτομάτως θεωρείται πως πρόκειται για θετικό(+).

```
2014
-500
+99
```

Για να χρησιμοποιήσουμε την αναπαράσταση σε οκταδικό θα πρέπει μπροστά από κάθε αριθμό να συμπεριλάβουμε το μηδέν. Όπως γνωρίζουμε οκταδικοί μπορούν να πάρουν τιμές από το 0 έως το 7. Παρομοίως κι εδώ μπορούμε να χρησιμοποιήσουμε αρνητικό και θετικό προσδιοριστή.

```
03736 //Είναι το δεκαδικό 2014
-0764 //Είναι το δεκαδικό -500
0143 //Είναι το δεκαδικό 99
```

Στους δεκαεξαδικούς αριθμούς θα πρέπει να χρησιμοποιήσουμε μπροστά από κάθε αριθμό το πρόθεμα «0x». Στο δεκαεξαδικό σύστημα χρησιμοποιούνται αριθμοί από το 0 έως το 9 και γράμματα από το A έως το F (λατινική αλφάβητος). Τα γράμματα μπορούν να είναι κεφαλαία η μικρά καθώς επίσης μπορούμε να συμπεριλάβουμε αρνητικό ή θετικό προσδιοριστή.

```
0x7de //Είναι το δεκαδικό 2014
-0x1f4 //Είναι το δεκαδικό -500
0x63 //Είναι το δεκαδικό 99
```

Για να χρησιμοποιήσουμε δυαδικούς αριθμούς θα πρέπει να προσθέσουμε στην αρχή το πρόθεμα «0b» ακολουθούμενο από αριθμούς 0 ή 1, που αντιπροσωπεύουν τον αριθμό μας. Παρόμοια με όλες τις άλλες περιπτώσεις κι εδώ μπορούμε να συμπεριλάβουμε θετικούς ή αρνητικούς προσδιοριστές.

```
0b11111011110 //Είναι το δεκαδικό 2014
-0b111110100 //Είναι το δεκαδικό -500
0b1100011 //Είναι το δεκαδικό 99
```

Τέλος υπάρχει και η επιστημονική αναπαράσταση αριθμών, όπου δεν χρειάζεται κανένα είδος πρόθεμα, αλλά μπορεί να χρησιμοποιηθεί θετικός ή αρνητικός προσδιοριστής.

```
2.014E3 //Είναι το δεκαδικό 2014
-5.0E2 //Είναι το δεκαδικό -500
9.9E1 //Είναι το δεκαδικό 99
```

Εάν για οποιοδήποτε λόγο θέλουμε να ελέγξουμε αν ένας αριθμός είναι ακέραιος μπορούμε να το κάνουμε με την εντολή «is_int()» η οποία παίρνει ως όρισμα εισόδου, μέσα στις παρενθέσεις της, έναν αριθμό και μπορεί να μας δώσει ως έξοδο «true» ή «false», ανάλογα αν ο αριθμός είναι ή δεν είναι ακέραιος.

```
is_int(14);
```

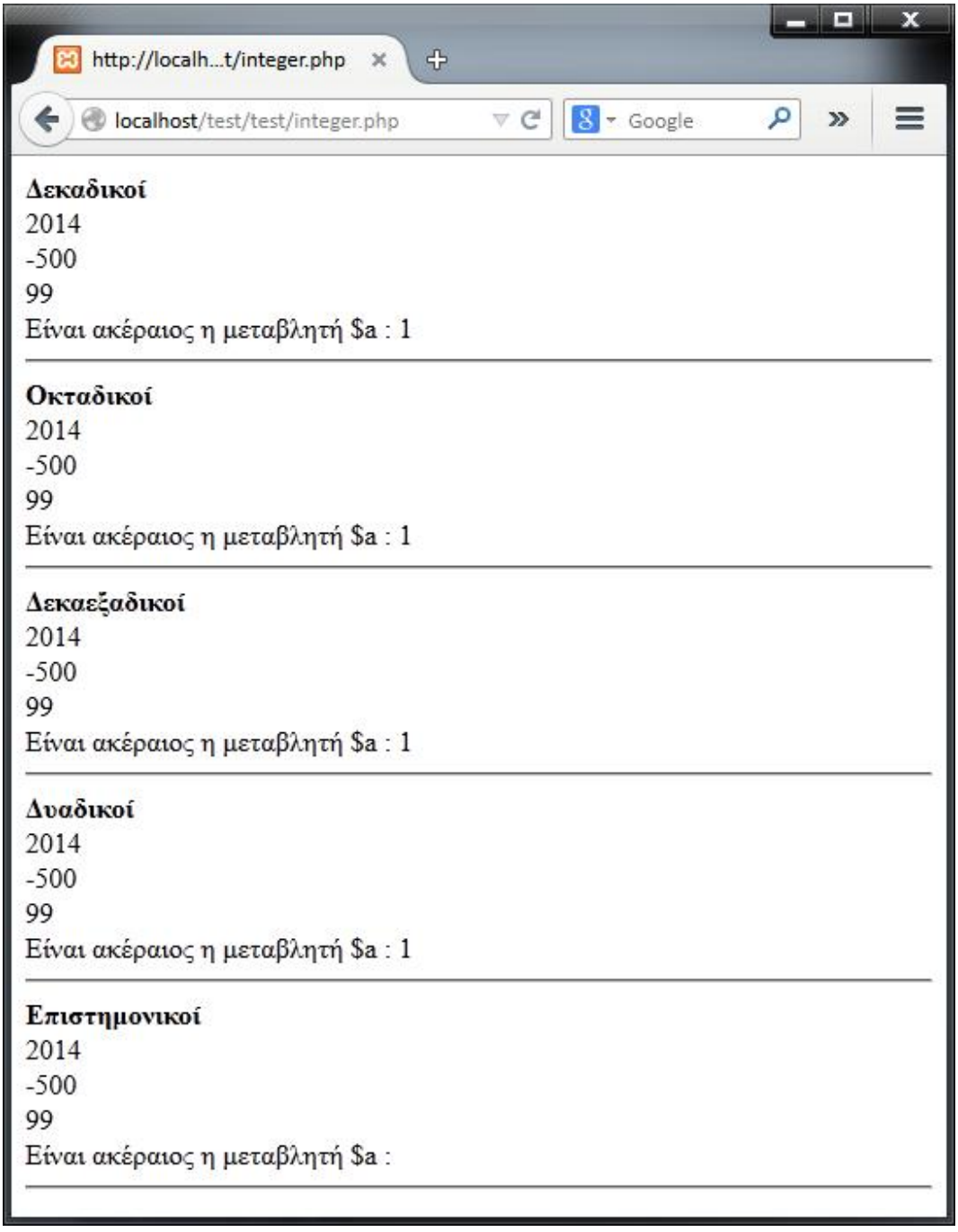
Η παραπάνω εντολή θα μας δώσει ως αποτέλεσμα «true». Με αυτόν τον τρόπο μπορούμε να χρησιμοποιήσουμε οποιοδήποτε είδος ελέγχου, όπως οι εντολές if, switch κ.α. και να ελέγξουμε την ροή του προγράμματος μας, με βάση το αν είναι ή δεν είναι ακέραιος το όρισμα της «is_int()». Εδώ θα πρέπει να σημειώσουμε ότι η εντολή αυτή λειτουργεί με όλες τις μορφές αναπαράστασης εκτός από την επιστημονική.

Παραθέτουμε ένα ολοκληρωμένο παράδειγμα χρήσης όλων των παραπάνω:

Αρχείο: integer.php

```php
<?php
 //Δεκαδικοί
 $a = 2014;
 $b = -500;
 $c = +99;
 echo "<b>Δεκαδικοί</b> <br/>";
 echo $a."<br/>".$b."<br/>".$c."<br/>";
 echo 'Είναι ακέραιος η μεταβλητή $a : '.is_int($a).'<br/><hr/>';

 //Οκταδικοί
 $a=03736; //Είναι το δεκαδικό 2014
 $b=-0764; //Είναι το δεκαδικό -500
 $c=0143; //Είναι το δεκαδικό 99
 echo "<b>Οκταδικοί</b> <br/>";
 echo $a."<br/>".$b."<br/>".$c."<br/>";
 echo 'Είναι ακέραιος η μεταβλητή $a : '.is_int($a).'<br/><hr/>';

 //Δεκαεξαδικοί
 $a=0x7de; //Είναι το δεκαδικό 2014
 $b=-0x1f4; //Είναι το δεκαδικό -500
 $c=0x63; //Είναι το δεκαδικό 99
 echo "<b>Δεκαεξαδικοί</b> <br/>";
 echo $a."<br/>".$b."<br/>".$c."<br/>";
 echo 'Είναι ακέραιος η μεταβλητή $a : '.is_int($a).'<br/><hr/>';

 //Δυαδικοί
 $a=0b11111011110; //Είναι το δεκαδικό 2014
 $b=-0b111110100; //Είναι το δεκαδικό -500
 $c=0b1100011; //Είναι το δεκαδικό 99
 echo "<b>Δυαδικοί</b> <br/>";
 echo $a."<br/>".$b."<br/>".$c."<br/>";
 echo 'Είναι ακέραιος η μεταβλητή $a : '.is_int($a).'<br/><hr/>';

 //Επιστημονικοί
 $a=2.014E3; //Είναι το δεκαδικό 2014
 $b=-5.0E2; //Είναι το δεκαδικό -500
 $c=9.9E1; //Είναι το δεκαδικό 99
 echo "<b>Επιστημονικοί</b> <br/>";
 echo $a."<br/>".$b."<br/>".$c."<br/>";
 echo 'Είναι ακέραιος η μεταβλητή $a : '.is_int($a).'<br/><hr/>';
?>
```

Αποτέλεσμα:



Εικόνα Α.51 Αποτέλεσμα εμφάνισης όλων των τρόπων χρήσης ενός ακέραιου αριθμού

Εδώ να σημειώσουμε ότι το «1» σημαίνει επίσης «true», ενώ το απόλυτο κενό, «false».

A.3.1.2 Πραγματικοί

Στην php εκτός από ακέραιους αριθμούς μπορούμε να χρησιμοποιήσουμε και αριθμούς με υποδιαστολή (άλλοτε γνωστοί και ως πραγματικοί αριθμοί) το εύρος των οποίων εξαρτάται από την αντίστοιχη πλατφόρμα, συνήθως όμως είναι από 1.7E-308 έως 1.7E+308 με 15 ψηφία ως μέγιστη ακρίβεια. Εάν θέλουμε παραπάνω ακρίβεια μπορούμε να χρησιμοποιήσουμε κάποιο επιπρόσθετο όπως το BC ή το GMP. Όπως με τους ακέραιους έτσι κι εδώ η php αναγνωρίζει τους πραγματικούς αριθμούς μέσα από μία πληθώρα αναπαραστάσεων, όπως:

- Τον κλασικό τρόπο που χρησιμοποιούμε στην καθημερινότητα μας, μόνο που αντί για κόμμα εδώ χρησιμοποιούμε την τελεία στην θέση του, επίσης μπορούμε να χρησιμοποιήσουμε θετικό ή αρνητικό προσδιοριστή:

```
3.14
0.0001
-30.023
```

- Και τον επιστημονικό τρόπο όπου μπορούμε επίσης να συμπεριλάβουμε θετικό ή αρνητικό προσδιοριστή:

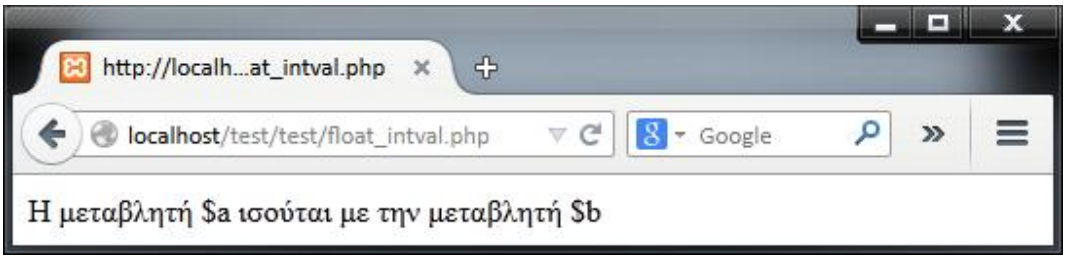
```
3.14E0 //Ο πραγματικός 3.14
1.0E-4 //Ο πραγματικός 0.0001
-3.0023E1 //Ο πραγματικός -30.023
```

Συνήθως σε πολλές περιπτώσεις τυχαίνει να θέλουμε να ελέγξουμε την ακεραιότητα μιας τιμής πραγματικού αριθμού μέσα σε μια συνθήκη όπως ο αριθμός «1.999999». Για να μπορέσουμε να το κάνουμε αυτό θα πρέπει να μετατρέψουμε τον αριθμό αυτό πρώτα σε έναν ακέραιο και ύστερα να συγκρίνουμε με βάση όλα τα ψηφία του ακέραιου πλέον που θα έχουμε μεταφέρει από την δεξιά πλευρά στην αριστερή της υποδιαστολής. Πιο συγκεκριμένα εάν θέλουμε να ελέγξουμε την τιμή 1.999999 θα πρέπει να την πολλαπλασιάσουμε με το 100000 έτσι ώστε δεξιά της υποδιαστολής να υπάρχουν μόνο μηδενικά, και μετά να μετατρέψουμε αυτή την τιμή σε ακέραιο με την εντολή «intval()», Παράδειγμα:

Αρχείο: float_intval.php

```php
<?php
 $a=1.999999;
 $b=1.999999;
 if(intval($a*1000000)==intval($b*1000000)){
  echo 'Η μεταβλητή $a ισούται με την μεταβλητή $b <br/>';
 }
?>
```

Αποτέλεσμα:



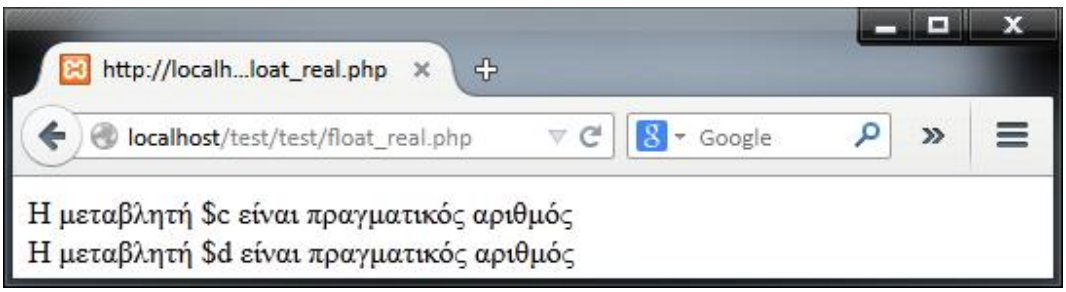
Εικόνα Α.52 Τρόπος χρήσης αριθμών με υποδιαστολή σε δομές ελέγχου

Όπως με τους ακέραιους έτσι κι εδώ μπορούμε να ελέγξουμε αν μια τιμή είναι πραγματικός αριθμός. Υπάρχουν δυο εντολές η «is_float()» και η «is_real()» οι οποίες κάνουν ακριβώς το ίδιο:

Αρχείο: float_real.php

```php
<?php
 $c=-30.023;
 $d=3.14E0; //Ο πραγματικός 3.14
 if(is_float($c)){
  echo 'Η μεταβλητή $c είναι πραγματικός αριθμός <br/>';
 }

 if(is_real($d)){
  echo 'Η μεταβλητή $d είναι πραγματικός αριθμός <br/>';
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.53 Έλεγχος μεταβλητής για το αν είναι πραγματικός αριθμός

A.3.1.3 Αλφαριθμητικά

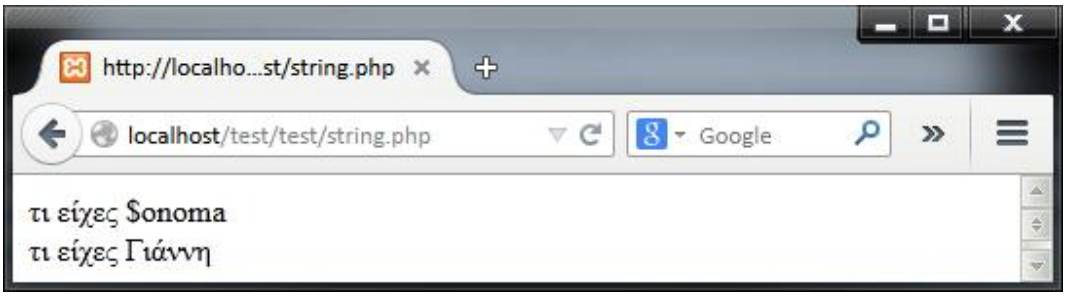
Τα αλφαριθμητικά χρησιμοποιούνται πολύ συχνά σε διαδικτυακές εφαρμογές και γι αυτό η php διαθέτει αυτή την δυνατότητα ενσωματωμένη στον πυρήνα της, δίνοντας μας έτσι την ευχέρεια να δημιουργούμε και να κάνουμε πράξεις με αλφαριθμητικά με μεγάλη ευκολία που σε άλλες γλώσσες αυτό δεν υποστηρίζεται.

Ένα αλφαριθμητικό είναι μια σειρά από χαρακτήρες είτε του ελληνικού αλφαβήτου είτε του λατινικού (είτε μίξη και των δύο), αριθμών και ειδικών συμβόλων, μπορούμε δηλαδή να γράψουμε ένα όνομα, μια πρόταση ή και ολόκληρο κείμενο μέσα σε ένα αλφαριθμητικό. Ο τρόπος με τον οποίο χρησιμοποιούμε τα αλφαριθμητικά είναι να γράψουμε αυτό που θέλουμε μέσα σε μονά ή διπλά εισαγωγικά, για παράδειγμα:

```
'Τι είχες Γιάννη;'
"Τι είχα πάντα;"
```

Η διαφορά τους βρίσκεται στο ότι στα μονά εισαγωγικά δεν μπορούμε να χρησιμοποιήσουμε ειδικούς χαρακτήρες αλφαριθμητικών (όπως αλλαγή γραμμής κτλ) ή μεταβλητές μέσα σε αυτά, αλλά μόνο απλό κείμενο. Αντίθετα στα διπλά εισαγωγικά εάν συμπεριλάβουμε το όνομα μιας μεταβλητής μέσα σε ένα αλφαριθμητικό τότε χρησιμοποιείται η τιμή της μεταβλητής αυτής ως αντικατάσταση, όπου βρεθεί στο αλφαριθμητικό, παράδειγμα:

```
<?php
 $onoma = "Γιάννη";
 echo 'τι είχες $onoma <br/>';
 echo "τι είχες $onoma <br/>";
?>
```



Εικόνα Α.54 Χρήση μεταβλητών μέσα σε αλφαριθμητικά

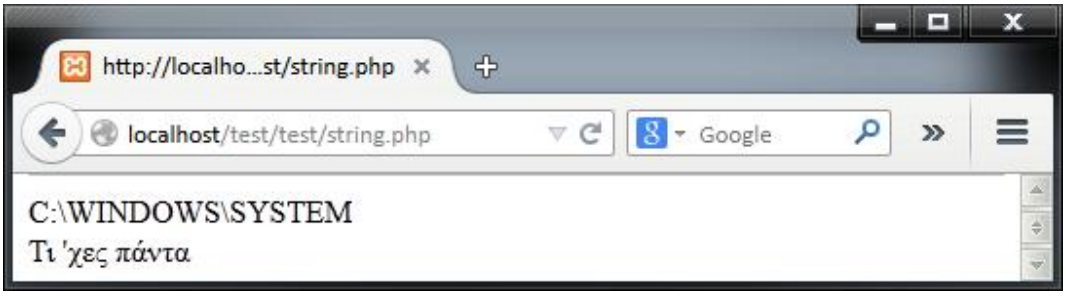
Οι ειδικοί χαρακτήρες που υποστηρίζονται μέσα στα διπλά εισαγωγικά είναι όλα αυτά που περιγράφονται στον παρακάτω πίνακα (Πίνακας Α.8).

Χαρακτήρας	Επεξήγηση
\"	Διπλά εισαγωγικά
\n	Νέα γραμμή
\r	
\t	Tab
\\	Οπίσθια μεσοκάθετος
\$	Δολάριο
\{	Αριστερή αγκύλη
\}	Δεξιά αγκύλη
\[	Αριστερό άγκιστρο
\]	Δεξιό άγκιστρο
Από \0 έως \777	Ένας χαρακτήρας ASCII σε οκταδική μορφή
Από \x0 έως \xff	Ένας χαρακτήρας ASCII σε δεκαεξαδική μορφή

Πίνακας Α.8 Ειδικοί χαρακτήρες αλφαριθμητικών μέσα σε διπλά εισαγωγικά

Στα μονά εισαγωγικά αναγνωρίζονται οι διπλές μεσοκάθετοι ως μοναδική μεσοκάθετος και το «\'» ως μονό εισαγωγικό.

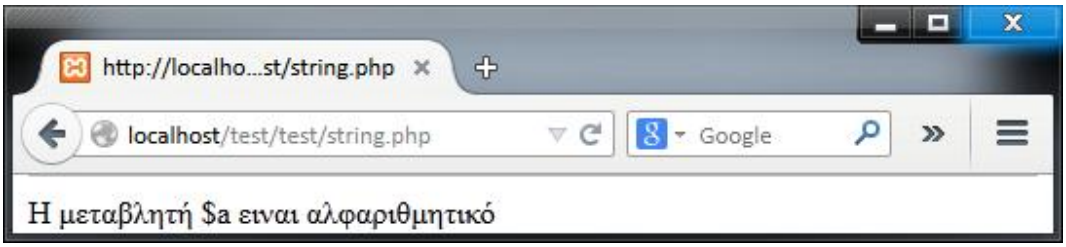
```
<?php
 $diadromi = 'C:\\WINDOWS\\SYSTEM ';
 $frasi = 'Τι \'χες πάντα';
?>
```



Εικόνα Α.55 Χρήση εισαγωγικών μέσα σε αλφαριθμητικά με μονά εισαγωγικά

Για να ελέγξουμε αν μια μεταβλητή είναι αλφαριθμητικό χρησιμοποιούμε την εντολή «is_string()» όπου ως όρισμα εισόδου μέσα στις παρενθέσεις της παίρνει μια μεταβλητή προς έλεγχο και επιστρέφει true ή false.

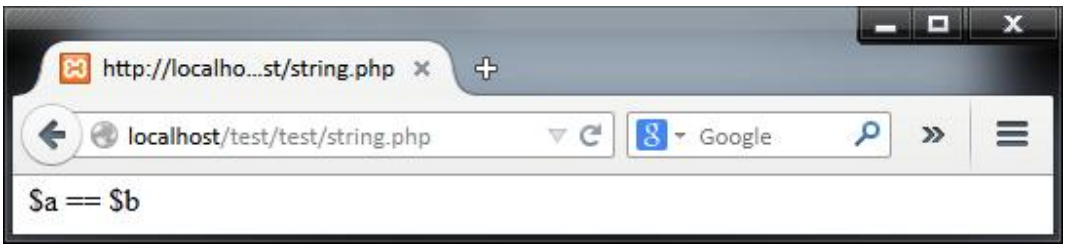
```
<?php
 $a = "Αλφαριθμητικό";
 if(is_string($a)){
   echo 'Η μεταβλητή $a είναι αλφαριθμητικό';
 }
?>
```



Πίνακας Α.9 Χρήση μεθόδου is_string()

Για να συγκρίνουμε δυο αλφαριθμητικά μπορούμε απλά να χρησιμοποιήσουμε τον αντίστοιχο τελεστή του διπλού ίσον «==», για παράδειγμα:

```
<?php
 $a = "Γειά σου";
 $b = "Γειά σου";
 if($a==$b){
  echo '$a == $b';
 }
?>
```



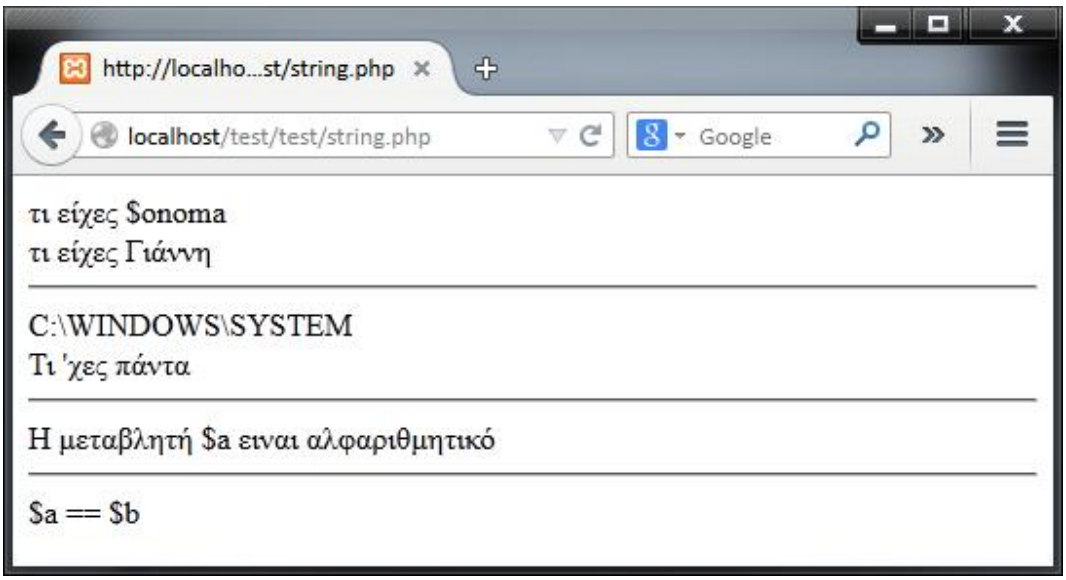
Πίνακας Α.10 Σύγκριση αλφαριθμητικών με τελεστή ισότητας

Όλα τα παραπάνω παραδείγματα σε ένα αρχείο:

Αρχείο: string.php

```php
<?php
 //'Τι είχες Γιάννη;'
 //"Τι είχα πάντα;"

 $onoma = "Γιάννη";
 echo 'τι είχες $onoma <br/>';
 echo "τι είχες $onoma <br/>";

 echo '<hr/>';

 $diadromi = 'C:\\WINDOWS\\SYSTEM ';
 $frasi = 'Τι \'χες πάντα';

 echo $diadromi."<br/>".$frasi;

 echo "<hr/>";

 $a = "Αλφαριθμητικό";
 if(is_string($a)){
  echo 'Η μεταβλητή $a ειναι αλφαριθμητικό';
 }

 echo '<br/>';
 echo '<hr/>';
 $a = "Γειά σου";
 $b = "Γειά σου";
 if($a==$b){
  echo '$a == $b';
 }
?>
```

Αποτέλεσμα:



Πίνακας Α.11 Αποτέλεσμα εκτέλεσης αρχείου string.php

A.3.1.4 Boolean

Μια μεταβλητή Boolean είναι μια μεταβλητή με περιεχόμενο το οποίο μπορεί να αναπαραστήσει αν κάτι είναι αληθές ή ψευδές. Αν για παράδειγμα θέλουμε να ρωτήσουμε κάποιον «βρέχει σήμερα;» και η απάντηση είναι «ναι» τότε στις γλώσσες προγραμματισμού η απάντηση που πήραμε είναι μια αληθής Boolean τιμή όπου μας απαντά σε ένα ερώτημα με ναι ή όχι. Συνήθως στις γλώσσες προγραμματισμού χρησιμοποιούνται οι τιμές «true» και «false» ως ναι και όχι αντίστοιχα. Στην php όμως το εύρος των τιμών αυξάνεται, και μαζί με αυτό και οι δυνατότητες που μας παρέχουν.

Οι παρακάτω τιμές στην php προσδιορίζουν το «όχι»:

- Η λέξη κλειδί false,
- Ο ακέραιος αριθμός 0,
- Ο πραγματικός 0.0,
- Ένα κενό αλφαριθμητικό "" ή ένα αλφαριθμητικό "0",
- Ένας κενός πίνακας με κανένα στοιχείο,
- Ένα αντικείμενο χωρίς τιμές ή λειτουργίες,
- Η τιμή NULL.

Μια τιμή η οποία δεν έχει κανέναν από τα παραπάνω περιεχόμενα τότε δεν θεωρείται false, και είναι true. Η php προσφέρει ωστόσο τις λέξεις κλειδιά «true» και «false» για ευκολία στην ανάγνωση αλλά μπορούν να χρησιμοποιηθούν και όλες οι παραπάνω τιμές.

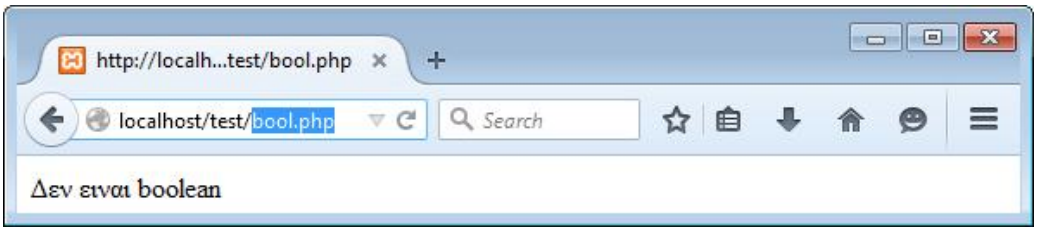
```
<?php
 $a = 5; // true
 $b = true; // true
 $c = ""; // false
 $d = false; //false
?>
```

Για να ελέγξουμε αν μια μεταβλητή είναι boolean αρκεί να χρησιμοποιήσουμε την εντολή «is_bool()» με μόνο όρισμα εισόδου την μεταβλητή που θέλουμε να ελέγξουμε.

Αρχείο: bool.php

```php
<?php
 $a = 5; // true
 if(is_bool($a)){
  echo "ειναι boolean!";
 }else{
  echo "Δεν ειναι boolean";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.56 Έλεγχος μεταβλητής με την εντολή is_bool

A.3.1.5 NULL

Υπάρχει μόνο ένα είδος κενής τιμής στην php και αυτή είναι η NULL. Το NULL το οποίο είναι case-insensitive (μπορεί να γραφτεί δηλαδή και με μικρά αλλά και με κεφαλαία και σημαίνει ακριβώς το ίδιο), αναπαριστά μια τιμή σε μια μεταβλητή της οποίας μεταβλητής δεν έχει αποδοθεί καμία τιμή. Όταν ορίζουμε μια μεταβλητή με αυτή την τιμή σημαίνει πως η μεταβλητή δεν έχει απολύτως τίποτα μέσα της.

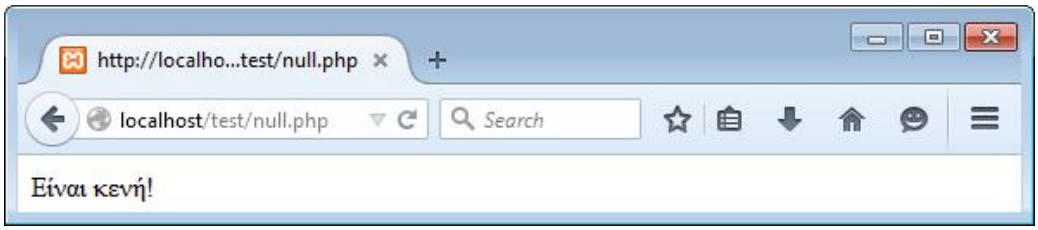
```php
$a = "Γεια σας!"; // Έχει τιμή
$a = null; //τώρα είναι κενή
$a = NuLl; //επίσης κενή
$a = NULL; // επίσης κενή
```

Μπορούμε να ελέγξουμε αν μια μεταβλητή είναι κενή με την εντολή is_null() η οποία δέχεται ως είσοδο μια μεταβλητή και επιστρέφει true αν η μεταβλητή είναι κενή και false αν δεν είναι.

Αρχείο: null.php

```php
<?php
 $a = null;
 if(is_null($a)){
  echo "Είναι κενή!";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.57 Έλεγχος μεταβλητής με την εντολή is_null

Α.3.2 Μεταβλητές

Οι μεταβλητές είναι ένας δεσμευμένος χώρος μνήμης RAM στον server όπου είναι εγκατεστημένη η php και τρέχει εκείνη την στιγμή το script μας. Μέσα σε αυτόν τον χώρο μνήμης αποδίδονται διάφορα είδη τιμών. Για να μπορέσουμε να σας εξηγήσουμε τι ακριβώς είναι οι μεταβλητές θα πρέπει να πάρουμε ένα παράδειγμα από την καθημερινή μας ζωή.

Καθημερινά στο σπίτι μας, χρησιμοποιούμε ένα σωρό από διαφορετικά «δοχεία» για να μπορέσουμε να αντεπεξέλθουμε στις διάφορες ανάγκες μας. Για παράδειγμα στην κουζίνα, το ποτήρι, το φλιτζάνι, η κανάτα, τα tapware, οι κατσαρόλες κ. α. εξυπηρετούν κάποια συγκεκριμένη χρήση. Φιλοξενούν μέσα σε αυτά κάποιο υλικό, είτε υγρό, είτε στερεό, είτε οτιδήποτε για προσωρινό διάστημα.

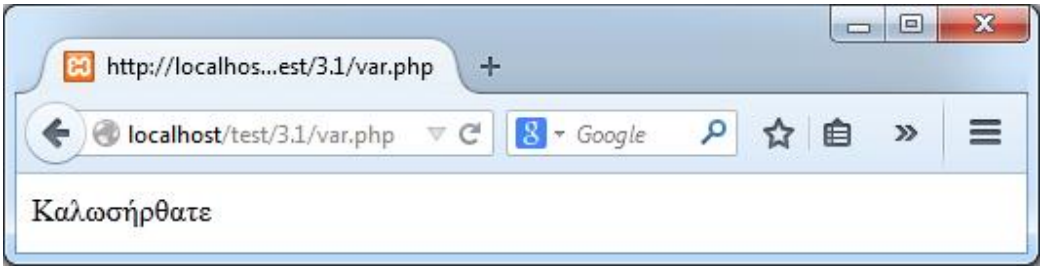
Στην ουσία οι μεταβλητές χρησιμοποιούνται σε κάθε γλώσσα για να δεσμεύσουν κομμάτια μνήμης στα οποία αποθηκεύονται προσωρινά αριθμοί, χαρακτήρες, κείμενα κ.α. Τα περιεχόμενα των μεταβλητών αυτών μπορούν να είναι οποιοδήποτε είδος τύπου δεδομένων. Βασικός κανόνας στην php είναι ότι τα ονόματα των μεταβλητών ξεκινούν με τον χαρακτήρα «$» (δολάριο). Έτσι αναγνωρίζει τις μεταβλητές ανάμεσα σε άλλα κείμενα.

Στο παρακάτω παράδειγμα, καταχωρείται στη μεταβλητή «$vartest» το (αλφαριθμητικό) κείμενο «Καλωσήρθατε»:

Αρχείο: var.php

```
<?php
 $vartest = "Καλωσήρθατε";
 echo $vartest;
?>
```

Αποτέλεσμα:



Εικόνα Α.58 Τύπωση περιεχομένου μεταβλητής

Α.3.2.1 Κανόνες ονομασίας μεταβλητών

Στην php υπάρχουν κάποιοι συγκεκριμένοι κανόνες με τους οποίους ονομάζουμε τις μεταβλητές μας, έτσι ώστε κατά την διάρκεια της εκτέλεσης η php να μην μπερδεύει το υπόλοιπο κείμενο με τις μεταβλητές που έχουμε.

- Τα ονόματα μεταβλητών θα πρέπει να ξεκινούν με τον χαρακτήρα «$» (δολάριο),
- Μετά το χαρακτήρα δολάριο το όνομα μεταβλητής μπορεί να ξεκινάει από γράμμα ή τον χαρακτήρα της κάτω παύλας «_»,
- Μια μεταβλητή στο σύνολο της ονομασίας της μπορεί να περιλαμβάνει αλφαριθμητικούς λατινικούς χαρακτήρες και κάτω παύλες (a-z,A-Z,0-9, και _),
- Μέσα στα ονόματα δεν επιτρέπεται η χρήση κενών χαρακτήρων « ». Αντιθέτως γι αυτόν τον σκοπό υπάρχει η κάτω παύλα, έτσι αν θέλουμε την μεταβλητή «var test» να την χρησιμοποιήσουμε στην php θα την γράφαμε κάπως έτσι: «$var_test» ή έτσι «$vartest».
- Η γλώσσα php είναι case sensitive και αυτό σημαίνει πως μια μεταβλητή με ίδιο όνομα αλλά με διαφορετικό τρόπο γραφής θεωρείται διαφορετική, πχ το «$vartest» και το «$VARTEST» ή το «$Vartest» είναι τελείως διαφορετικές μεταβλητές.

Οι μεταβλητές στην php δεν υπόκεινται σε μεταγλώττιση, επομένως μπορούμε να δημιουργήσουμε μια μεταβλητή δίνοντάς της αρχικά κάποιο περιεχόμενο βασιζόμενοι σε έναν από τους τύπους δεδομένων, αλλά στην συνέχεια να αλλάξουμε αυτό το περιεχόμενο σε κάποιο άλλο είδος τύπου δεδομένων. Για παράδειγμα θα μπορούσαμε στην αρχή να είχαμε μια μεταβλητή με περιεχόμενο έναν ακέραιο αριθμό, και στην συνέχεια να αλλάξουμε το περιεχόμενο της μεταβλητής αυτής σε ένα αλφαριθμητικό.

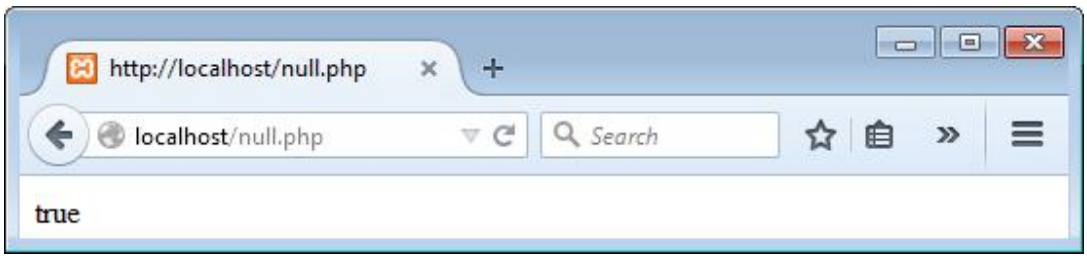
```php
<?php
 $ilikia = 35; //Δημιουργία ακέραιου 35
 $ilikia = "Κώστας"; //Αλλαγή σε αλφαριθμητικό
 $ilikia = false; //Αλλαγή σε boolean
?>
```

Μεταβλητές στις οποίες δεν έχει αποδοθεί κάποιο είδος τιμής, τότε ως αρχικοποιημένη τιμή διαθέτουν την τιμή «NULL». Ωστόσο είναι λάθος να χρησιμοποιούμε μεταβλητές που δεν τους έχουμε ορίσει τιμή. Το σωστότερο είναι να δημιουργούμε την μεταβλητή και αν δεν την χρειαζόμαστε προς το παρόν να την αρχικοποιούμε στην τιμή «null», χειροκίνητα. Για να ελέγξουμε αν μια μεταβλητή είναι «null» ακολουθούμε την παρακάτω διαδικασία:

Αρχείο: null.php

```php
<?php
 $var=null;
 if($var === null){
  echo "true";
 }
?>
```

Αποτέλεσμα:



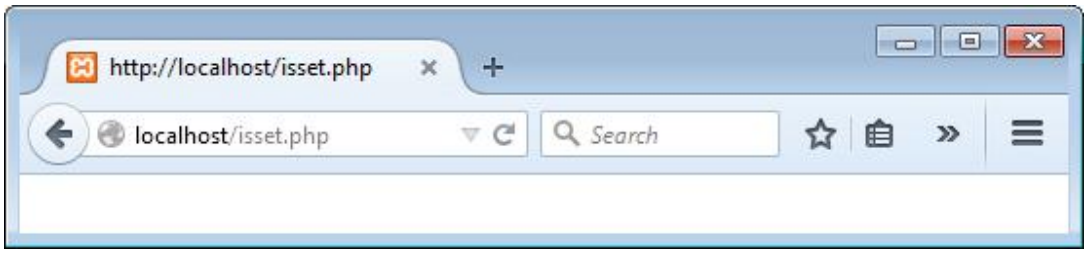
Εικόνα Α.59 Έλεγχος κενής μεταβλητής

Αν θέλουμε να ελέγξουμε αν μια μεταβλητή υπάρχει τότε χρησιμοποιούμε την εντολή «isset()» όπου προσθέτουμε μέσα στις παρενθέσεις την μεταβλητή που θέλουμε να ελέγξουμε, η εντολή αυτή επιστρέφει αντίστοιχα μια τιμή true ή false ανάλογα αν υπάρχει η μεταβλητή ή όχι.

Αρχείο: isset.php

```php
<?php
 if(isset($metavar)){
  echo "true";
 }
?>
```

Αποτέλεσμα:



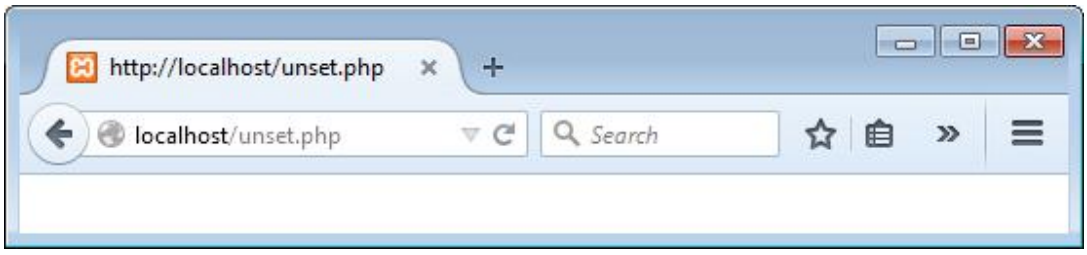
Εικόνα Α.60 Έλεγχος ύπαρξης μεταβλητής

Αν θέλουμε να αποδεσμεύσουμε μια θέση μνήμης διαγράφοντας μια μεταβλητή από αυτήν, τότε χρησιμοποιούμε την εντολή «unset()» όπου μέσα στις παρενθέσεις της εισάγουμε ως όρισμα την μεταβλητή που θέλουμε να αφαιρέσουμε.

Αρχείο: unset.php

```
<?php
 $metavar = "Κώστας";
 unset($metavar);
 echo $metavar;
?>
```

Αποτέλεσμα:



Εικόνα Α.61 Διαγραφή μεταβλητής από την μνήμη

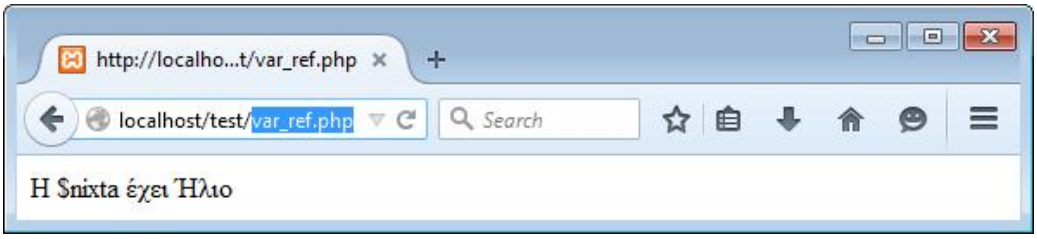
Α.3.2.2 Αναφορές μεταβλητών

Στην php είναι δυνατό να δημιουργήσουμε μια αναφορά προς μια μεταβλητή έτσι ώστε μια θέση μνήμης να διαθέτει δυο διαφορετικά ονόματα που οδηγούν σε αυτήν. Για να γίνει η μεταβλητή «$nixta» αναφορά της θέσης μνήμης της μεταβλητής «$imera» θα πρέπει να συνταχθεί:

Αρχείο: var_ref.php

```php
<?php
 $imera = "Ήλιο";
 $nixta = & $imera;
 echo "Η \$nixta έχει $nixta \n";
?>
```

Αποτέλεσμα:



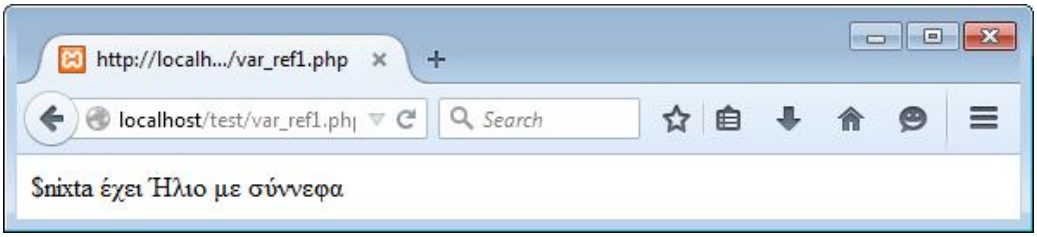
Εικόνα Α.62 Τύπωση περιεχομένου μεταβλητής αναφοράς

Αναθέτουμε δηλαδή την μεταβλητή «$imera» στην «$nixta» μόνο που πριν την ανάθεση χρησιμοποιούμε τον χαρακτήρα «&». Αυτό σημαίνει ότι η «$nixta» θα γίνει αναφορά της «$imera». Όπως βλέπουμε στο αποτέλεσμα χρησιμοποιούμε την αναφορά «$nixta» για να τυπώσουμε το περιεχόμενο της, παρ' όλα αυτά τυπώνεται το περιεχόμενο της μεταβλητής «$imera». Αυτό γίνεται γιατί πλέον η μεταβλητή «$nixta» είναι αναφορά της «$imera», έτσι αν αλλάξουμε την τιμή της «$nixta» αυτόματα θα αλλάξει και η τιμή της «$imera».

Αρχείο: var_ref1.php

```php
<?php
 $imera = "Ήλιο";
 $nixta = & $imera;
 $nixta = "Ήλιο με σύννεφα";
 echo "\$nixta έχει $nixta";
?>
```

Αποτέλεσμα:



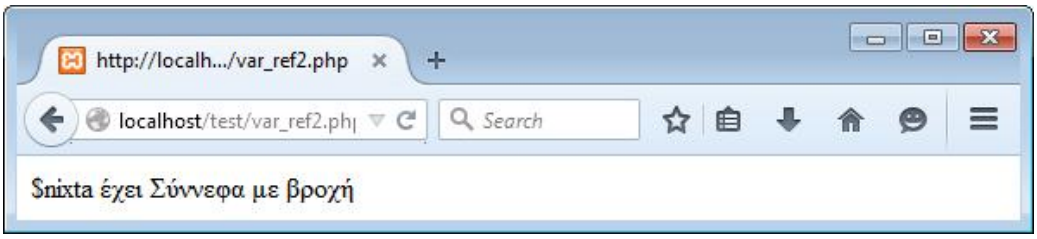
Εικόνα Α.63 Τύπωση αλλαγμένων περιεχομένων μεταβλητών με αναφορές

Το ίδιο ισχύει και αντίστροφα, αν αλλάξει η τιμή της «$imera» τότε όταν χρησιμοποιηθεί η αναφορά «$nixta» θα είναι σαν να χρησιμοποιούμε την μεταβλητή «$imera» στην θέση της.

Αρχείο: var_ref2.php

```
<?php
 $imera = "Ήλιο";
 $nixta = & $imera;
 $imera = "Σύννεφα με βροχή";
 echo "\$nixta έχει $nixta";
?>
```

Αποτέλεσμα:



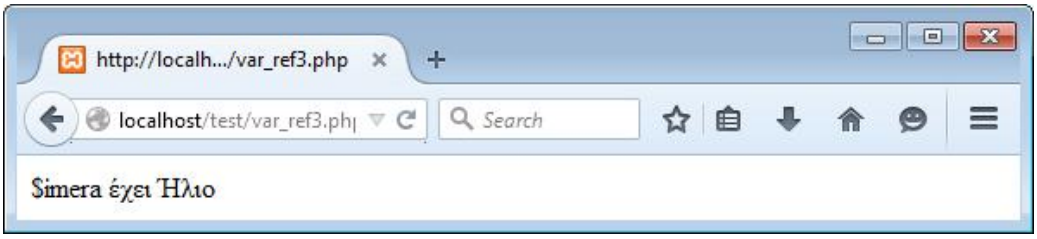
Εικόνα Α.64 Αντίστροφη χρήση αναφορών σε μεταβλητές

Αν διαγράψουμε μια αναφορά προς μια μεταβλητή αυτό δεν επηρεάζει την αρχική μεταβλητή ούτε την τιμή της στην μνήμη.

Αρχείο: var_ref3.php

```
<?php
 $imera = "Ήλιο";
 $nixta = & $imera;
 unset($nixta);
 echo "\$imera έχει $imera";
?>
```

Αποτέλεσμα:



Εικόνα Α.65 Διαγραφή αναφορών και επιρροή σε μεταβλητές

A.3.3 Τελεστές – Operators

Υπάρχουν 4 είδη τελεστών πράξεων στην php τα οποία χρησιμοποιούνται σε συνδυασμό με μια ή περισσότερες μεταβλητές (ή τιμές στις μεταβλητές) για να δώσουν ένα αποτέλεσμα. Στήλη Π = Προτεραιότητα, όπου Α = Αριστερή, Κ = Καμία, Δ = Δεξιά. Στήλη Α = Αύξοντας αριθμός σειράς προτεραιότητας.

Α	Π	Τελεστής	Πράξη
1	Α	,	Διαχωρισμός στοιχείων
2	Α	**or**	Λογικό «Η»
3	Α	**xor**	Λογικό «XOR»
4	Α	**and**	Λογικό «ΚΑΙ»
5	Α	=	Εκχώρηση τιμών
6	Α	?:	Συνθήκη
7	Α	||	Λογικό «Η»
8	Α	&&	Λογικό «ΚΑΙ»
9	Α	|	Δυαδικό «Η»
10	Α	^	Δυαδικό «XOR»
11	Α	&	Δυαδικό «ΚΑΙ»
12	Κ	==	Ισότητα
13	Κ	< ή <=	Μικρότερο, ή μικρότερο ίσο
14	Α	<<	Δυαδική μετατόπιση αριστερά
15	Α	+	Πρόσθεση
16	Α	*	Πολλαπλασιασμός
17	Δ	!	Λογικό «ΟΧΙ»
18	Κ	**instanceof**	Δοκιμή τύπου
19	Δ	~	Δυαδικό «ΟΧΙ»
20	Α	[	Πίνακας
21	Κ	**clone** ή **new**	Δημιουργία νέου αντικειμένου

Πίνακας Α.12 Τελεστές στην php

A.3.3.1 Αριθμητικοί τελεστές

Οι αριθμητικοί τελεστές αναλαμβάνουν να πραγματοποιήσουν μια πράξη μεταξύ δυο συμβαλλόμενων μερών και επιστρέφουν το αποτέλεσμα.

Τελεστής	Περιγραφή	Παράδειγμα	Αποτέλεσμα
+	Πρόσθεση	`$x=4;` `$x+4;`	8
-	Αφαίρεση	`$x=8;` `$x-2;`	6
*	Πολλαπλασιασμός	`$x=3;` `$x*2;`	15
/	Διαίρεση	`$x=15;` `$x/5;`	3
%	Υπόλοιπο διαίρεσης (Modulus)	`$x=5;` `$x%2;`	1

++	Αύξηση κατά ένα	$x=12; $x++;	13
--	Μείωση κατά ένα	$x=12; $x--;	24

Πίνακας Α.13 Αριθμητικοί Τελεστές

Α.3.3.2 Τελεστές σύγκρισης

Οι τελεστές σύγκρισης, αναλόγως την «πράξη» επιστρέφουν δυο τιμές ή true ή false.

Τελεστής	Περιγραφή	Παράδειγμα	Αποτέλεσμα
==	ίσον	3==4	false
===	ίσον και με ίδιο τύπο		
!=	όχι ίσον	3!=4	true
>	μεγαλύτερο	3>4	false
<	μικρότερο	3<4	true
>=	μεγαλύτερο ή ίσον	5>=4	false
<=	μικρότερο ή ίσον	5<=4	true

Πίνακας Α.14 Τελεστές σύγκρισης

Α.3.3.3 Τελεστές ορισμού τιμών

Στις πράξεις μέσω ορισμού τιμών, γίνεται μια πιο σύντομη χρήση του παραγόμενου κώδικα έτσι το x+=y είναι ίσο με το x=x+y.

Τελεστής	Παράδειγμα	Είναι ίσο με
=	$x=$y;	$x=$y;
+=	$x+=$y;	$x=$x+$y;
-=	$x-=$y;	$x=$x-$y;
*=	$x*=$y;	$x=$x*$y;
/=	$x/=$y;	$x=$x/$y;
%=	$x%=$y;	$x=$x%$y;

Πίνακας Α.15 Τελεστές ορισμού τιμών

Α.3.3.4 Λογικοί τελεστές

Οι λογικοί τελεστές χρησιμοποιούνται κυρίως για τον συνδυασμό συγκριτικών τελεστών. Ακολουθούν πιστά τα λογικά «και», «η» ,«όχι», και επιστρέφουν true ή false ανάλογα αν ισχύουν ή όχι.

Τελεστής	Περιγραφή	Παράδειγμα	Αποτέλεσμα
&&	Λογικό «ΚΑΙ»	$x=4;$y=8; ($x>0 && $y<20)	true
||	Λογικό «Η»	$x=12;$y=30; ($x==42 || $y==42)	false
!	Λογικό «ΟΧΙ»	$x=42;$y=43; !($x==$y)	true

Πίνακας Α.16 Λογικοί τελεστές

Α.3.3.5 Προτεραιότητα τελεστών

Όπως και στα καθημερινά μαθηματικά της ζωής μας έτσι και στην PHP υπάρχουν κάποιοι κανόνες σχετικά με την προτεραιότητα των πράξεων και συνεπώς των τελεστών.

```
$a = 3*20-2;
```

Η τιμή της $a θα είναι 58 ή 54 ; για να γίνει σαφέστερη η τοποθέτηση των πράξεων θα μπορούσαν να χρησιμοποιηθούν παρενθέσεις:

```
$a = (3*20)-2;
$a = 3*(20-2);
```

Στην πρώτη περίπτωση το 3 θα πολλαπλασιαστεί με το 20 και μετά θα αφαιρεθεί το 2 έτσι προκύπτει το 58. Ενώ στην δεύτερη περίπτωση πρώτα αφαιρείται το 2 από το 20 και το 18 πολλαπλασιάζεται με το 3 όπου μας δίνει αποτέλεσμα 54.

```
$a = 2-2-2;
```

Στην περίπτωση αυτή όπου έχουμε δύο ίδιες πράξεις, ισχύει ότι θα εκτελεστεί πρώτα η εντολή που βρίσκεται αριστερά. Επειδή το 2 που είναι στην μέση έχει ίδια πράξη και στις δυο πλευρές του και το «-» είναι με αριστερή προτεραιότητα, γίνεται πρώτα η αριστερή πράξη και έτσι έχουμε αποτέλεσμα το -2.

Γενικότερα ισχύει:

Προτεραιότητα	Τελεστής
Αριστερή	,
Αριστερή	or
Αριστερή	xor
Αριστερή	and
Δεξιά	print
Δεξιά	= += -= *= /= .= %= &= |= ^= <<= >>=
Αριστερή	?:
Αριστερή	||
Αριστερή	&&
Αριστερή	|
Αριστερή	^
Αριστερή	&
Καμία	== != === !==
Καμία	< <= > >=
Αριστερή	<< >>

Αριστερή	`+ - .`
Αριστερή	`* / %`
Δεξιά	`! ~ ++ -- (int) (float) (string) (array) (object) @`
Δεξιά	`[`
Καμία	`clone new`

Πίνακας Α.17 Προτεραιότητα τελεστών στην php

Α.4 ΒΑΣΙΚΕΣ ΕΝΤΟΛΕΣ

Σε όλες τις γλώσσες προγραμματισμού υπάρχουν κάποιες βασικές εντολές μέσω των οποίων καθορίζεται η ροή του προγράμματος του οποίου γράφουμε.

Εντολές αποφάσεων όπως οι if, else, elseif , switch και case μας δίνουν την δυνατότητα να διακλαδώσουμε την ροη του προγράμματος μας σε επιμέρους τμήματα με βάση κάποιες συνθήκες που θα πρέπει να πληρούνται. Αντίστοιχα οι βρόγχοι όπως οι while, do while, for και foreach καθώς επίσης και οι εντολές ελέγχου βρόγχων όπως οι break και οι continue, μας δίνουν την δυνατότητα να επαναλάβουμε μέρη του κώδικα μας όσες φορές εμείς θέλουμε στηρίζοντας το πλήθος των επαναλήψεων και πάλι σε κάποια συνθήκη ελέγχου.

Σε αυτό το κεφάλαιο θα δούμε αναλυτικά όλες τις εντολές αποφάσεων και επαναλήψεων, θα μάθουμε να τις χρησιμοποιούμε και να βγάζουμε συμπεράσματα με αυτές.

Α.4.1 if

Η εντολή «if» είναι η δομή απόφασης η οποία απαντά στο αν ισχύει ή δεν ισχύει μια συνθήκη μέσα σε αυτήν, έτσι μπορούμε να ελέγχουμε την ροή ενός προγράμματος

Ας θεωρήσουμε μια ηλιόλουστη ημέρα του χειμώνα πως θέλουμε να βγούμε έξω για βόλτα. Το κριτήριο απόφασης για το αν θα πάρουμε μαζί μας μπουφάν είναι αν η θερμοκρασία είναι κάτω από 20 βαθμούς Κελσίου.

Η λέξη «εάν» είναι η λέξη κλειδί στην χρήση της «if». Εάν ισχύει κάτι, τότε κάνε κάτι γι αυτό. Πχ… αν ισχύει ότι ο βαθμός του μαθήματος είναι μεγαλύτερος ή ίσος του 5 τότε ο φοιτητής πέρασε το μάθημα. Η δομή της έχει ως εξής:

```
if(συνθήκη){
  …εντολές που εκτελούνται μόνο αν ισχύει η συνθήκη….
 }
```

Για το παραπάνω παράδειγμα με την θερμοκρασία έχουμε:

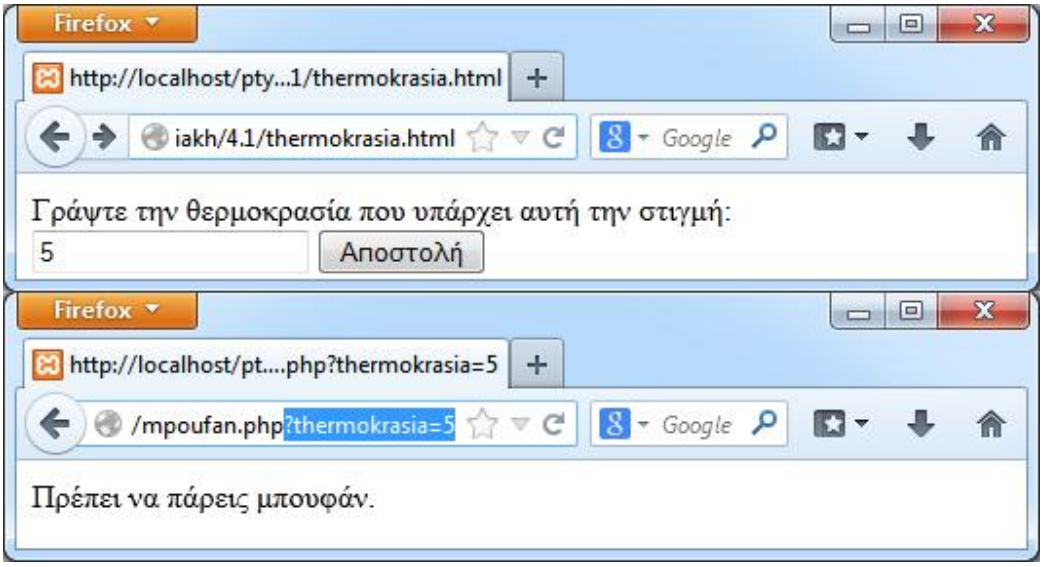
Αρχείο: temperature.html

```
<form method='get' action='jacket.php'>
    Γράψτε την θερμοκρασία που υπάρχει αυτή την στιγμή:
    <input type='text' name='thermokrasia' />
    <input type='submit' value='Αποστολή' />
</form>
```

Αρχείο: jacket.php

```
<?php
 $thermokrasia = $_GET['thermokrasia'];
 if($thermokrasia<20){
  echo "Πρέπει να πάρεις μπουφάν.";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.66 Εισαγωγή και εξαγωγή αποτελέσματος απόφασης

Η εντολή if μπορεί να συμπεριληφθεί μέσα σε μια άλλη if δημιουργώντας μια σειρά από ερωτήσεις. Για παράδειγμα να εκτελείται ένα κομμάτι εντολών μόνο εφόσον πληρούνται κάποιες προϋποθέσεις, όπως το να μπορέσει κάποιος να κάνει την πτυχιακή του εργασία θα πρέπει να έχει περάσει 25 μαθήματα και να βρίσκεται στο 8ο εξάμηνο σπουδών του.

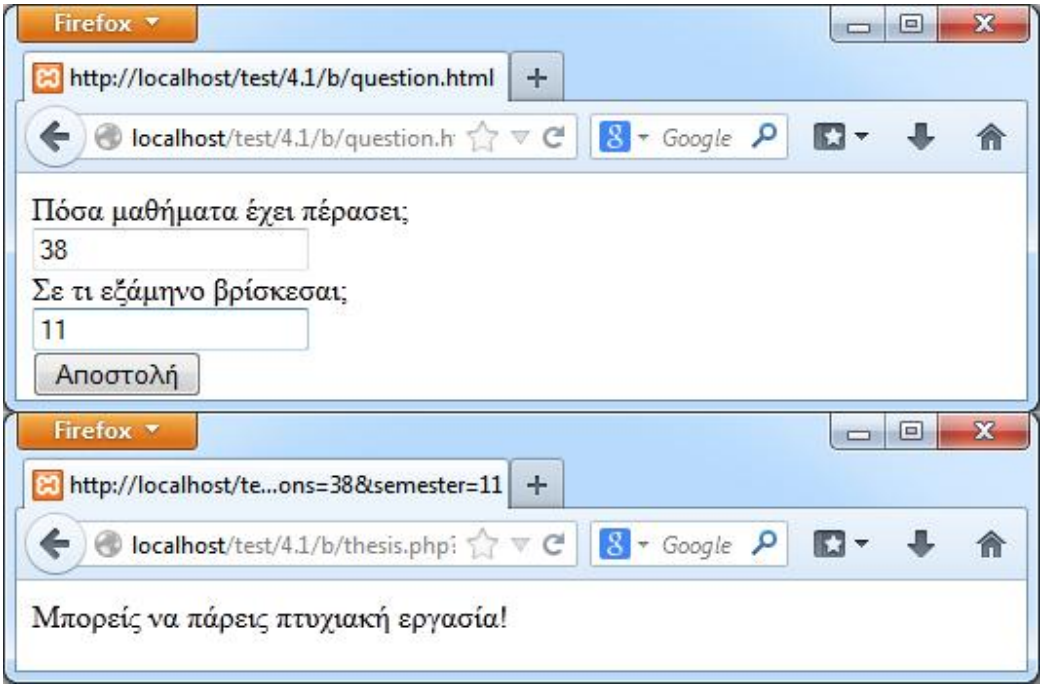
Αρχείο: question.html

```
<form method='get' action='thesis.php'>
 Πόσα μαθήματα έχει πέρασει;
 <br />
 <input type='text' name='lessons'/>
 <br />
 <br />
 Σε τι εξάμηνο βρίσκεσαι;
 <br />
 <input type='text' name='semester'/>
 <br />
 <input type='submit' value='Αποστολή'/>
</form>
```

Αρχείο: thesis.php

```
<?php
 $eksamhno = $_GET['semester'];
 $mathimata = $_GET['lessons'];
 if($mathimata>=25){
  if($eksamhno>=8){
   echo "Μπορείς να πάρεις πτυχιακή εργασία!";
  }
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.67 Αποτέλεσμα απόφασης με την χρήση δυο τιμών ως εισαγωγή

Το παραπάνω παράδειγμα θα μπορούσε να γραφτεί διαφορετικά χρησιμοποιώντας λογικούς τελεστές :

```
<?php
 $eksamhno = $_GET['semester'];
 $mathimata = $_GET['lessons'];
 if($eksamhno>=8 && $mathimata >=25){
  echo "μπορείς να πάρεις πτυχιακή εργασία!";
 }
?>
```

A.4.2 else και elseif

Το «else» συνήθως χρησιμοποιείται όταν κάποιος θέλει να λάβει υπόψη του όλες τις άλλες περιπτώσεις εκτός από αυτή που ελέγχει εκείνη την στιγμή. Δηλαδή στο παράδειγμα της ηλιόλουστης ημέρας του χειμώνα, θα μπορούσε να συμπεριληφθεί ένα αποτέλεσμα στην περίπτωση που η θερμοκρασία ήταν σε οποιοδήποτε άλλο επίπεδο θερμοκρασίας, π.χ.

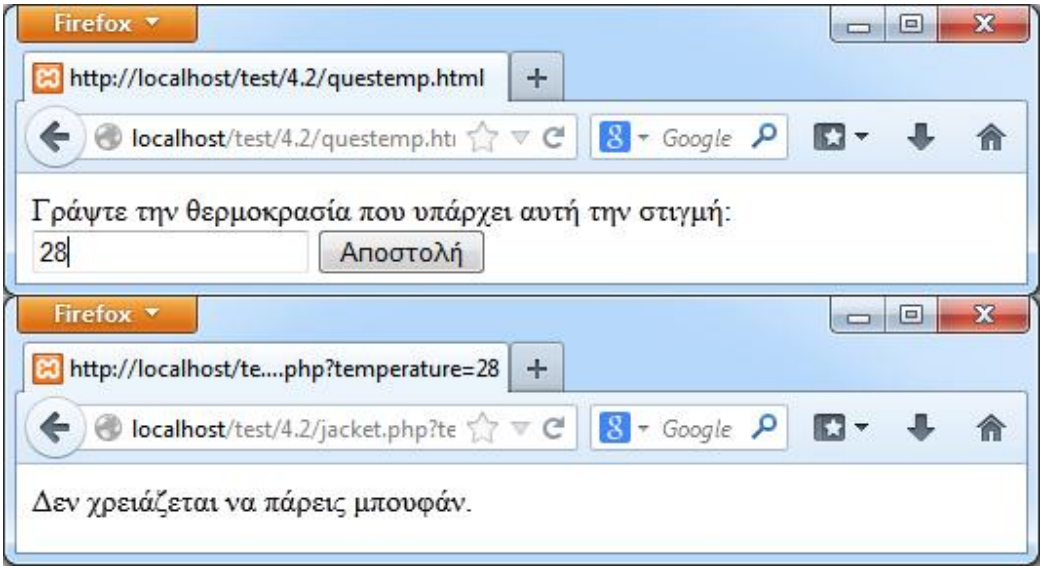
Αρχείο: questemp.html

```
<form method='get' action='jacket.php'>
 Γράψτε την θερμοκρασία που υπάρχει αυτή την στιγμή:
 <input type='text' name='temperature' />
 <input type='submit' value='Αποστολή' />
</form>
```

Αρχείο: jacket.php

```
<?php
 $thermokrasia = $_GET['temperature'];
 if($thermokrasia<20){
  echo "Πρέπει να πάρεις μπουφάν.";
 } else {
  echo "Δεν χρειάζεται να πάρεις μπουφάν.";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.68 Εξαγωγή σύνθετου αποτελέσματος

Έτσι με το παραπάνω παράδειγμα καλύπτουμε την περίπτωση που η θερμοκρασία μπορεί να είναι κάτω από 20 βαθμούς κελσίου, και επομένως θα πρέπει να πάρουμε μπουφάν, αλλά επίσης καλύπτουμε όλες τις υπόλοιπες περιπτώσεις όπου η θερμοκρασία θα μπορούσε να είναι 20 ή και πάνω από 20 βαθμούς κελσίου και στην προκειμένη δεν θα χρειαζόταν να πάρουμε μπουφάν.

Συμπεριλαμβάνοντας έτσι το «else», βοηθάμε διευκρινίζοντας το τι γίνεται σε όλες τις άλλες περιπτώσεις, από αυτή που το πρόγραμμα μας θα εκτελούσε ένα ευτυχές σενάριο. Αν το πρόγραμμα μας ήταν να μας υποδεικνύει πότε να πάρουμε μπουφάν με βάση την θερμοκρασία του περιβάλλοντος, τότε το να συμπεριλάβουμε το πότε ΔΕΝ χρειάζεται να πάρουμε μπουφάν, είναι προφανώς κάτι που απαιτείται καθώς ένας απλός χρήστης του προγράμματος δεν θα καταλάβει ότι το πρόγραμμα «λειτουργεί» αν δεν του εμφανίσει τίποτα ως αποτέλεσμα.

Τι γίνεται όμως όταν θέλουμε να διακρίνουμε αυτή την διαφορά σε περισσότερα μέρη ; Ας θεωρήσουμε πως μπουφάν φοράμε από τους 15 βαθμούς Κελσίου και κάτω. Από τους 15 έως τους 22 παίρνουμε ζακέτα, και από 22 και πάνω δεν παίρνουμε τίποτα.

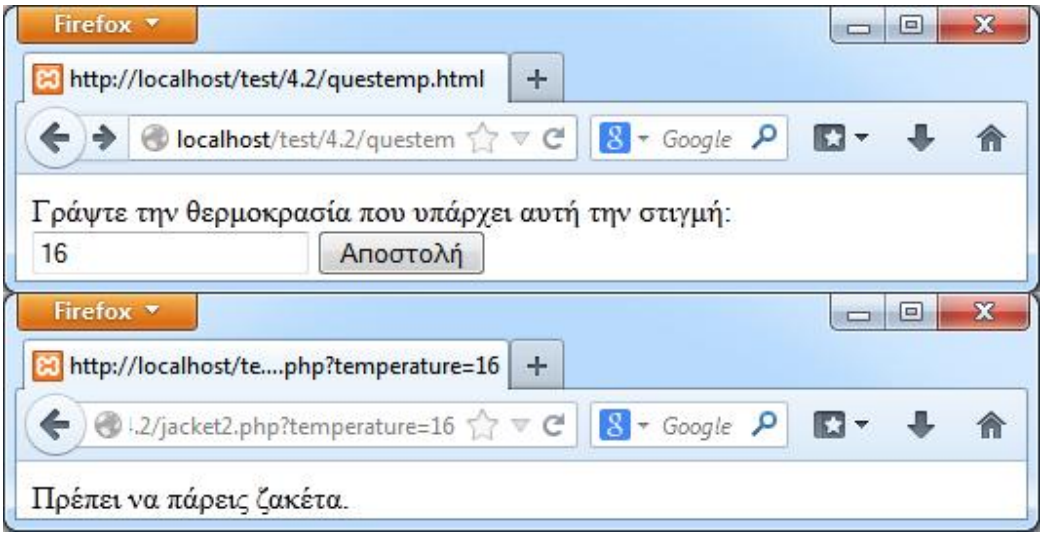
Αρχείο: questemp.html

```
<form method='get' action='jacket.php'>
 Γράψτε την θερμοκρασία που υπάρχει αυτή την στιγμή:
 <input type='text' name='temperature' />
 <input type='submit' value='Αποστολή' />
</form>
```

Αρχείο: jacket.php

```
<?php
 $thermokrasia = $_GET['temperature'];
 if($thermokrasia<15){
  echo "Πρέπει να πάρεις μπουφάν.";
 } elseif($thermokrasia>15 && $thermokrasia<22) {
  echo "Πρέπει να πάρεις ζακέτα.";
 } elseif($thermokrasia>=22){
  echo "Δεν χρειάζεται να πάρεις μπουφάν ή ζακέτα.";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.69 Είσοδος θερμοκρασίας και έξοδος εξειδικευμένου αποτελέσματος

Α.4.3 switch και case

Η εντολή «switch» είναι μια εντολή απόφασης που προσπαθεί να συντομεύσει την εντολή «if». Με την χρήση της «switch» αποφεύγουμε την χρήση πολλαπλών «if» τα οποία θα έκαναν πολύ δύσκολο τον έλεγχο του προγράμματος από την οπτική γωνία του προγραμματιστή, και συντακτικά θα ήταν δύσκολο να βρεθούν συγκεκριμένες περιπτώσεις.

Ο τρόπος σύνταξης της «switch» έχει συγκεκριμένο τρόπο και γίνεται ως εξής:

```
switch(μεταβλητή){
 case περίπτωση πρώτη:
  //...εντολές που εκτελούνται...
  break;
 case περίπτωση δεύτερη:
  //...εντολές που εκτελούνται στην δεύτερη περίπτωση...
 break;
 default:
  //...εντολές που εκτελούνται όταν δεν βρεθεί καμία από τις παραπάνω
περιπτώσεις...
}
```

- Όπου «μεταβλητή» εισάγουμε το όνομα της μεταβλητής που θέλουμε να μελετήσουμε.
- Όπου «περίπτωση πρώτη» εισάγουμε τον έλεγχο που θέλουμε να κάνουμε σε αυτή την περίπτωση, ότι θα βάζαμε δηλαδή στις παρενθέσεις της «if». Παρομοίως για την «περίπτωση δεύτερη», «περίπτωση τρίτη» κ.ο.κ.
- Η εντολή «break» σημαίνει ότι θα σταματήσει η php την εκτέλεση του κώδικα της «switch» στο σημείο όπου θα βρεθεί, διαφορετικά η php θα εκτελούσε όλες τις εντολές από το σημείο που βρήκε την «case» που ταιριάζει με την περίπτωση. Όλες οι εντολές για κάθε περίπτωση ξεκινούν από την άνω και κάτω τελεία του «case» και τελειώνουν στο «break».
- Το «default» χρησιμοποιείται για όλες τις άλλες περιπτώσεις που δεν υπάρχουν σε case, είναι δηλαδή το «else» της «switch».

Παίρνοντας το παράδειγμα της ενότητας Α.4.2 παρατηρούμε ότι ο κώδικας μας πλαισιώνεται από τρία διαφορετικά «if» τα οποία κάνουν δυσκολότερη την ανάγνωση του , καθώς επίσης και την περεταίρω διόρθωση ή ανάπτυξη. Οι γραμμές του κώδικα είναι πολλές και αν είχαμε πιο περίπλοκους ελέγχους να κάνουμε τόσο περισσότερο θα διογκώνονταν και η μάζα του παραγόμενου κώδικα. Ας δούμε το συγκεκριμένο παράδειγμα όμως εφαρμόζοντας «switch»:

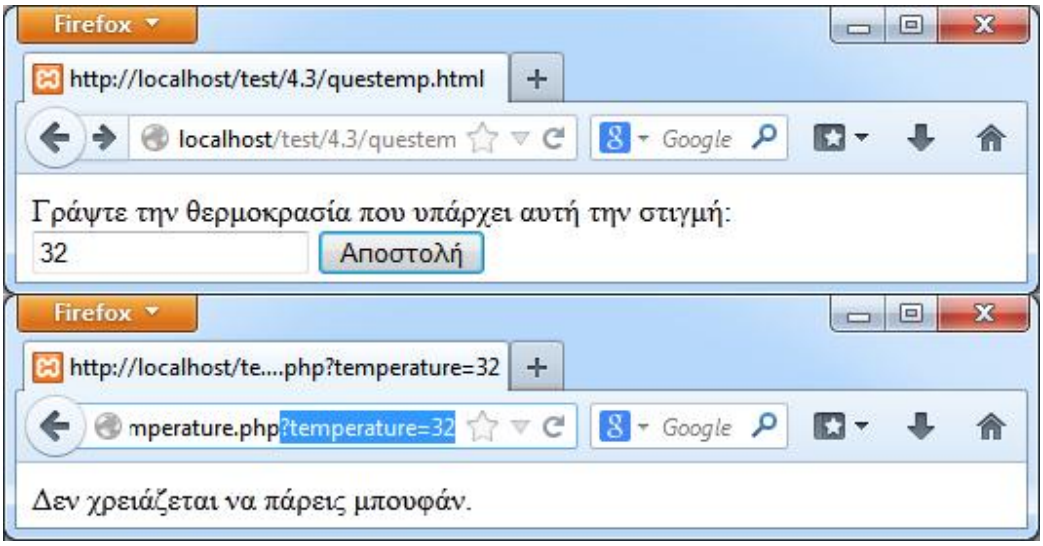
Αρχείο: questemp.html

```
<form method='get' action='temperature.php'>
 Γράψτε την θερμοκρασία που υπάρχει αυτή την στιγμή:
 <input type='text' name='temperature' />
 <input type='submit' value='Αποστολή' />
</form>
```

Αρχείο: temperature.php

```php
<?php
 $thermokrasia = $_GET['temperature'];
 switch($thermokrasia){
  case ($thermokrasia<15):
   echo "Πρέπει να πάρεις μπουφάν.";
   break;
  case ($thermokrasia>15 && $thermokrasia<22):
   echo "Πρέπει να πάρεις ζακέτα.";
   break;
  case ($thermokrasia>=22):
   echo "Δεν χρειάζεται να πάρεις μπουφάν.";
   break;
  default:
   echo "Δεν εισήχθηκε θερμοκρασία.";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.70 Εισαγωγή θερμοκρασίας και έξοδος αποτελέσματος με την case

Μας δίνεται μια πιο συγκεκριμένη δομή στον τρόπο με τον οποίο «εξετάζονται» οι περιπτώσεις, έτσι είναι ευκολότερη η επεξεργασία και η προσθήκη γραμμών κώδικα στο μέλλον.

A.4.4 while

Ο απλούστερος βρόγχος στην php είναι η βρόγχος «while». Η σύνταξη του βασίζεται σε μια συνθήκη, παρόμοια με την εντολή «if». Η διαφορά τους είναι ότι η «if» εκτελεί μόνο μια φορά το τμήμα κώδικα που την αφορά, ενώ η «while» για όσο είναι αληθής η συνθήκη της.

Γενικώς η «while» χρησιμοποιείται σε περιπτώσεις που δεν γνωρίζουμε για πόσες επαναλήψεις θα απαιτηθούν για να γίνει η συνθήκη ψευδής. Ορίζουμε την μεταβλητή που αντιπροσωπεύει τον μετρητή, πριν από το «while», όπως επίσης δεν ξεχνάμε να αυξήσουμε η μειώσουμε τον μετρητή μέσα του.

Ο τρόπος σύνταξης έχει ως εξής:

```
while(συνθήκη){
 ... εντολές που θα εκτελεστούν ...
}
```

Όπου «συνθήκη», η συνθήκη που θέλουμε για όσο ισχύει να επαναλαμβάνονται οι «... εντολές που θα εκτελεστούν ...».

Ας υποθέσουμε πως εισάγουμε έναν αριθμό και θέλουμε από το 1 έως και τον αριθμό που θα εισάγουμε, να τυπώσει κατά σειρά όλους τους αριθμούς.

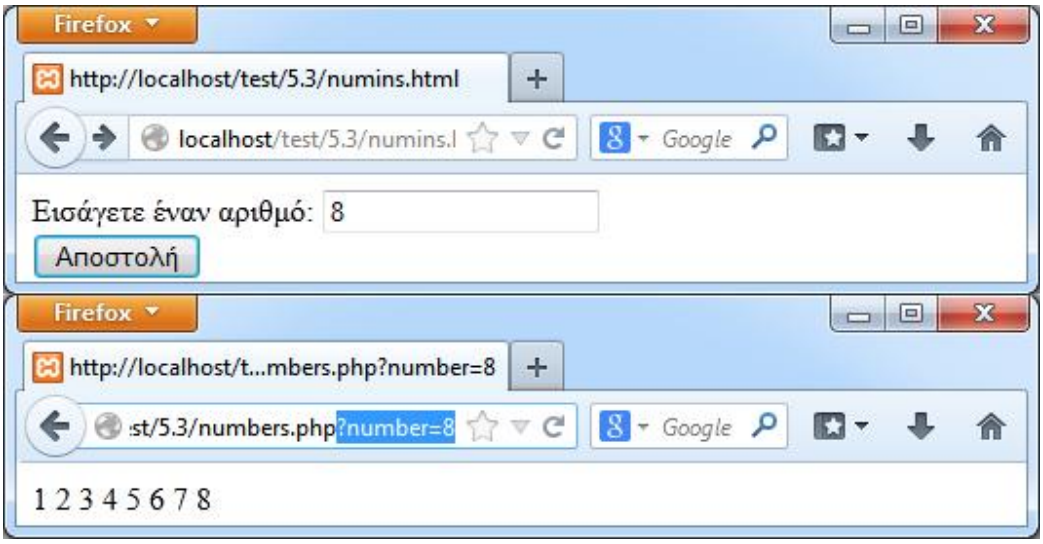
Αρχείο: numins.html

```
<form method='get' action='numbers.php'>
 Εισάγετε έναν αριθμό: <input type='text' name='number'/>
 <br />
 <input type='submit' value='Αποστολή'/>
</form>
```

Αρχείο: numbers.php

```
<?php
 $arithmos=$_GET['number'];
 $i=1;
 while($i<=$arithmos){
  echo $i."\n";
  $i++;
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.71 Εισαγωγή ενός αριθμού και έξοδος αλληλουχίας με την χρήση while

Α.4.5 do while

Το «do-while» είναι ένα είδος βρόγχου το οποίο εκτελείται τουλάχιστον μία φορά πριν αρχίσει να επαναλαμβάνεται ελέγχοντας αν ισχύει η συνθήκη του. Αυτό γίνεται γιατί ο έλεγχος της συνθήκης γίνεται στο τέλος και όχι στην αρχή κάθε επανάληψης, επομένως εκτελείται τουλάχιστον μία φορά. Ο τρόπος σύνταξης έχει ως εξής:

```
do{
  ... Εντολές που εκτελούνται...
}while(συνθήκη);
```

- Όπου «... Εντολές που εκτελούνται...» πηγαίνουν οι εντολές πού θέλουμε να εκτελεστούν τουλάχιστον μία φορά και ύστερα κατ' επανάληψη,
- Όπου «συνθήκη» πηγαίνει η συνθήκη που θέλουμε να ελέγξουμε στο τέλος, και ύστερα σε κάθε επανάληψη.

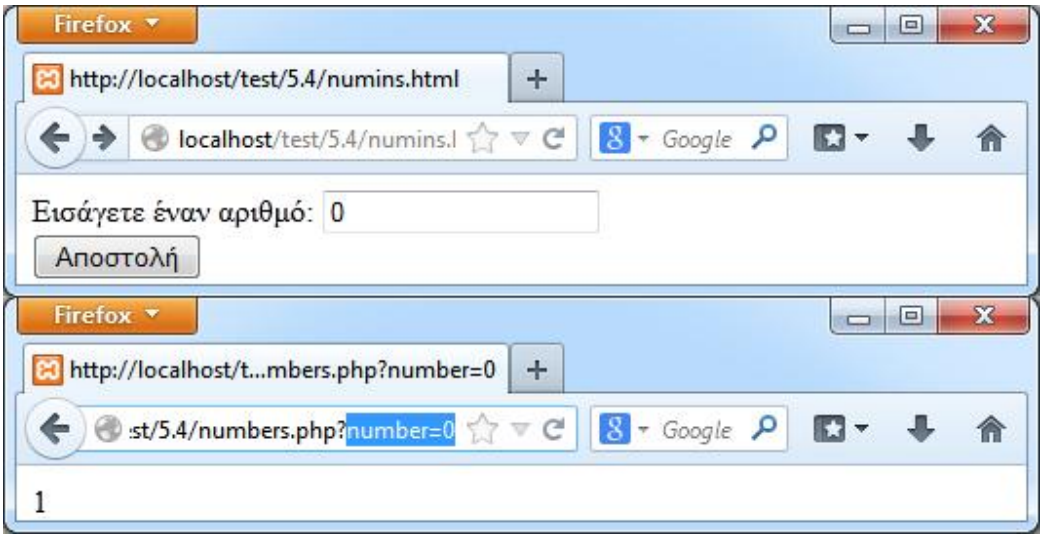
Αρχείο: numins.html

```
<form method='get' action='numbers.php'>
 Εισάγετε έναν αριθμό: <input type='text' name='number'/>
 <br />
 <input type='submit' value='Αποστολή'/>
</form>
```

Αρχείο: numbers.php

```php
<?php
 $arithmos=$_GET['number'];
 $i=1;
 do{
  echo $i."\n";
  $i++;
 }while($i<=$arithmos);
?>
```

Αποτέλεσμα:



Εικόνα Α.72 Εισαγωγή αριθμού και εξαγωγή αποτελέσματος εκτέλεσης του do-while

Στο παραπάνω παράδειγμα, παρόλο που εισάγαμε 0 για αριθμό βάση του οποίου θα ξεκινήσει η μέτρηση, τυπώθηκε ο αριθμός «1» που σημαίνει ότι εκτελέσθηκε ο βρόγχος τουλάχιστον μία φορά χωρίς να λάβει υπόψη την συνθήκη.

A.4.6 for

Στην ζωή μας καθημερινά εμφανίζεται μια φυσική διαδικασία ή μια υποχρέωση η οποία είναι επαναλαμβανόμενη . Ένα φυσικό παράδειγμα είναι η έμμηνος ρύση, ή οι εποχές της γης. Ένα παράδειγμα υποχρέωσης θα μπορούσε να ήταν η χρήση ενός χαπιού σε καθημερινή βάση.

Η δομή επανάληψης «for» αναλαμβάνει να κάνει κάτι επαναλαμβανόμενα για όσο ισχύει κάποια συνθήκη, π.χ. όσο υπάρχουν χάπια στο κουτί, για τόσο ο ασθενής θα επαναλαμβάνει καθημερινή χορήγηση. Η σύνταξη της for έχει ως εξής:

```
for(εντολή 1;συνθήκη;εντολή 2){
 // Εντολές που εκτελούνται σε κάθε επανάληψη
}
```

- Όπου «εντολή 1» μια εντολή όπου εκτελείται μία φορά μόνο στην αρχή όταν πρώτο μπει το script στον βρόγχο. Εδώ συνήθως ορίζεται ένας μετρητής.
- Όπου «συνθήκη» είναι η συνθήκη που θα ελέγχεται σε κάθε επανάληψη πριν μπει στην επανάληψη. Εδώ συνήθως ελέγχεται η τιμή του μετρητή.
- Όπου «εντολή 2» είναι μια εντολή όπου εκτελείται σε κάθε τέλος της κάθε επανάληψης. Εδώ συνήθως αυξάνεται ή μειώνεται η τιμή του μετρητή.

Ας υποθέσουμε πως εισάγοντας δυο αριθμούς θέλουμε να υπολογίσουμε για τον πρώτο το αποτέλεσμα του υψωμένο στον δεύτερο.

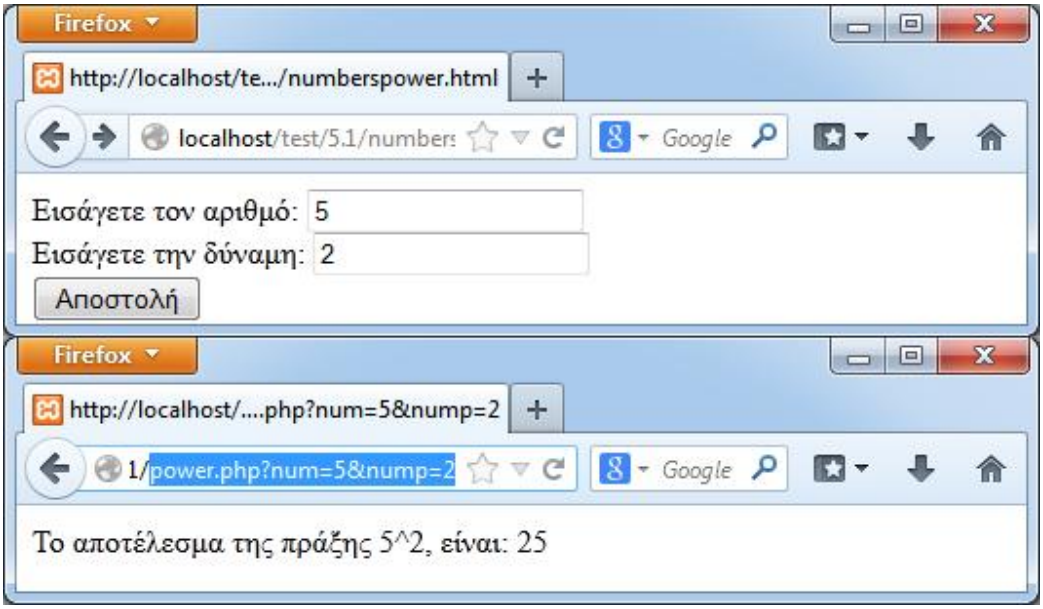
Αρχείο: numberspower.html

```
<form method='get' action='power.php'>
 Εισάγετε τον αριθμό: <input type='text' name='num'/>
 <br />
 Εισάγετε την δύναμη: <input type='text' name='nump'/>
 <br />
 <input type='submit' value='Αποστολή'/>
</form>
```

Αρχείο: power.php

```
<?php
 $arithmos = $_GET['num'];
 $dinami = $_GET['nump'];
 $result=1;
 for($i=1;$i<=$dinami;$i++){
  $result=$result*$arithmos;
 }
 echo "Το αποτέλεσμα της πράξης ".$arithmos."^".$dinami.", είναι:
".$result;
?>
```

Αποτέλεσμα:



Εικόνα Α.73 Εισαγωγή αριθμών και εξαγωγή αποτελέσματος πράξης δύναμης με for

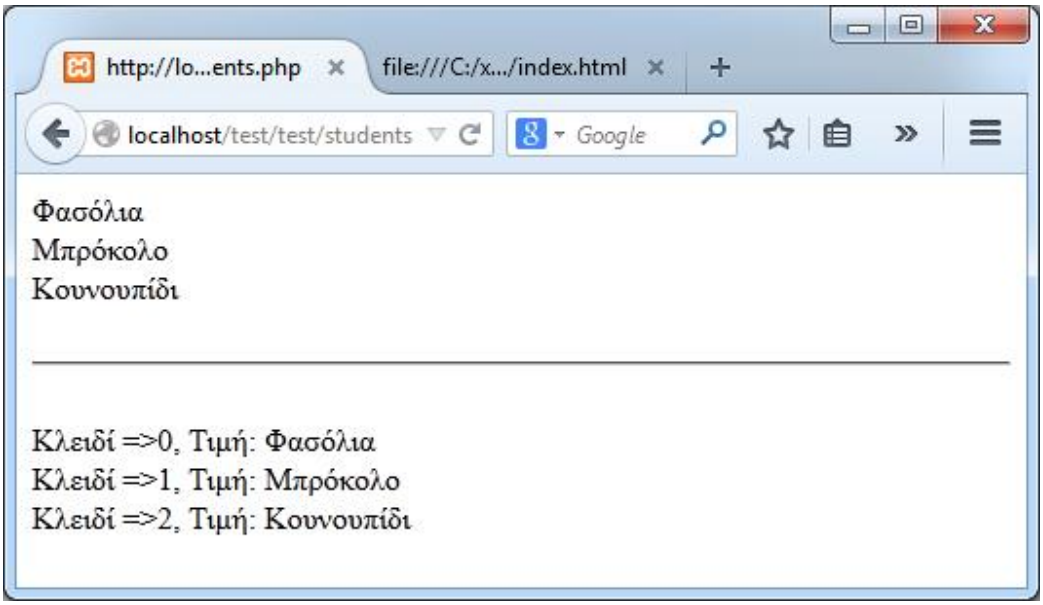
Α.4.7 foreach

Το «foreach» είναι μια δομή επανάληψης για διάσχιση πίνακα χωρίς μετρητές. Η δομή αυτή ξεκινά την επανάληψη παίρνοντας το πρώτο στοιχείο του πίνακα εκτελώντας όλες τις εντολές που βρίσκονται στην επανάληψη του και ομοίως ακολουθεί μέχρι να φτάσει στο τελευταίο του. Η διάσχιση γίνεται χρησιμοποιώντας τις τιμές που περιλαμβάνει ο πίνακας μέσα στις αντίστοιχες θέσεις ή και σε μερικές περιπτώσεις αν το ζητήσουμε μπορούμε να συμπεριλάβουμε και τους δείκτες (κλειδιά) των εκάστοτε θέσεων. Οι πίνακες επεξηγούνται αναλυτικά σε επόμενο κεφάλαιο.

Αρχείο: vegetables.php

```
<?php
 $pinakas = array("Φασόλια","Μπρόκολο","Κουνουπίδι");

 foreach($pinakas as $value){
  echo $value."<br/>";
 }

 echo "<br/><hr/><br/>";

 foreach($pinakas as $key => $value){
  echo "Κλειδί =>".$key.", Τιμή: ".$value."<br/>";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.74 Εκτέλεση foreach για τύπωση πίνακα.

Όπως παρατηρούμε στην Εικόνα Α.74 υπάρχουν δυο αποτελέσματα καθότι δύο και τα «foreach» μας. Στο πρώτο «foreach» εκτελούμε διάσχιση του πίνακα «$pinakas» χωρίς να ζητάμε να δοθούν και τα κλειδιά μαζί με τις τιμές. Στο δεύτερο «foreach» ζητάμε αυτή την αλλαγή δίνοντας το κάθε κλειδί σε μια μεταβλητή «$key». Έτσι μπορούμε να χρησιμοποιήσουμε και τα δυο, κλειδί και τιμή, κατά την διάρκεια της διάσχισης. Αυτό χρησιμεύει ιδιαίτερα όταν το κλειδί δεν είναι αριθμημένη σειρά από το 0 έως το μήκος του πίνακα (μείον ένα), αλλά κείμενο όπως «όνομα», «επώνυμο», κτλ.

Α.4.8 break και continue

Τα «break» και «continue», χρησιμοποιούνται για να διακόψουν την ροή ενός βρόγχου είτε σε κάποια συγκεκριμένη επανάληψη, είτε διακόπτοντας γενικώς τον βρόγχο.

continue

Το «continue» παρακάμπτει όλες τις εντολές που ακολουθούν και πάει κατευθείαν στην αρχή του βρόγχου. Ας θεωρήσουμε πως θέλουμε να τυπωθούν όλοι οι «μονοί» αριθμοί από το 1 έως έναν αριθμό που θα του πούμε, παρακάμπτοντας την τύπωση των «ζυγών». Ο Κώδικας που προκύπτει είναι ο εξής:

Αρχείο: choosenumber.html

```
<form method='get' action='shownumbers.php'>
 Εισάγετε έναν αριθμό: <input type='text' name='number' />
 <br/>
 <input type='submit' value='Αποστολή' />
</form>
```

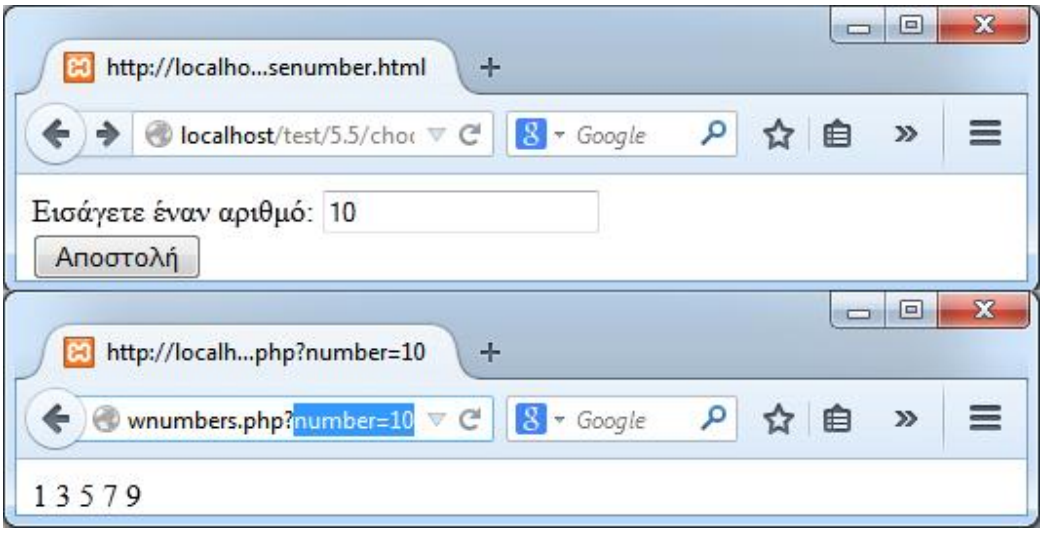
Αρχείο: shownumbers.php

```
<?php
 $arithmos = $_GET['number'];
 for($i=1;$i<=$arithmos;$i++){
  if(($i%2)==0){
   continue; // θα πάει άμεσα στη γραμμή for
  }
  echo $i."\n";
 }
?>
```

Παρατηρείστε πως η εντολή «echo» βρίσκεται έξω από το «if» αλλά παρόλα αυτά όταν ο αριθμός του «$i» είναι ζυγός, τότε δεν εκτελείται γιατί ισχύει το «if» και ο βρόγχος συνεχίζει με «continue».

Αν θέλουμε να ελέγξουμε αν ένας αριθμός είναι ζυγός ή μονός, διαιρούμε με τον αριθμό δύο με την πράξη του υπολοίπου (modulus) και ύστερα αν το υπόλοιπο είναι 0 είναι ζυγός αριθμός, ενώ αν είναι 1 είναι μονός.

Το αποτέλεσμα θα είναι:



Εικόνα Α.75 Εισαγωγή αριθμού και τύπωση αποτελεσμάτων με την χρήση continue

break

Το «break» προκαλεί την άμεση έξοδο από το βρόγχο και τη συνέχιση του κώδικα στην πρώτη γραμμή που ακολουθεί το βρόγχο. Ας δούμε ένα παράδειγμα:

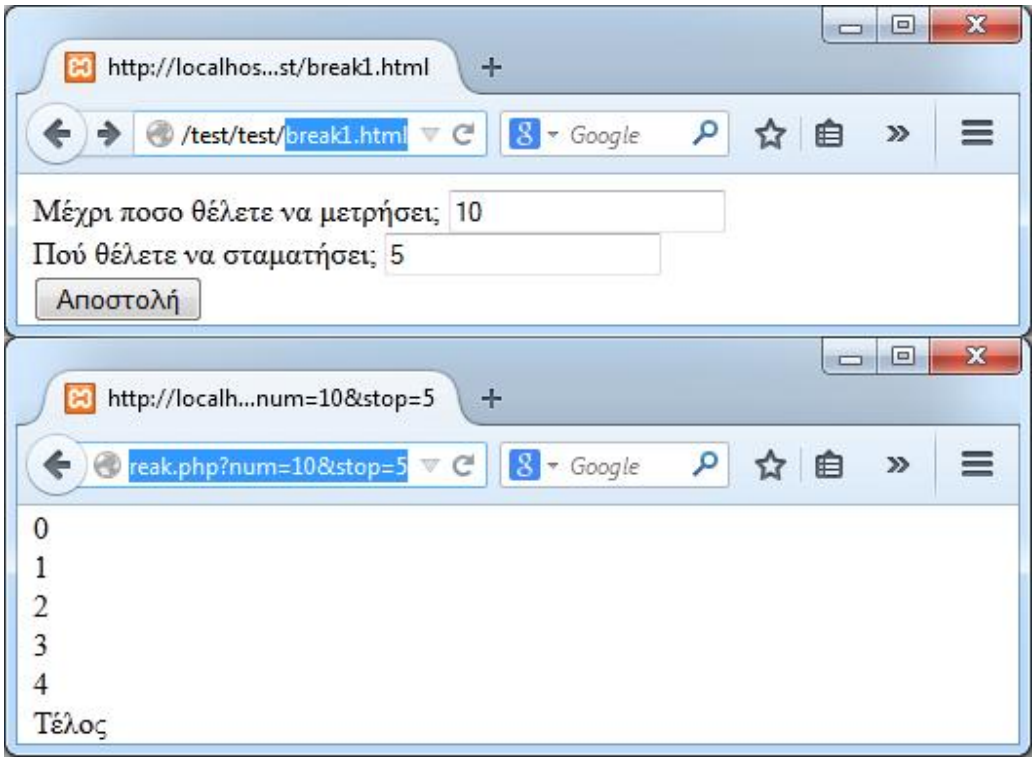
Αρχείο: break1.html

```
<form method='get' action='break.php'>
 Μέχρι ποσο θέλετε να μετρήσει; <input type='text' name='num'/><br/>
 Πού θέλετε να σταματήσει; <input type='text' name='stop'/><br/>
 <input type='submit' value='Αποστολή'/>
</form>
```

Αρχείο: break.php

```
<?php
 for($i=0;$i<$_GET['num'];$i++){
  if($i==$_GET['stop']){
   break;
  }
  echo $i."<br/>";
 }
 echo "Τέλος";
?>
```

Αποτέλεσμα:



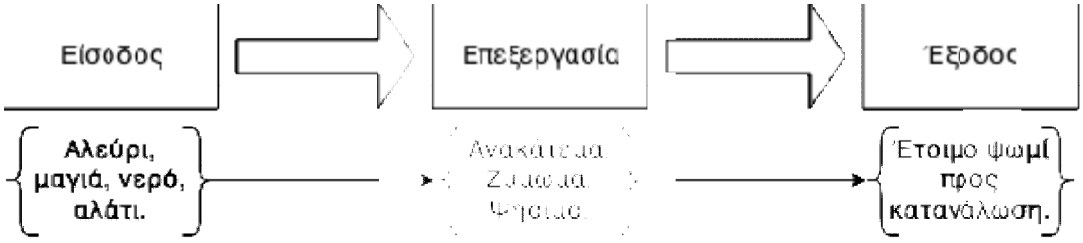
Εικόνα Α.76 Σπάσιμο βρόγχου με την εντολή break

Α.5 ΜΕΘΟΔΟΙ

Οι μέθοδοι (η αλλιώς και συναρτήσεις, functions στα αγγλικά) υπάρχουν σχεδόν σε όλες τις γλώσσες προγραμματισμού. Στην php υπάρχουν τόσο εσωτερικές μέθοδοι της ίδιας της php όσο και εξωτερικές από πρόσθετες λειτουργίες. Εκτός από τα δυο παραπάνω είδη μεθόδων υπάρχουν και οι ορισμένες από τον προγραμματιστή μέθοδοι οι οποίες εκτελούν ένα μικρό (ή και σε μερικές περιπτώσεις μεγάλο) σύνολο εντολών. Το σύνολο αυτό διατηρείται και εκτελείται αυτόνομα για όσες φορές εμείς το καλέσουμε, καθώς διαθέτει και ένα όνομα. Κάθε μέθοδος προαιρετικά μπορεί να παίρνει κάποια είσοδο τιμών κατά την εκτέλεση της και προαιρετικά επίσης μπορεί να επιστρέφει κάποιο αποτέλεσμα. Για να κατανοήσετε όλα τα παραπάνω θα προσπαθήσουμε να προσεγγίσουμε τις μεθόδους με ένα παράδειγμα από την καθημερινότητα μας.

Καθημερινά στην ζωή μας τρώμε ψωμί γιατί είναι απαραίτητο για την διατροφή μας. Για να κατασκευάσουμε εμείς ψωμί θα χρειαζόμασταν αλεύρι, μαγιά, νερό, αλάτι και μπόλικο χρόνο για να μπορέσουμε να το παρασκευάσουμε. Θα πρέπει να ανακατέψουμε τα συστατικά, να ζυμώσουμε την ζύμη μας και ύστερα να την ψήσουμε στον φούρνο περιμένοντας να γίνει. Φανταστείτε τώρα πως για όλη αυτή την διαδικασία υπάρχει μια έτοιμη μηχανή η οποία δέχεται ως είσοδο όλα τα συστατικά (αλεύρι, μαγιά, νερό, αλάτι) και παρασκευάζει αυτόματα το ψωμί περνώντας από όλα τα στάδια παρασκευής του (ανακάτεμα, ζύμωμα, ψήσιμο). Στο τέλος μας «επιστρέφει» το έτοιμο προς κατανάλωση ψωμί. Αυτή την συσκευή μπορούμε να την παρομοιάσουμε με μια μέθοδο, τα υλικά που παίρνει η συσκευή, ως οι παράμετροι εισόδου της μεθόδου, και το αποτέλεσμα που μας αποφέρει, δηλαδή το ψωμί, ως την έξοδο αυτής της μεθόδου (η αλλιώς το αποτέλεσμα της).

Η μέθοδοι είναι απτό παράδειγμα της λογικής , εισαγωγή ύστερα επεξεργασία και τέλος έξοδος, όπου κάποιος βλέπει μόνο την είσοδο και την έξοδο και όχι τι συμβαίνει ενδιάμεσα κατά την επεξεργασία.



Σχήμα Α.1 Τρόπος λειτουργίας μεθόδων

Α.5.1 Δημιουργία μεθόδου

Φανταστείτε τις μεθόδους (ή αλλιώς functions) σαν εργαλεία τα οποία βρίσκονται τοποθετημένα σε διάφορα ράφια. Όταν εσείς χρειάζεστε ένα εργαλείο, το αναζητείτε με βάση το όνομα του, π.χ. τρυπάνι. Ένα τρυπάνι όμως για να λειτουργήσει χρειάζεται μια πηγή ενέργειας, π.χ. μια μπαταρία, και μια κεφαλή. Στον προγραμματισμό χρησιμοποιούμε την ίδια λογική, το μόνο που αλλάζει είναι ο τρόπος με το οποίο το κάνουμε. Αν το τρυπάνι ήταν «function» ή σύνταξη του θα είχε ως εξής:

```
function tripani($mpataria, $kefalh){
 //Χρήση τρυπανιού
 return true;
}
```

Όπως παρατηρούμε υπάρχουν σημαντικά σημεία σύνταξης:

1. Όλες οι μέθοδοι έχουν ένα όνομα και στην συγκεκριμένη περίπτωση είναι το «tripani». Κάθε όνομα πρέπει να είναι ένα οποιοδήποτε αλφαριθμητικό ξεκινώντας με ένα γράμμα ή κάτω παύλα, και στην συνέχεια μπορεί να συνεχίζεται με κάτω παύλες, γράμματα ή αριθμούς. Δεν μπορεί ένα όνομα μεθόδου να ξεκινά με αριθμό. Οι μέθοδοι διακρίνονται μέσω τον ονομάτων που έχουν, και είναι αυστηρά σημαντικό κάθε μέθοδος να έχει διαφορετική ονομασία από κάποια άλλη ώστε να αποφευχθούν σφάλματα στην χρήση τους. Τα ονόματα των μεθόδων είναι case-insensitive, αυτό σημαίνει πως μια μέθοδο με το όνομα «tripani» είναι ίδια με την «TriPani» ή «TRIPANI».
2. Πριν το όνομα της μεθόδου πρέπει υπάρχει η λέξη «function».
3. Μαζί με το όνομα της μεθόδου υπάρχει ΠΑΝΤΟΝΤΕ και ένα ζεύγος παρενθέσεων «()».

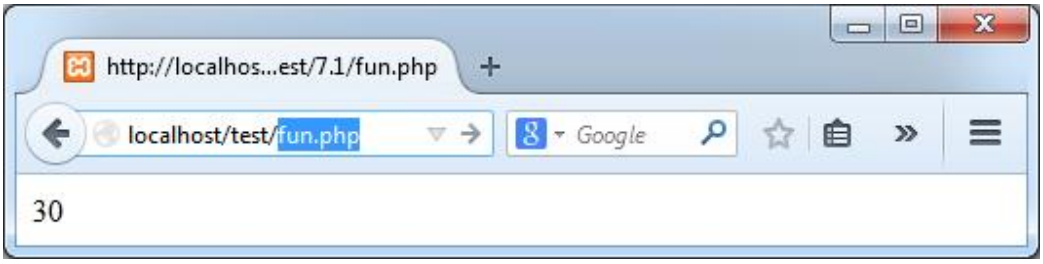
4. Μέσα στις παρενθέσεις μπορούν να υπάρχουν προαιρετικά ορίσματα εισόδου ή αλλιώς παράμετροι της μεθόδου, στην περίπτωση μας είναι τα $mpataria, $kefalh.
5. Το σύνολο των εντολών που θα εκτελεστούν από την μέθοδο κλείνονται μέσα σε αγκύλες «{…}».
6. Όλες οι μέθοδοι επιστρέφουν συνήθως κάτι ως αποτέλεσμα των πράξεων τους, επομένως το «return» είναι η εντολή (η οποία βρίσκεται μέσα στις αγκύλες) που καθορίζει το τι θα επιστρέψει η μέθοδος από εκεί που θα την καλέσουμε.
7. Μπορούμε να έχουμε πολλές μεθόδους γραμμένες μέσα στο ίδιο script ή ακόμη και σε διαφορετικά συμπεριλαμβάνοντας τα από διαφορετικές πηγές.
8. Οι εντολές που περιέχει μια μέθοδος εκτελούνται όταν την καλέσουμε με το όνομα της.
9. Κάθε μέθοδος μπορεί να εκτελεστεί άπειρες φορές μέσα σε ένα script.

Ας υποθέσουμε πως θέλουμε να φτιάξουμε μια μέθοδο η οποία να επιστρέφει το άθροισμα δυο, ορισμένων αριθμών - παραμέτρων. Ο κώδικας που προκύπτει είναι ο εξής:

Αρχείο: fun.php

```
<?php
 function prosthesi($a,$b){
  return ($a+$b);
 }

 $arithmosa = 5;
 $arithmosb = 25;

 echo prosthesi($arithmosa,$arithmosb);
?>
```

Αποτέλεσμα:



Εικόνα Α.77 Αποτέλεσμα εκτέλεσης μεθόδου με χρήση εισερχόμενων παραμέτρων

Όπως βλέπουμε οι μεταβλητές $arithmosa και $arithmosb «μεταθέτουν» το περιεχόμενο τους στις μεταβλητές μεθόδου $a και $b. Έτσι μέσα στην μέθοδο η $arithmosa είναι η $a και η $arithmosb είναι η $b.

- Μια μέθοδος μπορεί να έχει όσες παραμέτρους θέλουμε,
- Οι παράμετροι χωρίζονται με κόμμα,
- Το όνομα της κάθε παραμέτρου ακολουθεί τους κανόνες ονομασίας μεταβλητών,
- Κάθε παράμετρος πρέπει να έχει μοναδικό όνομα,
- Όλες οι παράμετροι καταχωρούνται μέσα στις παρενθέσεις,
- Όταν καλείται η συνάρτηση θα πρέπει να ορίζονται και τιμές για τις αντίστοιχες παραμέτρους.

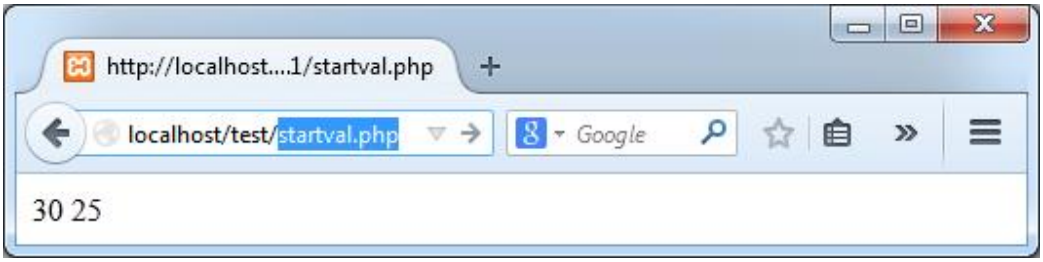
Όταν καλέσουμε την μέθοδο prosthesi στο τέλος του κώδικα μας δηλαδή στο echo, τότε θα εκτελεστεί με ορίσματα το 5 και το 25, όπου αν προσθέσουμε μας επιστρέφει 30, το οποίο και το αποτέλεσμα μας.

A.5.2 Αρχικές τιμές

Στις μεθόδους δίνεται η δυνατότητα ορισμού αρχικών τιμών. Παίρνοντας ως παράδειγμα το προηγούμενο με την πρόσθεση, θα μπορούσαμε να ορίσουμε προεπιλεγμένη αρχική τιμή στην οποία αν όταν κάποιος καλέσει την μέθοδο χωρίς να υπάρχει τιμή σε κάποιο όρισμα της, τότε να χρησιμοποιηθεί η αντίστοιχη αρχική της μεθόδου.

```
<?php
 function prosthesi($a,$b=20){
  return ($a+$b);
 }
 $arithmosa = 5;
 $arithmosb = 25;

 echo prosthesi($arithmosa,$arithmosb)."\n";
 echo prosthesi($arithmosa);
?>
```

Η παραπάνω εκτέλεση του κώδικα μας επιστρέφει το εξής αποτέλεσμα:



Εικόνα Α.78 Αρχικές τιμές σε εκτέλεση μεθόδου

Το πρώτο αποτέλεσμα προκύπτει από το κάλεσμα της μεθόδου δίνοντας όμως και δεύτερη παράμετρο. Το δεύτερο αποτέλεσμα προκύπτει μόνο δίνοντας την πρώτη παράμετρο και για δεύτερη ορίζεται η αρχική της που της έχουμε ορίσει.

Α.5.3 Εμβέλεια μεταβλητών

Η εμβέλεια των μεταβλητών καθορίζει κατά πόσο μία μεταβλητή θα είναι προσπελάσιμη από άλλες μεθόδους και κλάσεις.

Α.5.3.1 local

Μια μεταβλητή δηλωμένη μέσα σε μια μέθοδο είναι μια τοπική μεταβλητή της αντίστοιχης μεθόδου, αυτό σημαίνει πως η μεταβλητή αυτή είναι προσπελάσιμη μόνο μέσα στην μέθοδο, ζει και υπάρχει για όσο τρέχει η μέθοδος και μετά διαγράφεται από την μνήμη. Κάθε φορά που η ίδια μέθοδος τρέχει μια καινούργια τοπική μεταβλητή με το ίδιο όνομα δημιουργείται μέσα σε αυτήν και ζει για όσο χρονικό διάστημα χρειάζεται για να εκτελεστεί η μέθοδος. Στην php μόνο στις μεθόδους μπορούν να δημιουργηθούν τοπικές μεταβλητές σε αντίθεση με άλλες γλώσσες όπου σε βρόγχους και σε ελέγχους αποφάσεων μπορούσαν να κάνουν το ίδιο.

```
$vartest = "Κώστας";
```

Α.5.3.2 Global

Είναι μεταβλητές που είναι δηλωμένες έξω από μια συνάρτηση δηλαδή είτε στην κλάση ή λίγο πριν την συνάρτηση. Οι μεταβλητές αυτές είναι προσπελάσιμες από όλες τις άλλες συναρτήσεις και υπάρχουν για όσο εκτελείται το script.

```
global $vartest = "Γιάννης";
```

A.5.3.3 Static

Οι μεταβλητές αυτές είναι δηλωμένες μέσα σε μια συνάρτηση, είναι προσπελάσιμες μέσα από αυτή και ζουν ακόμα κι αν τελειώσει την εκτέλεση της η συνάρτηση. Έτσι όταν ξανά κληθεί η ίδια συνάρτηση η static μεταβλητή διατηρεί την τιμή της.

```
static $vartest = "Χρήστος";
```

A.6 ΑΛΦΑΡΙΘΜΗΤΙΚΑ

Όπως είδαμε και στην ενότητα Α.3.1 τα αλφαριθμητικά βρίσκονται εσωτερικά στην php για λόγους ευκολίας στην ανάπτυξη διαδικτυακών εφαρμογών. Σε αυτό το κεφάλαιο θα δούμε τι είδους πράξεις μπορούμε να κάνουμε με τα αλφαριθμητικά και πως μπορούμε να τα επεξεργαστούμε μέσα από την php.

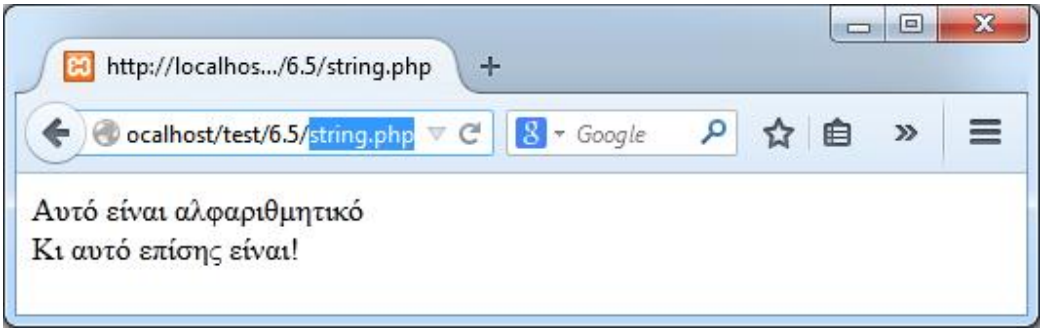
A.6.1 Προσπέλαση αλφαριθμητικών

Τα αλφαριθμητικά ή αλλιώς συμβολοσειρές είναι ένα σύνολο από χαρακτήρες που σχηματίζουν λέξεις και κείμενα. Είναι όλα αυτά στα οποία ένα script μπορεί να επικοινωνήσει μαζί μας, δηλαδή διαθέτει ένα σύνολο από κείμενα και προτάσεις που ένας χρήστης τις βλέπει και καταλαβαίνει περί τίνος πρόκειται, π.χ. ο τίτλος του κουμπιού «Αποστολή» είναι ένα αλφαριθμητικό. Γενικώς στον κώδικα οτιδήποτε κλείνεται μέσα σε μονά ή διπλά αυτάκια είναι αλφαριθμητικό. Ας δούμε ένα παράδειγμα:

Αρχείο: string.php

```
<?php
 $test = "Αυτό είναι αλφαριθμητικό";
 $test1 = 'Κι αυτό επίσης είναι!';
 echo $test."<br/>".$test1;
?>
```

Αποτέλεσμα:



Εικόνα Α.79 Χρήση αλφαριθμητικών σε μεταβλητές

Αν και έχουμε χρησιμοποιήσει πολύ αλφαριθμητικά σε προηγούμενες ενότητες, στην παρούσα ενότητα δεν προσπαθούμε να εξηγήσουμε τι είναι το αλφαριθμητικό, αλλά ποιους περιορισμούς έχουμε μέσα σε αυτό και ποίες πράξεις μπορούμε να κάνουμε.

Για παράδειγμα δεν μπορούμε να χρησιμοποιήσουμε μια σκέτη μεσοκάθετο «\» μέσα σε ένα αλφαριθμητικό γιατί ο συγκεκριμένος χαρακτήρας υποδηλώνει εντολή. Αν χρησιμοποιήσουμε αυτόν τον χαρακτήρα θα πρέπει να τον συνδυάσουμε με κάτι άλλο ακόμη.

Χαρακτήρας	Επεξήγηση
`\n`	Αφήνει μια γραμμή (LF)
`\r`	Αφήνει ένα enter (CR)
`\t`	Οριζόντιο tab (HT)
`\v`	Κάθετο tab (VT)
`\e`	Escape (ESC)
`\\`	Μέσοκάθετος
`\$`	Το σύμβολο του δολαρίου
`\"`	Το σύμβολο των διπλών αυτιών
`\[0-7]{1-3}`	Εισαγωγή χαρακτήρα με οκταδικό
`\x[0-9A-fa-f]{1,2}`	Εισαγωγή χαρακτήρα με δεκαεξαδικό

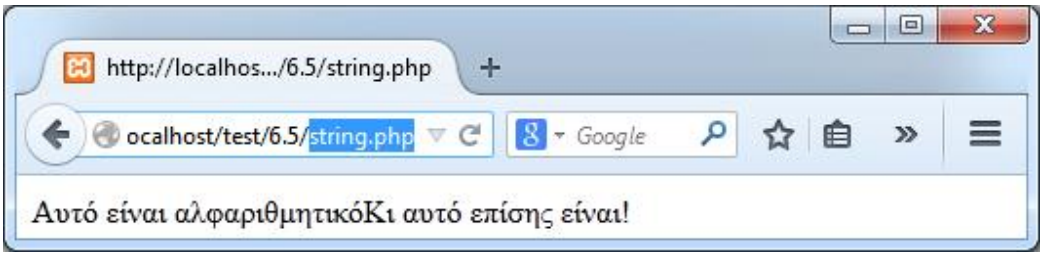
Πίνακας Α.18 Ειδικοί χαρακτήρες αλφαριθμητικών

Οι παραπάνω χαρακτήρες βοηθούν κυρίως στην ανάγνωση του πηγαίου κώδικα και όχι στην εμφάνιση του αποτελέσματος. Η Αλλαγή γραμμής για παράδειγμα με οποιαδήποτε τρόπο δεν θα εμφανιστεί στο παραγόμενο αποτέλεσμα άμεσα στον χρήστη, αλλά στον πηγαίο κώδικα που μπορεί να δει ο προγραμματιστής σε html. Στα αλφαριθμητικά δεν νοείται η πράξη της πρόσθεσης, αλλά η πράξη της συμπερίληψης, όταν θέλουμε να προσθέσουμε δυο αλφαριθμητικά δεν θα χρησιμοποιήσουμε το σύμβολο της πρόσθεσης, αλλά μία τελεία.

Αρχείο: string1.php

```
<?php
 $test = "Αυτό είναι αλφαριθμητικό";
 $test1 = 'Κι αυτό επίσης είναι!';
 echo $test.$test1;
?>
```

Αποτέλεσμα:



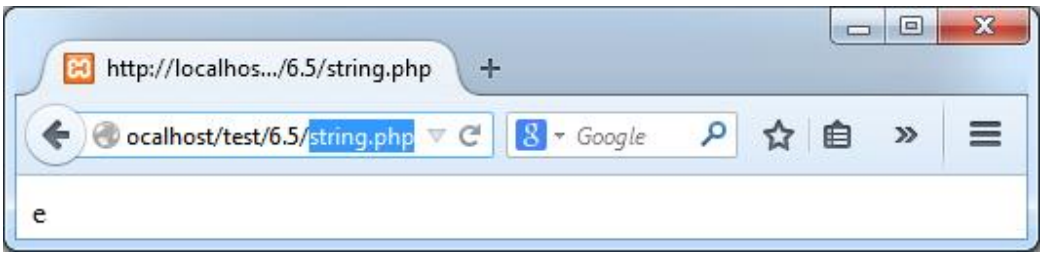
Εικόνα Α.80 Συμπερίληψη αλφαριθμητικών

Τέλος η προσπέλαση των αλφαριθμητικών γίνεται όπως σε έναν μονοδιάστατο πίνακα αλλά δεν υποστηρίζεται η Unicode κωδικοποίηση. Μπορούμε δηλαδή να χρησιμοποιήσουμε έναν μόνο χαρακτήρα μέσα από ένα αλφαριθμητικό, δίνοντας του την θέση στην οποία βρίσκεται. Ας δούμε ένα παράδειγμα:

Αρχείο: string3.php

```
<?php
 $test ="hello";
 echo $test[3];
?>
```

Αποτέλεσμα:



Εικόνα Α.81 Χρήση ενός μόνο χαρακτήρα από ένα αλφαριθμητικό

Αν θεωρήσουμε το αλφαριθμητικό έναν πίνακα, τότε κάθε χαρακτήρας βρίσκεται σε μία θέση μέσα σε αυτόν (ξεκινώντας πάντα από το 0).

A.6.2 Πράξεις με αλφαριθμητικά

Στα αλφαριθμητικά εφόσον έχουμε να κάνουμε με πολλά δεδομένα (λόγω πολλών χαρακτήρων ο ένας δίπλα στον άλλο), υπάρχουν εξ' ορισμού κάποιες μέθοδοι όπου μπορούν να μας κάνουν την ζωή ευκολότερη.

A.6.2.1 Συνάρτηση chr()

Είναι μια ανεξάρτητη συνάρτηση που παίρνει ως όρισμα έναν αριθμό και επιστρέφει τον χαρακτήρα ascii που αντιστοιχείται στον συγκεκριμένο αριθμό. Αυτό χρησιμεύει στις περιπτώσεις όπου υπάρχουν ειδικοί χαρακτήρες και δεν μπορούν να χρησιμοποιηθούν μέσα στον κώδικα.

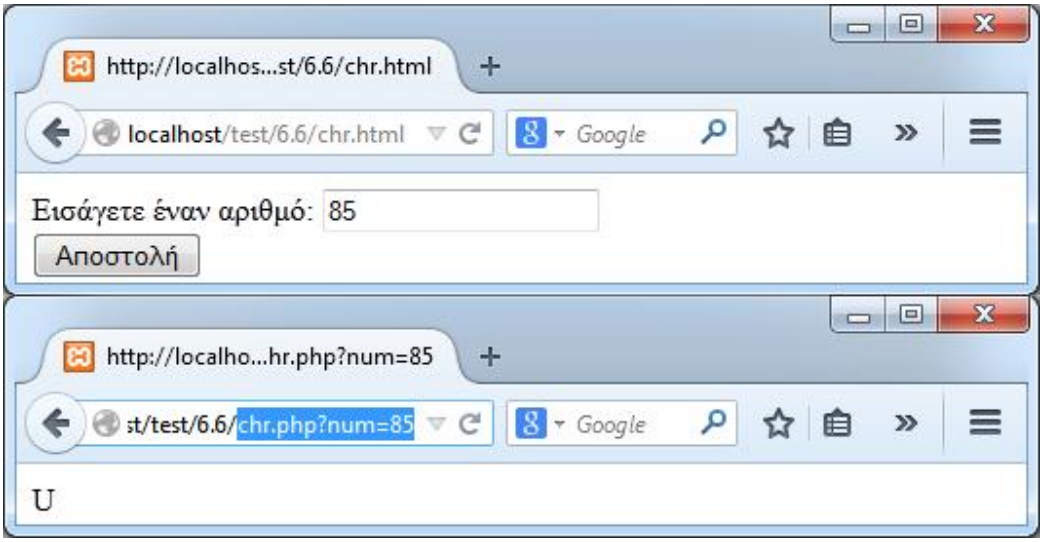
Αρχείο: chr.html

```
<form action='chr.php' method='get'>
 Εισάγετε έναν αριθμό: <input type='text' name='num'>
 <br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: chr.php

```
<?php
 echo chr($_GET['num']);
?>
```

Αποτέλεσμα:



Εικόνα A.82 Επιστροφή χαρακτήρα ascii με την χρήση της εντολής chr()

A.6.2.2 Συνάρτηση ord()

Αντίθετα με την «chr()» η «ord()» επιστρέφει τον αριθμό που αντιστοιχεί σε έναν ascii χαρακτήρα που της δώσαμε ως όρισμα.

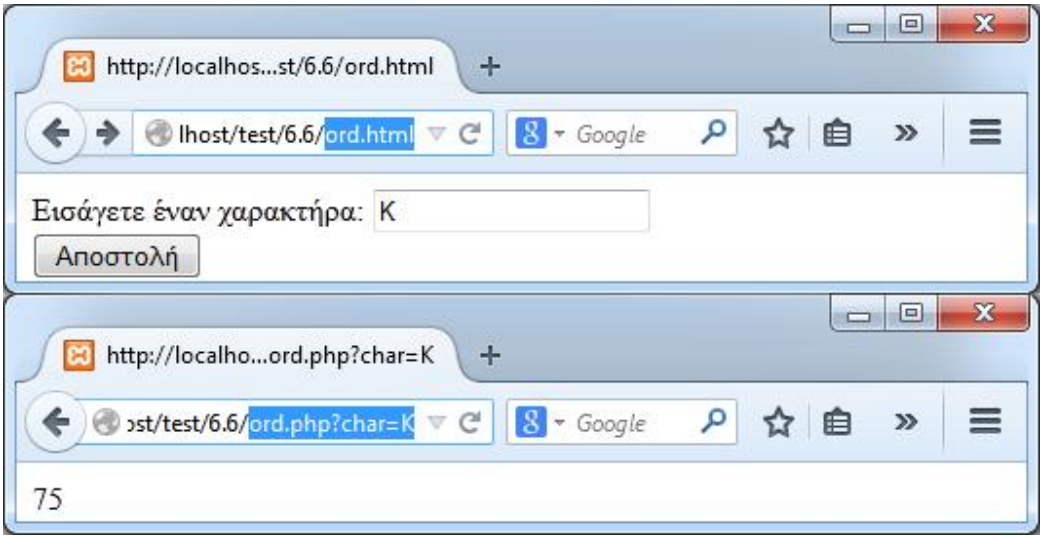
Αρχειο:ord.html

```
<form action='ord.php' method='get'>
 Εισάγετε έναν χαρακτήρα: <input type='text' name='char'>
 <br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο:ord.php

```
<?php
 echo ord($_GET['char']);
?>
```

Αποτέλεσμα:



Εικόνα Α.83 Επιστροφή αριθμού για την εισαγωγή χαρακτήρα στην ord()

A.6.2.3 Συνάρτηση explode()

Η συνάρτηση αυτή βοηθάει πολύ κατά την εισαγωγή δεδομένων από εξωτερικά αρχεία. Διαχωρίζει τα στοιχεία ενός αλφαριθμητικού στα οποία μεταξύ τους υπάρχει κάποιος διαχωριστικός χαρακτήρας π.χ. κόμμα, και ύστερα τα τοποθετεί σε έναν πίνακα τον οποίο επιστρέφει. Παίρνει δυο ορίσματα το ένα είναι ένας διαχωριστικός χαρακτήρας και το δεύτερο είναι το αλφαριθμητικό που θέλουμε να διαχωρίσει.

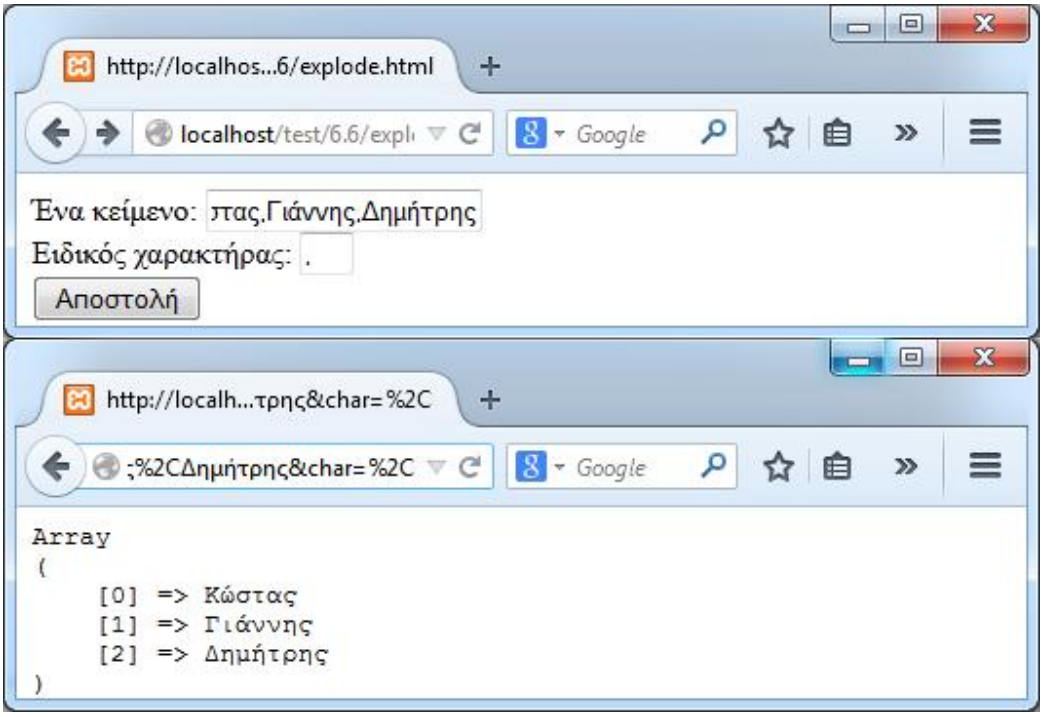
Αρχείο: explode.html

```
<form action='explode.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 Ειδικός χαρακτήρας: <input type='text' size='1' name='char'><br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: explode.php

```
<?php
 echo "<pre>";
 print_r(explode($_GET['char'],$_GET['text']));
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.84 Μετατροπή κειμένου σε πίνακα με την explode()

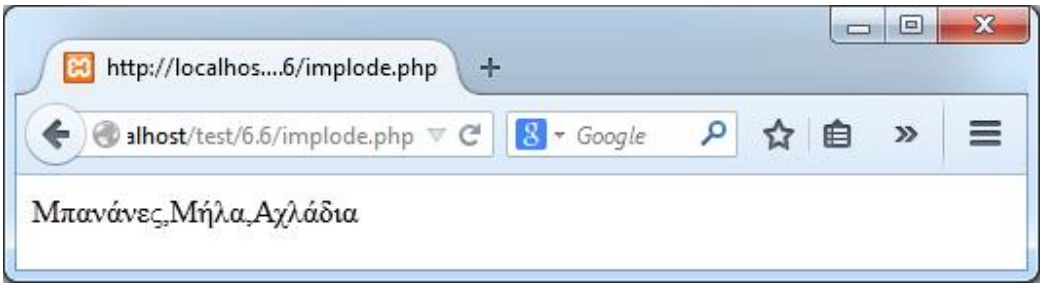
Α.6.2.4 Συνάρτηση implode()

Αντίθετα με την «explode()» η «implode()» παίρνει ως όρισμα έναν πίνακα, και επιστρέφει ένα αλφαριθμητικό με όλα τα στοιχεία του πίνακα διαχωρισμένα με ένα διαχωριστικό χαρακτήρα της επιλογής μας. Χρειάζεται δυο ορίσματα έναν χαρακτήρα διαχωρισμού και έναν πίνακα.

Αρχείο: implode.php

```php
<?php
 $pinakas = array("Μπανάνες","Μήλα","Αχλάδια");
 echo implode(",",$pinakas);
?>
```

Αποτέλεσμα:



Εικόνα Α.85 Μετατροπή πίνακα σε αλφαριθμητικό με την implode()

A.6.2.5 Συνάρτηση str_getcsv()

Τα αρχικά csv προέρχονται από τα Comma Seperated Values (Τιμές Διαχωρισμένες με Κόμμα). Αυτό που κάνει είναι να επιστρέφει έναν πίνακα για ένα αλφαριθμητικό το οποίο του δώσαμε ως είσοδο και περιέχει τέτοιες τιμές αποκλειστικά χωρισμένες με κόμμα. Δεν επιλέγουμε τον διαχωριστικό χαρακτήρα επομένως παίρνει μόνο μια παράμετρο, το αλφαριθμητικό.

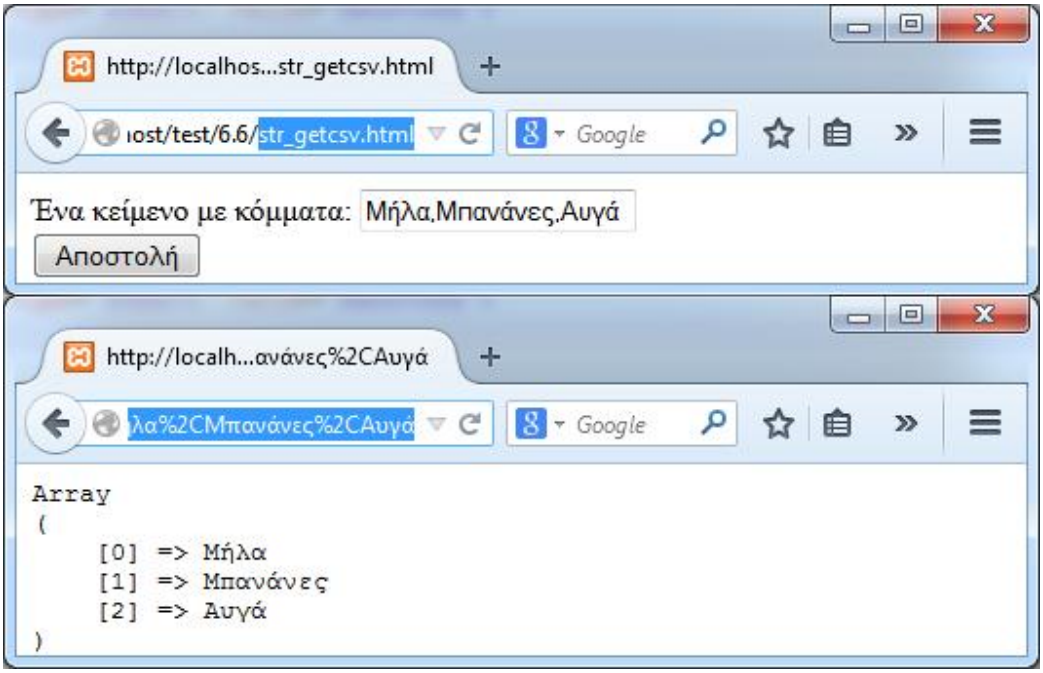
Αρχείο: str_getcsv.html

```php
<form action='str_getcsv.php' method='get'>
 Ένα κείμενο με κόμματα: <input type='text' name='text'>
 <br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: str_getcsv.php

```php
<?php
 echo "<pre>";
 print_r(str_getcsv($_GET['text']));
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.86 Μετατροπή csv κειμένου σε πίνακα με την str_getcsv()

Α.6.3 Επεξεργασία αλφαριθμητικών

Σε αυτή την ενότητα θα δούμε πως μπορούμε να επεξεργαστούμε το ίδιο το περιεχόμενο ενός αλφαριθμητικού αλλάζοντας του την δομή. Υπάρχουν συγκεκριμένες μέθοδοι που μπορούμε να επεξεργαστούμε ένα αλφαριθμητικό. Οι πιο βασικές περιγράφονται σε αυτή την ενότητα.

Α.6.3.1 Συνάρτηση substr()

Η συγκεκριμένη συνάρτηση χρειάζεται τρία ορίσματα. Μία αλφαριθμητική μεταβλητή, έναν αριθμό από όπου θα ξεκινήσει, και έναν αριθμό από όπου θα τελειώσει. Οι αριθμοί αυτοί οι δυο θα πρέπει να είναι μικρότεροι του μήκους του αλφαριθμητικού. Αυτό που κάνει είναι να επιστρέφει ένα κομμάτι από το αλφαριθμητικό που του δώσαμε, δεν επιδρά πάνω στα ορίσματα του.

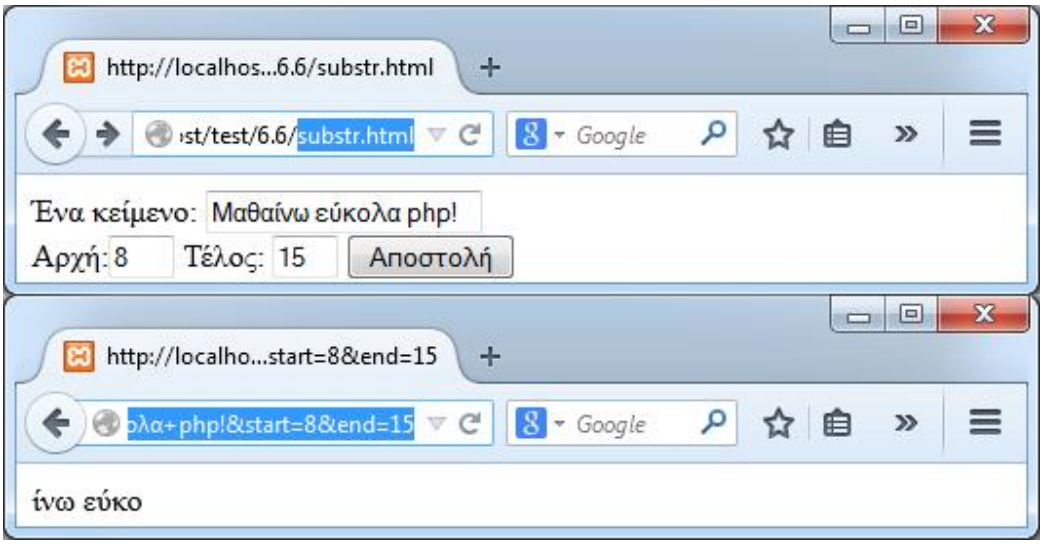
Αρχείο:substr.html

```
<form action='substr.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 Αρχή:<input type='text' name='start' size='2'>
 Τέλος: <input type='text' name='end' size='2'>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο:substr.php

```
<?php
 echo substr($_GET['text'],$_GET['start'],$_GET['end']);
?>
```

Αποτέλεσμα:



Εικόνα Α.87 Εξαγωγή υπό-αλφαριθμητικού με την substr()

Α.6.3.2 Συνάρτηση trim()

Αφαιρεί από ένα αλφαριθμητικό κενά και ειδικούς χαρακτήρες αριστερά και δεξιά του. Παίρνει ένα ή δυο ορίσματα, το πρώτο είναι το αλφαριθμητικό που θέλουμε να «καθαρίσουμε» και το δεύτερο προαιρετικά είναι ένας ειδικός χαρακτήρας που θέλουμε να αφαιρέσουμε. Αν δεν εισάγουμε χαρακτήρα διαγράφει όλα τα κενά. Το αποτέλεσμα του είναι επιστρεφόμενο και δεν επιδρά στα ορίσματα του.

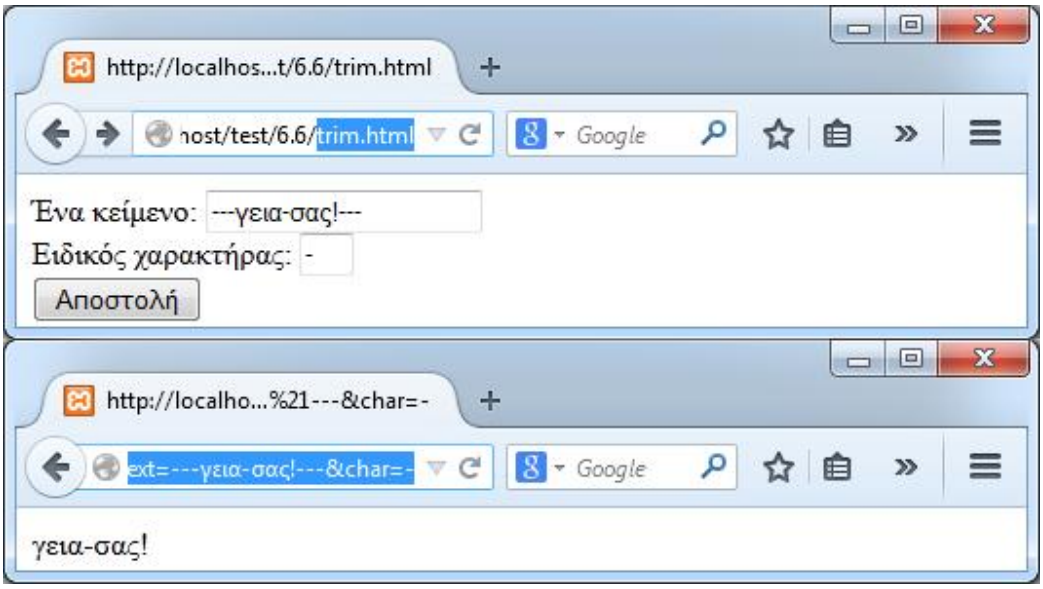
Αρχείο: trim.html

```
<form action='trim.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 Ειδικός χαρακτήρας: <input type='text' size='1' name='char'><br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: trim.php

```
<?php
 echo trim($_GET['text'],$_GET['char']);
?>
```

Αποτέλεσμα:



Εικόνα Α.88 Αφαίρεση ακραίων χαρακτήρων μέσω της trim()

Παρά του ότι του είπαμε ποιος είναι ο ειδικός χαρακτήρας, διέγραψε μόνο αυτούς που βρίσκονται αριστερά και δεξιά του αλφαριθμητικού και όχι ότι βρίσκεται μέσα σε αυτό.

Α.6.3.3 Συνάρτηση ltrim()

Παρομοίως με την «trim()» αφαιρεί δηλαδή όλους τους όμοιους χαρακτήρες που θα βρει σε ένα αλφαριθμητικό από την αριστερή μόνο πλευρά του. Το αποτέλεσμα είναι επιστρεφόμενο και δεν επιδρά σε κανένα όρισμα.

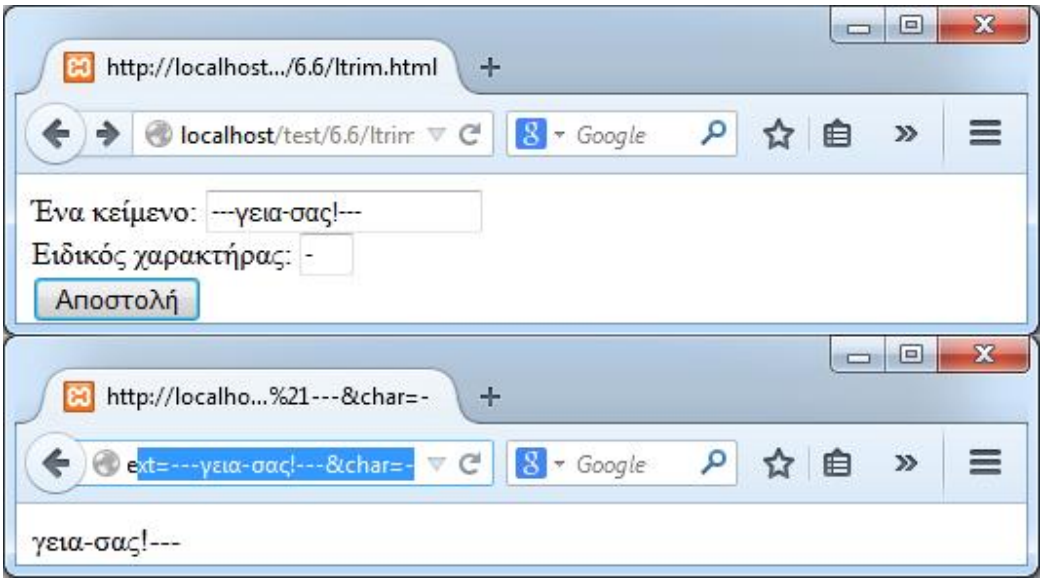
Αρχείο: ltrim.html

```
<form action='ltrim.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 Ειδικός χαρακτήρας: <input type='text' size='1' name='char'><br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: ltrim.php

```
<?php
 echo ltrim($_GET['text'],$_GET['char']);
?>
```

Αποτέλεσμα:



Εικόνα Α.89 Αφαίρεση χαρακτήρων από τα αριστερά με την ltrim()

Α.6.3.4 Συνάρτηση rtrim()

Παρομοίως με την «trim()» αφαιρεί δηλαδή όλους τους όμοιους χαρακτήρες που θα βρει σε ένα αλφαριθμητικό από την δεξιά μόνο πλευρά του. Το αποτέλεσμα είναι επιστρεφόμενο και δεν επιδρά σε κανένα όρισμα.

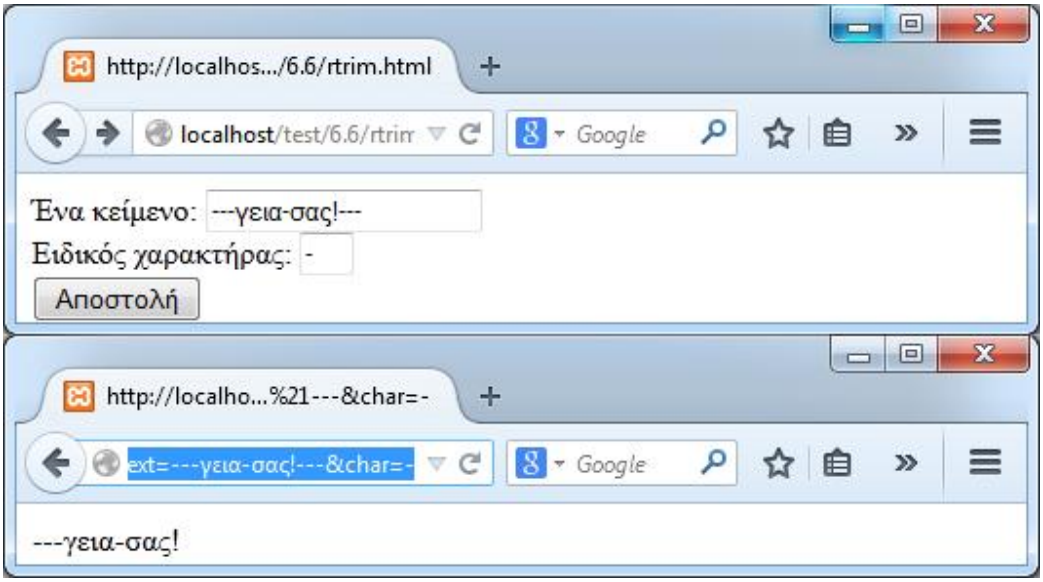
Αρχείο: rtrim.html

```
<form action='rtrim.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 Ειδικός χαρακτήρας: <input type='text' size='1' name='char'><br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: rtrim.php

```
<?php
 echo rtrim($_GET['text'],$_GET['char']);
?>
```

Αποτέλεσμα:



Εικόνα Α.90 Αφαίρεση χαρακτήρων από δεξιά με την rtrim()

Α.6.3.5 Συνάρτηση str_replace()

Η συγκεκριμένη συνάρτηση αντικαθιστά μια σειρά από χαρακτήρες που θα της δώσουμε με μία άλλη επίσης που θα της δώσουμε, μέσα σε μια αλφαριθμητική μεταβλητή. Παίρνει τρία ορίσματα, πρώτα το κείμενο που θέλουμε να βρούμε, μετά το κείμενο που θέλουμε να πάρει την θέση του, και τέλος την μεταβλητή ή το κείμενο που θέλουμε να γίνει αυτή η αλλαγή. Το αποτέλεσμα είναι επιστρεφόμενο και δεν επιδρά σε κανένα από τα τρία ορίσματα.

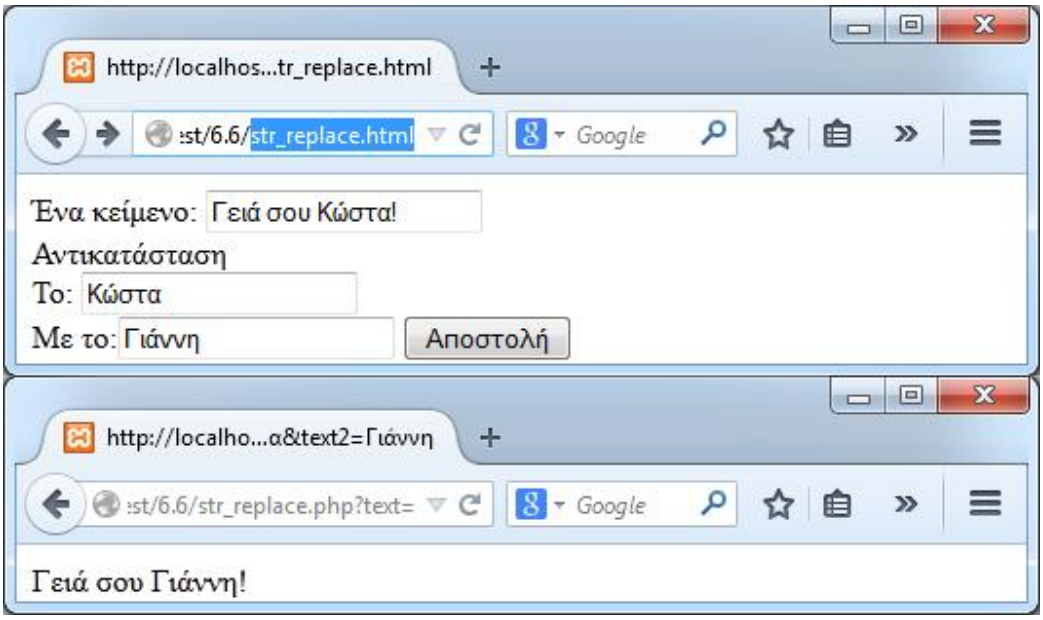
Αρχείο: str_replace.html

```
<form action='str_replace.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 Αντικατάσταση <br/>
 Το: <input type='text' name='text1'><br/>
 Με το:<input type='text' name='text2'>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: str_replace.php

```
<?php
 echo str_replace($_GET['text1'],$_GET['text2'],$_GET['text']);
?>
```

Αποτέλεσμα:



Εικόνα Α.91 Αντικατάσταση τμημάτων αλφαριθμητικού μέ την str_replace()

Α.6.3.6 Συνάρτηση str_shuffle()

Όπως και η μετάφραση του shuffle = τυχαία, έτσι και η συνάρτηση αυτό που κάνει είναι να ανακατεύει τυχαία τα περιεχόμενα μιας αλφαριθμητικής μεταβλητής. Ως παράμετρο παίρνει μόνο την μεταβλητή του αλφαριθμητικού μας. Η συγκεκριμένη συνάρτηση δεν λειτουργεί με Unicode χαρακτήρες και το αποτέλεσμα της επιστρέφεται, δεν επιδρά στην ίδια την μεταβλητή.

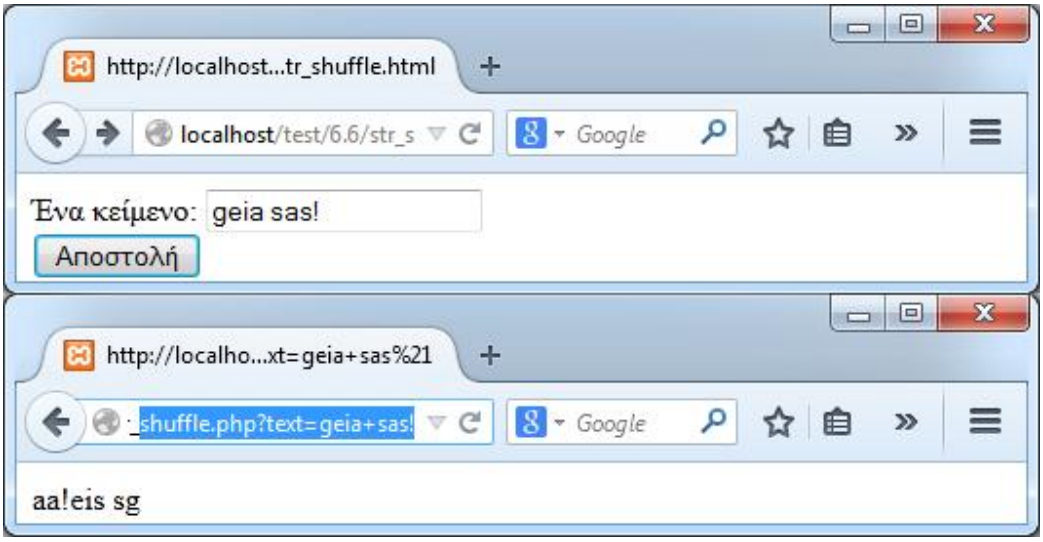
Αρχείο: str_shuffle.html

```
<form action='str_shuffle.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: str_shuffle.php

```
<?php
 echo str_shuffle($_GET['text']);
?>
```

Αποτέλεσμα:



Εικόνα Α.92 Τυχαία μεταβολή αλφαριθμητικού με την str_shuffle()

Α.6.3.7 Συνάρτηση strcmp()

Συγκρίνει δυο αλφαριθμητικά και επιστρέφει 0 αν είναι ίδια, αρνητικό αριθμό αν το πρώτο είναι μικρότερο του δεύτερου σε μέγεθος και θετικό αριθμό αν το πρώτο είναι μεγαλύτερο του δεύτερου επίσης σε μέγεθος.

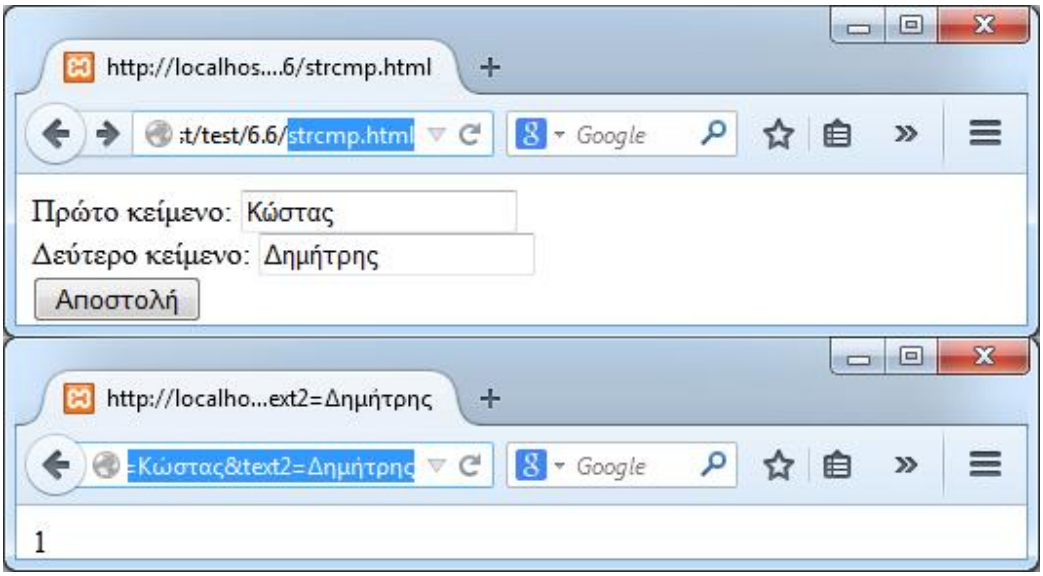
Αρχείο: strcmp.html

```
<form action='strcmp.php' method='get'>
 Πρώτο κείμενο: <input type='text' name='text1'>
 <br/>
 Δεύτερο κείμενο: <input type='text' name='text2'><br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: strcmp.php

```
<?php
 echo strcmp($_GET['text1'],$_GET['text2']);
?>
```

Αποτέλεσμα:



Εικόνα Α.93 Αποτέλεσμα σύγκρισης μεγέθους αλφαριθμητικών με την strcmp()

Α.6.3.8 Συνάρτηση str_split()

Η συγκεκριμένη συνάρτηση διαχωρίζει έναν – έναν τους χαρακτήρες ενός αλφαριθμητικού που του έχουμε εισάγει ως παράμετρο, και επιστρέφει έναν πίνακα με κάθε θέση του οποίου υπάρχει και από ένας χαρακτήρας του αλφαριθμητικού που του δώσαμε. Δεν λειτουργεί σωστά με Unicode χαρακτήρες. Το αποτέλεσμα είναι επιστρεφόμενο και δεν επιδρά στο όρισμα.

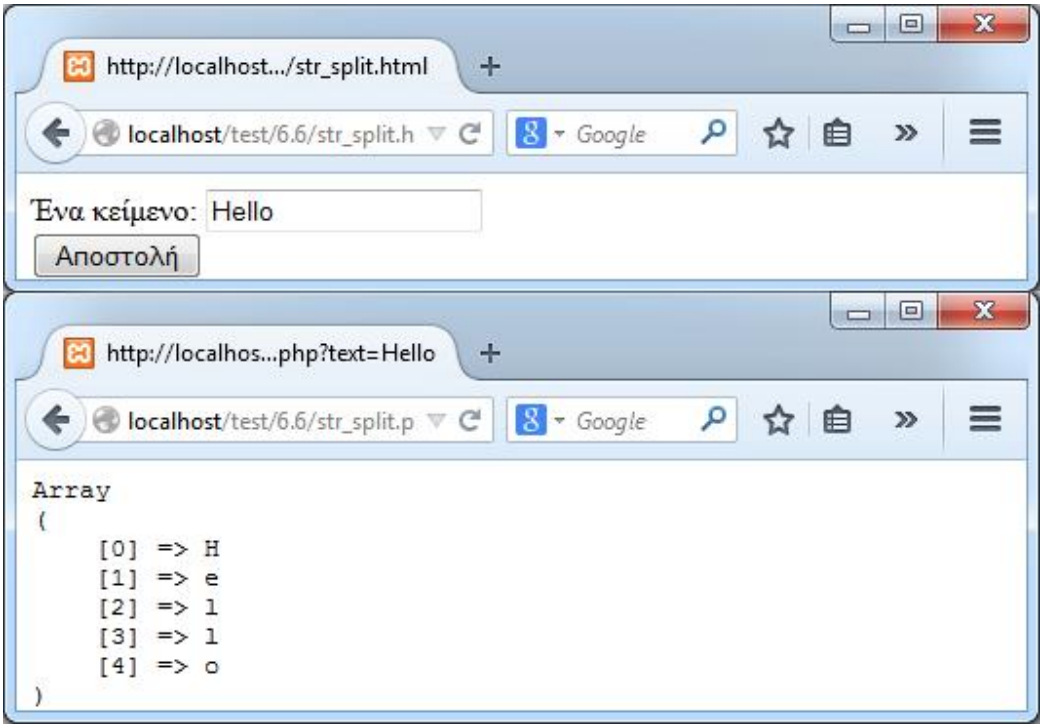
Αρχείο: str_split.html

```
<form action='str_split.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: str_split.php

```
<?php
 echo "<pre>";
 print_r(str_split($_GET['text']));
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.94 Διαχωρισμός αλφαριθμητικού σε πίνακα με την str_split()

Α.6.3.9 Συνάρτηση strrev()

Αντιστρέφει την σειρά του αλφαριθμητικού που παίρνει ως όρισμα. Δεν λειτουργεί με Unicode χαρακτήρες. Το αποτέλεσμα είναι επιστρεφόμενο και δεν επιδρά στο όρισμα.

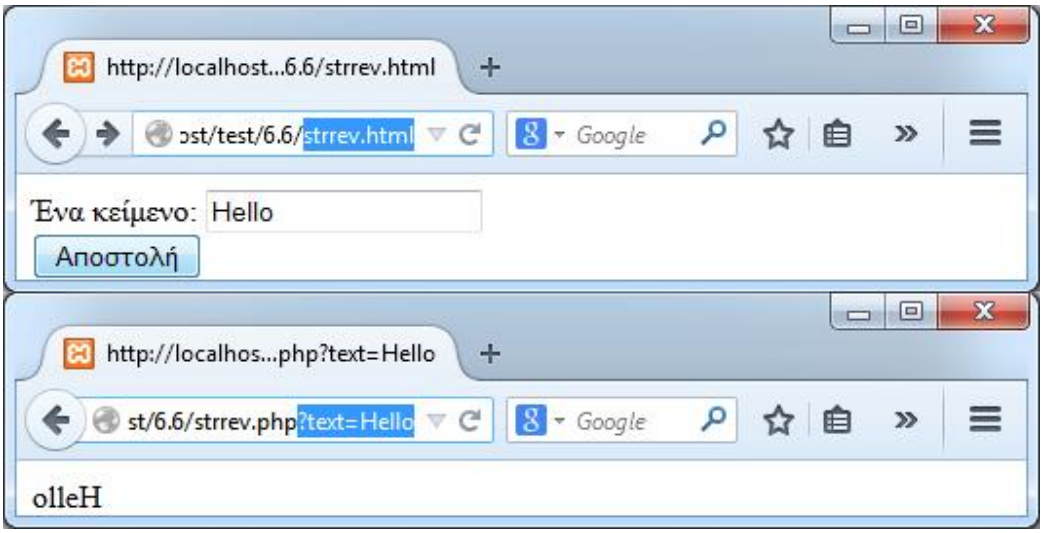
Αρχείο: strrev.html

```
<form action='strrev.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: strrev.php

```
<?php
 echo strrev($_GET['text']);
?>
```

Αποτέλεσμα:



Εικόνα Α.95 Αντιστροφή αλφαριθμητικού με την strrev()

Α.6.3.10 Συνάρτηση strtolower()

Μετατρέπει τα στοιχεία ενός αλφαριθμητικού που παίρνει ως όρισμα από κεφαλαία σε πεζά γράμματα. Δεν λειτουργεί με Unicode χαρακτήρες. Το αποτέλεσμα είναι επιστρεφόμενο και δεν επιδρά στο όρισμα.

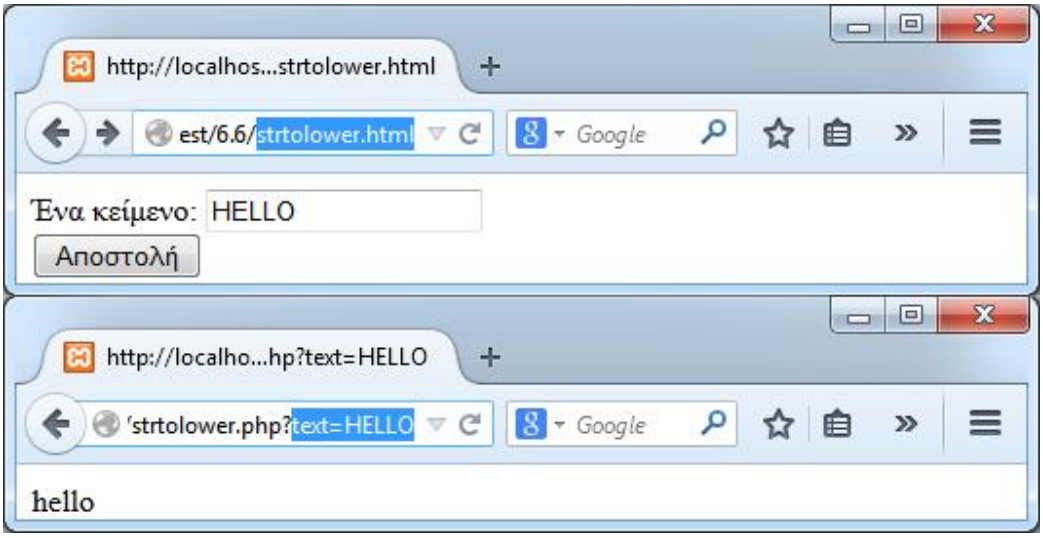
Αρχείο: strtolower.html

```
<form action='strtolower.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: strtolower.php

```
<?php
 echo strtolower($_GET['text']);
?>
```

Αποτέλεσμα:



Εικόνα Α.96 Μετατροπή από κεφαλαία σε πεζά με την strtolower()

Α.6.3.11 Συνάρτηση strtoupper()

Αντίθετα με την «strtolower()» μετατρέπει τα στοιχεία ενός αλφαριθμητικού που παίρνει ως όρισμα από πεζά σε κεφαλαία γράμματα. Δεν λειτουργεί με Unicode χαρακτήρες. Το αποτέλεσμα είναι επιστρεφόμενο και δεν επιδρά στο όρισμα.

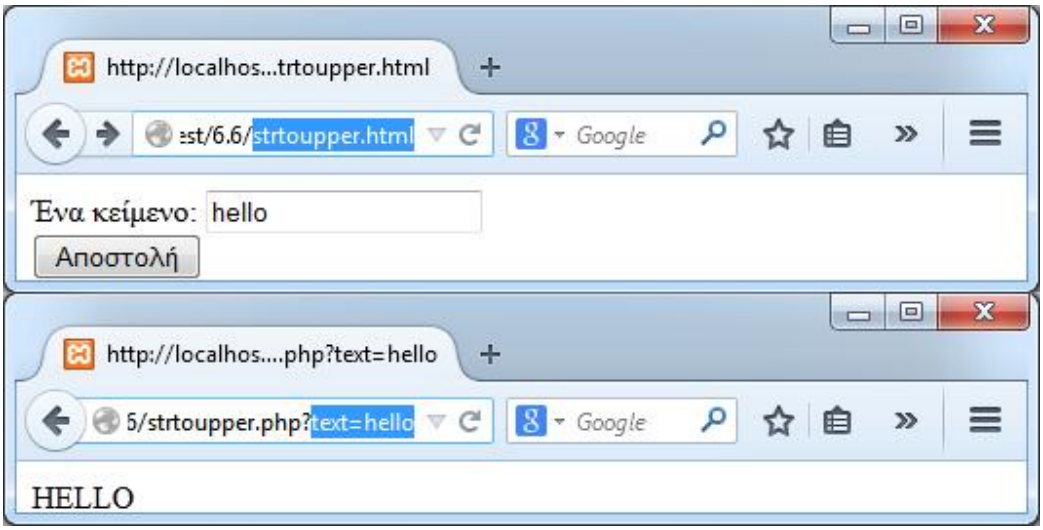
Αρχείο: strtoupper.html

```
<form action='strtoupper.php' method='get'>
 Ένα κείμενο: <input type='text' name='text'>
 <br/>
 <input type='submit' value='Αποστολή'>
</form>
```

Αρχείο: strtoupper.php

```
<?php
 echo strtoupper($_GET['text']);
?>
```

Αποτέλεσμα:



Εικόνα Α.97 Μετατροπή από πεζά σε κεφαλαία με την strtoupper()

Α.6.4 Κανονικές εκφράσεις

Οι κανονικές εκφράσεις στην php χρησιμοποιούνται έτσι ώστε να αναγνωριστούν συγκεκριμένα «μοτίβα» αλφαριθμητικών μέσα από άλλα αλφαριθμητικά , επιβεβαιώνοντας έτσι την ύπαρξη των πρώτων στα δεύτερα. Θα μπορούσαμε δηλαδή να αναγνωρίσουμε αν μέσα σε ένα κείμενο είναι γραμμένη μια διεύθυνση email, ένα τηλέφωνο ή οτιδήποτε άλλο του υποδείξουμε εμείς. Σε αυτή την ενότητα θα δούμε αναλυτικά πως αυτό

Α.6.4.1 Συνάρτηση preg_match

Στην preg_match πάντοτε χρησιμοποιείται ένα μοτίβο (pattern) το οποίο περιέχει σύμβολα μαζί με ειδικούς χαρακτήρες οι οποίοι αποτελούν και τους «κανόνες» του ζητούμενου αλφαριθμητικού. Το μοτίβο ξεκινά πάντα με «/» και τελειώνει με «/».

Παράδειγμα	Επεξήγηση
`^www`	Συμβολίζει ένα string που ξεκινά σε «www».
`com$`	Συμβολίζει ένα string που τελειώνει σε «com».
`w.rld`	Έναν οποιοδήποτε χαρακτήρα ανάμεσα στο «w» και το «rld».
`earth|world`	Θα πρέπει να περιλαμβάνει «earth» ή «world».
`a*bc`	Συμβολίζει πως το a θα μπορεί να εμφανίζεται από 0 έως άπειρες φορές. Πχ bc, abc, aabc, aaabc κ.α.
`[a-z]+`	Συμβολίζει πως θα εμφανιστεί ένας χαρακτήρας από a-z από 1 έως άπειρες φορές. Δηλαδή είναι υποχρεωτικό το να εμφανιστεί ένας τουλάχιστον.
`world?`	Υπάρχει ένα ή κανένα «d» μετά το «worl».
`world{1}`	Υπάρχει υποχρεωτικά ένα «d» μετά το «worl».
`world{1,}`	Υπάρχει ένα ή περισσότερα «d» μετά το «worl».

world{2,3}	Υπάρχουν δύο ή τρία «d» μετά το «worl».
^,{5}$	Ένα string με ακριβώς πέντε χαρακτήρες.
[a-z]	Έναν μικρό χαρακτήρα μόνο από το a έως το z.
[abc]	Έναν από τους χαρακτήρες «a»,«b»,«c».
[a-zA-Z]	Έναν κεφαλαίο ή μικρό χαρακτήρα από το a έως το z.
[^wW]	Να μην εμπεριέχεται το w ή το W.
world/i	Αναζήτηση με διάκριση στους πεζούς-κεφαλαίους χαρακτήρες.
\.	Ένας ειδικός χαρακτήρας να φαίνεται ακριβώς όπως είναι.

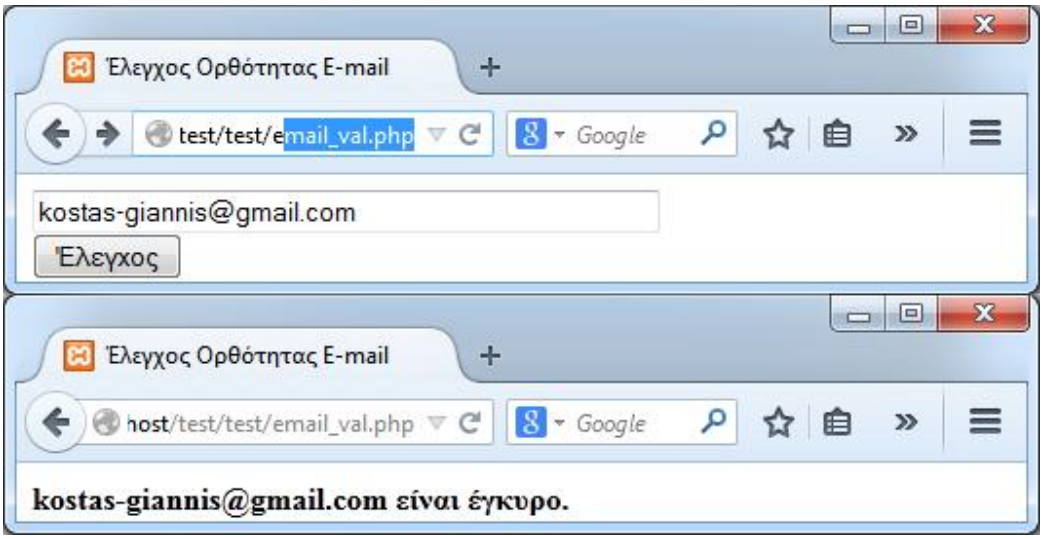
Πίνακας Α.19 Επεξήγηση αλφαριθμητικών μοτίβων

Ας υποθέσουμε πως θέλουμε να «ελέγξουμε» αν ένας χρήστης έχει εισάγει σωστά τα στοιχεία του σε μια φόρμα εγγραφής η οποία περιλαμβάνει μόνο το e-mail του.

Αρχείο: email_val.php

```
<html>
 <head>
  <title>Ελεγχος Ορθότητας E-mail</title>
 </head>
 <body>
  <?php
   if(isset($_POST['email'])){
    $email = $_POST['email'];

    $pattern='/^[a-zA-Z][a-zA-Z0-9-._]+@[a-zA-Z0-9-]+\.[a-zA-
Z.]{2,5}$/';

    if(!preg_match($pattern, $email)){
     echo "<strong>$email δεν είναι έγκυρο.</strong>";
    } else {
     echo "<strong>$email είναι έγκυρο.</strong>";
    }
   }else{?>
    <form action='email_val.php' method='post'>
     <input type='text' size='50' name='email' />
     <br/>
     <input type='submit' value='Ελεγχος'/>
    </form>
   <?php } ?>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Α.98 Έλεγχος σωστό email με την preg_match()

Ξέρουμε ότι μια διεύθυνση e-mail:

- Ξεκινά με ένα γράμμα πεζό ή κεφαλαίο

  `/^[a-zA-Z]`
- Μετά το πρώτο γράμμα μπορεί να περιέχει οποιοδήποτε λατινικό χαρακτήρα πεζό ή κεφαλαίο, αριθμούς , είτε την «τελεία», «παύλα», «κάτω παύλα». Αυτό θα μπορεί να επαναλαμβάνεται από 1 έως όσες φορές θέλουμε.

  `[a-zA-Z0-9-._]+`
- Ύστερα ο υποχρεωτικός χαρακτήρας «@»
- Μετά το «@» θα πρέπει να υπάρχει το όνομα του πάροχου υπηρεσίας email όπου μπορεί να είναι με πεζά η κεφαλαία γράμματα, αριθμούς και την παύλα. Τα παρακάτω θα πρέπει να υπάρχουν από μια φορά και παραπάνω.

  `[a-zA-Z0-9-]+`
- Ύστερα ακολουθεί μια υποχρεωτική «τελεία».

  `\.`
- Ορίζουμε το top-level domain του παρόχου που μπορεί να είναι από 2 έως 5 χαρακτήρες.

  `[a-zA-Z.]{2,5}`
- Και τέλος γράφουμε τον χαρακτήρα τέλους του μοτίβου «$».

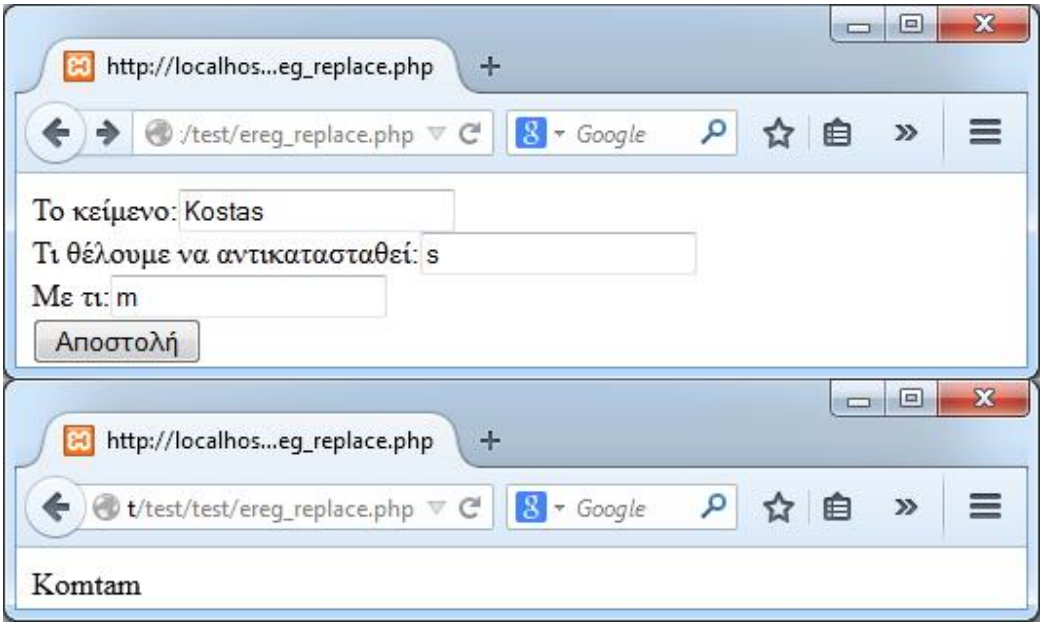
A.6.4.2 Συνάρτηση ereg_replace()

Παίρνει τρείς εισόδους αλφαριθμητικών, πρώτα αυτό που θέλουμε να αντικαταστήσουμε, δεύτερο ότι θέλουμε όταν αντικατασταθεί να υπάρχει, και τρίτο το αλφαριθμητικό στο οποίο θέλουμε να συμβούν όλα αυτά. Το αποτέλεσμα είναι επιστρεφόμενο και δεν επιδρά σε κανένα όρισμα.

Αρχείο: ereg_replace.php

```
<html>
 <head>
 </head>
 <body>
  <?php
   if(isset($_POST['str']) && isset($_POST['repl']) &&
isset($_POST['with'])){
    echo ereg_replace($_POST['repl'],$_POST['with'],$_POST['str']);
   } else {?>
    <form action='ereg_replace.php' method='post'>
     Το κείμενο:<input type='text' name='str'/><br/>
     Τι θέλουμε να αντικατασταθεί:<input type='text' name='repl'/>
<br/>
     Με τι:<input type='text' name='with'/><br/>
     <input type='submit' value='Αποστολή'/>
    </form>
  <?php } ?>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Α.99 Αντικατάσταση χαρακτήρα σε αλφαριθμητικό με την ereg_replace()

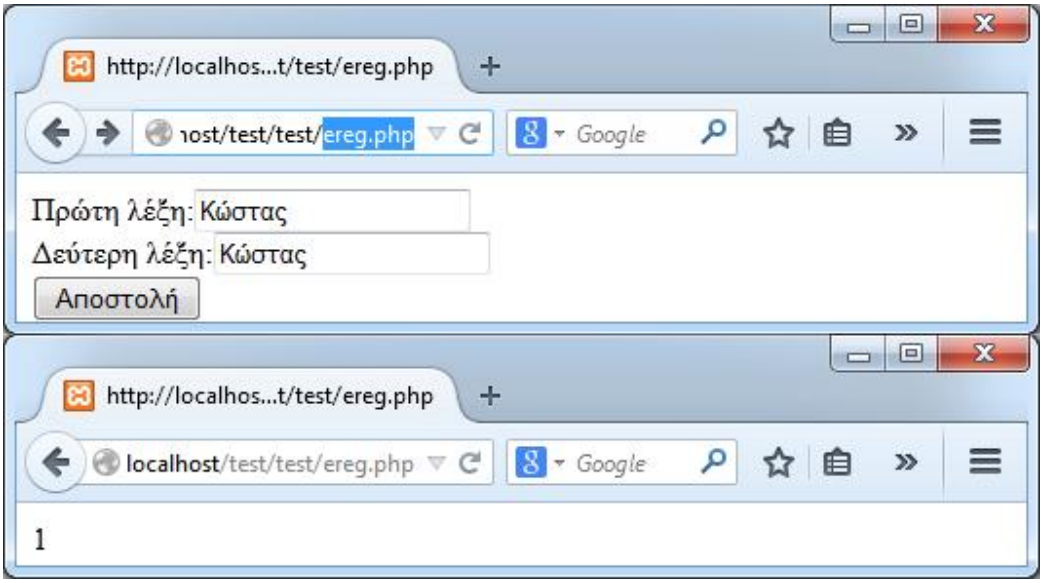
A.6.4.3 Συνάρτηση ereg()

Η συνάρτηση αυτή συγκρίνει δυο αλφαριθμητικά και πιο συγκεκριμένα το δεύτερο σύμφωνα με το πρώτο και επιστρέφει 1 αν είναι ίδια ή false σε διαφορετική περίπτωση.

Αρχείο: ereg.php

```
<html>
 <head>
 </head>
 <body>
  <?php
   if(isset($_POST['text1']) && isset($_POST['text2'])){
    echo ereg($_POST['text2'],$_POST['text1']);
   } else {?>
    <form action='ereg.php' method='post'>
     Πρώτη λέξη:<input type='text' name='text1'/><br/>
     Δεύτερη λέξη:<input type='text' name='text2'/> <br/>
     <input type='submit' value='Αποστολή'/>
    </form>
  <?php } ?>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Α.100 Σύγκριση αλφαριθμητικών με την ereg()

Α.7 ΠΙΝΑΚΕΣ

Ένας πίνακας είναι ένα σύνολο από στοιχεία διαφόρων τύπων δεδομένων οργανωμένα και ταξινομημένα με βάση κάποιο κλειδί – συντεταγμένη.

Φανταστείτε έναν πίνακα ως ένα λεωφορείο, κάθε θέση μέσα στο λεωφορείο έχει έναν ξεχωριστό αριθμό που μπορεί να φιλοξενήσει έναν επιβάτη. Δεν χρειάζεται όμως οπωσδήποτε αυτό που θα φιλοξενήσει να είναι επιβάτης, μπορεί ακόμη να είναι τα ψώνια του επιβάτη, ένα σακίδιο, ή οτιδήποτε άλλο μπορεί να τοποθετηθεί πάνω σε μια τέτοια θέση. Όπως είναι λογικό, ένας επιβάτης που αρχικά έκατσε στην θέση 30 του λεωφορείου μπορεί για οποιοδήποτε λόγο να σηκωθεί και να αλλάξει θέση πηγαίνοντας σε οποιαδήποτε άλλη βρει κενή.

Όπως στο λεωφορείο, και στην php δεν είναι υποχρεωτικό να φιλοξενεί κάθε θέση ενός πίνακα μόνο ένα συγκεκριμένο τύπο δεδομένων, αλλά μπορεί να φιλοξενεί μια ποικιλία από ακέραιους, δεκαδικούς, αλφαριθμητικά, αντικείμενα, ακόμη και άλλους πίνακες. Υπάρχουν δυο είδη πινάκων στην php αναλόγως το είδος των κλειδιών που διαθέτει.

Στο παρόν κεφάλαιο θα δούμε πως μπορούμε να δημιουργήσουμε έναν πίνακα, να τον προσπελάσουμε, να του προσθέσουμε στοιχεία, να διαγράψουμε και γενικά να κάνουμε όλες τις δυνατές πράξεις με πίνακες.

Α.7.1 Δημιουργία πίνακα

Συχνά εμφανίζεται στον προγραμματισμό η ανάγκη χρήσης ομαδοποιημένων μεταβλητών, είτε ίδιου τύπου είτε μεικτών τύπων.

Αν θέλαμε να χρησιμοποιήσουμε απλές μεταβλητές για να αντιπροσωπεύσουμε τα ονόματα φοιτητών ενός ΤΕΙ, τότε θα έπρεπε να χρησιμοποιήσουμε μία ξεχωριστή μεταβλητή για κάθε περίπτωση:

```
<?php
 $spoudastes1 = "Κωνσταντίνος";
 $spoudastes2 = "Δημήτρης";
 $spoudastes3 = "Αντώνης";
 $spoudastes4 = "Γιάννης";
?>
```

Για να μην χρησιμοποιούμε τόσες πολλές μεταβλητές και κυρίως για ευκολία στον προγραμματισμό, τόσο στην ανάπτυξη όσο και στην συντήρηση και την διόρθωση, χρησιμοποιούμε τους πίνακες.

Μια μεταβλητή είναι μια θέση στην μνήμη με όνομα στην οποία αποθηκεύεται προσωρινά μια τιμή. Ο πίνακας είναι μια θέση με όνομα στην μνήμη που αποθηκεύονται ένα σύνολο από τιμές. Έτσι μπορούμε να ομαδοποιήσουμε αυτές τις «αναφορές» των μεταβλητών στην μνήμη χρησιμοποιώντας μια «αναφορά» με κλειδιά , δηλαδή έναν πίνακα.

Το κλειδί ενός πίνακα, είναι αυτό που καθορίζει ποια τιμή χρησιμοποιούμε εκείνη την στιγμή. Με βάση το παραπάνω παράδειγμα θα μπορούσαμε να ορίσουμε ότι η μεταβλητή «$spoudastes1» έχει κλειδί «1» η μεταβλητή «$spoudastes2» έχει κλειδί «2» κτλ. Αυτό μας βοηθά να προσπελαύνουμε τις μεταβλητές αυτές με εύκολο τρόπο μέσω βρόγχων.

Η δημιουργία ενός πίνακα επιτυγχάνεται χρησιμοποιώντας ένα όνομα μεταβλητής μαζί με δυο άγκιστρα με μια τιμή μέσα σε αυτά, είτε ακέραια, είτε αλφαριθμητική. Η τιμή αυτή, που θα εισάγουμε μέσα στα άγκιστρα, ονομάζεται κλειδί του πίνακα, και μας δείχνει ποιο στοιχείο ακριβώς έχουμε την εκάστοτε στιγμή. Αφού αποφασίσουμε για το όνομα του πίνακα και των κλειδιού ύστερα σε εκείνη την θέση του πίνακα θα πρέπει να αναθέσουμε μια τιμή, προκείμενου ο πίνακας να δημιουργηθεί.

Υπάρχουν δυο είδη πινάκων στην php, οι αριθμημένοι και οι μη αριθμημένοι. Τα κλειδιά θέσεων ενός αριθμημένου πίνακα είναι ακέραιοι αριθμοί ξεκινώντας πάντα από το μηδέν. Αυτού του τύπου πίνακες χρησιμοποιούνται όταν θέλουμε να βρίσκουμε ένα στοιχείο με βάση την συντεταγμένη θέση του. Αντίθετα οι μη αριθμημένοι πίνακες διαθέτουν ως κλειδιά θέσεων, αλφαριθμητικά και όχι ακέραιους αριθμούς. Αυτού του είδους οι πίνακες λειτουργούν σαν πίνακες δυο στηλών όπου η πρώτη στήλη είναι το κλειδί και η δεύτερη, η τιμή μέσα στην θέση του αντίστοιχου κλειδιού.

Πρακτικά η php χρησιμοποιεί όλους τους πίνακες ως πίνακες με αλφαριθμητικά κλειδιά, η μόνη διαφορά με τα αριθμητικά είναι το διαφορετικό είδος τύπου

δεδομένων. Μερικοί πίνακες παρέχονται για αποκλειστική χρήση αριθμημένης διάταξης έτσι ώστε να υπάρχει μια διαδοχική συνοχή στην σειρά των στοιχειών που προσπελαύνουμε, ξεκινώντας πάντα από το μηδέν. Σε κάθε περίπτωση (αριθμημένων και μη) ένας πίνακας δεν μπορεί να διαθέτει δυο ίδια κλειδιά.

Η σειρά με την οποία εισάγονται τα στοιχεία σε έναν πίνακα, είναι και αυτή η οποία καθορίζει την σειρά των στοιχείων.

Για να δημιουργήσουμε έναν πίνακα θα πρέπει πρώτα να έχουμε κάτι να εισάγουμε σε αυτόν. Μπορούμε απλά να εισάγουμε μια τιμή σε μια θέση ενός πίνακα που ακόμη δεν υπάρχει και έτσι θα δημιουργηθεί. Αν χρησιμοποιήσουμε μια θέση χωρίς να έχει υπάρξει ήδη ο πίνακας από πριν, τότε δεν θα δημιουργηθεί νέος πίνακας.

```
<?php
 echo $pinakas[0]; //Ο πίνακας δεν υπάρχει ακόμη
 $pinakas[0] = 2014;
?>
```

A.7.1.1 Δημιουργία αριθμημένου πίνακα

Για να δημιουργήσουμε έναν αριθμημένο πίνακα τότε θα πρέπει με την σειρά να εισάγουμε στοιχεία σε κάθε θέση του που επιθυμούμε. Χρησιμοποιούμε πάντα την ίδια αναφορά-όνομα όταν θέλουμε να χρησιμοποιήσουμε τον συγκεκριμένο πίνακα, και πάντοτε όταν εισάγουμε στοιχεία προσέχουμε το εκάστοτε κλειδί:

```
<?php
 $spoudastes[0] = "Κωνσταντίνος";
 $spoudastes[1] = "Δημήτρης";
 $spoudastes[2] = "Αντώνης";
 $spoudastes[3] = "Γιάννης";
?>
```

Εναλλακτικά χρησιμοποιώντας την μέθοδο array() η οποία δημιουργεί έναν πίνακα με βάση τα ορίσματα που θα της εισάγουμε. Αυτός ο τρόπος δημιουργεί κλειδιά ξεκινώντας με πρώτο κλειδί το «0».

```
<?php
 $spoudastes = array("Κωνσταντίνος","Δημήτρης","Αντώνης","Γιάννης");
?>
```

A.7.1.2 Δημιουργία μη αριθμημένου πίνακα

Για να δημιουργήσουμε έναν πίνακα με αλφαριθμητικά κλειδιά αρκεί μόνο να αντικαταστήσουμε τον ακέραιο αριθμό με ένα αλφαριθμητικό:

```
<?php
 $spoudastes['Πρόεδρος'] = "Κωνσταντίνος";
 $spoudastes['Αντιπρόεδρος'] = "Δημήτρης";
 $spoudastes['Γραμματέας']= "Αντώνης";
?>
```

Εναλλακτικά χρησιμοποιώντας την μέθοδο array() μόνο που στην συγκεκριμένη περίπτωση χωρίζουμε τα κλειδιά με τις τιμές με ένα σύμβολο «=>» π.χ.

```
<?php
 $spoudastes = array(
    'Πρόεδρος' => "Κωνσταντίνος",
    'Αντιπρόεδρος' => "Δημήτρης",
    'Γραμματέας' => "Αντώνης"
 );
?>
```

A.7.1.3 Κενός πίνακας

Για να δημιουργήσουμε ένα κενό πίνακα αρκεί να χρησιμοποιήσουμε την μέθοδο array() χωρίς ορίσματα.

```
<?php
 $kenos = array();
?>
```

A.7.1.4 Αυτόματα κλειδιά

Μπορούμε να δημιουργήσουμε αυτόματα κλειδιά με το σύμβολο «=>» και μια σειρά από τιμές χωρισμένες με κόμμα:

```
<?php
 $imeres = array(1=>"ΔΕΥ","ΤΡΙ","ΤΕΤ","ΠΕΜ","ΠΑΡ","ΣΑΒ","ΚΥΡ");
?>
```

Στο παραπάνω παράδειγμα το κλειδί είναι 1 για το «ΔΕΥ», το κλειδί 2 για το «ΤΡΙ» … το κλειδί 7 για το «ΚΥΡ».

Αν χρησιμοποιήσουμε ως αρχικό κλειδί ένα αλφαριθμητικό τότε οι υπόλοιπες τιμές παίρνουν αριθμημένα κλειδιά ξεκινώντας 0.

```
<?php
 $imeres = array('Αρχηγός'=>"Κωνσταντίνος","Δημήτρης","Αντώνης");
?>
```

Το οποίο μας δίνει:

Κλειδί	Τιμή
Αρχηγός	Κωνσταντίνος
0	Δημήτρης
1	Αντώνης

Πίνακας Α.20 Δημιουργία πίνακα με ένα στοιχείο μη αριθμημένο

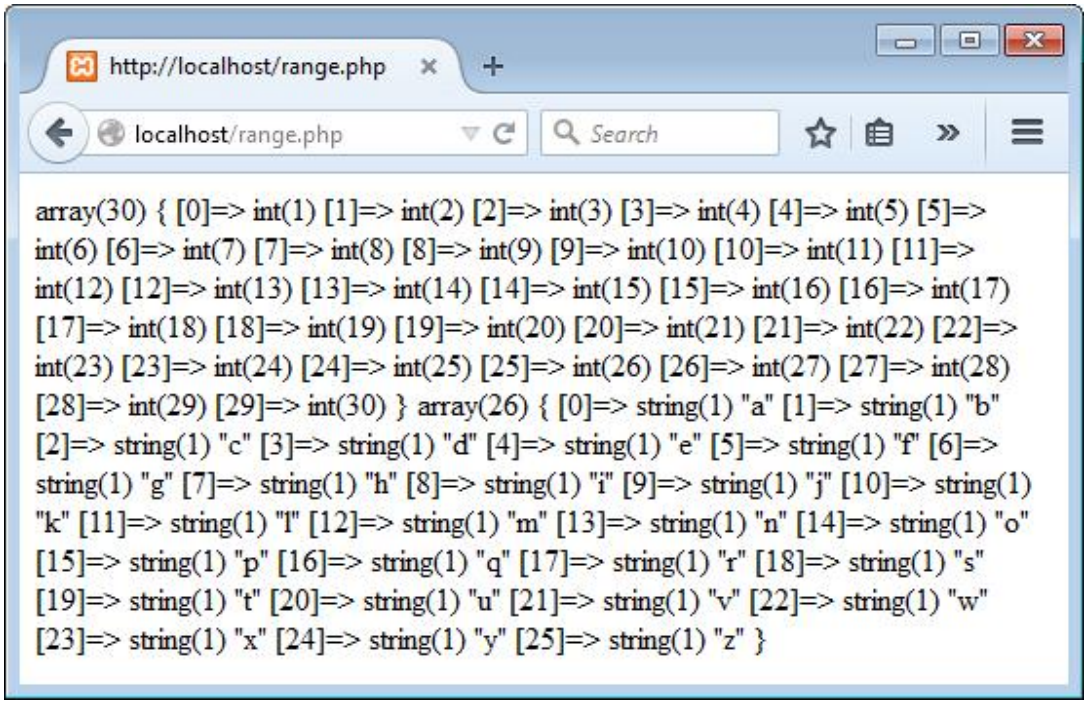
Α.7.1.5 Εύρος τιμών – range()

Η εντολή «range()» δημιουργεί έναν νέο πίνακα στηριγμένο σε ένα εύρος τιμών. Π.χ. μπορούμε να ορίσουμε ότι θέλουμε να δημιουργήσουμε έναν πίνακα με περιεχόμενο όλους τους αριθμούς από το 1 έως το 30 ή όλα τα γράμματα από το «a» έως το «z».

Αρχείο: range.php

```
<?php
 $minas = range(1,30);
 $alfabito = range('a','z');

 var_dump($minas);
 var_dump($alfabito);
?>
```

Αποτέλεσμα:



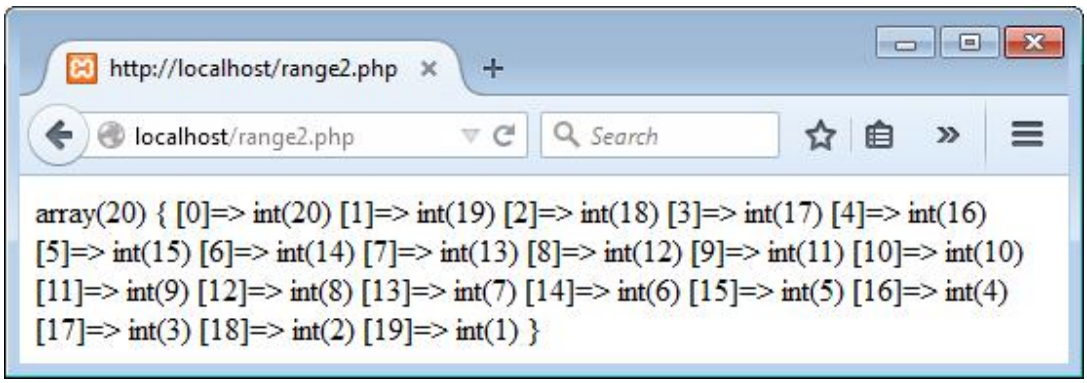
Εικόνα Α.101 Δημιουργία πίνακα με εύρος αριθμών και γραμμάτων

Μπορούμε επίσης να εισάγουμε και ανάποδες σειρές στα στοιχεία μας, δηλ , να ζητήσουμε ανάποδη αρίθμηση από το 10 στο 1 π.χ.

Αρχείο: range2.php

```
<?php
 $arithmisi = range(20,1);
 var_dump($arithmisi);
?>
```

Αποτέλεσμα:



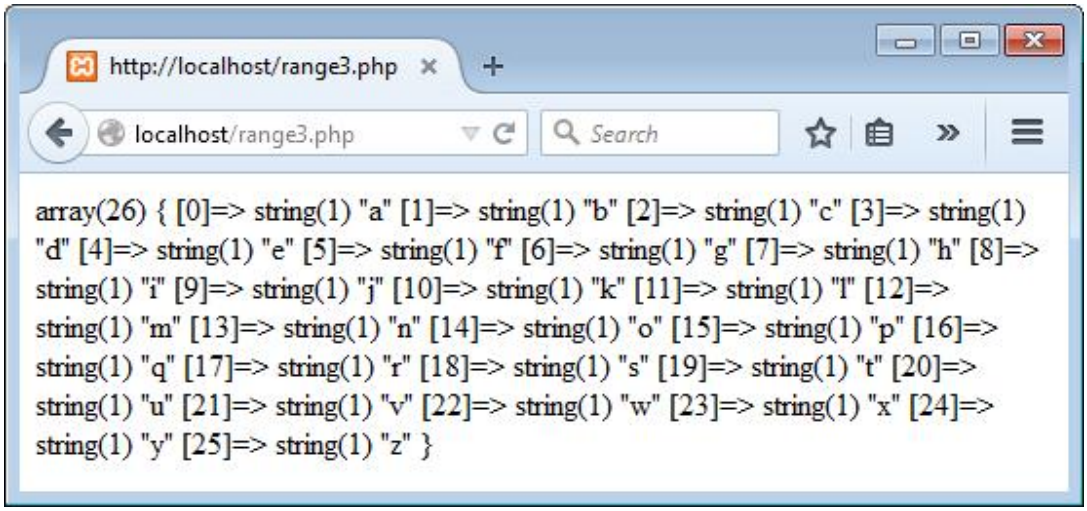
Εικόνα Α.102 Παράδειγμα χρήσης ανάποδου έυρους σε πίνακα

Αν προσπαθήσουμε να χρησιμοποιήσουμε εύρος τιμών σε μεγαλύτερες αλφαριθμητικές εισόδους εκτός από έναν χαρακτήρα, τότε ο μόνος χαρακτήρας που λαμβάνεται υπόψη είναι ο πρώτος:

Αρχείο: range3.php

```
<?php
 $evros = range("altitude","zombie");
 var_dump($evros);
?>
```

Αποτέλεσμα:



Εικόνα Α.103 Χρήση μεγαλύτερων αλφαριθμητικών για είσοδο στο range()

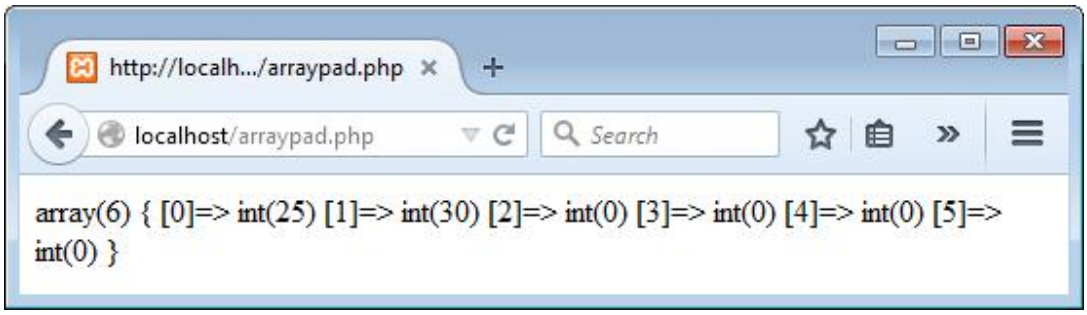
Α.7.1.6 Γέμισμα πίνακα – array_pad()

Για να δημιουργήσουμε έναν πίνακα με συγκεκριμένο αριθμό θέσεων όπου στην αρχή θα διαθέτουν κάποιες από τις θέσεις του τιμές ενώ άλλες δεν θα διαθέτουν τότε χρησιμοποιούμε την εντολή «array_pad()». Η «array_pad()» δέχεται τρία ορίσματα, το πρώτο είναι ο πίνακας που θέλουμε να εμπεριέχεται μέσα στο γέμισμα, το δεύτερο όρισμα είναι το πόσες τιμές θέλουμε να διαθέτει ο νέος πίνακας ως ελάχιστο πλήθος (π.χ. 10), και το τρίτο όρισμα είναι η τιμή όπου θα εισάγεται στις κενές θέσεις (π.χ. 0). Η array_pad() επιστρέφει ως έξοδο έναν νέο πίνακα βασιζόμενο στα εισαγόμενα στοιχεία, αφήνοντας ανεπηρέαστο τον πίνακα που του δώσαμε ως είσοδο.

Αρχείο: arraypad.php

```
<?php
 $pinakas = array(25, 30);
 $gemisma = array_pad($pinakas,6,0);

 var_dump($gemisma);
?>
```

Αποτέλεσμα:



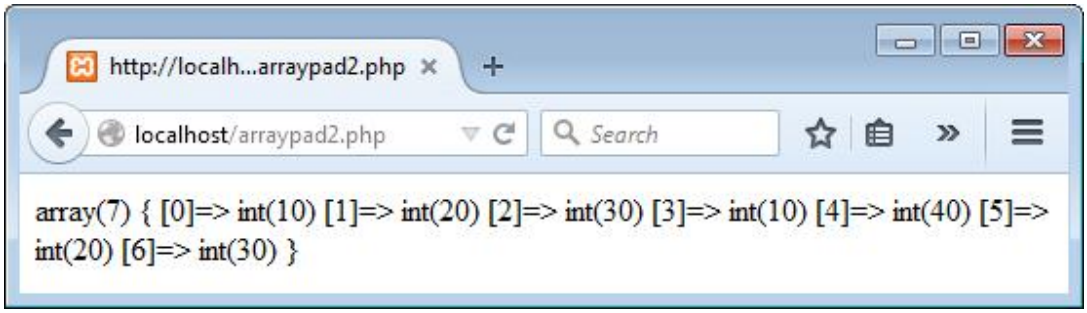
Εικόνα Α.104 Γέμισμα πίνακα με την array_pad()

Αν ο πίνακας που εισάγουμε στην πρώτη θέση είναι μεγαλύτερος από το ελάχιστο μήκος του πίνακα που έχουμε εισάγει στην δεύτερη θέση των παραμέτρων τότε ο πίνακας που θα προκύψει θα είναι αυτός που εισάγαμε:

Αρχείο: arraypad2.php

```
<?php
 $pinakas = array(10,20,30,10,40,20,30);
 $gemisma = array_pad($pinakas,6,0);

 var_dump($gemisma);
?>
```

Αποτέλεσμα:



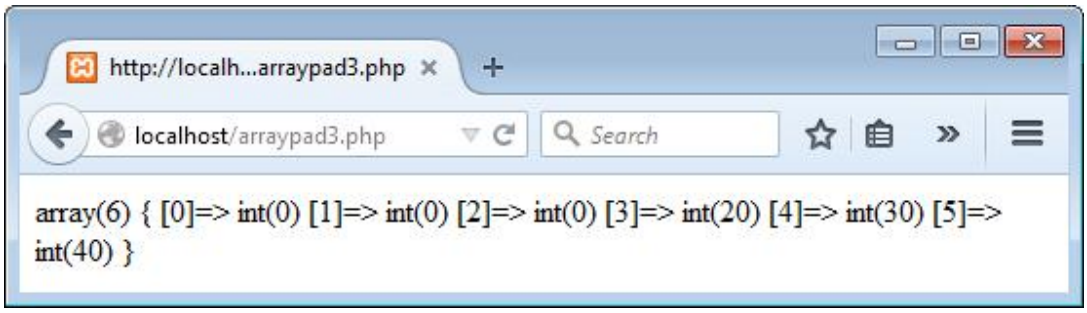
Εικόνα Α.105 Επιστροφή μεγαλύτερου πίνακα σε σχέση με τα ορίσματα στην array_pad()

Στα παραπάνω παραδείγματα οι νέες τιμές προστίθενται στο τέλος του παραγόμενου πίνακα. Αν θέλουμε όμως να εισάγουμε τις νέες τιμές. Στην αρχή του πίνακα και στο τέλος να προστεθούν οι τιμές που ήδη υπάρχουν τότε αρκεί ως δεύτερο όρισμα να εισάγουμε το ακέραιο αριθμό του πλήθος των στοιχείων που θέλουμε να έχει ο πίνακας, προσθέτοντας στην αρχή το αρνητικό σύμβολο «-».

Αρχείο: arraypad3.php

```php
<?php
 $times = array(20,30,40);
 $gemisma = array_pad($times,-6,0);

 var_dump($gemisma);
?>
```

Αποτέλεσμα:



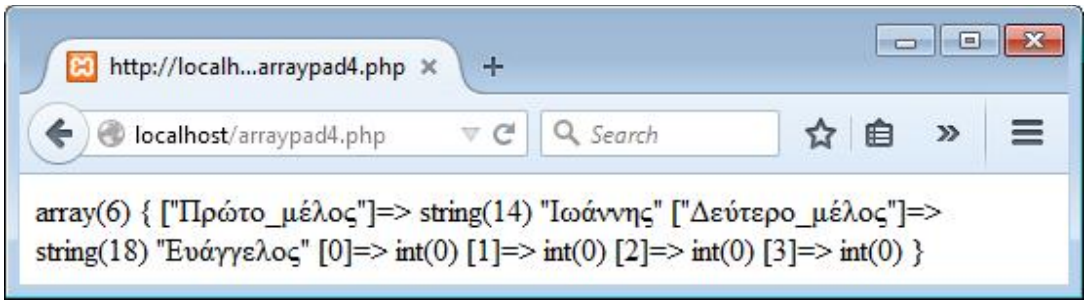
Εικόνα Α.106 Γέμισμα πίνακα προς την αρχή του πίνακα με την array_pad()

Αν χρησιμοποιήσουμε ως είσοδο έναν μη αριθμημένο πίνακα, τότε τα κλειδιά του μη αριθμημένου θα διατηρηθούν και τα νέα στοιχεία που θα προστεθούν θα διαθέτουν αριθμημένα κλειδιά ξεκινώντας από το 0.

Αρχείο: arraypad4.php

```php
<?php
 $omada = array(
  "Πρώτο_μέλος"=>"Ιωάννης",
  "Δεύτερο_μέλος"=>"Ευάγγελος");
 $gemisma = array_pad($omada,6,0);

 var_dump($gemisma);
?>
```

Αποτέλεσμα:



Εικόνα Α.107 Γέμισμα μη αριθμημένου πίνακα με την array_pad()

Το σύνολο των μεταβλητών που διαθέτει ένας πίνακας ονομάζεται μήκος των στοιχείων του πίνακα. Για «ρωτήσετε» τον πίνακα ποιο είναι το μήκος του, αρκεί να χρησιμοποιήσετε την μέθοδο «count()». Π,χ. count($omada);.

Για να αποκτήσουμε πρόσβαση σε μια μεταβλητή αρκεί μόνο να την καλέσουμε με το όνομα του πίνακα και τον δείκτη στον οποίο βρίσκεται π.χ. Γράφοντας «$omada[1]» θα χρησιμοποιηθεί το «Ευάγγελος».

Α.7.1.7 Πίνακες πολλαπλών διαστάσεων

Οι πίνακες όπως και στην καθημερινή μας ζωή δεν είναι μόνο μονοδιάστατοι. Το εβδομαδιαίο πρόγραμμα μαθημάτων, η λίστα βαθμολογιών, ένας πίνακας τιμών προϊόντων, είναι μερικά παραδείγματα πινάκων μεγαλύτερων διαστάσεων του ενός.

Ένας πίνακας δυο διαστάσεων απεικονίζεται με τον δημοφιλή τρόπο:

Όνομα	Επώνυμο	ΑΜ	Βαθμός
Ιωάννα	Δημητρίου	1256	5
Δημήτρης	Ιωαννίδης	1345	8
Μάριος	Παπαδόπουλος	1212	6

Πίνακας Α.21 Υποτιθέμενος πίνακας βαθμολογιών

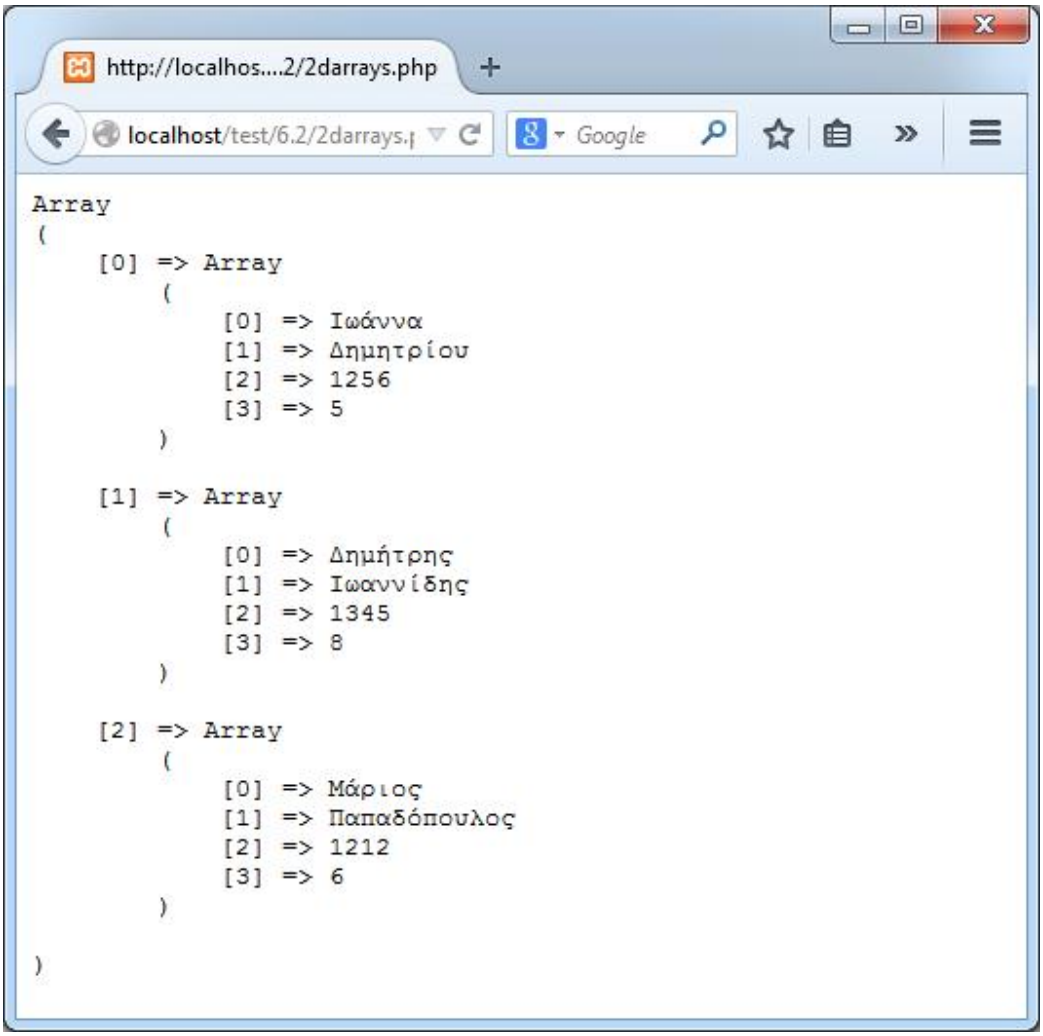
Κάθε γραμμή του πίνακα αποτελεί και μια εγγραφή του. Το πακέτο «Ιωάννα, Δημητρίου, 1256, 5» είναι μια ομάδα στοιχείων που αφορούν μια φοιτήτρια με όνομα «Ιωάννα», επώνυμο «Δημητρίου», Αριθμό μητρώου «1256» και βαθμό «5». Είναι δηλαδή ένας μονοδιάστατος πίνακας με 4 θέσεις όπου η θέση «0» αναπαριστά το «Όνομα» η θέση «1» το «Επώνυμο» κ.ο.κ.

Ένας πίνακας δύο διαστάσεων είναι ένας πίνακας όπου σε κάθε θέση του υπάρχει και από ένας άλλος πίνακας. Οι «γραμμές» που αναφέραμε παραπάνω είναι από μόνες τους ξεχωριστοί πίνακες, όπου συνθέτοντας τους όλους μαζί δημιουργείται ο πίνακας που συνολικά μας δίνει την εικόνα των δυο διαστάσεων. Ας δούμε ένα παράδειγμα χρήσης πινάκα με δυο διαστάσεις:

Αρχείο: 2darrays.php

```php
<?php
 $p1 = array("Ιωάννα","Δημητρίου","1256","5");
 $p2 = array("Δημήτρης","Ιωαννίδης","1345","8");
 $p3 = array("Μάριος","Παπαδόπουλος","1212","6");

 $pinakas2d = array($p1,$p2,$p3); //Οι πίνακες $p1,$p2,$p3 γίνονται γραμμές στον πίνακα $pinakas2d

 echo "<pre>";
 print_r($pinakas2d);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Πίνακας Α.22 Εκτύπωση πίνακα δυο διαστάσεων

A.7.2 Προσπέλαση στοιχείων

Ποιος είναι ο λόγος να διαθέτουμε έναν πίνακα χωρίς να μπορούμε να ανατρέξουμε σε αυτόν τα στοιχεία που διαθέτει; Παρακάτω θα δούμε πως αυτό επιτυγχάνεται καθώς επίσης θα δούμε διάφορες μεθόδους άντλησης πληροφοριών για τους πίνακες που διαθέτουμε.

A.7.2.1 Μέγεθος ενός πίνακα

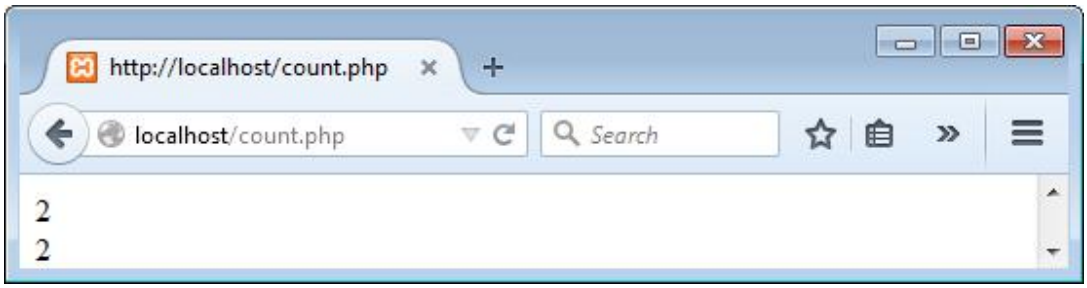
Το μέγεθος ή μήκος ενός πίνακα είναι το συνολικό πλήθος των στοιχείων που διαθέτει. Δυο εντολές μας δίνουν το μήκος ενός πίνακα.

Η «count()» όπου παίρνει ως όρισμα έναν πίνακα και επιστρέφει το συνολικό μήκος του.

Αρχείο: count.php

```
<?php
 //Αριθμημένος πίνακας
 $imeres = array("Δευτέρα","Τρίτη");
 $mikos = count($imeres);
 echo $mikos."<br/>";

 //Μη αριθμημένος πίνακας
 $omada = array(
  "Τεχνικός"=>"Παναγιώτης",
  "Υποστηριχτής"=>"Αριστομένης");
 $mikos = count($omada);
 echo $mikos;
?>
```

Αποτέλεσμα:



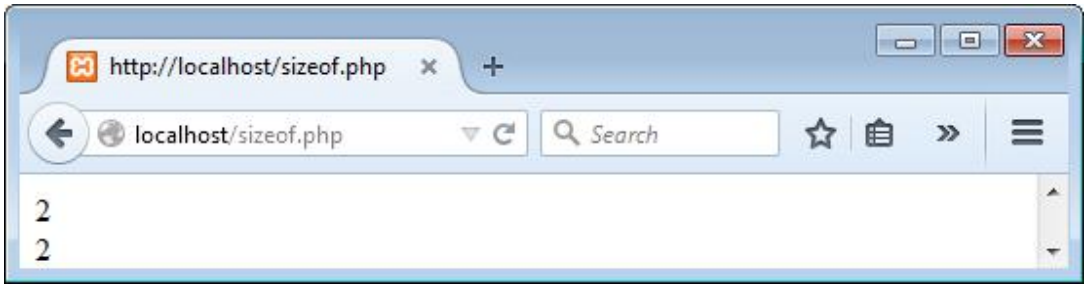
Εικόνα Α.108 Το μήκος πινάκων από την count()

Η «sizeof()» όπου επίσης παίρνει ως είσοδο έναν πίνακα και επιστρέφει το μήκος του.

Αρχείο: sizeof.php

```php
<?php
 //Αριθμημένος πίνακας
 $imeres = array("Δευτέρα","Τρίτη");
 $mikos = sizeof($imeres);
 echo $mikos."<br/>";

 //Μη αριθμημένος πίνακας
 $omada = array(
  "Τεχνικός"=>"Παναγιώτης",
  "Υποστηριχτής"=>"Αριστομένης");
 $mikos = sizeof($omada);
 echo $mikos;
?>
```

Αποτέλεσμα:



Εικόνα Α.109 Μήκος πινάκων από την sizeof()

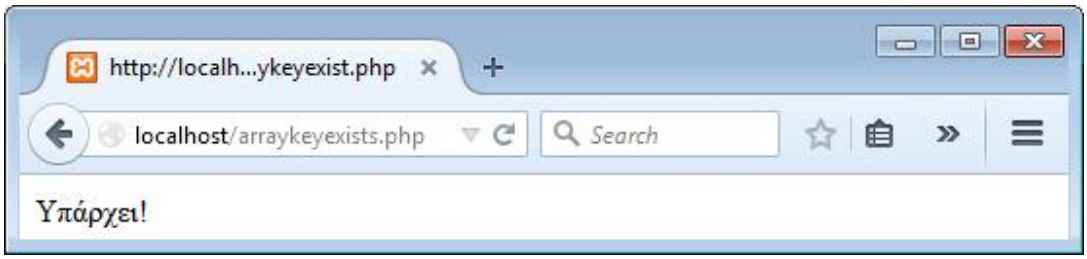
Α.7.2.2 Έλεγχος ύπαρξης ενός κλειδιού

Για να ελέγξουμε αν υπάρχει ένα κλειδί μέσα σε ένα πίνακα τότε θα χρησιμοποιήσουμε την εντολή «array_key_exists()». Η εντολή αυτή παίρνει ως πρώτο όρισμα ένα κλειδί και ως δεύτερο έναν πίνακα για να ελέγξει αν υπάρχει μέσα σε αυτόν το συγκεκριμένο κλειδί. Η εντολή αυτή επιστρέφει true αν υπάρχει το κλειδί μέσα στον πίνακα και false αν δεν υπάρχει. Αν χρησιμοποιούσαμε απλά την εντολή «if($pinakas['kleidi'])» τότε θα ήταν λάθος διότι δεν γίνεται έλεγχος του πίνακα μέσω της if.

Αρχείο: arraykeyexists.php

```php
<?php
 $car_price['suzuki'] = 3000;

 if(array_key_exists('suzuki',$car_price)){
  echo "Υπάρχει!";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.110 Εύρεση κλειδιού σε πίνακα με την array_key_exists()

Α.7.2.3 Προσπέλαση με for και foreach

Με βάση την δυνατότητα δεικτών στους πίνακες είναι πιο εύκολη η πρόσβαση στα στοιχεία τους. Έτσι μέσω μιας δομής επανάληψης είναι δυνατό να προσπελάσουμε όλα τα στοιχεία ενός πίνακα και να εκτελέσουμε συγκεκριμένες λειτουργίες για κάθε ένα από αυτά.

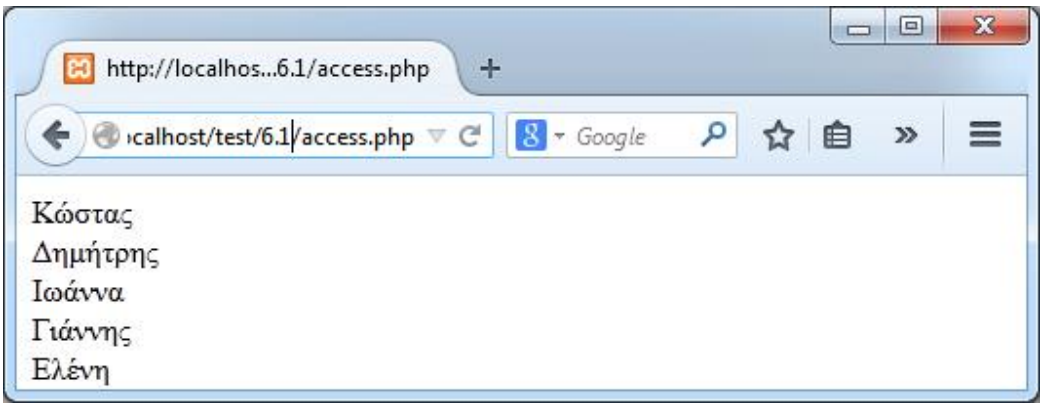
Έτσι ξέρουμε πως το πρώτο στοιχείο του πίνακα θα είναι πάντα στην θέση με δείκτη «0», ενώ το τελευταίο στην θέση «count($pinakas)-1».

Με βάση αυτή την λογική ας δούμε ένα παράδειγμα προσπέλασης αριθμημένου πίνακα μιας διάστασης.

Αρχείο: access.php

```
<?php
 $students = array("Κώστας","Δημήτρης","Ιωάννα","Γιάννης","Ελένη");
 for($i=0;$i<count($students);$i++){
    echo $students[$i]."<br/>";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.111 Προσπέλαση αριθμημένου πίνακα με την for

Το αποτέλεσμα επιβεβαιώνει την σωστή χρήση των δεικτών στους πίνακες. Ο πίνακας με μήκος «5» θέσεων, ξεκινά το μέτρημα των θέσεων του από την θέση 0 και τελειώνει στην θέση 4. Γι αυτό και χρησιμοποιούμε τον τελεστή σύγκρισης «<» στην συνθήκη επανάληψης. Η μεταβλητή «$i» θα αλλάξει διαδοχικά τιμές από το 0 που την ορίσαμε έως το 4.

Σχήμα Α.2 Επεξήγηση θέσεων ενός αριθμημένου πίνακα

Το παραπάνω παράδειγμα χρησιμοποιεί την εντολή for για την προσπέλαση του πίνακα, υπάρχει και η δυνατότητα προσπέλασης μέσω της foreach η οποία έχει εξηγηθεί σε προηγούμενη ενότητα για ευκολία εύρεσης.

Α.7.2.4 Προσπέλαση με την each()

Εκτός από την προσπέλαση με κάποιο είδος βρόγχου είναι δυνατό να προσπελαύνουμε κάποια από τα στοιχεία ενός πίνακα ή και ενδεχομένως όλα, όποτε εμείς θέλουμε χωρίς να βασιζόμαστε σε βρόγχους. Αυτό επιτυγχάνεται με την ενσωματωμένη δυνατότητα της php να υποστηρίζει δείκτες σε κάθε πίνακα της.

Κάθε πίνακας στην php διαθέτει έναν τρόπο να γνωρίζει πιο στοιχείο κάθε φορά χρησιμοποιούμε και κάνουμε πράξεις με αυτό, συγκεκριμένα διαθέτει έναν μοναδικό δείκτη, όπου μας δείχνει το στοιχείο στο οποίο βρισκόμαστε μέσα στον πίνακα. Αυτός ο δείκτης διαθέτει κάποιες διαφορετικές εντολές από όπου μπορούμε να περιηγούμαστε στα στοιχεία του πίνακα με μεγάλη ευκολία.

- current()
  Επιστρέφει το στοιχείο στο οποίο ο δείκτης μας δείχνει εκείνη την στιγμή.
- reset()
  Τοποθετεί τον δείκτη στο πρώτο στοιχείο του πίνακα.
- next()
  Τοποθετεί τον δείκτη στο επόμενο στοιχείο του πίνακα από ότι είμαστε.

- prev()

  Τοποθετεί τον δείκτη στο προηγούμενο στοιχείο του πίνακα από ότι είμαστε.
- end()

  Τοποθετεί τον δείκτη στο τέλος του πίνακα στον οποίο βρισκόμαστε.
- each()

  Επιστρέφει το κλειδί και την τιμή ενός στοιχείου σε έναν πίνακα και ύστερα μετακινεί τον δείκτη στο επόμενο στοιχείο.
- key()

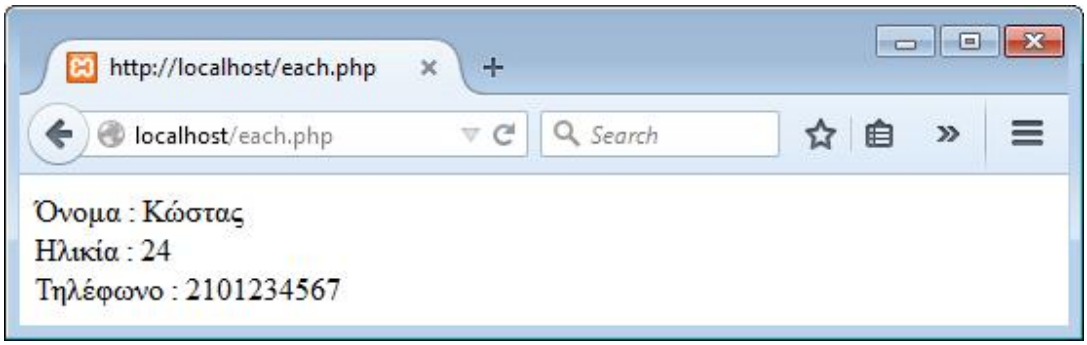
  Επιστρέφει το κλειδί το στοιχείο που ο δείκτης βρίσκεται εκείνη την στιγμή.

Η εντολή each() μπορεί εκτός από αυτόνομα, να χρησιμοποιηθεί και μέσα σε βρόγχους έτσι ώστε να προσπελάσουμε κάθε στοιχείο ενός πίνακα.

Αρχείο: each.php

```
<?php
 $epafi =
array("Όνομα"=>"Κώστας","Ηλικία"=>24,"Τηλέφωνο"=>2101234567);
 reset($epafi);

 while(list($kleidi, $timi)=each($epafi)){
  echo "{$kleidi} : {$timi} </br>";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.112 Προσπέλαση πίνακα με την each()

Αυτή η προσέγγιση δεν δημιουργεί αντίγραφο του πίνακα όπως στην εντολή foreach, αλλά χρησιμοποιεί μόνο δυο μεταβλητές ανάθεσης όπως τις ορίζουμε και έτσι είναι κατάλληλη για προσπέλαση τεράστιων μεγεθών πινάκων.

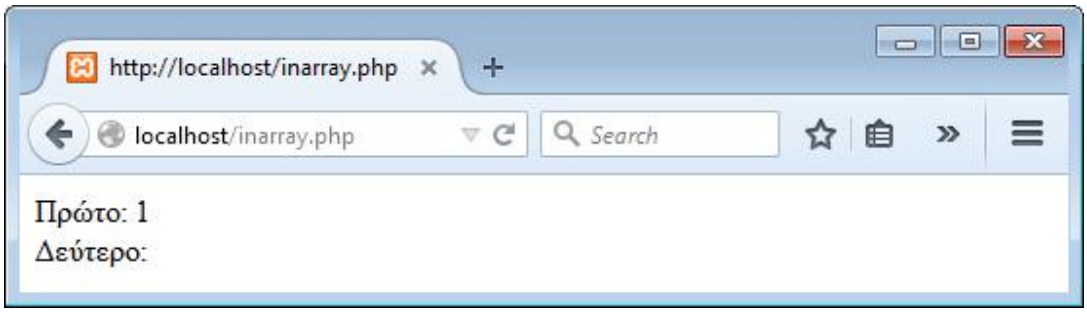
A.7.2.5 Αναζήτηση στοιχείων

Για να αναζητήσουμε ένα στοιχείο μέσα σε έναν πίνακα θα πρέπει να χρησιμοποιήσουμε την εντολή «in_array()». Η συγκεκριμένη εντολή δέχεται δύο ή τρία ορίσματα, ανάλογα την περίπτωση ταιριάσματος που θέλουμε να χρησιμοποιήσουμε, και επιστρέφει μια τιμή true ή false. Η πρώτη παράμετρος που δέχεται ως είσοδος είναι το στοιχείο που ψάχνουμε, η δεύτερη παράμετρος είναι ο πίνακας από όπου θέλουμε να βρούμε το στοιχείο, και η τρίτη παράμετρος είναι μια τιμή true ή false για το αν θέλουμε εκτός από το ταίριασμα του περιεχομένου να γίνει και ταίριασμα του τύπου δεδομένων. Η τρίτη παράμετρος μπορεί να απουσιάζει από την χρήση της εντολής κι έτσι αυτόματα να θεωρηθεί false.

Αρχείο: inarray.php

```
<?php
 $trofima = array(
    'Ντομάτα' => 6.1,
    'Φράουλα'   => 3.2,
    'Ζάχαρη' => '1.0');

 echo "Πρώτο: ".in_array(1.0,$trofima);
 echo "<br/>";
 echo "Δεύτερο:".in_array(1.0,$trofima,true);
?>
```

Αποτέλεσμα:



Εικόνα Α.113 Αναζήτηση στοιχείου μέσα σε πίνακα με την in_array()

Βλέπουμε πως στην πρώτη περίπτωση που δεν ψάχνουμε και για ταίριασμα τύπου δεδομένων το αποτέλεσμα είναι true, ενώ στην δεύτερη περίπτωση το αποτέλεσμα είναι τίποτα, δηλαδή false.

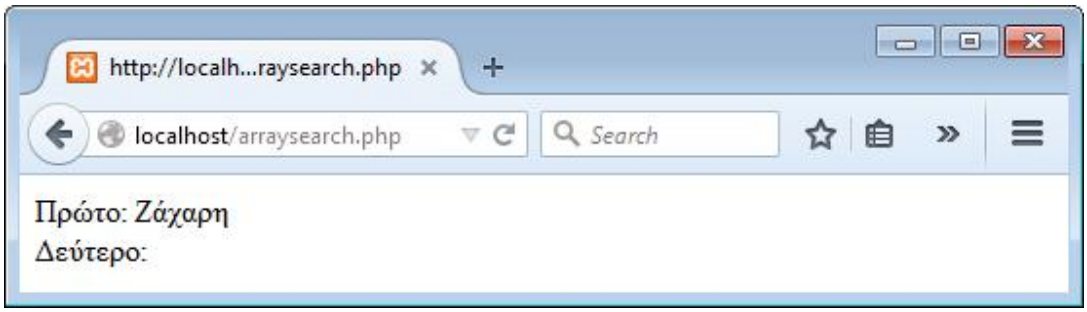
array_search()

Εκτός από την αναζήτηση για το αν υπάρχει ένα στοιχείο μπορούμε να αναζητήσουμε και την ακριβή θέση του στοιχείου με την εντολή array_search() η οποία δέχεται τα ίδια ορίσματα με την in_array() και επιστρέφει το κλειδί της θέσεως στην οποία βρέθηκε το στοιχείο το οποίο ψάχνουμε.

Αρχείο: arraysearch.php

```
<?php
 $trofima = array(
   'Ντομάτα' => 6.1,
   'Φράουλα'   => 3.2,
   'Ζάχαρη' => '1.0');

 echo "Πρώτο: ".array_search(1.0,$trofima);
 echo "<br/>";
 echo "Δεύτερο:".array_search(1.0,$trofima,true);
?>
```

Αποτέλεσμα:



Εικόνα Α.114 Επιστροφή θέσης εύρεσης στοιχείου με την array_search()

Α.7.2.6 Προσπέλαση πινάκων με δυο διαστάσεις

Οι στήλες αναφέρονται στις θέσεις των εσωτερικών πινάκων, ενώ οι γραμμές στον πίνακα-σύνολο που περιέχει τους εσωτερικούς πίνακες. Ας θεωρήσουμε πως οι γραμμές είναι η μεταβλητή «$i» και οι στήλες η μεταβλητή «$j». Για να χρησιμοποιήσουμε την τιμή «1345» από τον παρακάτω πίνακα θα πρέπει να πούμε σε ποια γραμμή και σε ποια στήλη βρίσκεται. Για λόγους ευκολίας θα χρησιμοποιούμε ΠΡΩΤΑ την γραμμή και μετά την στήλη… άρα βρίσκεται στην θέση «1,2», όπου $i=1 και $j=2.

Στήλη	0	1	2	3
Γραμμή	Όνομα	Επώνυμο	ΑΜ	Βαθμός
0	Ιωάννα	Δημητρίου	1256	5
1	Δημήτρης	Ιωαννίδης	1345	8
2	Μάριος	Παπαδόπουλος	1212	6

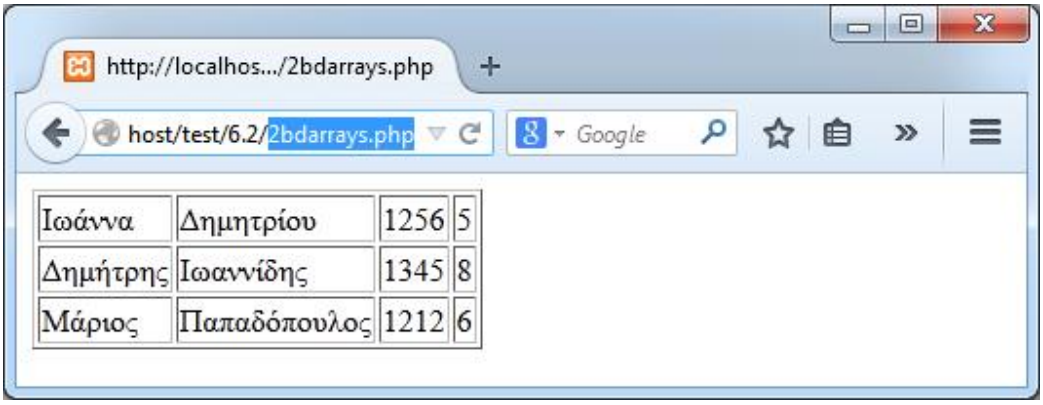
Εικόνα Α.115 Αναπαράσταση γραμμών και στηλών ενός πίνακα

Ας δούμε ένα παράδειγμα προσπέλασης πίνακα με δυο διαστάσεις.

Αρχείο: 2bdarrays.php

```
<?php
 $students = array(
  array("Ιωάννα","Δημητρίου","1256","5"),
  array("Δημήτρης","Ιωαννίδης","1345","8"),
  array("Μάριος","Παπαδόπουλος","1212","6")
 );

 echo "<table border='1'>";
 for($i=0;$i<count($students);$i++){
  echo "<tr>";
  for($j=0;$j<count($students[$i]);$j++){
   echo "<td>".$students[$i][$j]."</td>";
  }
  echo"</tr>";
 }

 echo "</table>";
?>
```

Αποτέλεσμα:



Εικόνα Α.116 Εκτύπωση δισδιάστατου πίνακα μέσω της for

Όπως βλέπουμε στον κώδικα ο πίνακας ονομάζεται «$students» ενώ για να ζητήσουμε μια θέση μέσα σε αυτόν, ανάλογα τις διαστάσεις του χρησιμοποιούμε δείκτες μέσα σε άγκιστρα π.χ. «$students[1][2]» το οποίο θα μας δώσει «1345».

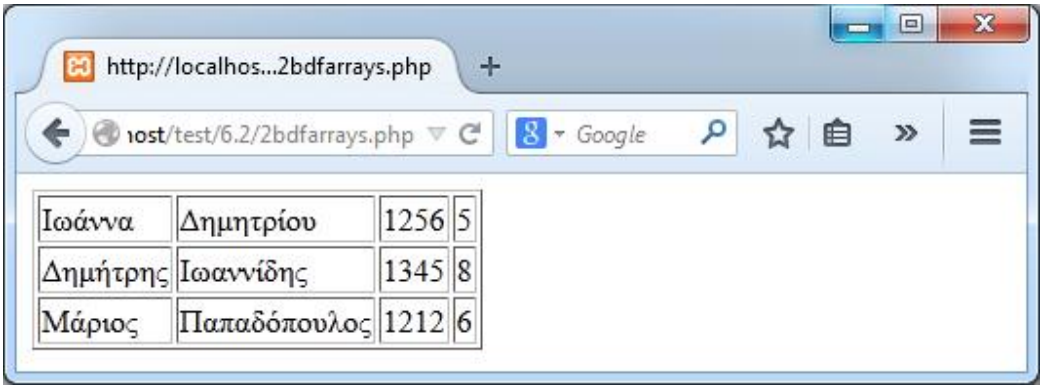
Στο δεύτερο for όταν θέλουμε να ζητήσουμε το μέγεθος του υπό-πίνακα ζητάμε απλά το μήκος της θέσης του πίνακα δηλαδή «$students[0]» είναι ίσο με το «array("Ιωάννα","Δημητρίου","1256","5")» και συνεπώς επιστρέφεται ο αριθμός 4.

Ας δούμε το ίδιο παράδειγμα προσπέλασης χρησιμοποιώντας «foreach» αντί για «for».

Αρχείο: 2bdfarrays.php

```
<?php
 $students = array(
  array("Ιωάννα","Δημητρίου","1256","5"),
  array("Δημήτρης","Ιωαννίδης","1345","8"),
  array("Μάριος","Παπαδόπουλος","1212","6")
 );

 echo "<table border='1'>";
 foreach($students as $value){
  echo "<tr>";
  foreach($value as $value2){
   echo "<td>".$value2."</td>";
  }
  echo "</tr>";
 }

 echo "</table>";
?>
```

Αποτέλεσμα:



Εικόνα Α.117 Εκτύπωση δισδιάστατου πίνακα μέσω της foreach

Η μεταβλητή «$value» γίνεται ο πίνακας που υπάρχει σε κάθε γραμμή του πίνακα «$students». Επομένως μετατρέπεται ως το όρισμα της δεύτερης «foreach» που από αυτήν θα πάρουμε τις τιμές.

Για λόγους καλύτερης κατανόησης κατά την πρώτη επανάληψη της πρώτης «foreach» το «$value» θα είναι ίσο με «array("Ιωάννα" , "Δημητρίου" , "1256" , "5")» επομένως η πρώτη επανάληψη της δεύτερης «foreach», η μεταβλητή «$value2» θα πάρει την τιμή «"Ιωάννα"».

A.7.3 Επεξεργασία ενός πίνακα

Η επεξεργασία ενός πίνακα είναι ένα αναπόσπαστο κομμάτι του προγραμματισμού καθώς με αυτόν τον τρόπο μπορούμε να επιλύσουμε την σύντομη επεξεργασία τεράστιων όγκων δεδομένων. Στις επόμενες υπό-ενότητες θα δούμε πως μπορούμε να επεξεργαστούμε έναν πίνακα αλλάζοντας το περιεχόμενο ή την δομή του.

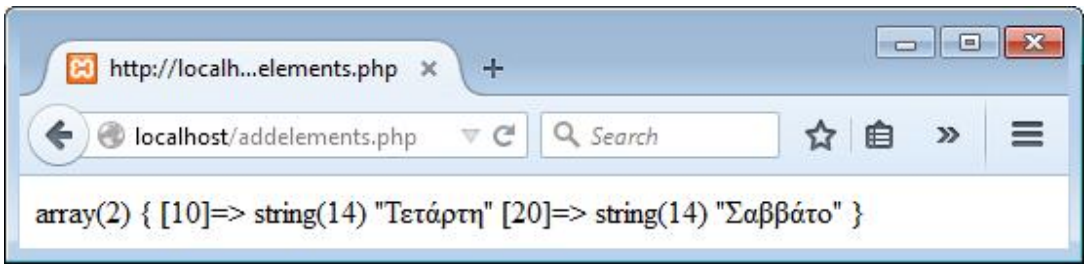
A.7.3.1 Προσθήκη νέων στοιχείων

Υπάρχουν δυο τρόποι για να εισάγουμε ένα νέο στοιχείο σε έναν πίνακα. Ο πρώτος είναι απλά να καλέσουμε την θέση με το κλειδί στην οποία θέλουμε να προσθέσουμε το στοιχείο και να της αποδώσουμε μια τιμή.

Αρχείο: addelements.php

```
<?php
 $pinakas[10] = "Τετάρτη";
 $pinakas[20] = "Σαββάτο";
 var_dump($pinakas);
?>
```

Αποτέλεσμα:



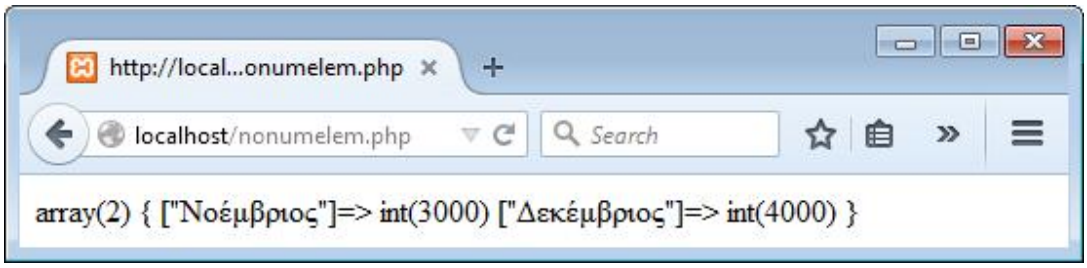
Εικόνα A.118 Αποτέλεσμα προσθήκης νέων στοιχείων σε πίνακα

Ή εναλλακτικά με την χρήση μη αριθμημένων πινάκων.

Αρχείο: nonnumelem.php

```
<?php
 $pinakas["Νοέμβριος"] = 3000;
 $pinakas["Δεκέμβριος"] = 4000;
 var_dump($pinakas);
?>
```

Αποτέλεσμα:



Εικόνα Α.119 Εισαγωγή στοιχείων σε μη αριθμημένο πίνακα

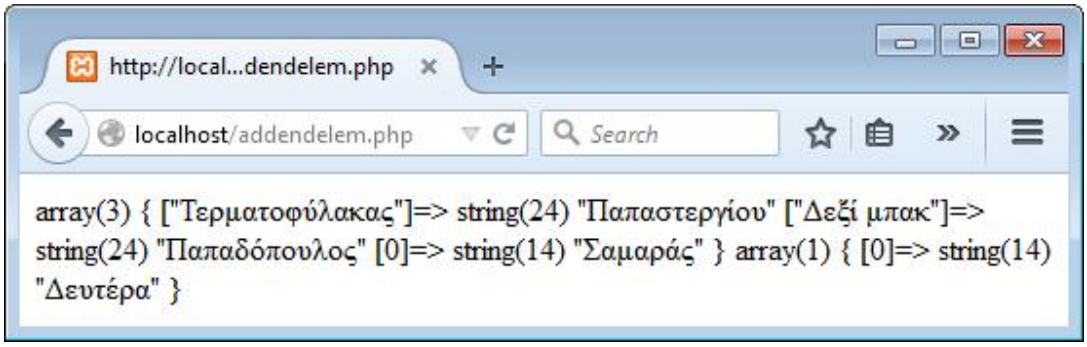
Προσέξτε στα παραπάνω παραδείγματα πως για νέες τιμές χρησιμοποιούμε διαφορετικά κλειδιά. Αν χρησιμοποιούσαμε ένα κλειδί που ήδη υπάρχει, τότε θα αντικαθιστούνταν ή ήδη υπάρχουσα τιμή του με την νέα που του αναθέτουμε.

Ο δεύτερος τρόπος εισαγωγής, γίνεται στο τέλος ενός πίνακά. Για να είναι δυνατή όμως αυτή η εισαγωγή θα πρέπει πρώτα ο πίνακας να υπάρχει από πριν. Σε περίπτωση που δεν υπάρχει δημιουργείται εκείνη την στιγμή αριθμημένος πίνακας με πρώτο κλειδί το «0».

Αρχείο: addendelem.php

```
<?php
 $paiktes = array(
  "Τερματοφύλακας"=>"Παπαστεργίου",
  "Δεξί μπακ"=>"Παπαδόπουλος");
 $paiktes[] = "Σαμαράς"; //εισαγωγή στο τέλους ήδη υπάρχον

 var_dump($paiktes);

 $imeres[]="Δευτέρα"; //δημιουργία νέου πίνακα
 var_dump($imeres);
?>
```

Αποτέλεσμα:



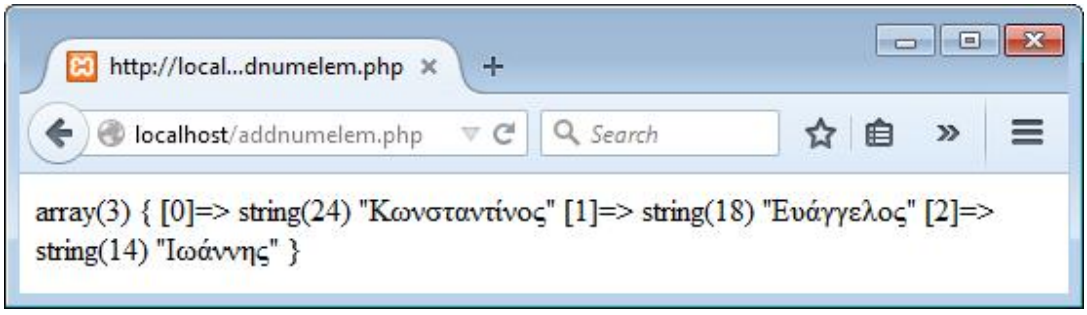
Εικόνα Α.120 Εισαγωγή στοιχείων στο τέλος ενός πίνακα

Παρά που ο πίνακας είναι μη αριθμημένος, το κλειδί που χρησιμοποιείται για τις νεοεισερχόμενες τιμές είναι αριθμός και συγκεκριμένα ο πρώτος διαθέσιμος δηλαδή το «0». Αν επιχειρήσουμε να προσθέσουμε μια τιμή μέσα σε έναν αριθμημένο πίνακα, τότε η νέα τιμή θα προστεθεί στην επόμενη κατά σειρά θέση όπου το κλειδί θα το επιτρέπει.

Αρχείο: addnumelem.php

```
<?php
 $omada = array("Κωνσταντίνος","Ευάγγελος");
 $omada[] = "Ιωάννης";

 var_dump($omada);
?>
```

Αποτέλεσμα:



Εικόνα Α.121 Εισαγωγή στοιχείου σε αριθμημένο πίνακα

Α.7.3.2 Διαγραφή και εισαγωγή στοιχείων με την array_splice()

Η εντολή array_splice() μας δίνει την δυνατότητα τόσο της διαγραφής στοιχείων όσο και της εισαγωγής σε έναν πίνακα καθώς επίσης και της δημιουργίας νέου πίνακα με τα διαγραμμένα στοιχεία. Παίρνει 4 παραμέτρους (1 προαιρετική) :

1. Έναν πίνακα στον οποίο επιδρούμε,
2. Έναν ακέραιο αριθμό ως σημείο εκκίνησης μέσα στον πίνακα, το οποίο συμβολίζει το σημείο από όπου θα ξεκινήσουμε να επηρεάζουμε τα στοιχεία. Αν πρόκειται για διαγραφή τότε είναι το σημείο στο οποίο θα ξεκινήσουμε να διαγράφουμε στοιχεία, αν πρόκειται για εισαγωγή τότε είναι το σημείο στο οποίο θα ξεκινήσουμε να εισάγουμε στοιχεία,
3. Έναν ακέραιο αριθμό, ο οποίος συμβολίζει το μήκος από το σημείο εκκίνησης και ύστερα στο οποίο θα επιδράσει η αλλαγή μας,
4. Μόνο σε περίπτωση εισαγωγής, ένας πίνακας με τα στοιχεία που επιθυμούμε να εισάγουμε.

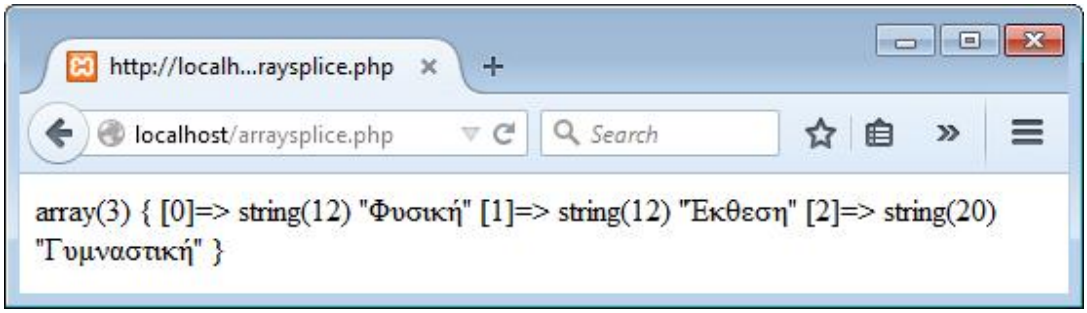
Σε περίπτωση που κάνουμε διαγραφή και όχι εισαγωγή, λείπει η τέταρτη παράμετρος εισόδου, και η εντολή επιστρέφει τα διαγραμμένα στοιχεία σε μορφή νέου πίνακα: Αν δεν συμπεριλάβουμε το μήκος στην διαγραφή τότε η διαγραφή ξεκινά από το τέλος του πίνακα.

Αν απλά μας ενδιαφέρει να διαγράψουμε και δεν μας νοιάζει να διατηρήσουμε τις διαγραμμένες τιμές τότε απλά δεν αντιστοιχίζουμε την εκτέλεση της εντολής σε κάποια μεταβλητή.

Αρχείο: arraysplice.php

```
<?php
 $mathimata =
array("Φυσική","Χημεία","Μαθηματικά","Ιστορία","Έκθεση");
 $diegramenos = array_splice($mathimata,1,3);

 var_dump($mathimata);
?>
```

Αποτέλεσμα:



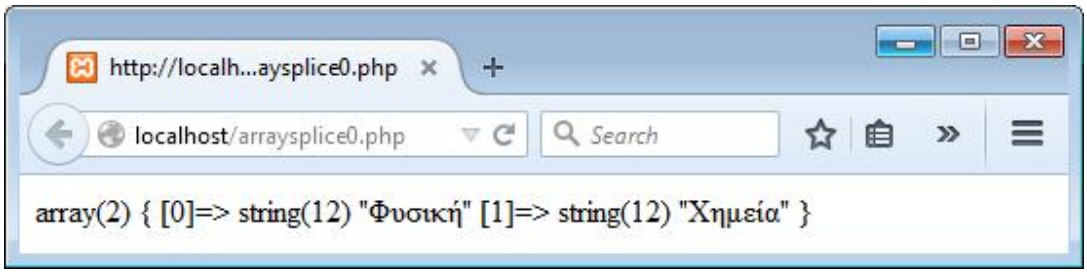
Εικόνα Α.122 Αφαίρεση τμημάτων πίνακα με την array_splice()

Μπορούμε αν θέλουμε να δηλώσουμε μόνο έναν αριθμό ως όρισμα μαζί με τον πίνακα στην εκτέλεση της array_splice(). Αυτό θα διαγράψει όσα στοιχεία βρίσκονται από το σημείο που του υποδείξαμε ως το τέλος του πίνακα, έτσι αν του δώσουμε 2 τα στοιχεία «Μαθηματικά», «Ιστορία», «Έκθεση» θα διαγραφούν από τον πίνακα.

Αρχείο: arraysplice0.php

```
<?php
 $mathimata =
array("Φυσική","Χημεία","Μαθηματικά","Ιστορία","Έκθεση");
 $diegramenos = array_splice($mathimata,2);

 var_dump($mathimata);
?>
```

Αποτέλεσμα:



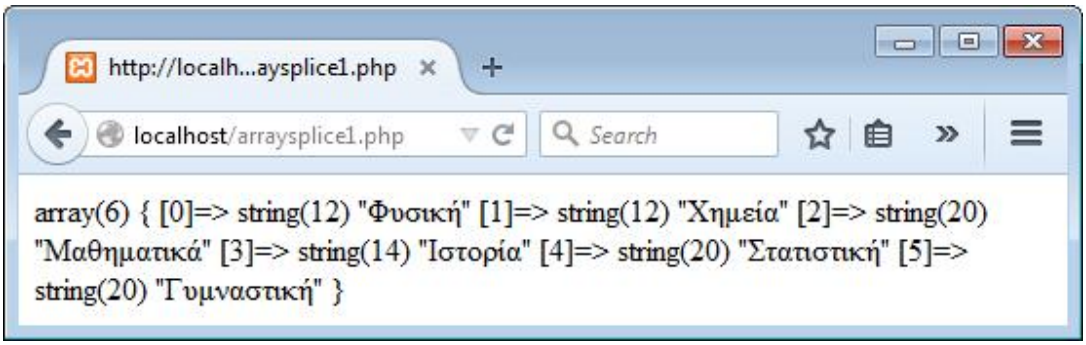
Εικόνα Α.123 Διαγραφή στοιχείων με προσαρμοσμένο μήκος

Για να εισάγουμε στοιχεία απλά χρησιμοποιούμε την εντολή με 4 παραμέτρους όπως εξηγήθηκαν παραπάνω.

Αρχείο: arraysplice1.php

```php
<?php
 $mathimata =
array("Φυσική","Χημεία","Μαθηματικά","Ιστορία","Έκθεση" );
 $nea=array("Στατιστική","Γυμναστική");
 $diegramenos = array_splice($mathimata,4,2,$nea);

 var_dump($mathimata);
?>
```

Αποτέλεσμα:



Εικόνα Α.124 Εισαγωγή νέων στοιχείων σε έναν πίνακα με την array_splice()

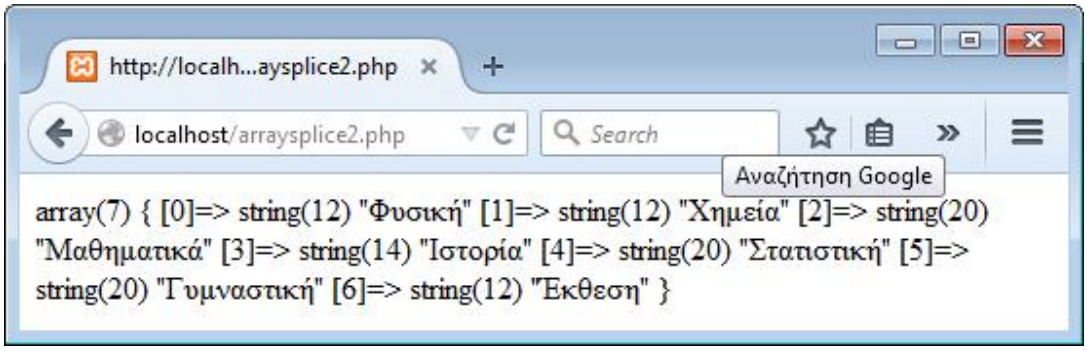
Αν χρησιμοποιήσουμε κατά την εισαγωγή θέση σε συνδυασμό με μήκος, όπου ήδη υπάρχουν στοιχεία, τότε τα σημεία στα οποία υπάρχει «σύγκρουση» αντικαθίστανται από τα νέα εισερχόμενα στοιχεία με την σειρά που είχαν πριν εισαχθούν.

Αν θέλουμε κατά την διάρκεια της εισαγωγής να μην διαγραφούν στοιχεία, και τα υπάρχουσα που έρχονται σε «σύγκρουση» να μετακινηθούν προς τα δεξιά. Τότε εισάγουμε σημείο εκκίνησης στις παραμέτρους αλλά στο μήκος εισάγουμε τιμή «0». Με αυτή την διαδικασία οι θέσεις οι οποίες δεν έχουν κάποιες τιμές διαγράφονται ενώ αυτές που ήδη έχουν τιμές πριν την εισαγωγή διατηρούνται.

Αρχείο: arraysplice2.php

```php
<?php
 $mathimata =
array("Φυσική","Χημεία","Μαθηματικά","Ιστορία","Έκθεση" );
 $nea=array("Στατιστική","Γυμναστική");
 $diegramenos = array_splice($mathimata,4,0,$nea);

 var_dump($mathimata);
?>
```

Αποτέλεσμα:



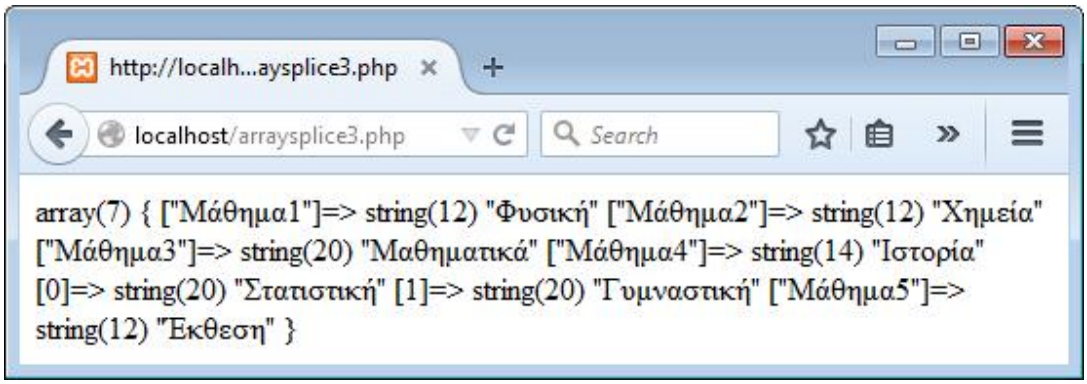
Εικόνα Α.125 Εισαγωγή νέων στοιχείων σε πίνακα διατηρώντας και όλα τα παλαιά

Τέλος να σημειώσουμε πως η array_splice() λειτουργεί και σε μη αριθμημένους πίνακες. Όπου δίνοντας τιμές για σημείο εκκίνησης και μήκος, τα αλφαριθμητικά κλειδιά μετατρέπονται σε αριθμητικά μόνο για την εκτέλεση της εντολής επιδρόντας όμως στην ήδη υπάρχουσα σειρά των στοιχείων που έχει ο πίνακας.

Αρχείο: arraysplice3.php

```
<?php
 $mathimata = array(
  "Μάθημα1" => "Φυσική",
  "Μάθημα2" => "Χημεία",
  "Μάθημα3" => "Μαθηματικά",
  "Μάθημα4" => "Ιστορία",
  "Μάθημα5" => "Έκθεση" );
 $nea=array("Στατιστική","Γυμναστική");
 $diegramenos = array_splice($mathimata,4,0,$nea);

 var_dump($mathimata);
?>
```

Αποτέλεσμα:



Εικόνα Α.126 Εισαγωγή νέων στοιχείων σε μη αριθμημένο πίνακα

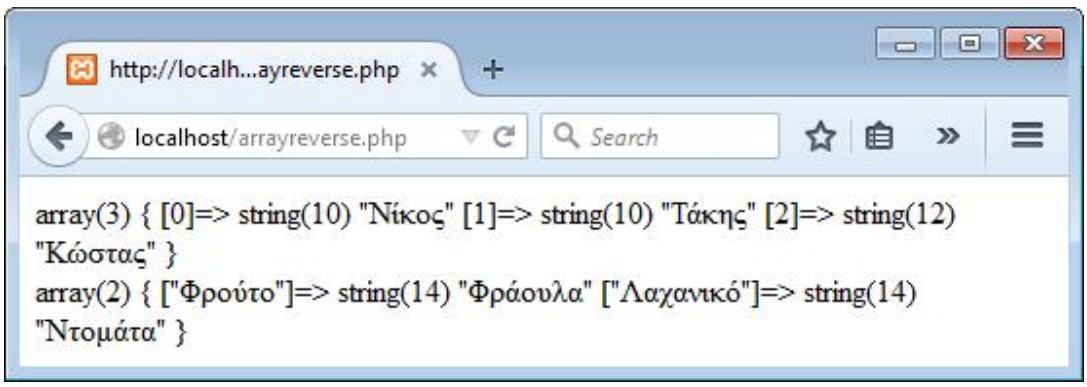
A.7.3.3 Αντιστροφή – array_reverse() & array_flip()

Στην php μπορούμε την σειρά των στοιχείων ενός πίνακα να την αλλάξουμε σε ακριβώς την αντίθετη από ότι είναι. Αυτό επιτυγχάνεται με την εντολή «array_reverse()», η οποία παίρνει ως είσοδο μια παράμετρο, τον πίνακα που θέλουμε να αντιστρέψουμε. Αν εισάγουμε έναν αριθμημένο πίνακα τότε τα αριθμητικά κλειδιά ξανά παίρνουν αριθμούς από την αρχή ξεκινώντας από το 0, έτσι μια τιμή που ήταν τελευταία και είχε πχ. κλειδί 10 με την αντιστροφή θα έχει κλειδί 0. Αν εισάγουμε μη αριθμημένο πίνακα τότε τα αλφαριθμητικά κλειδιά μένουν ανεπηρέαστα.

Αρχείο: arrayreverse.php

```
<?php
 //Αριθμημένος πίνακας
 $numbered = array("Κώστας","Τάκης","Νίκος");
 $reversed = array_reverse($numbered);
 var_dump($reversed);
 echo "<br/>";

 //Μη αριθμημένος πίνακας
 $unumbered = array(
   'Λαχανικό' => 'Ντομάτα',
   'Φρούτο'   => 'Φράουλα');
 $reversed = array_reverse($unumbered);
 var_dump($reversed);
?>
```

Αποτέλεσμα:



Εικόνα A.127 Αντιστροφή τιμών πίνακα με την array_reverse()

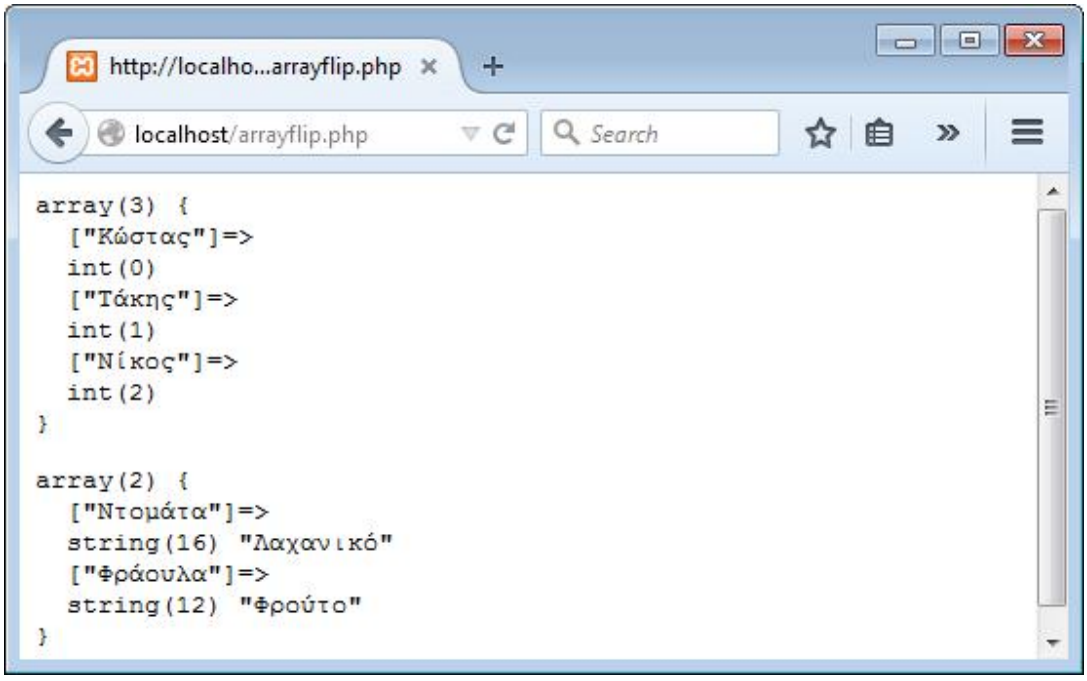
Εκτός από την αντίστροφη σειρά των στοιχείων υπάρχει και η αντιστροφή μεταξύ τιμών και κλειδιών. Δηλαδή η τιμή μιας θέσης να γίνει το κλειδί της ενώ το κλειδί της να γίνει η τιμή της. Τα στοιχεία τα οποία οι τιμές τους δεν είναι ούτε αλφαριθμητικά αλλά ούτε ακέραιοι αριθμοί, παραμένουν ως έχουν στο νέο αποτέλεσμα. Απαραίτητη

προϋπόθεση για να λειτουργήσει σωστά είναι κάθε τιμή που υπάρχει σε έναν πίνακα να είναι μοναδική σε σχέση με όλες τις άλλες τιμές, διότι κατά την αλλαγή σε κλειδί , δεν μπορούν να υπάρχουν διπλότυπα κλειδιά.

Αρχείο: arrayflip.php

```php
<?php
 //Αριθμημένος πίνακας
 $numbered = array("Κώστας","Τάκης","Νίκος");
 $fliped = array_flip($numbered);
 echo "<pre>";
 var_dump($fliped);
 echo "</pre>";

 //Μη αριθμημένος πίνακας
 $unumbered = array(
   'Λαχανικό' => 'Ντομάτα',
   'Φρούτο'   => 'Φράουλα');
 $fliped = array_flip($unumbered);
 echo "<pre>";
 var_dump($fliped);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.128 Αντιστροφή τιμών με κλειδιά σε έναν πίνακα με την array_flip()

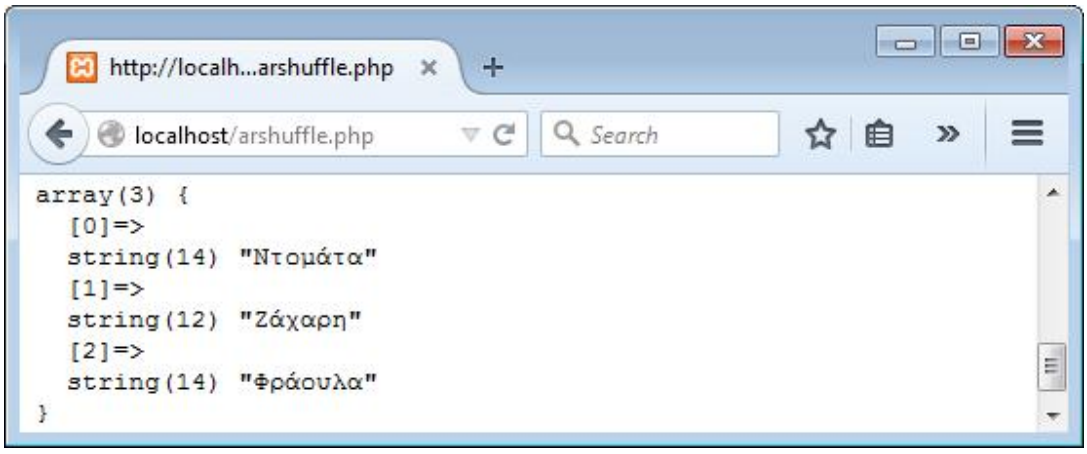
A.7.3.4 Τυχαία σειρά στοιχείων – shuffle()

Παρόμοια με την αντιστροφή πίνακα και την αντιστροφή κλειδιών-τιμών, μπορεί να υπάρξει και η εντελώς τυχαία σειρά των στοιχείων ενός πίνακα. Αυτό γίνεται με την εντολή «shuffle()» ή οποία δέχεται έναν πίνακα ως παράμετρο εισόδου στον οποίο και επιδρά. Αλλάζει τα στοιχεία σε μια τυχαία εντελώς σειρά από ότι είναι. Αντικαθιστά οποιοδήποτε κλειδί υπάρχει, με αριθμούς ξεκινώντας από το 0.

Αρχείο: arshuffle.php

```php
<?php
 //Αριθμημένος πίνακας
 $numbered = array(
   "Δευτέρα","Τρίτη",
   "Τετάρτη","Πέμπτη",
   "Παρασκευή","Σαββάτο",
   "Κυριακή");
 shuffle($numbered);
 echo "<pre>";
 var_dump($numbered);
 echo "</pre>";

 //Μη αριθμημένος πίνακας
 $unumbered = array(
   'Λαχανικό' => 'Ντομάτα',
   'Φρούτο'   => 'Φράουλα',
   'Γλυκαντικό' => 'Ζάχαρη');
 shuffle($unumbered);
 echo "<pre>";
 var_dump($unumbered);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα A.129 Τυχαία σειρά περιεχομένων ενός πίνακα με την shuffle()

Μετά την χρήση της shuffle είναι λογικό η σειρά πλέον να μην είναι ίδια όπως στο αποτέλεσμα μας καθότι εδώ χρησιμοποιούμε τυχαιότητα.

A.7.3.5 Μεταβλητές και πίνακες list(), extract(), compact()

Μια από τις πιο πολυσυζητημένες ικανότητες της php σε σύγκριση με ότι έχει να κάνει με έναν πίνακα είναι οι σχέσεις των πινάκων μεταξύ μεταβλητών και πως αυτά τα δυο μπορούν να αλληλεπιδράσουν αρμονικά τόσο για την εισαγωγή στοιχείων από μεταβλητές σε πίνακα όσο και την εξαγωγή μεταβλητών από τιμές πινάκων.

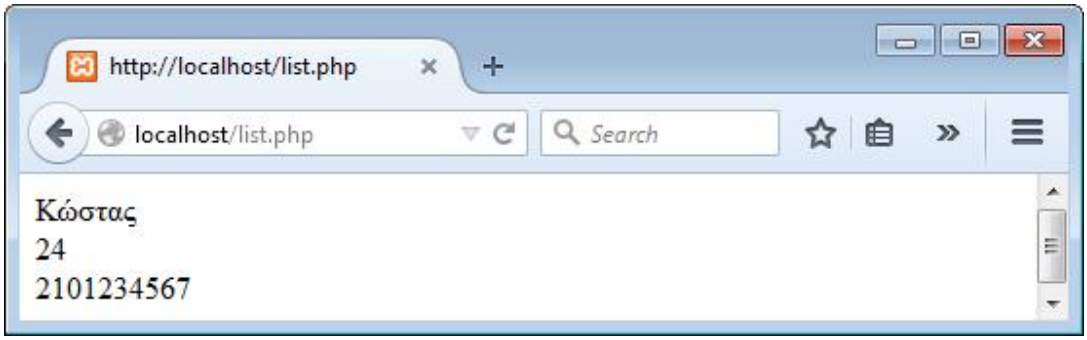
A.7.3.5.1 list()

Υπάρχει η δυνατότητα αντιγραφής όλων των τιμών ενός πίνακα σε μεταβλητές, χρησιμοποιώντας την εντολή list(). Η list() παίρνει τόσα ορίσματα, όσες και οι μεταβλητές που θα προκύψουν από τον πίνακα που έχουμε σκοπό να χρησιμοποιούμε γι αυτήν την αντιγραφή. Η σειρά με την οποία θα αντιγραφούν τα στοιχεία από τον πίνακα είναι ακριβώς ίδια με την σειρά στην οποία θα εισάγουμε τις μεταβλητές ως παραμέτρους μέσα στην list().

Αρχείο: list.php

```
<?php
 $epafi = array("Κώστας",24,2101234567);
 list($onoma,$ilikia,$tilefono) = $epafi;
 echo $onoma."<br/>".$ilikia."<br/>".$tilefono;
?>
```

Αποτέλεσμα:



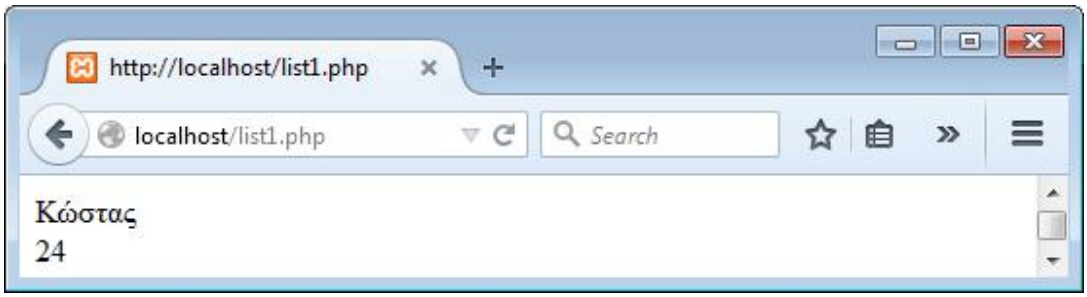
Εικόνα Α.130 Μετατροπή πίνακα σε μεταβλητές

Αν υπάρχουν περισσότερες τιμές στον πίνακα από ότι μεταβλητές μέσα στο list() τότε οι επιπλέον τιμές του πίνακα που δεν μπορούν να αντιστοιχηθούν σε μεταβλητές, και έτσι αγνοούνται.

Αρχείο: list1.php

```
<?php
 $epafi = array("Κώστας",24,2101234567);
 list($onoma,$ilikia) = $epafi;
 echo $onoma."<br/>".$ilikia;
?>
```

Αποτέλεσμα:



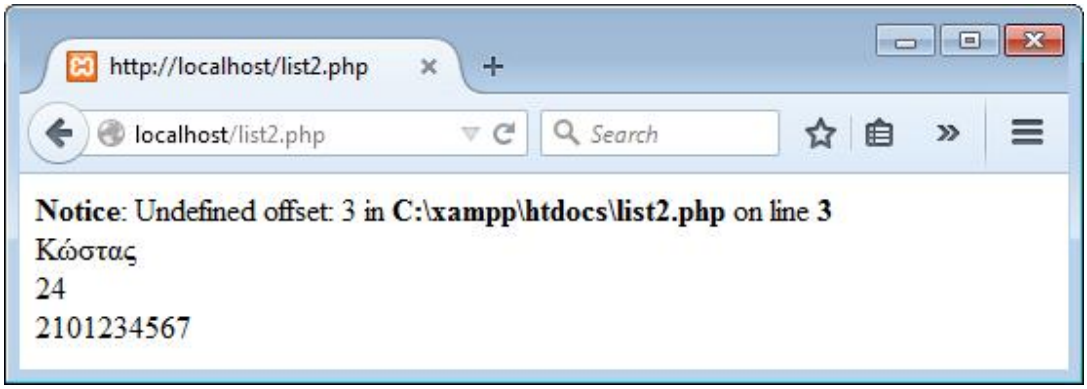
Εικόνα Α.131 Ελλιπής αντιστοίχηση πίνακα σε μεταβλητές

Δεν μπορούν να υπάρχουν περισσότερες μεταβλητές μέσα στην list() από όσα στοιχεία μέσα στον πίνακα διότι αυτό θα οδηγήσει σε σφάλμα.

Αρχείο: list2.php

```
<?php
 $epafi = array("Κώστας",24,2101234567);
 list($onoma,$ilikia,$tilefono,$odos) = $epafi;
 echo $onoma."<br/>".$ilikia."<br/>".$tilefono;
?>
```

Αποτέλεσμα:



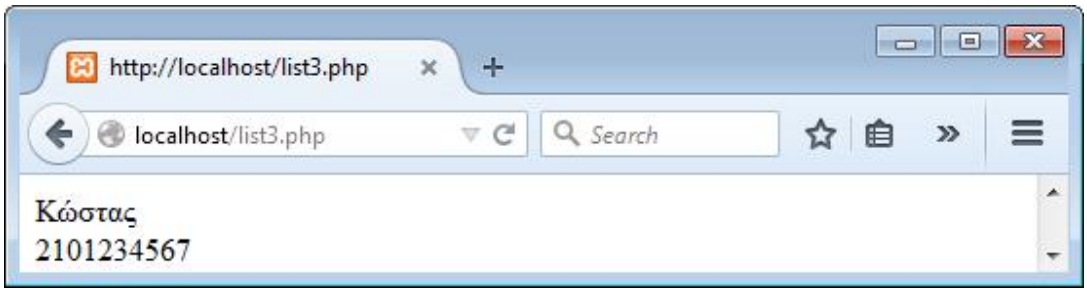
Εικόνα Α.132 Παράδειγμα εσφαλμένης αντιστοίχησης πίνακα σε μεταβλητές

Δυο συνεχόμενα κόμματα χωρίς να υπάρχει μια μεταβλητή ανάμεσα τους, σε εκείνο το σημείο αγνοούνται οι κατά σειρά τιμές του πίνακα που αντιστοιχούν σε εκείνα τα κενά.

Αρχείο: list3.php

```
<?php
 $epafi = array("Κώστας",24,2101234567);
 list($onoma,,$tilefono) = $epafi;
 echo $onoma."<br/>".$tilefono;
?>
```

Αποτέλεσμα:



Εικόνα Α.133 Διάφορες επιπλέον χρήσεις της εντολής list()

Α.7.3.5.2 extract()

Η εντολή extract() δημιουργεί μεταβλητές βασιζόμενη σε έναν μοναδικό μη αριθμημένο πίνακα με κλειδιά. Παίρνει τρία ορίσματα:

1. Έναν πίνακα από τον οποίο θα εξάγουμε τιμές,
2. Μία τιμή η οποία δηλώνει πως θα συμπεριφερθεί η εντολή (δείτε τον παρακάτω πίνακα).
3. Ένα αλφαριθμητικό που συμβολίζει ένα πρόθεμα.

Παίρνοντας όλα αυτά τα στοιχεία ως είσοδο δεν επιστρέφει κάτι, αλλά δημιουργεί τοπικές μεταβλητές με το ανάλογο περιεχόμενο.

Οι εντολές που μπορεί να πάρει το δεύτερο όρισμα είναι οι εξής:

- **EXTR_OVERWRITE**

  Αν υπάρχει σύγκρουση, τότε η ήδη υπάρχουσα μεταβλητή διαγράφεται και ισχύει η καινούργια.

- **EXTR_SKIP**

  Αν υπάρχει σύγκρουση, δεν διαγράφεται η ήδη υπάρχουσα μεταβλητή.

- **EXTR_PREFIX_SAME**

  Αν υπάρχει σύγκρουση, τότε δημιουργείται η νέα μεταβλητή με πρόθεμα την τιμή από το τρίτο όρισμα.

- **EXTR_PREFIX_ALL**

  Όλες οι νέες μεταβλητές δημιουργούνται με πρόθεμα το τρίτο όρισμα.

- **EXTR_PREFIX_INVALID**

  Πρόσθεσε πρόθεμα μόνο στις μεταβλητές που έχουν λανθασμένο όνομα.

- **EXTR_IF_EXISTS**

  Αντικαθίστανται οι μεταβλητές μόνο εφόσον υπάρχουν ήδη.

- **EXTR_PREFIX_IF_EXISTS**

  Δημιουργούνται μεταβλητές με πρόθεμα μόνο εάν οι εκδόσεις των αντίστοιχων μεταβλητών χωρίς πρόθεμα υπάρχουν ήδη.

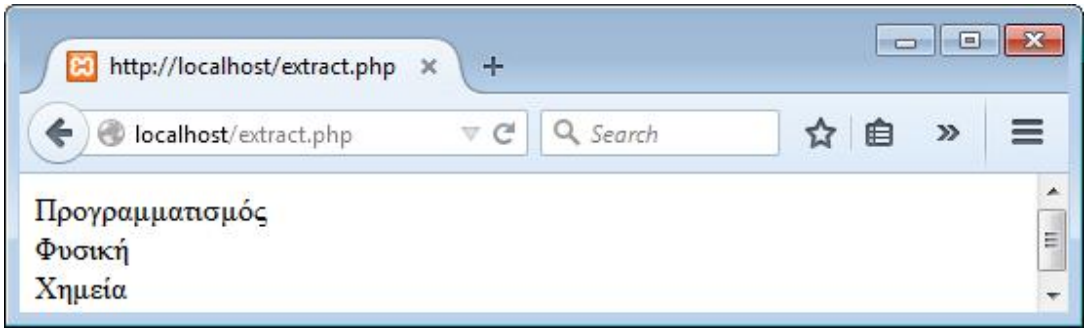
- **EXTR_REFS**

  Εξαγωγή μεταβλητών ως αναφορές. Αυτό σημαίνει ότι όλες οι μεταβλητές που θα δημιουργηθούν θα συνεχίζουν να είναι αναφορές προς τον πίνακα από όπου προκύψαν, κι αν αλλάξει το περιεχόμενο του αυτόματα αλλάζει και το περιεχόμενο των μεταβλητών.

Αρχείο: extract.php

```php
<?php
 $Μάθημα1 = "Προγραμματισμός";
 $mathimata = array(
  "Μάθημα1" => "Φυσική",
  "Μάθημα2" => "Χημεία",
  "Μάθημα3" => "Μαθηματικά",
  "Μάθημα4" => "Ιστορία",
  "Μάθημα5" => "Έκθεση" );

 extract($mathimata,EXTR_PREFIX_ALL,"νέο");

 echo $Μάθημα1."<br>";
 echo $νέο_Μάθημα1."<br>";
 echo $νέο_Μάθημα2;

?>
```

Αποτέλεσμα:



Εικόνα Α.134 Εξαγωγή μεταβλητών από πίνακα με την εντολή extract()

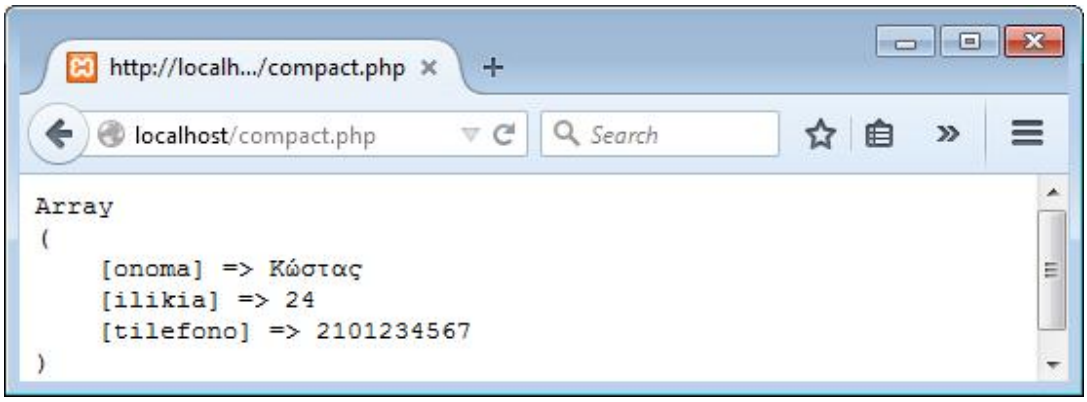
A.7.3.5.3 compact()

Η αντίστροφη χρήση της extract() είναι η compact(), όπου μπορούμε να δημιουργήσουμε έναν πίνακα βασιζόμενοι σε ένα σύνολο μεταβλητών. Η compact() παίρνει ως είσοδο μία λίστα από αλφαριθμητικές τιμές χωρισμένες με κόμμα, όπου κάθε τιμή είναι και ένα όνομα μεταβλητής. Ο πίνακας που επιστρέφεται είναι μη αριθμημένος και ως κλειδιά έχει τα ονόματα τα μεταβλητών ενώ ως τιμές έχει τις τιμές των μεταβλητών. Οποιαδήποτε άλλο όνομα το οποίο δεν αντιστοιχεί σε όνομα μεταβλητής ή δεν είναι σωστά γραμμένο με βάση τους κανόνες ονομασίας μεταβλητών, τότε απλά παραλείπεται.

Αρχείο: compact.php

```
<?php
 $onoma = "Κώστας";
 $ilikia = 24;
 $tilefono = 2101234567;

 $pinakas = compact("onoma","ilikia","tilefono");

 echo "<pre>";
 print_r($pinakas);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.135 Μετατροπή μεταβλητών σε πίνακα

Α.7.4 Πράξεις με πίνακες

Εκτός από την επεξεργασία ενός πίνακα αλλάζοντας το περιεχόμενο του ή την δομή του, είναι δυνατή και η αλληλεπίδραση διαφορετικών πινάκων μεταξύ τους. Παρακάτω θα δούμε τις βασικότερες πράξεις μεταξύ πινάκων στην PHP.

Α.7.4.1 Πράξεις μεταξύ πινάκων

Οι βασικότερες πράξεις μέσω των οποίων αλληλεπιδρών οι μεταβλητές στην php δρουν επίσης με παρόμοιο τρόπο στο πώς δυο πίνακες συμπεριφέρονται μεταξύ τους.

Πράξη	Επεξήγηση	Αποτέλεσμα
`$a + $b`	Ένωση	Ενώνονται οι πίνακες $a και $b
`$a == $b`	Ισότητα	Ελέγχεται αν οι δυο πίνακες έχουν ίδια κλειδιά και τιμές και επιστρέφεται true.
`$a === $b`	Ταυτότητα	Ελέγχονται για ίδιες τιμές και κλειδιά και επίσης για ίδιους τύπους και επιστρέφεται true.
`$a != $b`	Ανισότητα	Αν οι δυο πίνακες δεν είναι ίσοι μεταξύ τους επιστρέφεται true.
`$a <> $b`	Ανισότητα	Παρόμοια με παραπάνω.
`$a !== $b`	Μη ταυτότητα	Ελέγχεται αν οι δυο πίνακες δεν έχουν ίδιους τύπους (ας έχουν ίδια κλειδιά και τιμές) και επιστρέφεται true.

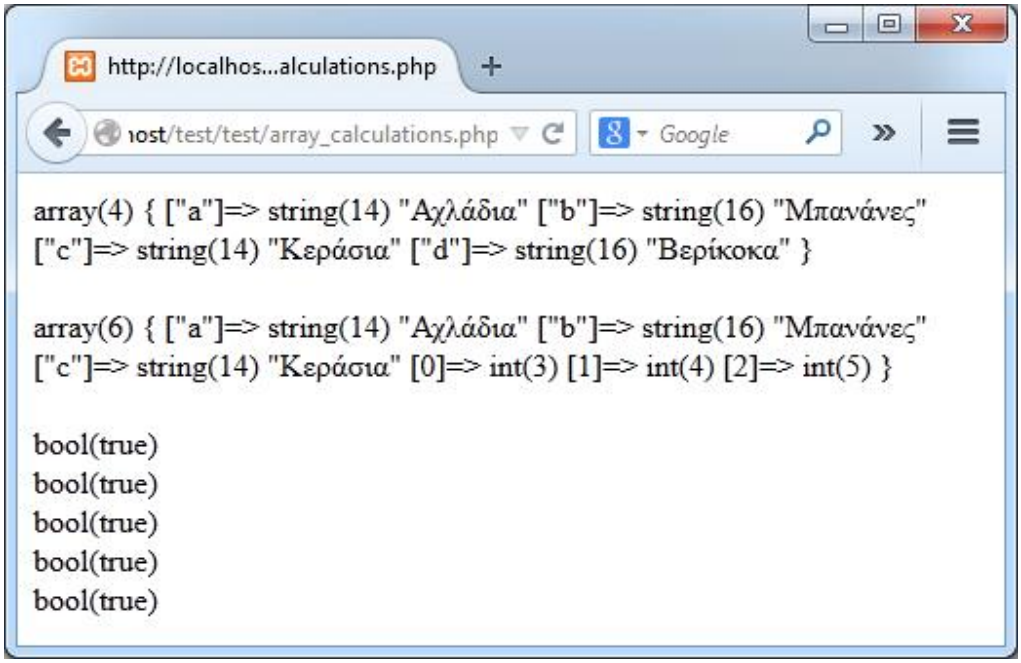
Πίνακας Α.23 Βασικές πράξεις μεταξύ πινάκων

Ας δούμε ένα παράδειγμα με όλα τα παραπάνω:

Αρχείο: array_calculations.php

```php
<?php
 $a = array("a"=>"Αχλάδια","b"=>"Μπανάνες","c"=>"Κεράσια");
 $b = array("a"=>"Αχλάδια","b"=>"Μπανάνες","d"=>"Βερίκοκα");
 $c = array(3,4,5);
 $c1 = array(3,4,5);
 $d = array(3.0,4.0,5.2);

 //Θα προστεθούν όσα στοιχεία με κριτήριο το κλειδί τους υπάρχουν στο
b αλλα δεν υπάρχουν στο a.
 var_dump($a+$b);
 echo "<br/><br/>";
 var_dump($a+$c);

 echo "<br/><br/>";

 var_dump($c==$c1); //Ισότητα
 echo "<br/>";
 var_dump($c===$c1); //Ταυτότητα
 echo "<br/>";
 var_dump($c!=$d); //Ανισότητα
 echo "<br/>";
 var_dump($c<>$d); //Ανισότητα
 echo "<br/>";
 var_dump($c!==$d); //Μη ταυτότητα
?>
```

Αποτέλεσμα:



Εικόνα Α.136 Αποτέλεσμα εκτέλεσης όλων των πράξεων μεταξύ δυο πινάκων

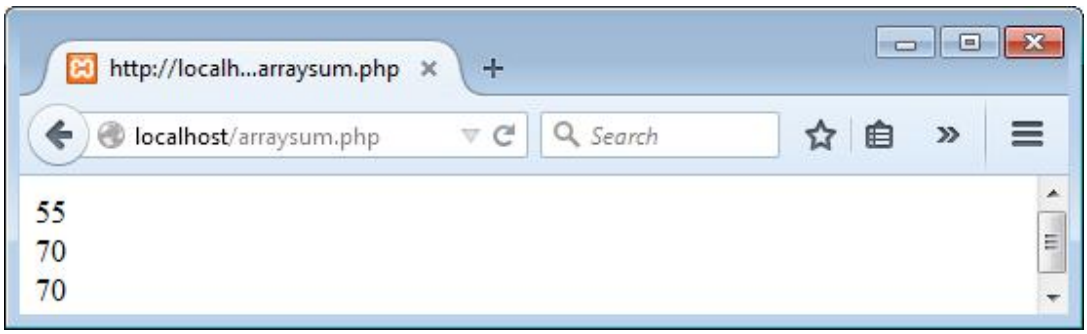
A.7.4.2 Υπολογισμός αθροίσματος

Η εντολή array_sum() μας προσφέρει το άθροισμα όλων τον τιμών ενός αριθμημένο ή και μη αριθμημένου πίνακα. Δέχεται ως παράμετρο εισόδου έναν πίνακα και επιστρέφει το συνολικό του άθροισμα. Αν της εισάγουμε έναν πίνακα όπου έχει μόνο αριθμητικές τιμές θα υπολογίσει το άθροισμα και θα το επιστρέψει. Αν της εισάγουμε έναν πίνακα που έχει και αριθμούς αλλά και αλφαριθμητικά, τότε θα υπολογίσει μόνο τους αριθμούς στο άθροισμα και θα το επιστρέψει.

Αρχείο: arraysum.php

```
<?php
 $a = range(1,10);
 echo array_sum($a);
 echo "<br/>";

 $b = array("Πρώτο"=>40,"Δεύτερο"=>30);
 echo array_sum($b);
  echo "<br/>";

 $c = array("Όνομα"=>"Ιωάννης", "Ηλικία" => 20, 40,10.0);
 echo array_sum($c);
?>
```

Αποτέλεσμα:



Εικόνα Α.137 Υπολογισμός αθροίσματος πίνακα

A.7.4.3 Διαφορά μεταξύ δυο πινάκων

Μια ακόμη γνωστή λειτουργία είναι να ελέγξουμε δυο πίνακες που ακριβώς διαφέρουν, ποια στοιχεία είναι μεταξύ τους διαφορετικά και όχι ίδια.

A.7.4.3.1 array_diff()

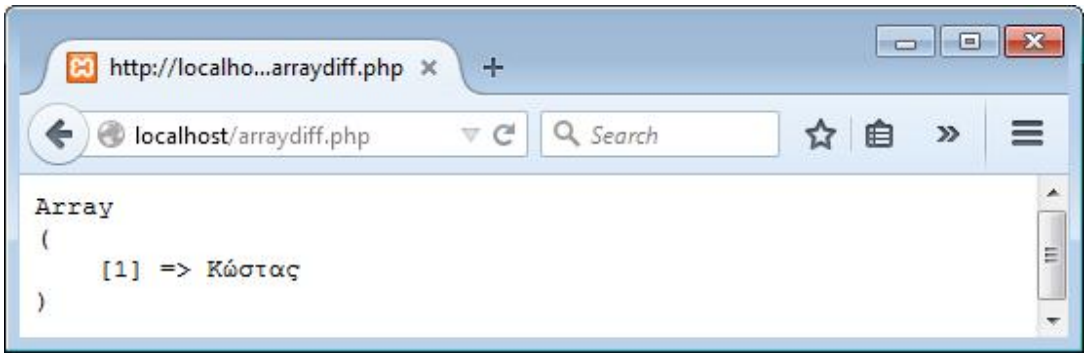
Η εντολή array_diff() δέχεται ως είσοδο όσους πίνακες θέλουμε να ελέγξουμε μεταξύ τους. Η εντολή ελέγχει συγκριτικά τον πρώτο πίνακα που εισάγουμε ως παράμετρο με

όλους τους άλλους, και όσα στοιχεία του πρώτου πίνακα δεν υπάρχουν στους άλλους πίνακες επιστρέφονται ομαδοποιημένα ως πίνακας με τα ίδια ακριβώς κλειδιά.

Αρχείο: arraydiff.php

```php
<?php
 $a = array("Νίκος","Κώστας","Γιάννης");
 $b = array("Νίκος",10,20,30);
 $c = array(0.0, 30,10,false,"Γιάννης",30);

 $diafora = array_diff($a,$b,$c);
 echo "<pre>";
 print_r($diafora);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.138 Διαφορά μεταξύ πινάκων

Κάθε στοιχείο ελέγχεται μέσω του τελεστή του τριπλού ίσον που σημαίνει πως εκτός από την ισότητα περιεχομένου ελέγχεται και αν το περιεχόμενο είναι ίδιου τύπου. Έτσι το 10 ως ακέραιος και το "10" ως αλφαριθμητικό δεν είναι ίδιο.

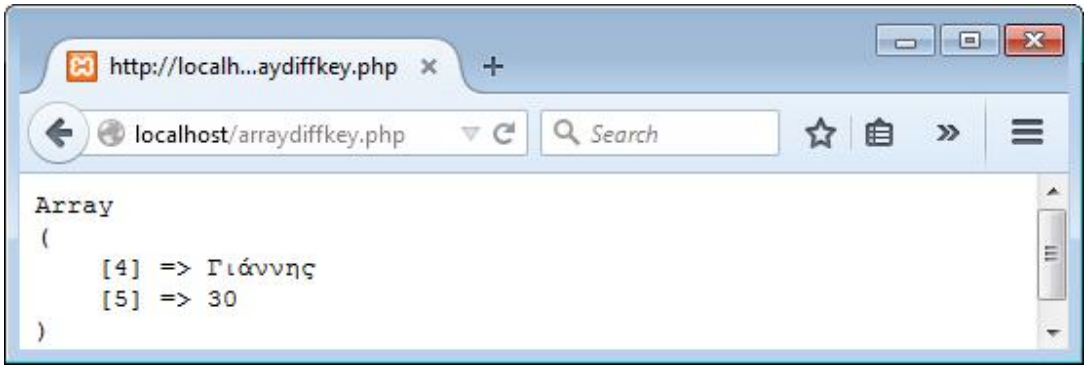
Α.7.4.3.2 array_diff_key()

Εκτός από το να ελέγξουμε την διαφορά τιμών μεταξύ πινάκων μπορούμε να ελέγξουμε και την διαφορά κλειδιών. Αυτό επιτυγχάνεται με την εντολή array_diff_key() όπου διασταυρώνονται όλα τα κλειδιά του πρώτου πίνακα με όλους τους άλλους που θα εισάγουμε. Αν κάποια από τα κλειδιά του πρώτου πίνακα δεν βρεθούν σε κανέναν από όλους τους άλλους τότε επιστρέφονται, ομαδοποιημένα σε πίνακα διατηρώντας την τιμή τους.

Αρχείο: arraydiffkey.php

```php
<?php
 $a = array(0.0, 30,10,false,"Γιάννης",30);
 $b = array("Νίκος","Κώστας","Γιάννης");
 $c = array("Νίκος",10,20,30);

 $diafora = array_diff_key($a,$b,$c);
 echo "<pre>";
 print_r($diafora);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.139 Διαφορά μεταξύ πινάκων εστιασμένη στα κλειδιά

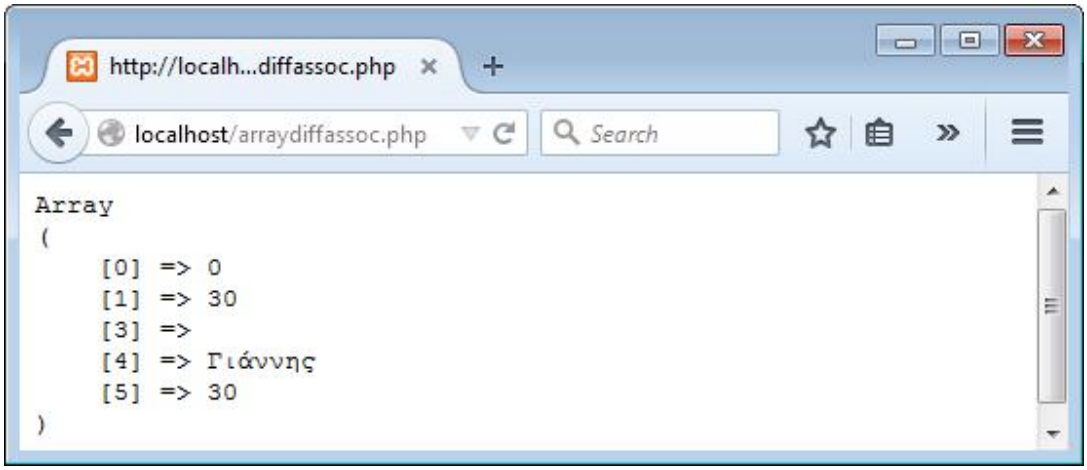
Α.7.4.3.3 array_diff_assoc()

Αν θέλουμε να ελέγξουμε ολοκληρωτικά ένα σύνολο από πίνακες λαμβάνοντας υπόψη τόσο τα κλειδιά τους όσο και την τιμή τους, τότε θα χρησιμοποιήσουμε την εντολή array_diff_assoc() η οποία δέχεται ως είσοδο ένα σύνολο από πίνακες και ελέγχει την ομοιότητα του πρώτου με όλους τους υπόλοιπους.

Αρχείο: arraydiffassoc.php

```php
<?php
 $a = array(0.0, 30,10,false,"Γιάννης",30);
 $b = array("Νίκος","Κώστας","Γιάννης");
 $c = array("Νίκος",20,10,30);

 $diafora = array_diff_assoc($a,$b,$c);
 echo "<pre>";
 print_r($diafora);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.140 Διαφορά πινάκων σε συνδυασμό κλειδιών και τιμών

Εδώ θα πρέπει να σημειώσουμε ξανά πως τα αποτελέσματα προέρχονται από τον πρώτο μόνο πίνακα , δηλαδή επιστρέφονται οι τιμές και τα κλειδιά του πρώτου πίνακα που δεν υπάρχουν σε όλους τους υπόλοιπους.

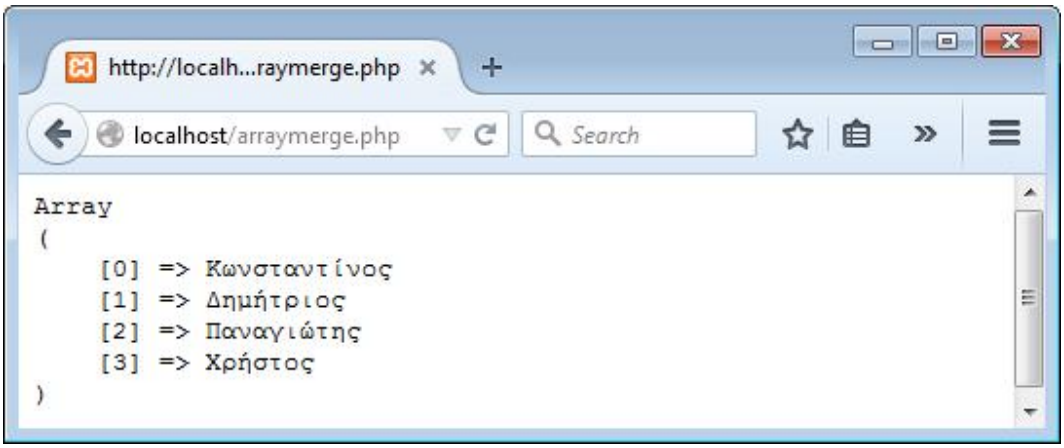
Α.7.4.4 Ένωση

Η εντολή array_merge() επιστρέφει την ένωση μεταξύ πινάκων. Παίρνει ως ορίσματα εισόδου ένα σύνολο από πίνακες από τους οποίους θέλουμε να εξάγουμε την ένωση τους και επιστρέφει έναν νέο πίνακα με τα στοιχεία όλων. Αν δυο η περισσότερα στοιχεία έχουν το ίδιο μη αριθμημένο κλειδί, τότε το τελευταίο αντικαθιστά όλα τα προηγούμενα. Αν ως όρισμα εισάγουμε μόνο έναν αριθμημένο πίνακα στην array_merge() τότε αυτό που επιστρέφεται είναι ο ίδιος αριθμημένος πίνακας αλλά με κλειδιά ξεκινώντας από το 0.

Αρχείο: arraymerge.php

```
<?php
 $foitites = array("Κωνσταντίνος","Δημήτριος","Ιωάννης");
 $kathigites = array("Παναγιώτης","Χρήστος","Γεώργιος");

 $tei = array_merge($foitites,$kathigites);

 echo "<pre>";
 print_r($tei);
 echo "</pre>";
?>
```

Αποτέλεσμα:



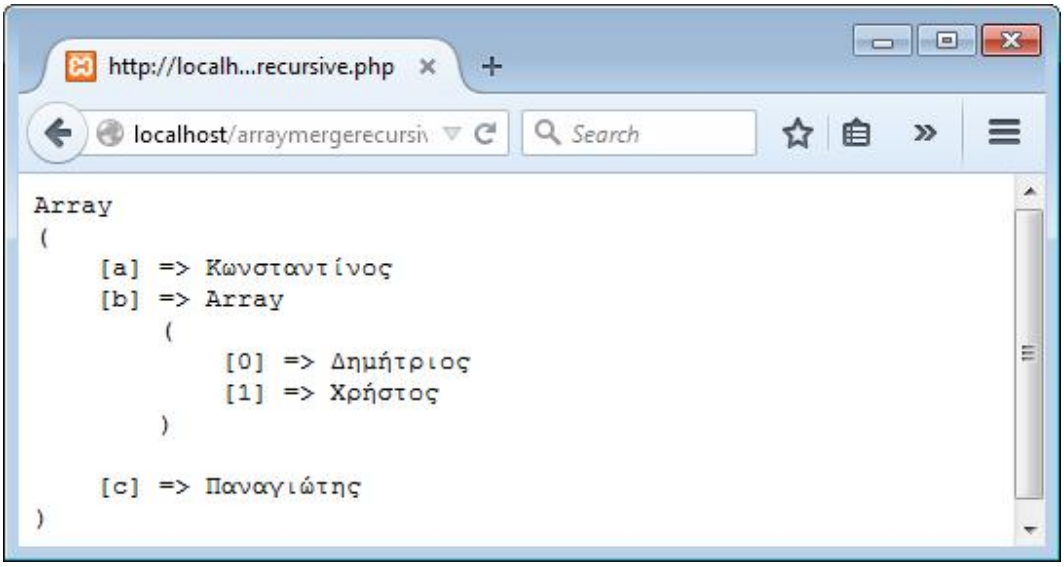
Εικόνα Α.141 Ένωση δύο πινάκων

Επίσης η εντολή array_merge_recursive() επιστρέφει την ένωση μεταξύ πινάκων όπου όταν βρεθούν ίδια κλειδιά αυτά διατηρούνται ακολουθώντας την διαδοχική σειρά των στοιχείων. Κάθε πίνακας προστίθεται στο τέλος του προηγούμενου και αυτό συμβαίνει αναδρομικά. Αν δυο στοιχεία έχουν διαφορετική τιμή αλλά ίδιο κλειδί τότε στην επιστροφή του επιστρέφεται ένας πίνακας, στην θέση της τιμής του ίδιου κλειδιού.

Αρχείο: arraymergerecursive.php

```php
<?php
 $foitites = array(1=>"Κωνσταντίνος",4=>"Δημήτριος");
 $kathigites = array(1=>"Παναγιώτης",4=>"Χρήστος",0=>"Γεώργιος");

 $tei = array_merge_recursive($kathigites,$foitites);

 echo "<pre>";
 print_r($tei);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.142 Ένωση πινάκων διατηρώντας τα διπλότυπα

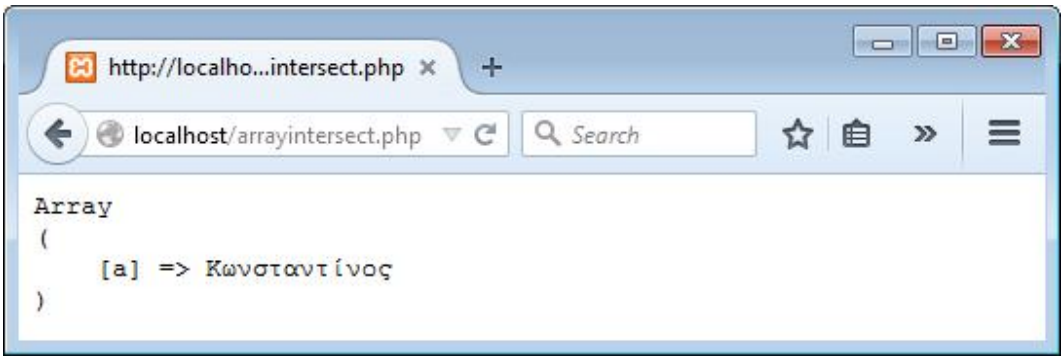
Α.7.4.5 Τομή

Η εντολή array_intersect() δέχεται ως ορίσματα ένα πλήθος πινάκων, ελέγχει τις τιμές τους, και επιστρέφει τα κοινά στοιχεία που υπάρχουν σε κάθε πίνακα, δηλαδή την τομή τους. Μαζί με τα όμοια στοιχεία που βρέθηκαν σε όλους τους πίνακες επιστρέφονται τα κλειδιά του πρώτου πίνακα.

Αρχείο: arrayintersect.php

```
<?php
 $foitites = array("a"=>"Κωνσταντίνος","e"=>"Δημήτριος");
 $kathigites = array("b"=>"Κωνσταντίνος","w"=>"Χρήστος");

 $tei = array_intersect($foitites,$kathigites);

 echo "<pre>";
 print_r($tei);
 echo "</pre>";
?>
```

Αποτέλεσμα:



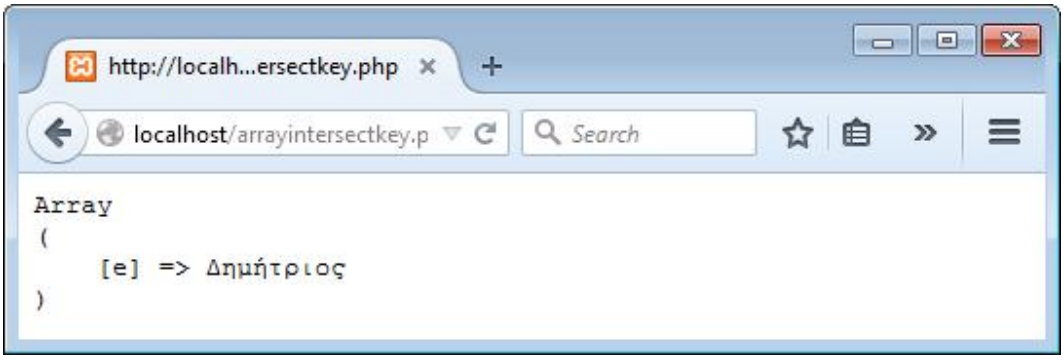
Εικόνα Α.143 Τομή πινάκων με βάση τις τιμές τους

Παρομοίως η εντολή array_intersect_key() επιστρέφει την τομή ενός πλήθους πινάκων βασιζόμενη όμως στα κλειδιά και όχι στις τιμές. Για τα όμοια κλειδιά που ανακαλύπτονται επιστρέφεται η τιμή που βρίσκεται μόνο στον πρώτο πίνακα.

Αρχείο: arrayintersectkey.php

```
<?php
 $foitites = array("a"=>"Κωνσταντίνος","e"=>"Δημήτριος");
 $kathigites = array("b"=>"Κωνσταντίνος","e"=>"Χρήστος");

 $tei = array_intersect_key($foitites,$kathigites);

 echo "<pre>";
 print_r($tei);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.144 Τομή πινάκων με βάση τα κλειδιά τους

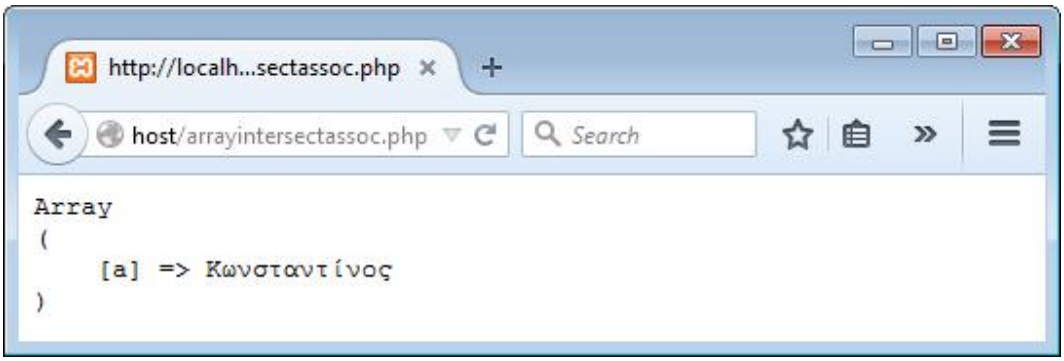
Τέλος η εντολή array_intersect_assoc() εστιάζει τόσο στις τιμές όσο και στα κλειδιά των πινάκων και επιστρέφει την τομή τους. Θα πρέπει να βρεθούν δηλαδή τόσο ίδιο

κλειδί όσο και τιμή σε διαφορετικά στοιχεία έτσι ώστε να θεωρηθούν τομή των πινάκων.

Αρχείο: arrayintersectassoc.php

```
<?php
 $foitites = array("a"=>"Κωνσταντίνος","e"=>"Δημήτριος");
 $kathigites = array("a"=>"Κωνσταντίνος","e"=>"Χρήστος");

 $tei = array_intersect_assoc($foitites,$kathigites);

 echo "<pre>";
 print_r($tei);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.145 Τομή πινάκων με βάση συνδυασμό κλειδιού και τιμής

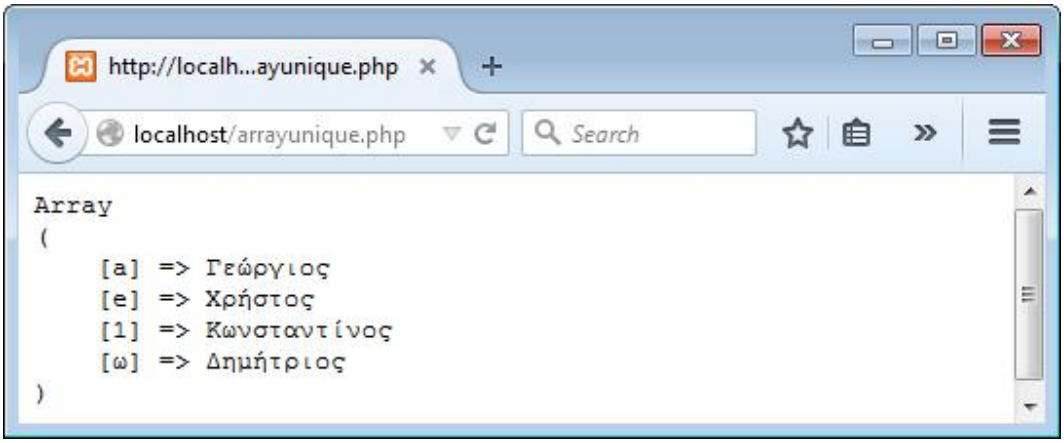
Α.7.4.6 Μοναδικότητα

Η εντολή array_unique() δέχεται ως όρισμα έναν μοναδικό πίνακα, και επιστρέφει έναν νέο πίνακα στον οποίο έχουν αφαιρεθεί οι διπλότυπες τιμές του πίνακα που εισάγαμε.

Αρχείο: arrayunique.php

```php
<?php
 $foitites = array(
        "a"=>"Κωνσταντίνος",
        "e"=>"Δημήτριος",
        "a"=>"Γεώργιος",
        "e"=>"Χρήστος",
        1=>"Κωνσταντίνος",
        2=>"Χρήστος",
        "ω"=>"Δημήτριος",
        "ε"=>"Γεώργιος");

 $tei = array_unique($foitites);

 echo "<pre>";
 print_r($tei);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.146 Μοναδικές τιμές ενός πίνακα σε συνδυασμό κλειδιού-τιμής

Προαιρετικά μπορούμε να εισάγουμε και ένα δεύτερο όρισμα που καθορίζει τον τρόπο με τον οποίο θα συμπεριφερθεί η εντολή, ο παρακάτω πίνακας επεξηγεί την λειτουργία τους:

- **SORT_REGULAR**
  Ελέγχει τα στοιχεία κανονικά χωρίς να αλλάζει τύπους.
- **SORT_NUMERIC**
  Ελέγχει τα στοιχεία αριθμητικά.
- **SORT_STRING**
  Ελέγχει τα στοιχεία ως αλφαριθμητικά.
- **SORT_LOCALE_STRING**
  Ελέγχει τα στοιχεία ως αλφαριθμητικά με βάση την εκάστοτε γλώσσα.

Α.7.4.7 Στοίβες

Στον προγραμματισμό υπάρχουν πολλές δομές δεδομένων από τις οποίες λαμβάνουμε διαφορετική συμπεριφορά και διαφορετικά αποτελέσματα.

Μια από αυτές τις δομές είναι η στοίβα. Σε μία στοίβα από πιάτα, το πρώτο πιάτο που εισάγεται στην στοίβα πηγαίνει στον πάτο της και επομένως είναι αυτό το οποίο θα βγει τελευταίο από αυτήν. Ο μόνος τρόπος για να πάρουμε ένα πιάτο από την στοίβα είναι παίρνοντας το στοιχείο που βρίσκεται στην κορυφή της, δηλαδή αυτό που μπήκε τελευταίο στην στοίβα. Αντίστοιχα για να προσθέσουμε ένα νέο στοιχείο στην στοίβα θα πρέπει να το εισάγουμε τελευταίο και πάνω από όλα τα υπόλοιπα, δεν μπορούμε απλά να το σπρώξουμε στην θέση που θέλουμε γιατί τότε θα σπάσουν όλα τα πιάτα.

Κάπως έτσι λειτουργεί και η στοίβα στον προγραμματισμό, είναι μια δομή LIFO (Last In First Out – Ότι μπαίνει τελευταίο βγαίνει πρώτο). Στην php κάθε πίνακας που δημιουργείται μπορεί να συμπεριφερθεί σαν στοίβα αρκεί να χρησιμοποιήσουμε τις κατάλληλες εντολές

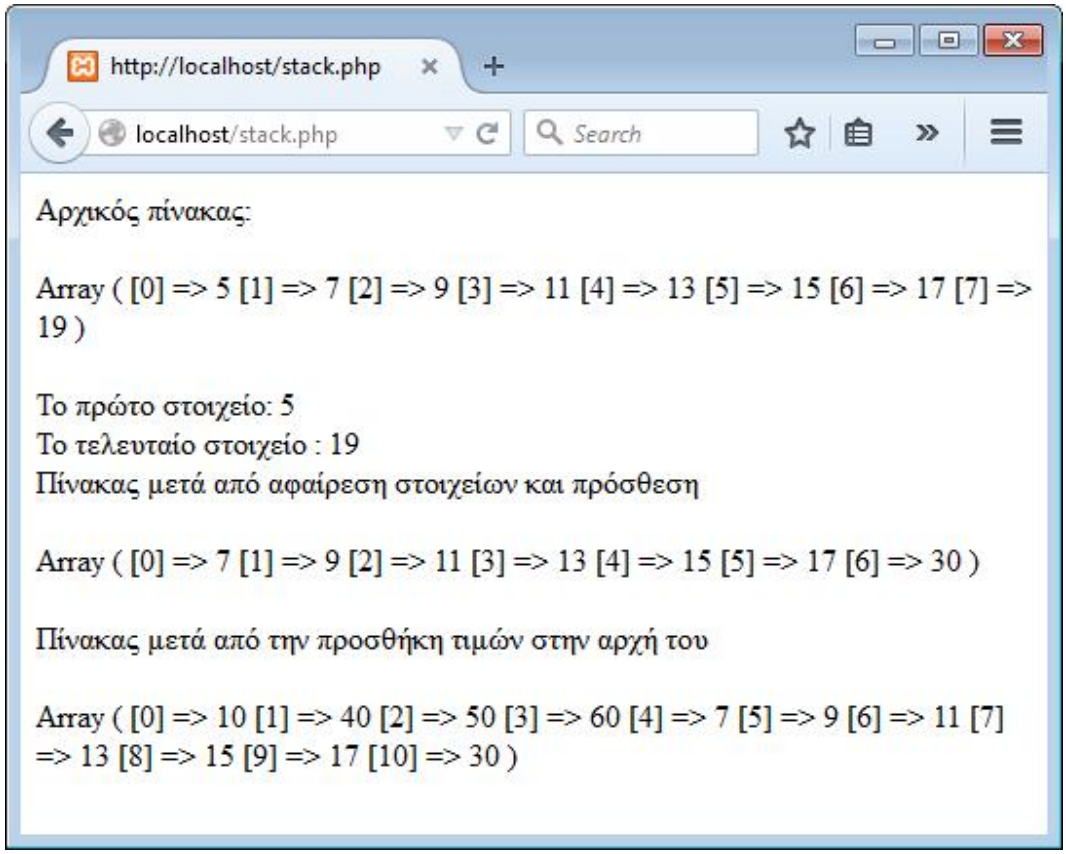
- Η εντολή **array_shift()** δέχεται έναν πίνακα ως όρισμα , επαναφέρει τον δείκτη του στην αρχή του και επιστρέφει την τιμή του στοιχείου σε αυτή την θέση. Το στοιχείο επιστρέφεται έχοντας αφαιρεθεί από την στοίβα. Τα αριθμημένα κλειδιά αναπροσαρμόζονται.
- Η εντολή **array_push()** δέχεται ως όρισμα έναν πίνακα και μια τιμή οποιαδήποτε τύπου δεδομένων και την εισάγει στον πίνακα που εισάγαμε ως πρώτο όρισμα, η θέση του νέου στοιχείο θα είναι ακριβώς στο τέλος του πίνακα κάθε φορά.
- Η εντολή **array_pop()** δέχεται ως όρισμα έναν πίνακα και επιστρέφει το τελευταίο του στοιχείο, επίσης μετά την εκτέλεση της επαναφέρει τον δείκτη του πίνακα στην πρώτη του θέση. Το στοιχείο επιστρέφεται έχοντας αφαιρεθεί από τον στοίβα. Τα αριθμημένα κλειδιά αναπροσαρμόζονται.
- Η εντολή **array_unshift()** δέχεται ως πρώτο όρισμα έναν πίνακα και ως δεύτερο ένα σύνολο από τιμές και ύστερα εισάγει αυτές τις τιμές στην αρχή του πίνακα αλλάζοντας ταυτόχρονα όλα (ΜΟΝΟ) τα αριθμητικά κλειδιά των υπόλοιπων στοιχείων έτσι ώστε οι νεοεισερχόμενες τιμές να ξεκινήσουν με

κλειδιά από «μηδέν». Τα μη αριθμητικά κλειδιά δεν θα επηρεαστούν από την εκτέλεση της εντολής.

Αρχείο: stack.php

```
<?php
function tiposi($pinakas){
  echo "<br/>";
  echo "<br/>";
  print_r($pinakas);
  echo "<br/>";
  echo "<br/>";
 }

 $stiba = range(5,20,2);
 echo "Αρχικός πίνακας:";
 tiposi($stiba);

 $shift = array_shift($stiba); //έξοδος πρώτου στοιχείου
 echo "Το πρώτο στοιχείο: ".$shift."<br/>";

 $pop = array_pop($stiba);//έξοδος τελευταίου στοιχείου
 echo "Το τελευταίο στοιχείο : ".$pop."<br/>";

 array_push($stiba,30); //προσθήκη στοιχείου στο τέλος
 echo "Πίνακας μετά από αφαίρεση στοιχείων και πρόσθεση";
 tiposi($stiba);

 array_unshift($stiba,10,40,50,60); //εισαγωγή νέων τιμών
 echo "Πίνακας μετά από την προσθήκη τιμών στην αρχή του";
 tiposi($stiba);
?>
```

Αποτέλεσμα:



Εικόνα Α.147 Δομή στοίβας στην php

Χρησιμοποιούμε την μέθοδο «tiposi» για μεγαλύτερη ευκολία στην τύπωση αποτελεσμάτων. Για να μάθετε περισσότερα για τις μεθόδους ανατρέξτε στο ανάλογο κεφάλαιο.

Α.7.5 Ταξινόμηση

Συχνά κατά την χρήση πινάκων στις γλώσσες προγραμματισμού, εκφράζεται η ανάγκη για ταξινόμηση των στοιχείων που διαθέτει με κάποια κριτήρια, όπως π.χ. σε ένα σχολείο θα μπορούσαν να ταξινομηθούν οι μαθητές κατά φθίνουσα σειρά με βάση τον μέσο όρο τους, δίνοντας έτσι μια λίστα με τους 10 καλύτερους μαθητές του σχολείου.

Στην php η ταξινόμηση μπορεί να γίνει είτε χειροκίνητα είτε χρησιμοποιώντας κάποιες από τις έτοιμες λειτουργίες – μεθόδους που διαθέτει. Εμείς θα εξετάσουμε όλες τις έτοιμες μεθόδους που περιγράφονται στον παρακάτω πίνακα:

Αύξουσα	Φθίνουσα	Επεξήγηση
sort()	rsort()	Ταξινόμηση με την τιμή σε καινούργιο

		αριθμημένο πίνακα με κλειδιά από το 0.
asort()	arsort()	Ταξινόμηση με την τιμή.
ksort()	krsort()	Ταξινόμηση με τα κλειδιά.
array_multisort()		Ταξινόμηση πολλών πινάκων ταυτόχρονα

Πίνακας Α.24 Εντολές ταξινόμησης στην php

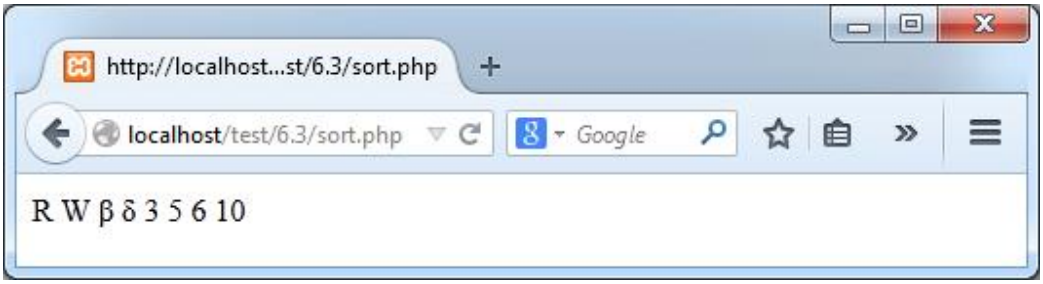
Α.7.5.1 Αύξουσα ταξινόμηση τιμών– sort()

Η συνάρτηση «sort()» ταξινομεί τις τιμές ενός πίνακα με κλειδιά σε αύξουσα σειρά. Δηλαδή από το Α … έως το Ω και από το 0 έως το 9. Τα κεφαλαία πάνε πριν τα πεζά και όλα τα γράμματα πάνε πριν τους αριθμούς. Επίσης οι λατινικοί χαρακτήρες πάνε πριν τους ελληνικούς.

Αρχείο: sort.php

```
<?php
 $pinakas = array('W', 3 , 5, 'δ', 6, 'β', 'R', 10);
 sort($pinakas);

 foreach($pinakas as $value){
  echo $value." ";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.148 Αποτέλεσμα εκτέλεσης sort()

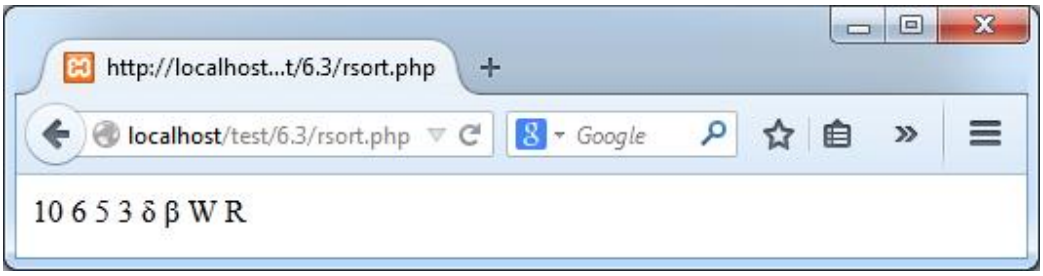
Α.7.5.2 Φθίνουσα ταξινόμηση τιμών – rsort()

Η συνάρτηση «rsort ()» ταξινομεί τις τιμές ενός πίνακα με κλειδιά σε φθίνουσα σειρά. Δηλαδή από το Ω … έως το Α και από το 9 έως το 0. Τα πεζά πάνε πριν τα κεφαλαία και όλοι οι αριθμοί πάνε πριν τα γράμματα. Επίσης οι ελληνικοί χαρακτήρες πάνε πριν τους λατινικούς.

Αρχείο: rsort.php

```php
<?php
 $pinakas = array('W', 3 , 5, 'δ', 6, 'β', 'R', 10);
 rsort($pinakas);

 foreach($pinakas as $value){
  echo $value." ";
 }
?>
```

Αποτέλεσμα:



Εικόνα Α.149 Φθίνουσα ταξινόμηση με την rsort()

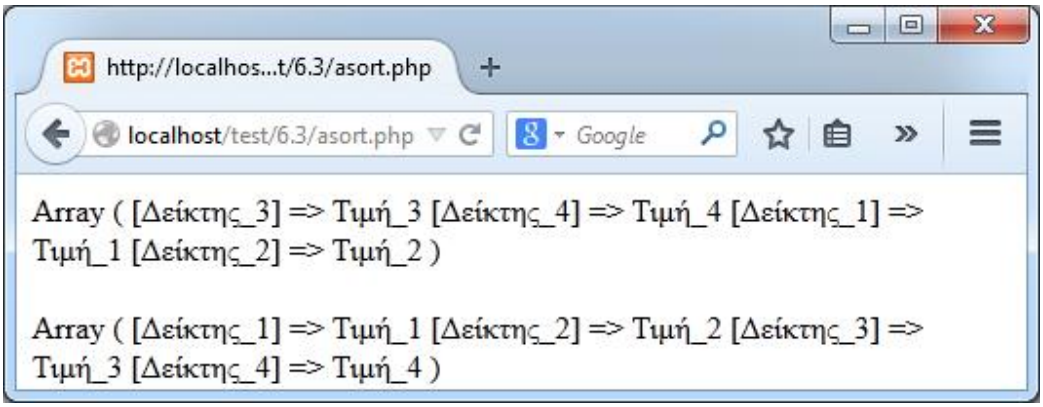
Α.7.5.3 Αύξουσα ταξινόμηση τιμών σε μη αριθμημένο πίνακα– asort()

Είναι μια συνάρτηση ταξινόμησης σε μη αριθμημένους πίνακες η οποία κάνει κατά αύξουσα σειρά ταξινόμηση. Η συνάρτηση εστιάζει στις τιμές του πίνακα και όχι στα κλειδιά της, διατηρώντας έτσι την ονομασία των κλειδιών.

Αρχείο: assort.php

```php
<?php
 $pinakas = array(
  "Δείκτης_3" => "Τιμή_3",
  "Δείκτης_4" => "Τιμή_4",
  "Δείκτης_1" => "Τιμή_1",
  "Δείκτης_2" => "Τιμή_2"
 );

 print_r($pinakas); //Τύπωση αρχικού πίνακα
 echo "<br/><br/>"; //κενός χώρος
 asort($pinakas); //ταξινόμηση τιμών διατηρώντας τους δείκτες
 print_r($pinakas); //εκτύπωση πίνακα
?>
```

Αποτέλεσμα:



Εικόνα Α.150 Αύξουσα ταξινόμηση σε μη αριθμημένο πίνακα με την asort()

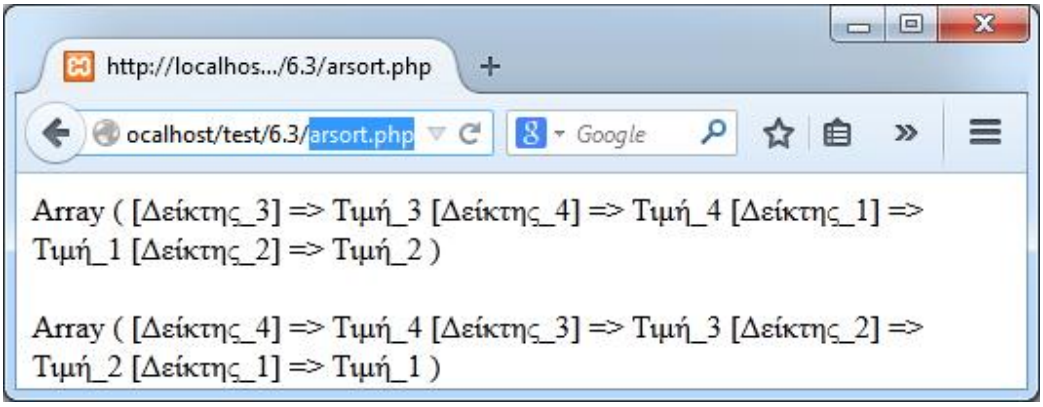
Α.7.5.4 Φθίνουσα ταξινόμηση τιμών σε μη αριθμημένο πίνακα– arsort()

Είναι μια συνάρτηση ταξινόμησης σε μη αριθμημένους πίνακες η οποία κάνει κατά φθίνουσα σειρά ταξινόμηση. Η συνάρτηση εστιάζει στις τιμές του πίνακα και όχι στα κλειδιά της, διατηρώντας έτσι την ονομασία των κλειδιών.

Αρχείο: arsort.php

```
<?php
 $pinakas = array(
  "Δείκτης_3" => "Τιμή_3",
  "Δείκτης_4" => "Τιμή_4",
  "Δείκτης_1" => "Τιμή_1",
  "Δείκτης_2" => "Τιμή_2"
 );

 print_r($pinakas); //Τύπωση αρχικού πίνακα
 echo "<br/><br/>"; //κενός χώρος
 arsort($pinakas); //ταξινόμηση τιμών διατηρώντας τους δείκτες
 print_r($pinakas); //εκτύπωση πίνακα
?>
```

Αποτέλεσμα:



Εικόνα Α.151 Φθίνουσα ταξινόμηση σε μη αριθμημένο πίνακα με την arsort()

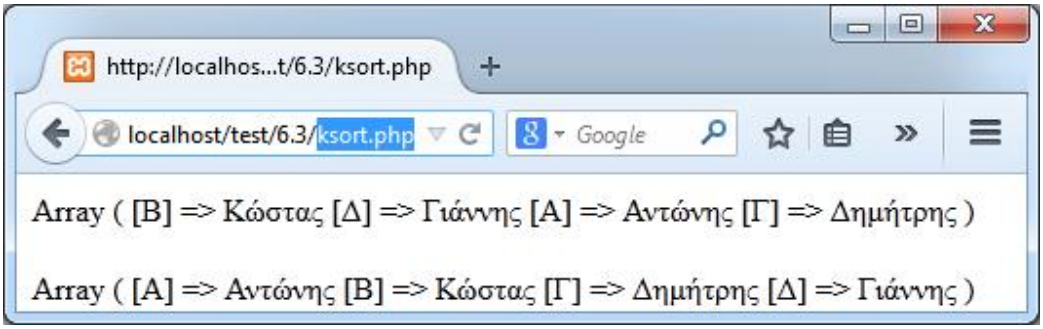
Α.7.5.5 Αύξουσα ταξινόμηση κλειδιών – ksort()

Όπως μπορούμε να ταξινομήσουμε πίνακες με βάση τις τιμές τους, έτσι μπορούμε να ταξινομήσουμε και πίνακες με βάση τα κλειδιά τους. Η «ksort()» πραγματοποιηθεί ταξινόμηση κλειδιών κατά αύξουσα σειρά.

Αρχείο: ksort.php

```
<?php
 $pinakas = array(
  "Β" => "Κώστας",
  "Δ" => "Γιάννης",
  "Α" => "Αντώνης",
  "Γ" => "Δημήτρης"
 );

 print_r($pinakas); //Τύπωση αρχικού πίνακα
 echo "<br/><br/>"; //κενός χώρος
 ksort($pinakas); //ταξινόμηση δεικτών διατηρώντας τις τιμές
 print_r($pinakas); //εκτύπωση πίνακα
?>
```

Αποτέλεσμα:



Εικόνα Α.152 Αύξουσα ταξινόμηση κλειδιών με την ksort()

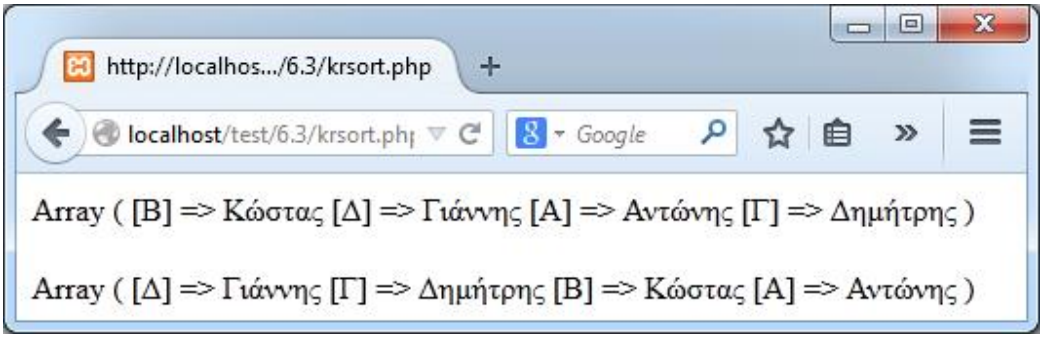
A.7.5.6 Φθίνουσα ταξινόμηση κλειδιών – krsort()

Όπως μπορούμε να ταξινομήσουμε πίνακες με βάση τις τιμές τους, έτσι μπορούμε να ταξινομήσουμε και πίνακες με βάση τα κλειδιά τους. Η «krsort()» πραγματοποιηθεί ταξινόμηση κλειδιών κατά φθίνουσα σειρά.

Αρχείο: krsort.php

```
<?php
 $pinakas = array(
  "Β" => "Κώστας",
  "Δ" => "Γιάννης",
  "Α" => "Αντώνης",
  "Γ" => "Δημήτρης"
 );

 print_r($pinakas); //Τύπωση αρχικού πίνακα
 echo "<br/><br/>"; //κενός χώρος
 krsort($pinakas); //ταξινόμηση δεικτών διατηρώντας τις τιμές
 print_r($pinakas); //εκτύπωση πίνακα
?>
```

Αποτέλεσμα:



Εικόνα Α.153 Φθίνουσα ταξινόμηση κλειδιών με την krsort()

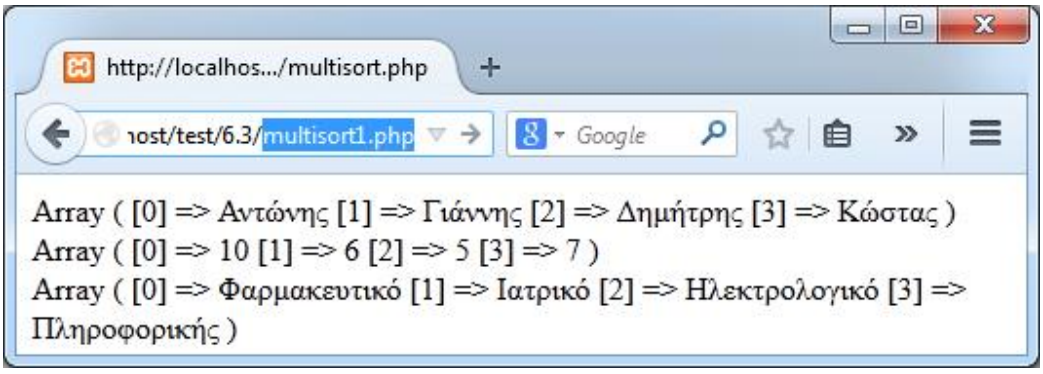
A.7.5.7 Πολλαπλή ταξινόμηση πινάκων – array_multisort()

Η «array_multisort()» μας δίνει την δυνατότητα να μπορέσουμε να ταξινομήσουμε πολλούς πίνακες ταυτόχρονα με βάση τον πρώτο που θα εισάγουμε. Γίνεται δηλαδή ταξινόμηση του πρώτου πίνακα και τα αντιστοιχιζόμενα στοιχεία στους υπόλοιπους πίνακες μετακινούνται με βάση την θέση που άλλαξαν τα στοιχεία του πρώτου πίνακα. Αν θεωρήσουμε πως για τον παρακάτω πίνακα ο «"Κώστας"» έχει βαθμό «7» και βρίσκεται στο τμήμα «"Πληροφορικής"» τότε θα ταξινομηθεί το όνομα του και τα υπόλοιπα στοιχεία θα πάρουν την ανάλογη θέση που βρίσκεται και το όνομα. Η αντιστοίχηση γίνεται κατά αύξουσα από προεπιλογή.

Αρχείο: multisort1.php

```php
<?php
 $onoma = array("Κώστας","Γιάννης","Δημήτρης","Αντώνης");
 $bathmos = array(7,6,5,10);
 $tmhma =
array("Πληροφορικής","Ιατρικό","Ηλεκτρολογικό","Φαρμακευτικό");

 array_multisort($onoma,$bathmos,$tmhma);
 print_r($onoma);
 echo "<br />";
 print_r($bathmos);
 echo "<br />";
 print_r($tmhma);
 echo "<br />";
?>
```

Αποτέλεσμα:



Εικόνα Α.154 Ταξινόμηση πολλαπλών πινάκων με την array_multisort()

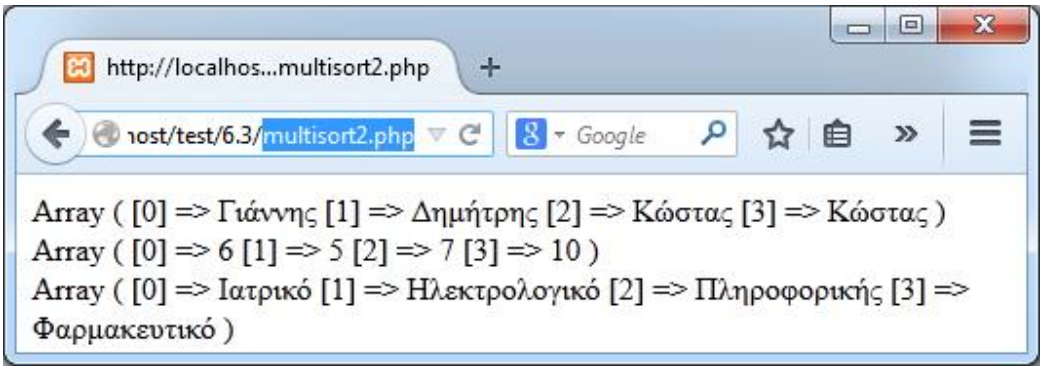
Όπως βλέπουμε ο «"Κώστας"» βρίσκεται 4ος όπως επίσης και ο βαθμός του αλλά και το τμήμα του. Δεν ταξινομούνται δηλαδή οι υπόλοιποι πίνακες κατά αύξουσα αλλά μόνο ο πρώτος και οι υπόλοιποι αντιστοιχίζονται.

Αν ο πρώτος πίνακας διαθέτει τιμές που είναι πανομοιότυπες … π.χ. υπάρχουν δύο Κώστηδες, τότε η «array_multisort» στο σημείο των δυο πανομοιότυπων θα ταξινομήσει με βάση τον δεύτερο πίνακα.

Αρχείο: multisort2.php

```php
<?php
 $onoma = array("Κώστας","Γιάννης","Δημήτρης","Κώστας");
 $bathmos = array(7,6,5,10);
 $tmhma =
array("Πληροφορικής","Ιατρικό","Ηλεκτρολογικό","Φαρμακευτικό");

 array_multisort($onoma,$bathmos,$tmhma);
 print_r($onoma);
 echo "<br />";
 print_r($bathmos);
 echo "<br />";
 print_r($tmhma);
 echo "<br />";
?>
```

Αποτέλεσμα:



Εικόνα Α.155 Ταξινόμηση πολλαπλών πινάκων με όμοια στοιχεία.

Στο σημείο των δύο Κώστηδων η ταξινόμηση συνέχισε με βάση τον βαθμό τους και έτσι ταξινομήθηκε πρώτα αυτός με τον μικρότερο και ύστερα αυτός με τον μεγαλύτερο (αύξουσα από προεπιλογή).

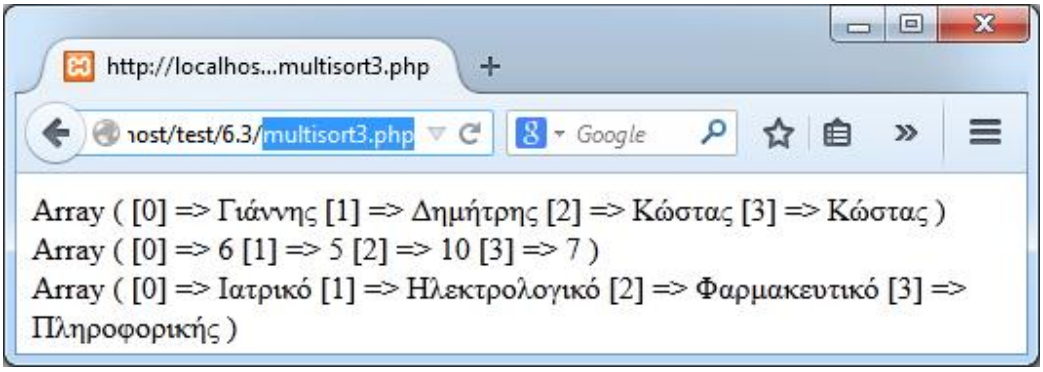
Αν θέλουμε να αλλάξουμε τρόπο με τον οποίο γίνεται η ταξινόμηση από αύξουσα σειρά σε φθίνουσα, τότε μπορούμε να χρησιμοποιήσουμε ένα προαιρετικό στοιχείο μετά από τον κάθε πίνακα στην εντολή «array_multisort».

- «**SORT_ASC**», ταξινομεί κατά αύξουσα σειρά (προεπιλογή,
- «**SORT_DESC**», ταξινομεί κατά φθίνουσα σειρά.

Αρχείο: multisort3.php

```php
<?php
 $onoma = array("Κώστας","Γιάννης","Δημήτρης","Κώστας");
 $bathmos = array(7,6,5,10);
 $tmhma =
array("Πληροφορικής","Ιατρικό","Ηλεκτρολογικό","Φαρμακευτικό");

 array_multisort($onoma, SORT_ASC, $bathmos, SORT_DESC , $tmhma);
 print_r($onoma);
 echo "<br />";
 print_r($bathmos);
 echo "<br />";
 print_r($tmhma);
 echo "<br />";
?>
```

Αποτέλεσμα:



Εικόνα Α.156 Διαφορετικό είδος ταξινόμησης σε πολλαπλούς πίνακες

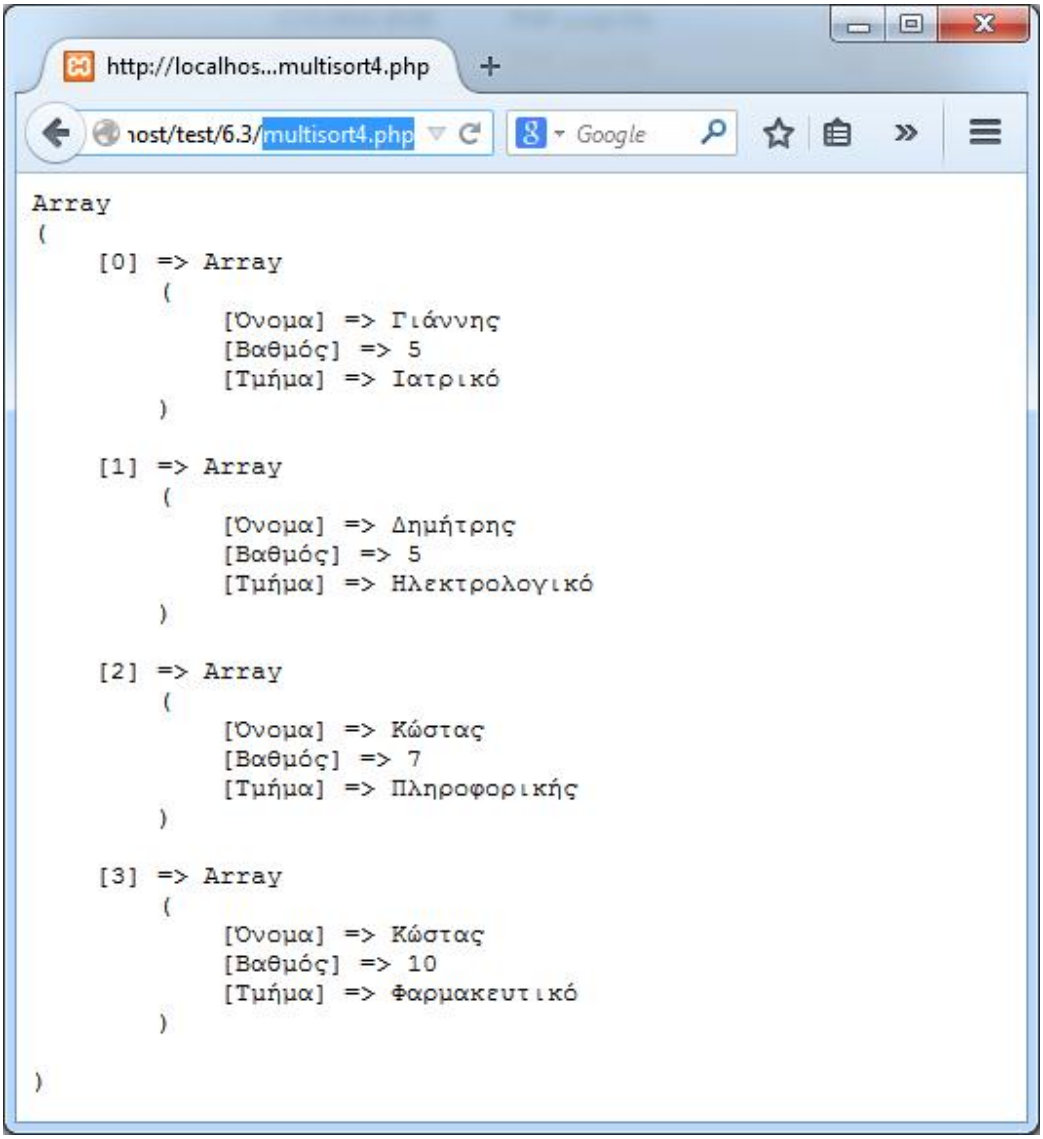
Εδώ ο Κώστας με τον μεγαλύτερο βαθμό ταξινομήθηκε πρώτος ενώ αυτός με τον μικρότερο δεύτερος λόγω του ότι χρησιμοποιήσαμε φθίνουσα ταξινόμηση στον δεύτερο πίνακα.

Η «array_multisort» μας δίνει την δυνατότητα επίσης να ταξινομήσουμε πολυδιάστατους πίνακες. Η συνάρτηση ταξινομεί τα στοιχεία βάση του πρώτου στοιχείου κάθε ένθετου πίνακα. Αν υπάρξουν όμοια στοιχεία, τότε συνεχίζει την ταξινόμηση με βάση το δεύτερο στοιχείο και πάει λέγοντας.

Αρχείο: multisort4.php

```php
<?php
 $foithtes = array(
  array(
   'Όνομα' => "Κώστας",
   'Βαθμός' => 7,
   'Τμήμα' => 'Πληροφορικής'
  ),
  array(
   'Όνομα' => "Γιάννης",
   'Βαθμός' => 5,
   'Τμήμα' => 'Ιατρικό'
  ),
  array(
   'Όνομα' => "Δημήτρης",
   'Βαθμός' => 5,
   'Τμήμα' => 'Ηλεκτρολογικό'
  ),
  array(
   'Όνομα' => "Κώστας",
   'Βαθμός' => 10,
   'Τμήμα' => 'Φαρμακευτικό'
  )
 );

 array_multisort($foithtes);
 echo "<pre>";
 print_r($foithtes);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.157 Ταξινόμηση πολυδιάστατου πίνακα

Α.8 ΑΝΤΙΚΕΙΜΕΝΑ

Κάθε μέρα στην ζωή μας χρησιμοποιούμε αντικείμενα. Ένα αυτοκίνητο, μια οδοντόβουρτσα, ένα κινητό τηλέφωνο, όλα αυτά είναι αντικείμενα που αλληλεπιδρούμε καθημερινά μαζί τους. Κάθε αντικείμενο έχει κάποιες λειτουργίες και κάποια χαρακτηριστικά, π.χ. ο αριθμός πορτών του αυτοκινήτου, το χρώμα του, τα κυβικά του, η τελική του ταχύτητα, η έναρξη του, το φρενάρισμα, η αλλαγή ταχύτητας κ.ο.κ.

Ένα αντικειμενοστραφές λογισμικό έχει φτιαχτεί έτσι ώστε να μπορούν να δημιουργηθούν αυτόνομα σύνολα αντικειμένων με ιδιότητες και λειτουργίες. Όπου αναφερόμαστε σε ιδιότητες και χαρακτηριστικά τότε αναφερόμαστε σε μεταβλητές, ενώ αντίστοιχα όπου υπάρχουν λειτουργίες τότε μιλάμε για μεθόδους. Όλες οι μέθοδοι ενός αντικειμένου αποτελούν το περιβάλλον του.

Έτσι ένα μεγάλο πρόβλημα μπορεί να χωριστεί σε επιμέρους μικρότερα προβλήματα καθιστώντας έτσι ευκολότερη την αντιμετώπιση και διαχείριση του.

Α.8.1 Κλάσεις

Οι κλάσεις είναι κάτι σαν το «καλούπι» στο οποίο τα αντικείμενα μας θα πάρουν την αντίστοιχη μορφή όταν θα δημιουργηθούν. Εδώ Αποφασίζουμε για το πώς θα ονομάζεται ο τύπος του αντικειμένου μας, τι χαρακτηριστικά (ιδιότητες) θα έχει καθώς και ποια θα είναι η συμπεριφορά του (δηλ. οι λειτουργίες του).

Μια κλάση στον προγραμματισμό αντιπροσωπεύει ένα σύνολο οντοτήτων ή αντικειμένων είτε είναι πραγματικά είτε θεωρητικά. Συνήθως χρησιμοποιούμε τις κλάσεις όταν πρόκειται να προσπελάσουμε πολλαπλά από αυτά τα αντικείμενα έτσι ώστε να είναι πιο εύκολα διαχωρίσιμα. Έτσι η κλάση «άνθρωπος» αναφέρεται σε πολλούς ανθρώπους, αλλά χρησιμοποιείται σε ονομαστική ενικού. Χρησιμοποιώντας αυτόν τον τρόπο μπορούμε να δείχνουμε την σχέση μεταξύ αλληλεπιδρώμενων αντικειμένων διευκολύνοντας κατά πολύ τον τρόπο της ανάπτυξης καθώς και την μελλοντική συντήρηση ή επέκταση. Μπορούμε έτσι να αλλάξουμε εύκολα την δομή ενός αντικειμένου προσθέτοντας νέες λειτουργίες ,αφαιρώντας ή προσθέτοντας ιδιότητες ή να διορθώσουμε λάθη χωρίς να χρειάζεται να αλλάξει το περιβάλλον του.

Οι μεταβλητές της κλάσης αντιπροσωπεύουν χαρακτηριστικά ή ιδιότητες της οντότητας αντικειμένου που αντιπροσωπεύει, ενώ όπως είπαμε και προηγουμένως οι μέθοδοι της αντιπροσωπεύουν τις λειτουργίες του. Ας δούμε ένα μικρό παράδειγμα:

- Οντότητα: Φοιτητής
- Ιδιότητες: Όνομα, Επώνυμο, Εξάμηνο, Χρωστούμενα μαθήματα κ.α.
- Λειτουργίες: Δήλωση μαθημάτων, Συμμετοχή σε εργασία, κ.α.

Η οντότητα φοιτητής με όνομα «Κώστας» είναι μέλος της κλάσης Φοιτητής με τα χαρακτηριστικά Όνομα, Επώνυμο, Εξάμηνο, Χρωστούμενα μαθήματα, και λειτουργίες Δήλωση μαθημάτων , Συμμετοχή σε εργασία κ.α.

Η διαδικασία με την οποία αναπτύσσουμε αντικείμενα στον αντικειμενοστραφή προγραμματισμό έχει μια συγκεκριμένη σειρά προκειμένου να λειτουργήσει σωστά.

Αρχικά πρέπει να αποφασίσουμε πως θα ονομάζεται η ομάδα αντικειμένων που έχουμε σκοπό να δημιουργούμε κάθε φορά, και με βάση αυτό το όνομα να κατασκευάσουμε το «καλούπι» με το οποίο θα τα δημιουργούμε. Το καλούπι αυτό στις γλώσσες προγραμματισμού ονομάζεται τύπος αντικειμένων και αντικατοπτρίζεται από το περιεχόμενο μιας κλάσης (class). Μια κλάση περιλαμβάνει όλα αυτά τα χαρακτηριστικά (ιδιότητες και λειτουργίες) που καθιστούν το αντικείμενο, αυτό που είναι. Οπότε το όνομα της κλάσης είναι και το όνομα του τύπου αντικειμένων που έχουμε σκοπό να δημιουργήσουμε. Αφού κατασκευάσουμε πλήρως την κλάση και συμπεριλάβουμε μέσα σε αυτήν και τους ανάλογους κατασκευαστές, τότε για να δημιουργήσουμε ένα αντικείμενο βασιζόμενοι στον συγκεκριμένο τύπο, αρκεί να καλέσουμε το όνομα της κλάσης, δίνοντας αντίστοιχα ορίσματα για τον κατασκευαστή της. Η γραμμή κώδικα που διαθέτει όλα αυτά, ονομάζεται στιγμιότυπο αντικειμένου, και είναι η στιγμή κατά την οποία ένα αντικείμενο δημιουργείται βασιζόμενο στο καλούπι του (κλάση).

1. Αρχικά αποφασίζουμε πως θα ονομάσουμε την ομάδα αντικειμένων μας.
2. Μετά, δημιουργούμε την κλάση μας συμπεριλαμβάνοντας μέσα όλα τα χαρακτηριστικά (ιδιότητες και λειτουργίες) των αντικειμένων μας.
3. Ύστερα δημιουργούμε ένα αντικείμενο βασιζόμενοι στον νέο τύπο που δημιουργήσαμε.

A.8.1.1 Δημιουργία κλάσης

Για να δημιουργήσουμε μια απλή κλάση χρησιμοποιούμε την εξής σύνταξη:

```
<?php
 class όνομα {
 }
?>
```

Χρησιμοποιούμε την λέξη κλειδί «class» και όπου όνομα βάζουμε το όνομα που θέλουμε να έχει η κλάση μας χρησιμοποιώντας λατινικά γράμματα.

```
<?php
 class oxima{
 }
?>
```

Για να είναι χρήσιμες οι κλάσεις θα πρέπει να έχουν ιδιότητες (=μεταβλητές) και λειτουργίες (=μέθοδοι).

```
<?php
 class oxima{
  var $montelo;
  var $kybika;
  function leitourgia(){

  }
 }
?>
```

Οι ιδιότητες είναι τα $montelo και $kybika, και η λειτουργία η leitourgia() η οποία δεν κάνει απολύτως τίποτα.

A.8.1.2 Ενθυλάκωση

Στις αντικειμενοστραφής γλώσσες προγραμματισμού υποστηρίζεται η ενθυλάκωση η οποία είναι γνωστή και ως απόκρυψη δεδομένων.

Η πρόσβαση στα δεδομένα ενός αντικειμένου γίνεται μόνο μέσα από το περιβάλλον του. Η ενθυλάκωση πραγματοποιείται μέσω των εντολών δήλωσης πρόσβασης , και έτσι εξασφαλίζεται πως μια κλάση δεν θα έχει άμεση πρόσβαση στα δεδομένα μιας άλλης κλάσης, αλλά αυτά θα «φιλτράρονται» χρησιμοποιώντας κάποια δημόσια λειτουργία της δεύτερης.

Εντολή	Περιγραφή	Παράδειγμα
public	Δημόσια	**public** $rodes; **public function** ekkinhsh(){}
private	Ιδιωτικό	**private** $rodes; **private function** ekkinhsh(){}

Πίνακας Α.25 Πρόσβαση στοιχείων αντικειμένου

Για παράδειγμα:

```php
<?php
 class oxima{
  private $montelo;
  public $kybika;
  public function leitourgia(){

  }
 }
?>
```

Η ιδιότητα $montelo είναι προσπελάσιμη μόνο από την κλάση oxima ενώ η ιδιότητα $kybika και η λειτουργία leitourgia() είναι προσπελάσιμες από οποιαδήποτε άλλη κλάση.

A.8.2 Κατασκευαστές και καταστροφείς

Εφόσον δημιουργηθούν οι κλάσεις θα πρέπει με κάποιο τρόπο να «πούμε» στο script ποια αντικείμενα θα δημιουργηθούν, και για τα αντικείμενα αυτά, ποίες κλάσεις θα χρησιμοποιηθούν . Για να δημιουργηθεί ένα αντικείμενο μέσα σε μια κλάση χρησιμοποιείται ένας κατασκευαστής. Αντίστοιχα όταν πλέον ένα αντικείμενο παύει να χρησιμοποιείται, παύει αυτομάτως και να υφίστανται Στο τέλος της ύπαρξης τους ένας καταστροφέας αναλαμβάνει δράση.

A.8.2.1 Κατασκευαστές

Ο κατασκευαστής αναλαμβάνει να εισάγει τιμές στις ιδιότητες της κλάσης (δηλαδή τις μεταβλητές της), δίνοντας τους έτσι τα μοναδικά χαρακτηριστικά ενός αντικειμένου. Την στιγμή δηλαδή που δημιουργείται το αντικείμενο τότε καλείται ο κατασκευαστής της κλάσης και αυτό που κάνει είναι να αντιστοιχίζει τιμές στις μεταβλητές της.

Ένας κατασκευαστής στην php5 δηλώνεται με την δεσμευμένη λέξη «__construct», ενώ στις παλαιότερες εκδόσεις php ένας κατασκευαστής αρκούσε να έχει το ίδιο όνομα με την κλάση. Η σύνταξη του έχει ως εξής:

```
function __construct(παράμετροι){
 ...εντολές...
}
```

Όπως βλέπουμε παραπάνω ο κατασκευαστής έχει πολλές ομοιότητες με μία μέθοδο:

1. Πρέπει να έχει την λέξη «function» πριν από το «__construct»,

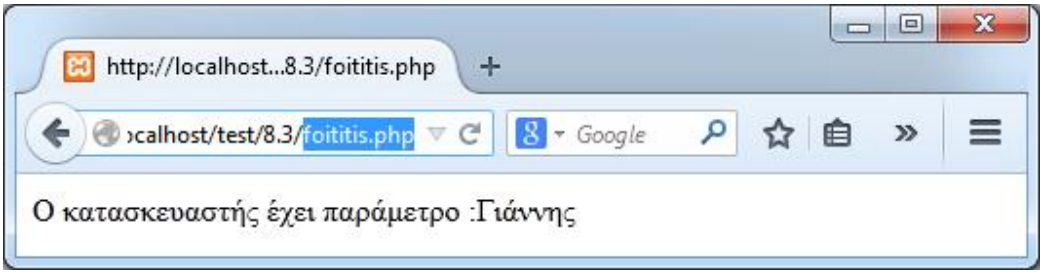
2. Παίρνει παραμέτρους μέσα στην παρένθεση η οποία ακολουθείται του «__construct»,
3. Όλες οι εντολές του πλαισιώνονται από αγκύλες.
4. Δεν έχει «return».

Ας δούμε ένα παράδειγμα κατασκευαστή:

Αρχείο: foititis.php

```
<?php
 class foititis{
  private $onoma="";
  function __construct($onom="Χωρίς όνομα"){
   $this->onoma = $onom;
   echo "Ο κατασκευαστής έχει παράμετρο :".$onom;
  }
 }

 $a1 = new foititis("Γιάννης");
?>
```

Αποτέλεσμα:



Εικόνα Α.158 Δημιουργία και τύπωση αντικειμένου με την php

- πρώτα συντάσσεται η κλάση και ύστερα χρησιμοποιείται,
- το «foititis» είναι η κλάση που αντιπροσωπεύει φοιτητές,
- το «$onoma» είναι μια ιδιότητα της κλάσης που αντιπροσωπεύει το όνομα του κάθε φοιτητή,
- το «__construct» είναι ο κατασκευαστής του φοιτητή ο οποίος έχει μία παράμετρο την «$onom», στην οποία αν δεν δοθεί κάποια τιμή γι αυτήν κατά την διάρκεια που καλέσουμε τον κατασκευαστή θα δοθεί η προεπιλεγμένη τιμή «Χωρίς όνομα».
- το «$this» είναι μια δεσμευμένη εντολή που χρησιμοποιείται για να έχουμε πρόσβαση μέσα από την κλάση στις ιδιότητες και της λειτουργίες της. Στην

συγκεκριμένη γραμμή ζητάμε να δοθεί η τιμή της παραμέτρου «$onom» του κατασκευαστή στην ιδιότητα της κλάσης «foititis», «$onoma».

- η γραμμή «$a1 = new foititis("Γιάννης");» καλείται στιγμιότυπο του αντικειμένου, και είναι η στιγμή που το script θα καλέσει τον κατασκευαστή από την κλάση «foititis» και θα δημιουργήσει το αντικείμενο.

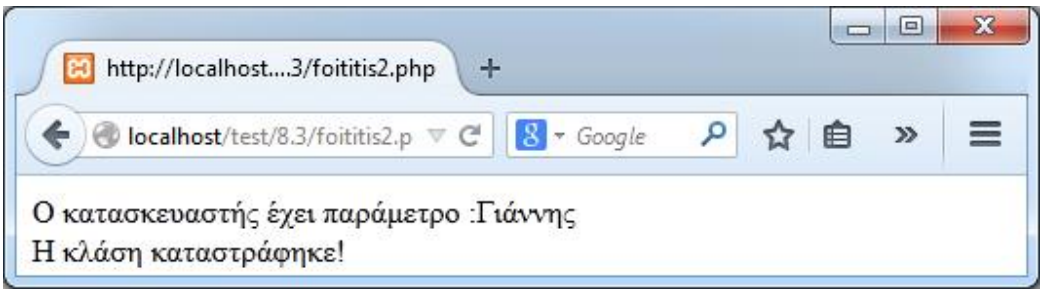
A.8.2.2 Καταστροφείς

Το αντίθετο από αυτό που είναι ένας κατασκευαστής, είναι ο καταστροφέας. Οι μέθοδοι καταστροφής είναι νέες στην php5, και επιτρέπουν την εκτέλεση κάποιο μέρους κώδικα πριν καταστραφεί ολόκληρη η κλάση. Όταν οι αναφορές προς μια κλάση (δηλαδή όλα της τα αντικείμενα) έχουν ακυρωθεί ή φύγουν από το πεδίο δράσης, τότε αυτόματα καλείται ο καταστροφέας της κλάσης και εκτελείται το μέρος του κώδικα που υπάρχει μέσα σε αυτόν. Χρησιμοποιείται η δεσμευμένη λέξη «__destruct» ως μέθοδος παρόμοια με τον κατασκευαστή.

Αρχείο: foititis2.php

```php
<?php
 class foititis{
  private $onoma="";
  function __construct($onom="Χωρίς όνομα"){
   $this->onoma = $onom;
   echo "Ο κατασκευαστής έχει παράμετρο :".$onom;
  }
  function __destruct(){
    echo "<br /> Η κλάση καταστράφηκε!";
  }
 }

 $a1 = new foititis("Γιάννης");
?>
```

Αποτέλεσμα:



Εικόνα A.159 Κατασκευή και καταστροφή κλάσης.

Ο καταστροφέας εκτελείται στο τέλος της χρήσης της κλάσης, όταν δηλαδή σταματάμε να χρησιμοποιούμε τα στοιχεία της και πλέον είναι ελεύθερη από εμάς να αποσυρθεί.

Α.8.3 Αντικείμενα

Τα αντικείμενα μας βοηθούν –πολύ- στην ανάπτυξη ενός λογισμικού. Η έννοια τους μπορεί να είναι θεωρητική αλλά ο τρόπος αντιμετώπισης στην πράξη και ιδιαίτερα στην συγγραφή του κώδικα είναι αυτό που κάνει την αντικειμενοστραφή ανάπτυξη λογισμικού ακόμη πιο χρήσιμη και ενδιαφέρουσα.

Για να δημιουργηθεί ένα αντικείμενο θα πρέπει να στηριχθεί στις ιδιότητες και τις λειτουργίες κάποιας κλάσης. Από κάθε κλάση μπορούν να δημιουργηθούν ένα ή και περισσότερα αντικείμενα, ενδεχομένως και κανένα (αλλά αυτό δεν βγάζει νόημα).

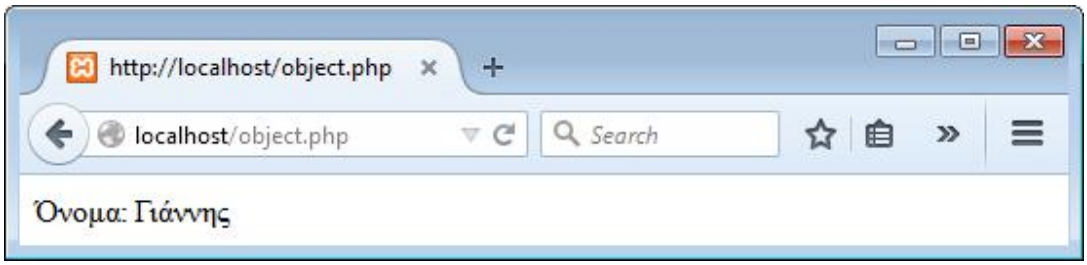
Η στιγμή που θέλουμε να δημιουργήσουμε το αντικείμενο μας στον κώδικα γίνεται με την εντολή «new». Όταν θέλουμε να χρησιμοποιήσουμε μεταβλητές ή λειτουργίες της κλάσης από ΤΗΝ ΙΔΙΑ την κλάση χρησιμοποιούμε την δεσμευμένη λέξη «$this» ακολουθουμένη με τον τελεστή «->». Αν θέλουμε για παράδειγμα να χρησιμοποιήσουμε το όνομα του φοιτητή μέσα από την ίδια την κλάση σε κάποια συνάρτηση το φωνάζουμε ως «$this->onoma;»

Ας δούμε την χρήση των εσωτερικών μεταβλητών ενός αντικειμένου μέσα από ένα παράδειγμα.

Αρχείο: object.php

```php
<?php
 class student{
  private $name="";
  function __construct($n1="Χωρίς όνομα"){
   $this->name = $n1;
  }

  function get_name(){
   return $this->name;
  }
 }

 $a1 = new student("Γιάννης");
 echo "Όνομα: ".$a1->get_name();
?>
```

Αποτέλεσμα:



Εικόνα Α.160 Χρήση ιδιοτήτων ενός αντικειμένου μέσω μεθόδων

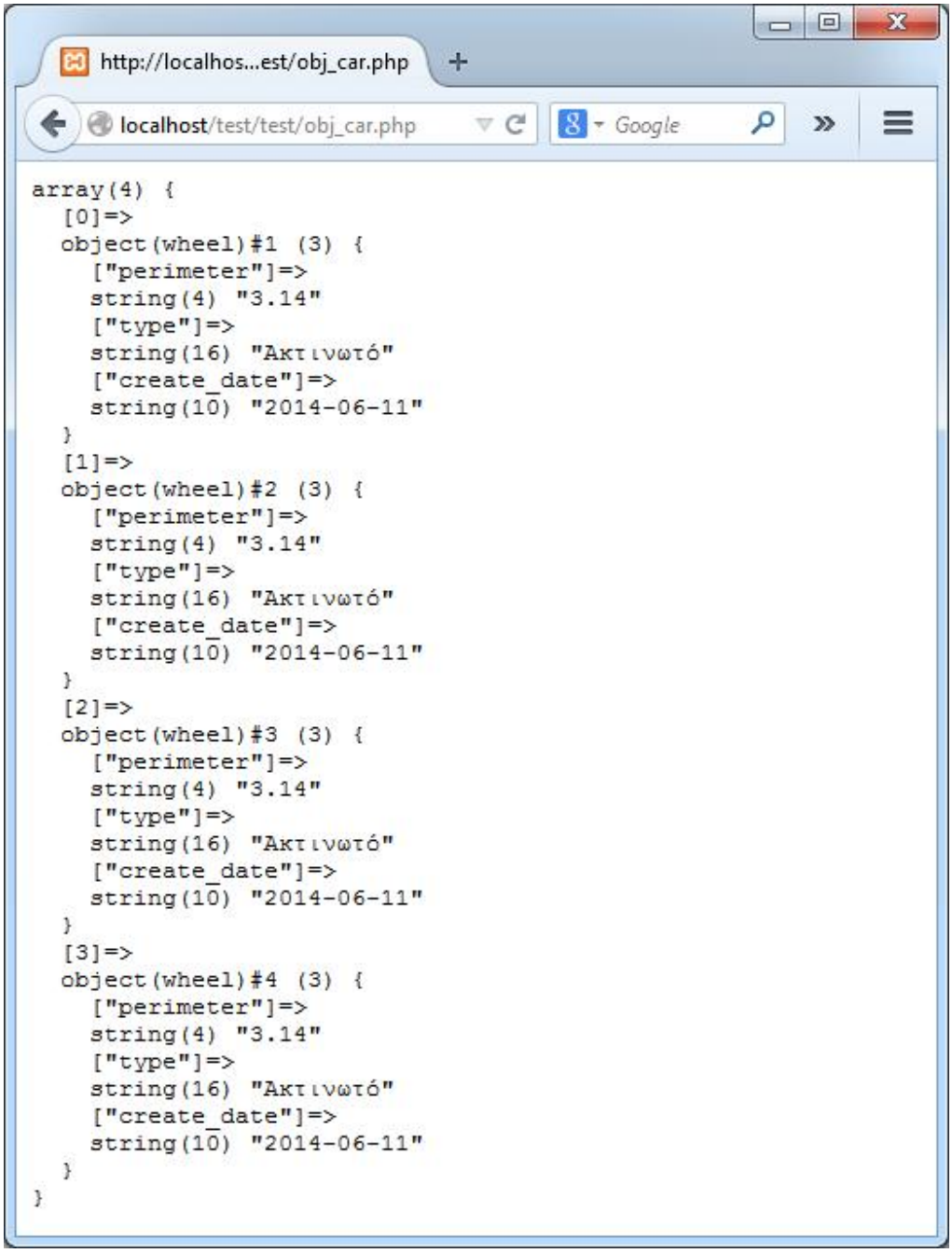
Α.8.3.1 Αντικείμενα μέσα σε αντικείμενα

Αφού δημιουργήσουμε μια κλάση και την εισάγουμε μέσα σε ένα αρχείο php τότε η κλάση η συγκεκριμένη μπορεί να χρησιμοποιηθεί ως τύπος μεταβλητής. Αυτό σημαίνει ότι μπορεί να χρησιμοποιηθεί σε εισόδους/εξόδους μεθόδων, μέσα σε άλλες κλάσεις, και άλλα σχετικά. Ας δούμε πως ένα αντικείμενο εξαρτάται από άλλα αντικείμενα, μέσα από ένα παράδειγμα:

Αρχείο: obj_car.php

```php
<?php
 class wheel{
  public $perimeter;
  public $type;
  public $create_date;

  function __construct($p1,$t1,$c1){
    $this->perimeter=$p1;
    $this->type=$t1;
    $this->create_date=$c1;
  }
 }

 class car{
  public $wheels;
  public $doors;
  public $model;
  public $price;
  function __construct($w1,$d1,$m1,$p1){
   $this->wheels = $w1;
   $this->doors = $d1;
   $this->model = $m1;
   $this->price = $p1;
  }
}
 $roda;
 for($i=0;$i<4;$i++){
  $roda[$i] = new wheel("3.14","Ακτινωτό","2014-06-11");
 }

 $a = new car($roda,5,"Fiat 2008",1000.52);

 echo "<pre>";
 var_dump($a->wheels);
 echo "</pre>";
?>
```

Αποτέλεσμα:



Εικόνα Α.161 Χρήση αντικειμένων μέσα σε αντικείμενα

Όπως παρατηρούμε με το παραπάνω βλέπουμε πως το αντικείμενο «car» μέσα στις ιδιότητες του έχει τέσσερα αντικείμενα «wheel». Γενικώς στον προγραμματισμό ισχύει πως όσο περισσότερο διασπάμε ένα πρόβλημα σε μικρότερα υπό-προβλήματα, τότε τόσο πιο εύκολα επιλύσιμο είναι συνολικά το πρόβλημα αυτό. Διαχωρίζοντας έτσι ένα αυτοκίνητο σε επιμέρους αντικείμενα όπως οι ρόδες, οι πόρτες κτλ. αυτό μας

βοηθά έτσι ώστε μια αλλαγή που επηρεάζει μόνο ένα από αυτά π.χ. τις ρόδες, τότε να μπορεί να γίνει εύκολα και χωρίς μεγάλη δυσκολία.

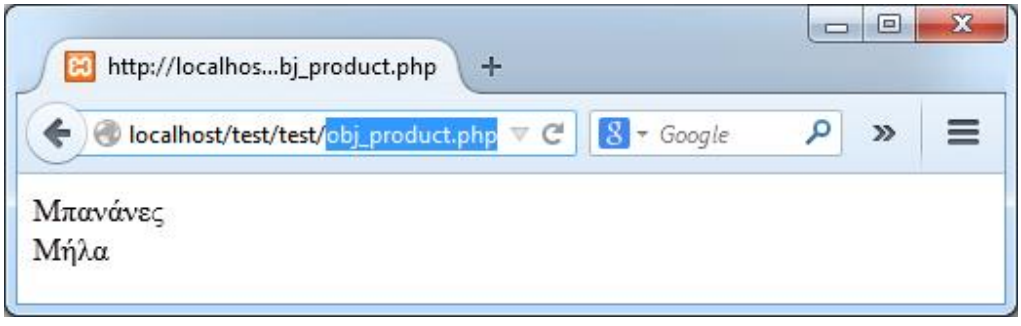
A.8.3.2 Πρόσβαση σε ιδιότητες private

Είναι πιθανό να έχουμε private μεταβλητές μέσα σε μια κλάση, και να έχουμε πρόσβαση σε αυτές έξω από την κλάση μέσω μεθόδων. Ας δούμε πως γίνεται αυτό:

Αρχείο: obj_product.php

```
<?php
 class product{
  private $name;
  private $price;

  function __construct($n1,$p1){
    $this->name=$n1;
    $this->type=$p1;
  }

  function get($name){
   return $this->$name;
  }

  function change($name,$value){
   $this->$name=$value;
  }
 }

 $a = new product("Μπανάνες",3.2);

 echo $a->get("name")."<br/>";
 $a->change("name","Μήλα");
 echo $a->get("name");
?>
```

Αποτέλεσμα:



Εικόνα Α.162 Πρόσβαση σε όλες τις private ιδιότητες μια κλάσης μέσω λειτουργιών

Όπως βλέπουμε στο παραπάνω κώδικα, η μέθοδος «get» παίρνει ένα όρισμα, το όνομα που θέλουμε να χρησιμοποιήσουμε. Όπως είναι γνωστό όλες οι μέθοδοι μιας κλάσης

έχουν δικαίωμα πρόσβασης στις private μεταβλητές τις. Εκμεταλλευόμαστε αυτό το πλεονέκτημα στην ανάκτηση μιας ιδιότητας της κλάσης που δεν μπορούμε άμεσα να προσπελάσουμε.

Παρομοίως η μέθοδος «change» χρειάζεται δυο παραμέτρους. Το όνομα της ιδιότητας και μια τιμή γι' αυτήν. Ζητώντας με τον ίδιο τρόπο την μεταβλητή, μέσα από την κλάση, μας δίνεται η δυνατότητα να αλλάξουμε την τιμή της χωρίς να την προσπελάσουμε άμεσα, αλλά μέσω της μεθόδου αυτής.

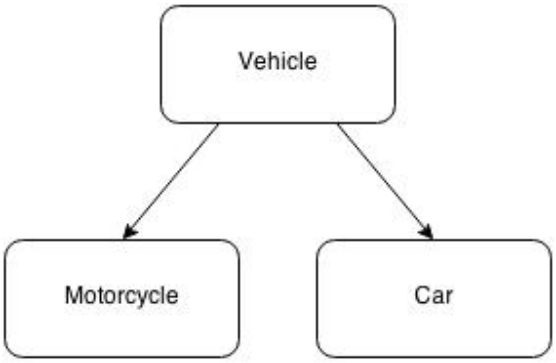
Το αντικείμενο αρχικά είχε ως ιδιότητα ονόματος «Μπανάνες» και ύστερα αυτό γίνεται «Μήλα» μέσω της χρήσης των μεθόδων αυτών.

A.8.4 Υπερκλάσεις και υποκλάσεις

Οι υπερκλάσεις και υποκλάσεις είναι ένας τρόπος ιεράρχησης των κλάσεων σε σχέση με την συγγένεια που διαθέτουν οι κλάσεις μεταξύ τους. Σε αυτή την ενότητα ανακαλύπτουμε τι είναι η κληρονομικότητα, καθώς επίσης μαθαίνουμε την πρόσβαση μεταβλητών ανάμεσα στις κλάσεις και την υπερκάλυψη.

A.8.4.1 Κληρονομικότητα

Όπως στην πραγματικότητα ο πατέρας μεταδίδει τα γονίδια του στα παιδιά του, και έτσι τα παιδιά ενός γονιού μοιάζουν σε αυτόν, κάπως έτσι και στον αντικειμενοστραφή προγραμματισμό χρησιμοποιείται η κληρονομικότητα. Μόνο που στην συγκεκριμένη περίπτωση εμείς αποφασίζουμε ποιές ιδιότητες και ποιές λειτουργίες θα κληρονομηθούν.



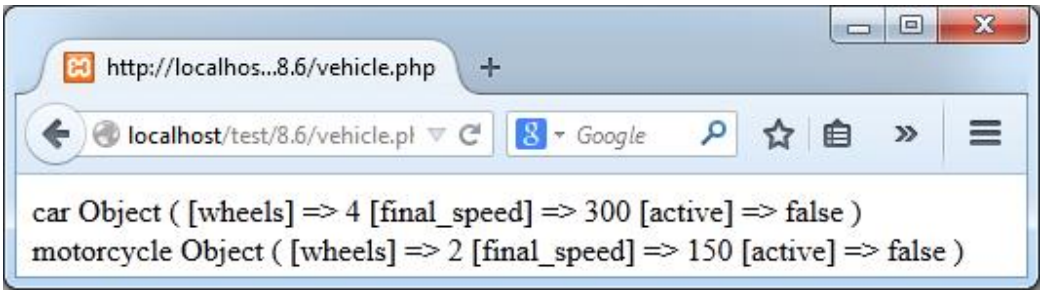
Σχήμα A.3 Κληρονομικότητα αντικειμένων

Ένα όχημα για παράδειγμα γνωρίζουμε ότι διαθέτει ρόδες για να μπορέσει να μετακινηθεί, επίσης γνωρίζουμε ότι θα έχει κάποια τελική ταχύτητα. Γενικεύουμε έτσι όλα τα οχήματα που έχουν ρόδες ως «όχημα»(vehicle). Με αυτή την γενίκευση μπορούμε να ειδικεύσουμε αργότερα σε περεταίρω «κατηγορίες», π.χ. αυτοκίνητο (car) ή μηχανάκι (motorcycle). Ένα αυτοκίνητο γνωρίζουμε πως θα έχει τέσσερις ρόδες (wheels) , ενώ ένα μηχανάκι θα έχει δυο, καθώς η τελική ταχύτητα είναι ανεξάρτητη (max speed) και για τα δυο. Έτσι προκύπτει ο παρακάτω κώδικας:

Αρχείο: vehicle.php

```php
<?php
 class vehicle{
  public $wheels;
  public $final_speed;
  public $active='false';

  function __construct($w,$fs){
   $this->wheels=$w;
   $this->final_speed = $fs;
  }

  function start_engine(){
    $this->active = 'true';
    return true;
  }
 }

 class car extends vehicle{
  function __construct($fs){
   parent::__construct(4,$fs);
  }
 }

 class motorcycle extends vehicle{
  function __construct($fs){
   parent::__construct(2,$fs);
  }

  public function startWheelie(){
   $this->wheels=1;
   return $this->wheels;
  }

  public function endWheelie(){
   $this->wheels=2;
   return $this->wheels;
  }
 }

 $a = new car(300);
 $b = new motorcycle(150);

 print_r($a);
 echo "<br />";
 print_r($b);

 ?>
```

Αποτέλεσμα:



Εικόνα Α.163 Κληρονομικότητα στην php

Όπως παρατηρούμε στον παραπάνω κώδικα υπάρχουν τρείς διαφορετικές κλάσεις οι οποίες όμως έχουν μια «σχέση» μεταξύ τους. Η κλάση «vehicle» είναι η κλάση-πατέρας από την οποία οι υπόλοιπες κλάσεις θα κληρονομήσουν στοιχεία. Η κλάση «car» και «motorcycle», είναι οι κλάσεις-παιδιά που κληρονομούν όλα τα στοιχεία της κλάσης-πατέρα.

- Όταν μία κλάση κληρονομεί στοιχεία μιας άλλης κλάσης, η κλάση-παιδί ονομάζεται: υποκλάση (subclass), παραγόμενη κλάση (derived) ή θυγατρική κλάση (child), ενώ η κλάση-πατέρας: υπερκλάση (superclass), γονική κλάση (parent), ή βασική κλάση (base).
- Για να ορίσουμε ότι μία κλάση είναι υποκλάση μιας υπερκλάσης χρησιμοποιούμε την εντολή «extends».
- Το μέλος που βρίσκεται αριστερά από το «extends» είναι το όνομα της υποκλάσης που θέλουμε να δημιουργήσουμε και το μέλος που βρίσκεται δεξιά είναι το όνομα της υπερκλάσης που θέλουμε από αυτό να κληρονομήσει στοιχεία.
- Μια υποκλάση κληρονομεί όλα τα στοιχεία (ιδιότητες και λειτουργίες) της υπερκλάσης τα οποία έχουν δείκτη προσβασιμότητας protected ή public, αλλά ποτέ δεν κληρονομεί τον κατασκευαστή, μπορεί όμως να έχει πρόσβαση σε αυτόν.
- Οι υποκλάσεις μπορούν να έχουν δικές τους ιδιότητες ή λειτουργίες οι οποίες να μην έχουν καμία σχέση με την υπερκλάση που προέρχονται.
- Μια κλάση μπορεί να κληρονομεί μόνο μια άλλη κλάση. Δεν μπορεί δηλαδή μια κλάση-παιδί να έχει κλάση-πατέρα και κλάση-μητέρα αλλά μόνο ένα από τα δυο.

- Μια υποκλάση μπορεί να γίνει υπερκλάση μιας άλλης υποκλάσης.
- Όταν θέλουμε μέσα από μια υποκλάση να έχουμε πρόσβαση σε ιδιότητες ή λειτουργίες μιας υπερκλάση τότε χρησιμοποιούμε το πρόθεμα «parent::»

A.8.4.2 Υπερκάλυψη

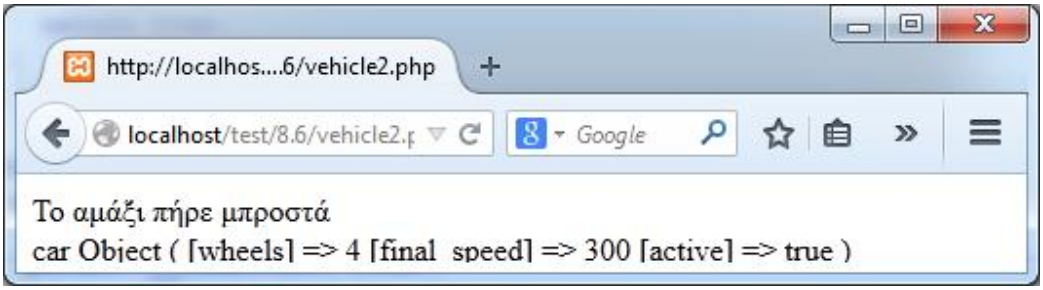
Η υπερκάλυψη είναι μια δυνατότητα των υποκλάσεων να «υπερκαλύπτουν» μια ήδη υπάρχουσα ιδιότητα ή λειτουργία σε μια υπερκλάση κάνοντας παραπάνω λειτουργίες ή προσαρμόζοντας μια ιδιότητα της.

Έτσι αν μια υποκλάση έχει μια μέθοδο με όνομα «start_engine()» η οποία με ακριβώς ίδιο όνομα υπάρχει και στην υπερκλάση, τότε για να χρησιμοποιήσουμε την μέθοδο της υπερκλάσης χρησιμοποιούμε το πρόθεμα «parent::» ενώ αν θέλουμε να χρησιμοποιήσουμε την μέθοδο της υποκλάσης που βρισκόμαστε χρησιμοποιούμε το πρόθεμα «$this->». Ας δούμε το παράδειγμα με τα οχήματα.

Αρχείο: vehicle2.php

```php
<?php
 class vehicle{
  public $wheels;
  public $final_speed;
  public $active='false';

  function __construct($w,$fs){
   $this->wheels=$w;
   $this->final_speed = $fs;
  }

  function start_engine(){ //αυτή είναι η μέθοδος της υπερκλάσης
    $this->active = 'true';
    return true;
  }
 }

 class car extends vehicle{
  public function start_engine(){
   parent::start_engine(); // καλείται η μέθοδος της υπερκλάσης
   echo "Το αμάξι πήρε μπροστά";
  }
  function __construct($fs){
   parent::__construct(4,$fs);
  }
 }

 $a = new car(300);

 $a->start_engine(); //εδώ θα γίνει χρήση της μεθόδου στο car
 echo "<br/>";
 print_r($a);

 ?>
```

Αποτέλεσμα:



Εικόνα Α.164 Υπερκάλυψη λειτουργιών

Εδώ βλέπουμε πως μέσα από μια λειτουργία μιας υποκλάσης μπορούμε να καλέσουμε μια λειτουργία της υπερκλάσης της χρησιμοποιώντας ακριβώς το ίδιο όνομα. Αν δεν δημιουργούσαμε την καινούργια μέθοδο μέσα στο «car» όταν θα καλούσαμε την μέθοδο «$a->start_engine();», τότε το script θα χρησιμοποιούσε την μέθοδο που

βρίσκεται στην υπερκλάση. Τώρα όμως που υπάρχει αντίστοιχη μέθοδος στην υποκλάση, καλείται αυτή. Το αν θα την συνδέσουμε με την μέθοδο της υπερκλάσης είναι θέμα δικό μας θα μπορούσαμε να μην συμπεριλάβουμε το «parent::start_engine();» και έτσι η μέθοδος να κάνει κάτι τελείως διαφορετικό από ότι έκανε η πρόγονος της.

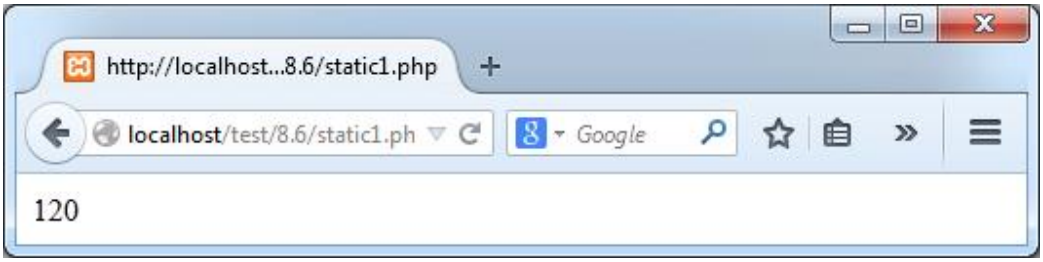
A.8.4.3 Σταθερές

Οι σταθερές (static) μεταβλητές είναι ένας τρόπος να μοιραζόμαστε μια μεταβλητή και τις αλλαγές αυτής, άμεσα με όλες τις υποκλάσεις μιας υπερκλάσης που την διαθέτει. Ας δούμε ένα σύντομο παράδειγμα χρήσης:

Αρχείο: static1.php

```
<?php
 class vehicle{
  static $max_speed = 120;
 }
 echo vehicle::$max_speed;
?>
```

Αποτέλεσμα:



Εικόνα A.165 Χρήση σταθερών μεταβλητών σε κλάσεις

Ας υποθέσουμε πως κάθε όχημα διαθέτει έναν μέγιστο όριο ταχύτητας με το οποίο μπορεί να κινηθεί στους δρόμους της Ελλάδας, και το οποίο είναι κοινό για όλα τα οχήματα. Πριν το 2010 το μέγιστο όριο ταχύτητας με βάση τον κώδικα οδικής κυκλοφορίας ήταν 120 χλμ την ώρα, ενώ με βάση κάποιες αλλαγές που πραγματοποιήθηκαν το 2011 το όριο αυτό άλλαξε σε 130 χλμ την ώρα.

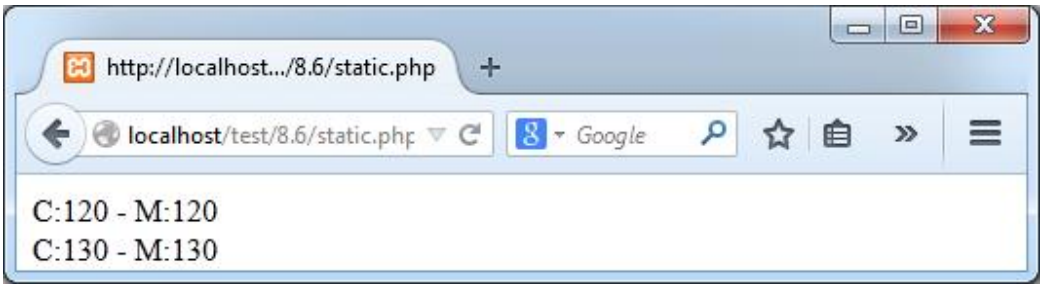
Αν η μεταβλητή που αντικατοπτρίζει αυτόν τον περιορισμό ήταν διαφορετική για κάθε όχημα με βάση την στιγμή της δημιουργίας του τότε όλα τα οχήματα που είχαν δημιουργηθεί από το 2010 και πίσω θα είχαν μέγιστο όριο ταχύτητας τα 120 χλμ/ώρα

ενώ όλα τα υπόλοιπα που δημιουργήθηκαν από το 2011 και μετά θα είχαν διαφορετικό όριο ταχύτητας. Οι στατικές μεταβλητές προσπαθούν να επιλύσουν αυτό το πρόβλημα με την άμεση ισχύ της μεταβλητής σε όλες τις κλάσεις την στιγμή ακριβώς που αλλάζει. Έτσι η αλλαγή σε 130 χλμ/ώρα επηρεάζει όλα τα οχήματα ανεξαιρέτως το πότε δημιουργήθηκαν.

Αρχείο: static.php

```php
<?php
 class vehicle{
  public $wheels;
  public $final_speed;
  public $active='false';
  public static $max_speed=120;

  function __construct($w,$fs){
   $this->wheels=$w;
   $this->final_speed = $fs;
  }

  function get_maxspeed(){
   return static::$max_speed;
  }

  function start_engine(){ //αυτή είναι η μέθοδος της υπερκλάσης
    $this->active = 'true';
    return true;
  }
 }

 class car extends vehicle{
  function __construct($fs){
   parent::__construct(4,$fs);
  }
 }

  class motorcycle extends vehicle{
  function __construct($fs){
   parent::__construct(2,$fs);
  }
 }

 $a = new car(300);
 $b = new motorcycle(150);

 echo "C:".$a->get_maxspeed()." - M:".$b->get_maxspeed()."<br/>";
//τύπωση μέγιστου ορίου

 vehicle::$max_speed=130; //αλλαγή μέγιστου ορίου

 echo "C:".$a->get_maxspeed()." - M:".$b->get_maxspeed()."<br/>";
//τύπωση ξανά

 ?>
```

Αποτέλεσμα:



Εικόνα Α.166 Επίδραση σταθερών μεταβλητών σε αντικείμενα

Όπως βλέπουμε στον κώδικα η λέξη «static» είναι η λέξη κλειδί στην προσπέλαση στατικών μεταβλητών ή και μεθόδων. Τόσο οι στατικές μεταβλητές όσο και οι στατικές μέθοδοι λειτουργούν αυτόνομα και χωρίς να χρειάζεται να έχει δημιουργηθεί κάποιο αντικείμενο για να «τρέξουν». Ο τρόπος με τον οποίο μπορούμε να καλέσουμε μια ιδιότητα ή μια μέθοδο χωρίς να έχει δημιουργηθεί αντικείμενο είναι «όνομα_κλάσης::όνομα_μεθόδου(Ορίσματα);», ως αντικατάσταση στο συγκεκριμένο παράδειγμα χρησιμοποιούμε μια μέθοδο για να αποκτήσουμε πρόσβαση στην στατική ιδιότητα. Για να την χρησιμοποιήσουμε όμως χρειαζόμαστε να την προσπελάσουμε ως στατική «static::$max_speed;».

Α.8.4.4 Διαρκές μεταβλητές

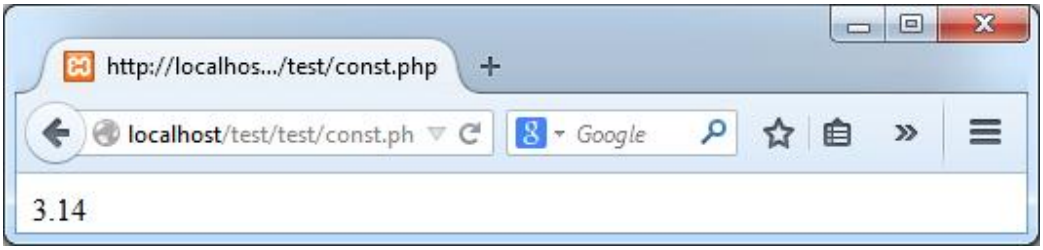
Μια διαρκής μεταβλητή χρησιμοποιείται με παρόμοιο τρόπο όπως η στατική μόνο που δεν μπορεί να είναι private ή protected αλλά μόνο public εξ' ορισμού.

Αυτού του είδους οι μεταβλητές χρησιμοποιούνται για να δηλωθούν τιμές οι οποίες δεν αλλάζουν όπως για παράδειγμα η περίμετρος ενός κύκλου. Επίσης οι διαρκές μεταβλητές δεν έχουν το σύμβολο «$» του δολαρίου και επομένως δεν μπορούν να χρησιμοποιηθούν σε συνθήκες.

Αρχείο: const.php

```
<?php
 class car{
  const pi=3.14;
 }

 echo car::pi;
?>
```

Αποτέλεσμα:



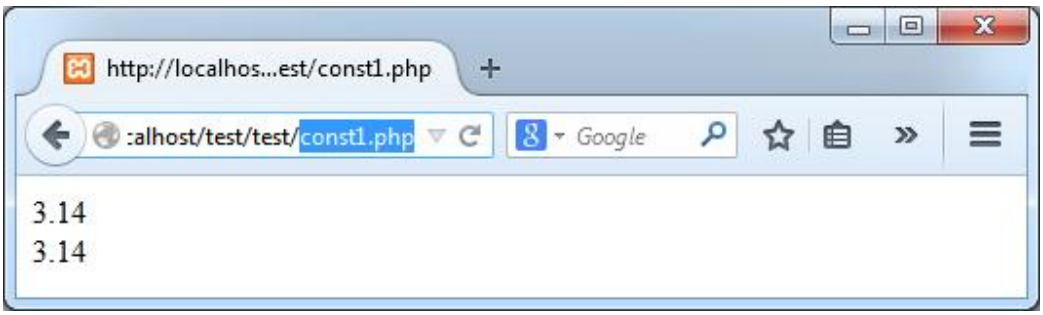
Εικόνα Α.167 Χρήση διαρκών μεταβλητών

Όπως βλέπουμε παραπάνω η χρήση της διαρκής μεταβλητής έξω από την κλάση γίνεται με το όνομα του αντικειμένου το πρόθεμα «::» και το όνομα της διαρκής μεταβλητής δηλ «car::pi». Όταν θέλουμε να προσπελάσουμε την μεταβλητή μέσα από την ίδια την κλάση τότε χρησιμοποιούμε το πρόθεμα «self::».

Αρχείο: const1.php

```php
<?php
 class car{
  const pi=3.14;

  public function test(){
   echo self::pi;
  }
 }

 $c = new car();

 echo $c::pi;
 echo "<br/>";
 echo $c->test();
?>
```

Αποτέλεσμα:



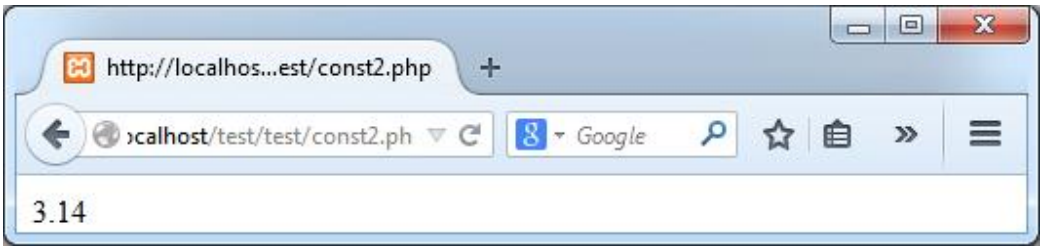
Εικόνα Α.168 Προσπέλαση διαρκών μεταβλητών από την ίδια κλάση

Εδώ πρέπει να σημειώσουμε πως μια υποκλάση κληρονομεί τις διαρκές μεταβλητές τις υπερκλάσης της.

Αρχείο const2.php

```
<?php
 class vehicle{
  const pi=3.14;
 }

 class car extends vehicle{
  public function test(){
   echo parent::pi;
  }
 }

 $c = new car();
 echo $c->test();
?>
```

Αποτέλεσμα:



Εικόνα Α.169 Κληρονομικές διαρκές μεταβλητές

Γενικώς ισχύει ότι κάθε υποκλάση μπορεί να φτιάξει τις δικές της διαρκές μεταβλητές όπως επίσης και να επικαλύψει υπάρχουσες από υπερκλάσεις.

Α.8.5 Αφηρημένες κλάσεις

Οι αφηρημένες κλάσεις, είναι κλάσεις οι οποίες δεν μπορούν να χρησιμοποιηθούν για να δημιουργήσουν αντικείμενα παρά μόνο να υπάρξουν ως υπερκλάσεις κάποιας υποκλάσης. Μια αφηρημένη κλάση είναι ένας τρόπος ορισμού ιδιοτήτων και λειτουργιών που θα μπορούν τα «παιδιά» της να χρησιμοποιήσουν. Έτσι μπορεί να έχει δικές τις μεταβλητές και ιδιότητες, καθώς επίσης και υποκλάσεις.

Παρομοίως μέσα σε μια αφηρημένη κλάση θα πρέπει να υπάρχουν αφηρημένες λειτουργίες (μέθοδοι) οι οποίες το μόνο που διαθέτουν είναι την υπογραφή τους και όχι σώμα εντολών (δηλ δεν έχουν άγκιστρα και εντολές παρά μόνο παραμέτρους μέσα στην παρένθεση). Έτσι μια υποκλάση μπορεί να χρησιμοποιήσει το όνομα αυτής της μεθόδου και να υλοποιήσει την λειτουργικότητα της.

Προσοχή! Για να είναι μια κλάση αφηρημένη θα πρέπει τουλάχιστον να περιέχει μια αφηρημένη μέθοδο.

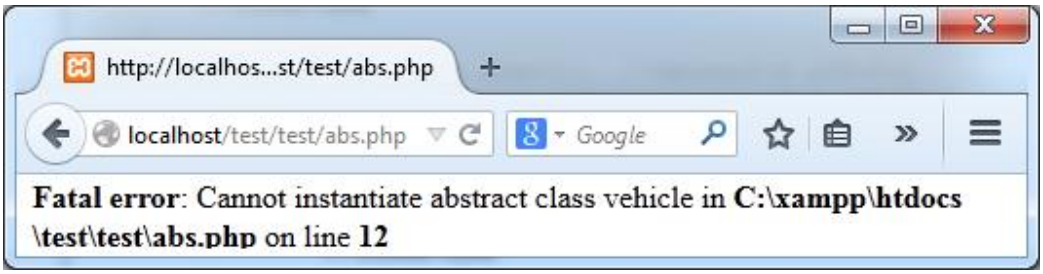
- Αν μια υποκλάση μιας αφηρημένης κλάσης δεν υλοποίει όλες τις αφηρημένες μεθόδους της, ή έχει κάποια δική της αφηρημένη μέθοδο τότε και η υποκλάση γίνεται αφηρημένη.
- Αν η υποκλάση μιας αφηρημένης κλάσης υλοποίει όλες τις αφηρημένες μεθόδους της υπερκλάσης που προέρχεται , τότε είναι μια κανονική κλάση.

Μια αφηρημένη κλάση ή μέθοδο δηλώνεται με την λέξη κλειδί «abstract» ύστερα «class» και ένα όνομα κλάσης.

Αρχείο: abs.php

```
<?php
 abstract class vehicle{
  private $wheels,$max_speed;

  function kanoniki(){
   echo "λειτουργεί";
  }

  abstract function afirimeni();
 }

 $a = new vehicle();
 $a->kanoniki();
?>
```

Αποτέλεσμα:



Εικόνα Α.170 Σφάλμα χρήσης λανθασμένης δημιουργίας αντικειμένου

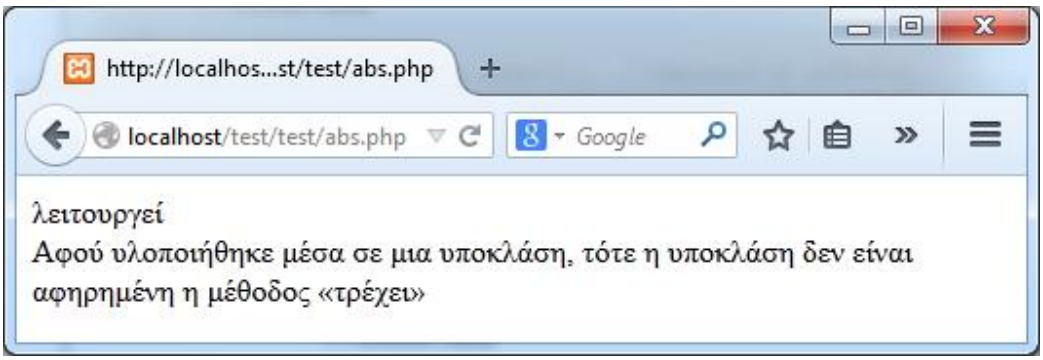
Μέσα στο παραπάνω παράδειγμα έχουμε μια «κανονική» μέθοδο μέσα στην αφηρημένη κλάση και μια «αφηρημένη» μέθοδο. Εμείς προσπαθήσαμε να δημιουργήσουμε ένα αντικείμενο μέσα από την συγκεκριμένη κλάση και ύστερα να καλέσουμε την «κανονική» μέθοδο της. Όπως βλέπετε και εσείς η php μας έβγαλε

σφάλμα χρήσης αφηρημένης μεθόδου για δημιουργία αντικειμένου. Ας δούμε την σωστή χρήση:

Αρχείο: abs1.php

```
<?php
 abstract class vehicle{
  private $wheels,$max_speed;

  function kanoniki(){
   echo "λειτουργεί";
  }

  abstract function afirimeni(); //αφηρημένη μέθοδος
 }
 class car extends vehicle{ // υποκλάση car
  function afirimeni(){ //υλοποιεί την αφηρημένη μέθοδο
   echo "Αφού υλοποιήθηκε μέσα σε μια υποκλάση, τότε η υποκλάση δεν
είναι αφηρημένη η μέθοδος «τρέχει»";
  }
 }
 $a = new car(); //Φτιάχνουμε ένα αντικείμενο της υποκλάσης
 $a->kanoniki();
 echo "<br/>";
 $a->afirimeni();
?>
```

Αποτέλεσμα:



Εικόνα Α.171 Χρήση αφηρημένων κλάσεων για δημιουργία αντικειμένων

Όπως βλέπουμε σε αυτό το παράδειγμα, δεν λειτουργεί μόνο η αφηρημένη μέθοδος αλλά και η κανονική, κι αυτό γίνεται γιατί μια υποκλάση της αφηρημένης κλάσης υλοποιεί τις αφηρημένες μεθόδους τις (στην συγκεκριμένη περίπτωση την μια που έχει) και έτσι η κλάση παιδί μπορεί να υποστηρίξει τα στιγμιότυπα αντικείμενων κληρονομώντας ταυτόχρονα όλες τις υπάρχουσες ιδιότητες και λειτουργίες της αφηρημένης κλάσης.

Αν δεν υλοποιούσαμε την αφηρημένη μέθοδο της αφηρημένης κλάσης τότε η υποκλάση θα ήταν κι αυτή αφηρημένη, επομένως δεν θα μπορούσαμε να δημιουργήσουμε στιγμιότυπα αντικειμένων.

Α.8.6 Διεπαφές

Μια διεπαφή χρησιμοποιείται για να ορίσει αποκλειστικά και μόνο υπογραφές μεθόδων χωρίς σώμα εντολών και χωρίς άγκιστρα. Είναι κάποιο είδος κλάσης το οποίο μπορεί να περιλαμβάνει μόνο υπογραφές μεθόδων και μεταβλητές οι οποίες μπορούν να είναι μόνο διαρκής.

Όπως σε μια αφηρημένη μέθοδο, έτσι και σε μια διεπάφη δεν μπορούμε να δημιουργήσουμε αντικείμενα με αυτήν, αλλά μια διεπαφή μπορεί να κληρονομήσει στοιχεία από μια η περισσότερες άλλες διεπαφές. Αυτό σημαίνει ότι μπορεί να έχει και πατέρα και μητέρα, πράγμα που οι απλές κλάσεις δεν μπορούν να το κάνουν αυτό. Οι απλές κλάσεις όμως μπορούν να «υλοποιήσουν» μια διεπαφή όπως γίνεται και στις αφηρημένες κλάσεις αλλά χωρίς να «κληρονομούνται στοιχεία».

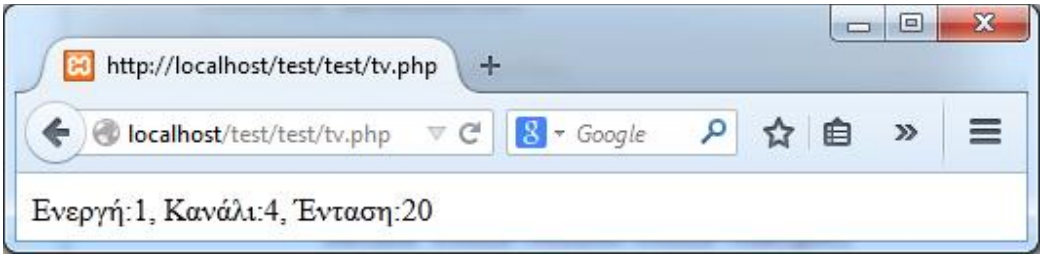
Ότι ισχύει για τις αφηρημένες κλάσεις και τις υποκλάσεις τους, έτσι κι εδώ όταν μια κλάση υλοποιεί μια διεπαφή θα πρέπει να υποχρεωτικά να υλοποιήσει όλες τις μεθόδους τις προκειμένου να μπορεί η κλάση να λειτουργήσει σωστά.

Ας υποθέσουμε πως μια διεπαφή τηλεκοντρόλ, μας δίνει κάποιες λειτουργίες όπως το άνοιγμα μιας τηλεόρασης, την αλλαγή καναλιού και την αλλαγή έντασης της τηλεόρασης. Για να χρησιμοποιηθεί αυτή η διεπαφή θα πρέπει να υπάρχει μια αντίστοιχη κλάση που να την χρησιμοποιεί. Η λέξεις κλειδιά στην προκειμένη περίπτωση είναι «interface» για την δημιουργία της διεπαφής και «implements» για την υλοποίηση της.

Αρχείο: tv.php

```php
<?php
 interface tilekontrol{
  function anoigmaTileorasis();
  function allagiKanaliou($a);
  function allagiEntasis($a);
 }

 class tileorasi implements tilekontrol{
  public $katastasi,$kanali,$entasi;

  function anoigmaTileorasis(){ //υλοποίηση πρώτης μεθόδου
   $this->katastasi=true;
  }
  function allagiKanaliou($a){ // υλοποίηση δεύτερης
   $this->kanali = $a;
  }
  function allagiEntasis($a){ // υλοποίηση τρίτης
   $this->entasi = $a;
  }
 }

 $a = new tileorasi; //δημιουργία αντικείμενου με βάση την κλάση
 $a->anoigmaTileorasis();
 $a->allagiKanaliou(4);
 $a->allagiEntasis(20);

 echo "Ενεργή:".$a->katastasi.", Κανάλι:".$a->kanali.", Ένταση:".$a-
>entasi;
?>
```

Αποτέλεσμα:



Εικόνα Α.172 Δημιουργία διεπαφής και υλοποίηση

Α.9 MySQL & PHP

Στις διαδικτυακές εφαρμογές εμφανίζεται η ανάγκη για αποθήκευση δεδομένων. Το όνομα ενός χρήστη, ο κωδικός του, η πιστωτική του κάρτα, η ηλικία του και άλλα σχετικά είναι μερικά από αυτά τα στοιχεία που αποθηκεύονται σε μια βάση δεδομένων.

Οι τύποι δεδομένων που αποθηκεύουμε αποτελούν συνήθως, αριθμούς και κείμενο. Η λίστα επαφών στο κινητό μας για παράδειγμα είναι μια μικρή βάση δεδομένων όπου μέσα σε αυτή κρατάμε το όνομα, το επώνυμο, το σταθερό τηλέφωνο, το κινητό, το email και άλλα σχετικά που έχουν να κάνουν με την εκάστοτε επαφή. Στο κινητό όλη αυτή η πληροφορία αποθηκεύεται σε αρχεία τα οποία βρίσκονται μέσα σε αυτό. Στο διαδίκτυο όμως, χρησιμοποιούνται διαδικτυακοί servers οι οποίοι αποτελούν server βάσεων δεδομένων και το μόνο που κάνουν είναι η αποθήκευση και η ανάκτηση πληροφοριών.

Ακριβώς εκεί χρησιμοποιείται και στις διαδικτυακές εφαρμογές. Ο κώδικας μας λέει πως λειτουργεί η εφαρμογή, αλλά όλα τα δεδομένα που δημιουργούν οι χρήστες της, αποθηκεύονται έξω από αυτή, και συγκεκριμένα σε μια βάση δεδομένων.

Σε αυτό το κεφάλαιο θα μάθουμε να χρησιμοποιούμε την MySQL (My Structured Query Language). Είναι ένα είδος βάσεων δεδομένων ανοιχτού κώδικα όπου καθένας μπορεί να την προμηθευτεί και να την χρησιμοποιήσει για οποιοδήποτε λόγο θέλει. Μέσω αυτής θα προσπαθήσουμε να σας δείξουμε πως συνδέεται μια διαδικτυακή εφαρμογή σε php με αυτήν.

Παρακάτω δεν θα δούμε περίπλοκες εντολές της SQL καθώς αυτό είναι ένα τελείως ξεχωριστό κομμάτι εκπαίδευσης που έχει να κάνει αποκλειστικά με την γλώσσα SQL. Στις παρακάτω ενότητες θα δούμε πως μπορούμε να εκτελέσουμε απλές εντολές σε συνδυασμό με την php, όπως να εισάγουμε δεδομένα σε μια βάση, να εξάγουμε τα αποτελέσματα ενός ερωτήματος και να επεξεργαστούμε στοιχειωδώς μια βάση. Για περεταίρω ανάλυση της SQL θα πρέπει να ανατρέξετε σε σχετικό οδηγό.

A.9.1 Μια σύντομη προεπισκόπηση στον κώδικα SQL

Πριν προχωρήσουμε στην διασύνδεση μιας βάσης δεδομένων με την php θα πρέπει πρώτα να θυμηθούμε επιγραμματικά μερικές από τις πιο βασικές εντολές της SQL.

Ο κώδικας SQL διαθέτει μια σειρά από εντολές εισαγωγής, εξαγωγής, διαγραφής και ενημέρωσης εγγραφών σε πίνακες που διαθέτει μια βάση δεδομένων. Κάθε φορά που εκτελούμε μια διαδικασία μέσα σε μια βάση δεδομένων, τότε αυτή η «πράξη» ονομάζεται ερώτημα (query). Ο τρόπος με τον οποίο κάνουμε όλες αυτές τις πράξεις

ερωτημάτων είναι μέσω της γλώσσας SQL τα αρχικά της οποίας προέρχονται από τα Structure Query Language, δηλαδή σε ελεύθερη μετάφραση: «Δομημένη Γλώσσα Ερωτημάτων».

Η SQL διαθέτει ένα σύνολο από εντολές που εκτελούν διάφορες βασικές δυνατότητες όπως η εισαγωγή, η εξαγωγή, η ενημέρωση, η εύρεση συγκεκριμένων τιμών κ.α. Στην συγκεκριμένη ενότητα θα εξετάσουμε τις πιο βασικές ενέργειες της SQL, μερικές από αυτές είναι:

- Ανάγνωση δεδομένων
- Εισαγωγή δεδομένων
- Ενημέρωση δεδομένων
- Διαγραφή δεδομένων
- Γενική διαχείριση ΒΔ.

Η γλώσσα SQL είναι case insensitive , δεν κάνει διάκριση ανάμεσα σε κεφαλαία και πεζά γράμματα. Στα παραδείγματα που ακολουθούν, οι εντολές της SQL θα γράφονται με κεφαλαία γράμματα ώστε να ξεχωρίζουν.

A.9.1.1 Βάση δεδομένων

Παρακάτω θα δούμε όλες τις εντολές που αλληλεπιδρούν με τις βάσεις δεδομένων.

Η εντολή SHOW DATABASES μας επιστρέφει μια λίστα με όλες τις βάσεις δεδομένων που υπάρχουν στον MYSQL server όπου βρισκόμαστε:

```
SHOW DATABASES;
```

Με την εντολή CREATE μπορούμε να δημιουργήσουμε μια βάση δεδομένων, εάν δεν υπάρχει:

```
CREATE DATABASE onoma_basis;
```

Η εντολή USE μας επιτρέπει να διαλέξουμε βάση δεδομένων στην οποία θα εργαστούμε:

```
USE onoma_basis;
```

Η εντολή DROP μας επιτρέπει να διαγράψουμε μια βάση δεδομένων από την βάση στην οποία βρισκόμαστε, εφόσον και μόνο το όνομα που εισάγουμε στην DROP συμπίπτει με μια βάση που ήδη υπάρχει:

```
DROP DATABASE onoma_basis;
```

Όπου «onoma_basis» εισάγουμε το επιθυμητό όνομα.

A.9.1.2 Πίνακες

Παρακάτω θα δούμε όλες τις εντολές που αλληλεπιδρούν με τους πίνακες μιας βάσης δεδομένων. Για να εκτελέσουμε κάποια από τις παρακάτω εντολές θα πρέπει να έχουμε χρησιμοποιήσει πρώτα την εντολή «USE» που αναγράφεται στην αμέσως προηγούμενη ενότητα.

Η εντολή SHOW TABLES μας επιστρέφει μια λίστα με τους ήδη υπάρχοντες πίνακες μέσα στην επιλεγμένη βάση δεδομένων:

```
SHOW TABLES;
```

Η εντολή CREATE TABLE δημιουργεί έναν νέο πίνακα εφόσον δεν υπάρχει μέσα στην βάση δεδομένων στην οποία βρισκόμαστε. Η εντολή παίρνει και πληθώρα στοιχείων μέσα της ως παραμέτρους σχετικά με τα πεδία του πίνακα, τον τύπο (type) τους, και το αν θα έχουν πρωτεύων κλειδί (PRIMARY KEY):

```
CREATE TABLE onoma_pinaka(
 pedio1 type(megethos) NOT NULL PRIMARY KEY AUTO_INCREMENT,
 pedio2 type(megethos) NOT NULL,
 pedio3 type(megethos) NULL
);
```

- Όπου «onoma_pinaka» εισάγουμε το επιθυμητό όνομα του πίνακα που θέλουμε να δημιουργήσουμε,
- Όπου «pedio1», «pedio2»,«pedio3» εισάγουμε τα ονόματα των πεδίων που θέλουμε ο πίνακας μας να διαθέτει,
- Όπου «type» εισάγουμε τον τύπο που θέλουμε το κάθε πεδίο να είναι, ενώ μέσα στις παρενθέσεις του, το μέγεθος του πεδίου, εισάγουμε δηλαδή έναν ακέραιο αριθμό.

- Το NOT NULL σημαίνει ότι το πεδίο δεν μπορεί να είναι κενό, άρα για κάθε εγγραφή θα υπάρχει μια τιμή, αντίθετα το NULL σημαίνει πως μπορεί να είναι κενό,
- Το PRIMARY KEY σημαίνει πως είναι πρωτεύον κλειδί το συγκεκριμένο πεδίο στο οποίο το έχουμε γράψει, και τέλος
- Το AUTO_INCREMENT σημαίνει αυτόματη αρίθμηση για ένα πεδίο το οποίο όταν δημιουργείται μια νέα εγγραφή θα αυξάνεται κατά ένα για κάθε έγγραφή δημιουργώντας έτσι μια ξεχωριστή τιμή για κάθε εγγραφή.

Η εντολή ALTER TABLE χρησιμοποιείται σε συνδυασμό με κάποια δευτερεύουσα εντολή για να επεξεργαστεί την δομή ενός πίνακα, όπου «onoma_pinaka» εισάγουμε το όνομα του ήδη υπάρχον πίνακα που θέλουμε να επεξεργαστούμε.

Η δευτερεύουσα εντολή ADD προσθέτει ένα νέο πεδίο που δεν υπάρχει στον ήδη υπάρχον πίνακα, όπου «pedio4» εισάγουμε το όνομα του νέου πεδίου και ύστερα συνεχίζουμε με τις ιδιότητες που θέλουμε να έχει:

```
ALTER TABLE onoma_pinaka ADD pedio4 varchar(255);
```

Η δευτερεύουσα εντολή DROP διαγράφει ένα ήδη υπάρχουν πεδίο από έναν ήδη υπάρχον πίνακα, όπου «pedio4» εισάγουμε το όνομα του πεδίου που θέλουμε να διαγράψουμε:

```
ALTER TABLE onoma_pinaka DROP pedio4;
```

Η δευτερεύουσα εντολή ADD INDEX προσθέτει δείκτη στο πεδίο ενός πίνακα, όπου «onoma_pinaka» τοποθετούμε το όνομα του πίνακα στον οποίο βρίσκεται το πεδίο, όπου «onoma_index» τοποθετούμε ένα νέο όνομα για τον δείκτη, όπου «pedio3» τοποθετούμε το όνομα το πεδίου:

```
ALTER TABLE onoma_pinaka ADD INDEX onoma_index(pedio3);
```

Η δευτερεύουσα εντολή DROP INDEX διαγράφει έναν δείκτη ενός πεδίου που ανήκει σε έναν πίνακα, όπου «onoma_index» τοποθετούμε το όνομα του δείκτη:

```
ALTER TABLE onoma_pinaka DROP INDEX onoma_index;
```

Η δευτερεύουσα εντολή ADD PRIMARY KEY μετατρέπει ένα πεδίο σε πρωτεύον κλειδί ενός πίνακα, όπου «pedio2» προσθέτουμε το όνομα του πεδίου που θέλουμε να κάνουμε πρωτεύον κλειδί. ΠΡΟΣΟΧΗ! Αν πίνακας που προσπαθούμε να προσθέσουμε το νέο πρωτεύον κλειδί, έχει ήδη ένα πρωτεύον κλειδί, δεν μπορούμε να προσθέσουμε νέο, θα πρέπει πρώτα να διαγράψουμε το παλαιό:

```
ALTER TABLE onoma_pinaka ADD PRIMARY KEY (pedio2);
```

Η δευτερεύουσα εντολή DROP PRIMARY KEY μετατρέπει ένα πεδίο πρωτεύοντος κλειδιού σε απλό πεδίο. Για να μπορέσουμε να διαγράψουμε ένα κλειδί που είναι AUTO_INCREMENT θα πρέπει πρώτα να αφαιρέσουμε αυτή την ιδιότητα του με την εντολή CHANGE:

```
ALTER TABLE onoma_pinaka DROP PRIMARY KEY;
```

Η δευτερεύουσα εντολή CHANGE αναλαμβάνει να αλλάξει την δομή ενός πεδίου στην μορφή που εμείς επιθυμούμε, όπου «pedio1» τοποθετούμε το όνομα του πεδίου που έχει μέχρι εκείνη την στιγμή ύστερα στο «neo_pedio» τοποθετούμε το όνομα που θέλουμε να έχει από εδώ και ύστερα το πεδίο, και ύστερα όλες τις άλλες ιδιότητες που θέλουμε να διατηρήσει, όσες δεν συμπεριλάβουμε καταργούνται και ακολουθούνται οι προεπιλεγμένες τιμές:

```
ALTER TABLE onoma_pinaka CHANGE pedio1 neo_pedio int(11);
```

Η εντολή TRUNCATE TABLE διαγράφει όλα τα δεδομένα ενός πίνακα, δηλαδή όλες τις εγγραφές του, και δεν επηρεάζει καθόλου τα πεδία ή τον ίδιο τον πίνακα, όπου «onoma_pinaka» εισάγουμε το όνομα του πίνακα που θέλουμε να αδειάσουμε:

```
TRUNCATE TABLE onoma_pinaka;
```

Η εντολή DESCRIBE ή DESC μας δίνει κάποιες πληροφορίες σχετικά με τον πίνακα που θα της εισάγουμε, όπου «onoma_pinaka» εισάγουμε το όνομα του πίνακα για τον οποίο θέλουμε τις πληροφορίες:

```
DESCRIBE onoma_pinaka;
DESC onoma_pinaka;
```

Η εντολή DROP TABLE διαγράφει έναν ήδη υπάρχον πίνακα από την βάση δεδομένων στην οποία βρισκόμαστε, όπου «onoma_pinaka» εισάγουμε το όνομα του πίνακα που θέλουμε να διαγράψουμε:

```
DROP TABLE onoma_pinaka;
```

A.9.1.3 Ερωτήματα

Η εντολή SELECT FROM μας επιστρέφει ένα σύνολο εγγραφών μέσα από ένα πίνακα που θα επιλέξουμε, βασιζόμενη σε κάποια κριτήρια. Για παράδειγμα χρησιμοποιώντας τον χαρακτήρα «*» (αστερίσκο), επιστρέφονται όλες οι γραμμές του πίνακα. Όπου «onoma_pinaka» εισάγουμε το όνομα του πίνακα για τον οποίο θέλουμε να επιστραφούν οι γραμμές του:

```
SELECT * FROM onoma_pinaka;
```

Χρησιμοποιώντας την SELECT προσδιορίζοντας αντίστοιχα τα πεδία που θέλουμε πριν το FROM, μας επιστρέφει ένα σύνολο γραμμών μέσα σε ένα πίνακα μόνο για τα επιλεγμένα πεδία που έχουμε διαλέξει, όπου «pedio2», και «pedio3» εισάγουμε τα ονόματα των πεδίων που θέλουμε να επιλεχθούν καθώς επίσης κι άλλα αν θέλουμε, ύστερα στο «onoma_pinaka» εισάγουμε το όνομα τον πίνακα από τον οποίο θα επιλεγούν τα στοιχεία:

```
SELECT pedio2, pedio3 FROM onoma_pinaka;
```

Η εντολή DISTINCT σε συνδυασμό με την SELECT επιστρέφει ένα σύνολο εγγραφών για όσα πεδία επιλέξουμε, αφαιρώντας τις διπλότυπες τιμές, όπου «pedio2» τοποθετούμε το όνομα του πεδίου που θέλουμε να επιλεχθεί, όπου «onoma_pinaka» το όνομα του πίνακα, αντίστοιχα:

```
SELECT DISTINCT (`pedio2`) FROM onoma_pinaka;
```

Η δευτερεύουσα εντολή WHERE μας δίνει την δυνατότητα να περιορίσουμε το εύρος του ερωτήματος μας διαλέγοντας συγκεκριμένες εγγραφές. Δίνοντας ένα αλφαριθμητικό, αριθμό ή γενικώς μια τιμή στην εντολή αυτή ψάχνει για εγγραφές που θα ταιριάζει αυτή η τιμή στο πεδίο που της έχουμε ορίσει, έτσι επιστρέφεται ένα μικρό

σύνολο από εγγραφές, ενδεχομένως και καμία. Όπου «onoma_pinaka» μπαίνει το όνομα του πίνακα μας, όπου «pedio2» μπαίνει το πεδίο για το οποίο θέλουμε να ταιριάξουμε την τιμή, και όπου «0» εισάγουμε την τιμή που θέλουμε να ταιριάξουμε:

```
SELECT * FROM `onoma_pinaka` WHERE pedio2=0;
```

Παρόμοια με την WHERE υπάρχει και η LIKE όπου στην θέση του πεδίου με την ισότητα τοποθετούμε απλά την εντολή με ένα αλφαριθμητικό που θέλουμε να ταιριάξει. Όπου χρησιμοποιήσουμε το σύμβολο «%» θεωρείται μπαλαντέρ.

```
SELECT * FROM onoma_pinaka WHERE pedio2 LIKE "%τιμή%";
```

Μπορούμε να μετρήσουμε τα επιστρεφόμενα αποτελέσματα ενός ερωτήματος με την εντολή COUNT.

```
SELECT COUNT(*) FROM onoma_pinaka;
```

Μπορούμε να επιστρέψουμε τα αποτελέσματα ενός ερωτήματος ταξινομημένα κατά αύξουσα ή φθίνουσα σειρά. Με την δευτερεύουσα εντολή ORDER BY. Η εντολή χρειάζεται να τις υποδείξουμε τον πίνακα και το πεδίο (ή πεδία) στο οποίο θα γίνει η ταξινόμηση, η οποία μπορεί να είναι κατά αύξουσα (ASC) ή φθίνουσα (DESC).

```
SELECT * FROM onoma_pinaka ORDER BY pedio2 ASC, pedio3 DESC;
```

Μπορούμε να εισάγουμε μια νέα γραμμή σε έναν πίνακα με την εντολή INSERT INTO. Η εντολή χρειάζεται το όνομα του πίνακα στον οποίο θα εισάγουμε την νέα γραμμή, προαιρετικά τα πεδία στα οποία θα εισάγουμε τιμές αλλιώς τα χρησιμοποιεί όλα, και τις νέες τιμές που θέλουμε να εισάγουμε στα παραπάνω πεδία:

```
INSERT INTO onoma_pinaka (neo_pedio,pedio2,pedio3)
VALUES (4,"τιμή1","τιμή2");
```

Όπως καταλαβαίνουμε από την παραπάνω εντολή κάθε πεδίο θα αντιστοιχηθεί και με μια τιμή μετά το VALUES, έτσι θα εισαχθεί η νέα γραμμή στον πίνακα. Αν θέλουμε μπορούμε να παραλείψουμε το πρώτο πεδίο καθώς μπορεί να είναι αυτόματα ρυθμισμένο να αυξάνεται κατά ένα σε κάθε νέα γραμμή.

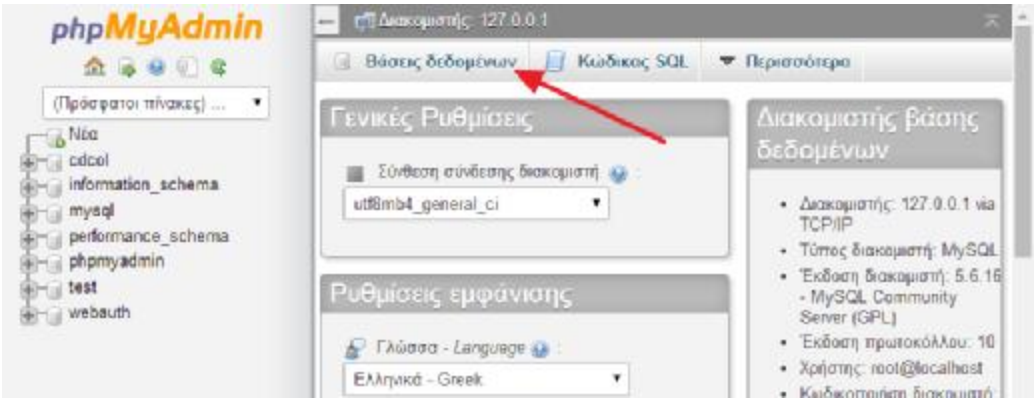
Υπάρχουν πολλοί συνδυασμοί και εντολές της SQL που δεν θα εξηγήσουμε σε αυτόν τον οδηγό, καθώς πρόκειται αυστηρά για επεξήγηση λειτουργικότητας των εντολών της php. Μπορείτε να ανατρέξετε σχετικούς οδηγούς στην ιστοσελίδα μας για την SQL ή μπορείτε να αναζητήσετε σχετικό υλικό στο Διαδίκτυο.

A.9.2 Οδηγός χρήσης phpmyadmin

Αν ακολουθήσατε τον οδηγό εγκατάστασης του XAMPP στο πρώτο κεφάλαιο, θα παρατηρήσατε πως το «κουμπάκι» start στην mysql υπάρχει γι αυτόν ακριβώς τον λόγο. Αν έχουμε ενεργοποιημένη την MySQL και επισκεφθούμε από το πρόγραμμα περιήγησης μας την διεύθυνση «http://localhost/phpmyadmin» θα δούμε ένα διαχειριστικό κομμάτι της mysql όπου μπορούμε να εκτελέσουμε βασικές εντολές της SQL δημιουργώντας βάσεις δεδομένων εισάγοντας δεδομένα σε αυτές και άλλα πολλά.

Βήμα πρώτο

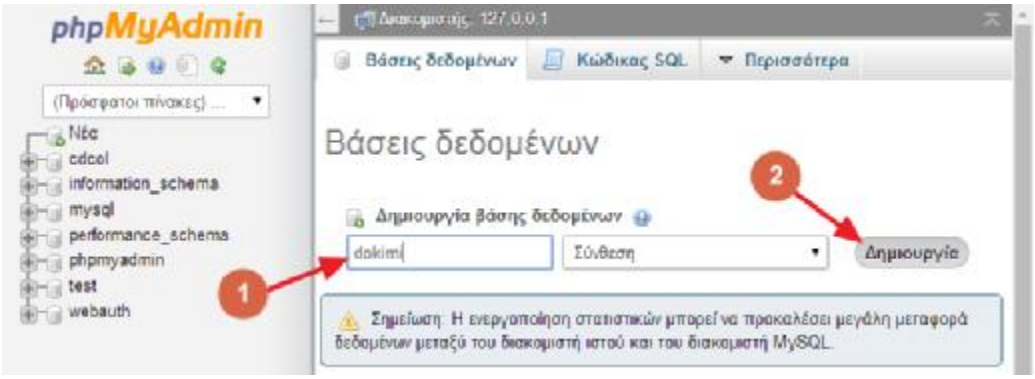
Αφού επισκεφτούμε την διεύθυνση «http://localhost/phpmyadmin» μας εμφανίζεται η παρακάτω οθόνη, κάνουμε κλικ στην επιλογή «Βάσεις Δεδομένων».



Εικόνα A.173 Αρχική οθόνη phpmyadmin

Βήμα δεύτερο

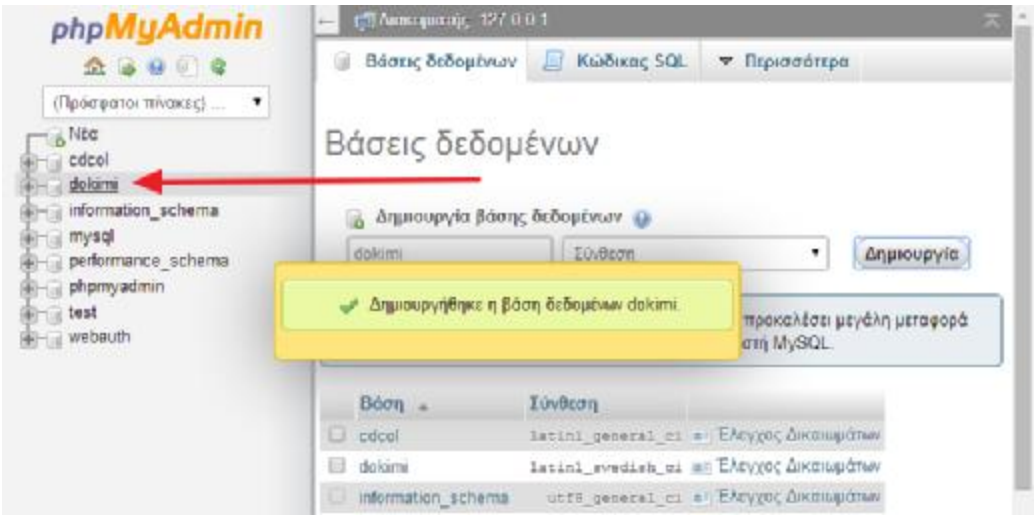
Μας ανοίγει η παρακάτω οθόνη όπου εκεί γράφουμε ένα όνομα μέσα στο πεδίο «Όνομα βάσης δεδομένων» και ύστερα κάνουμε κλικ στο «Δημιουργία».



Εικόνα Α.174 Δημιουργία βάσης δεδομένων

Βήμα τρίτο

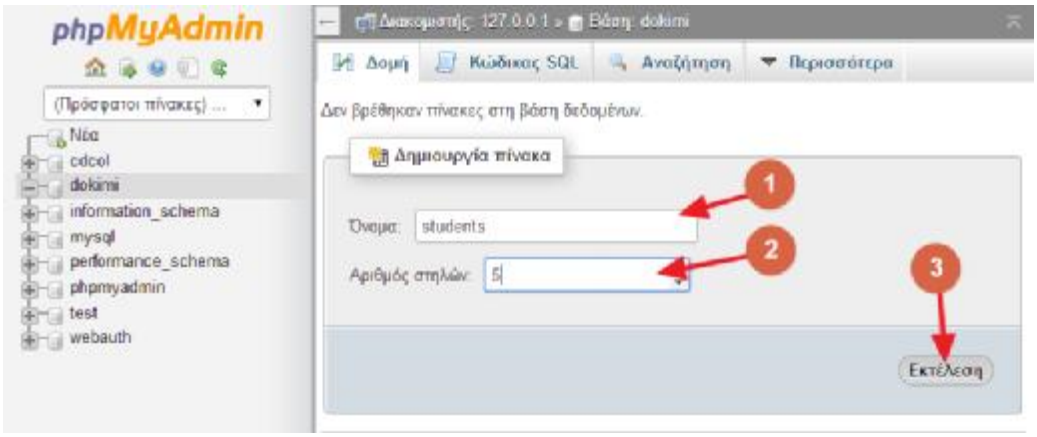
Ενημερωνόμαστε για την δημιουργία, και ακριβώς από κάτω βλέπουμε στην λίστα με τις βάσεις δεδομένων το όνομα της βάσης που μόλις δημιουργήσαμε. Τώρα μπορούμε να περιηγηθούμε μέσα στην βάση δεδομένων κάνοντας κλικ στο όνομα της αριστερά από την λίστα.



Εικόνα Α.175 Επιβεβαίωση δημιουργίας βάσης δεδομένων

Βήμα τέταρτο

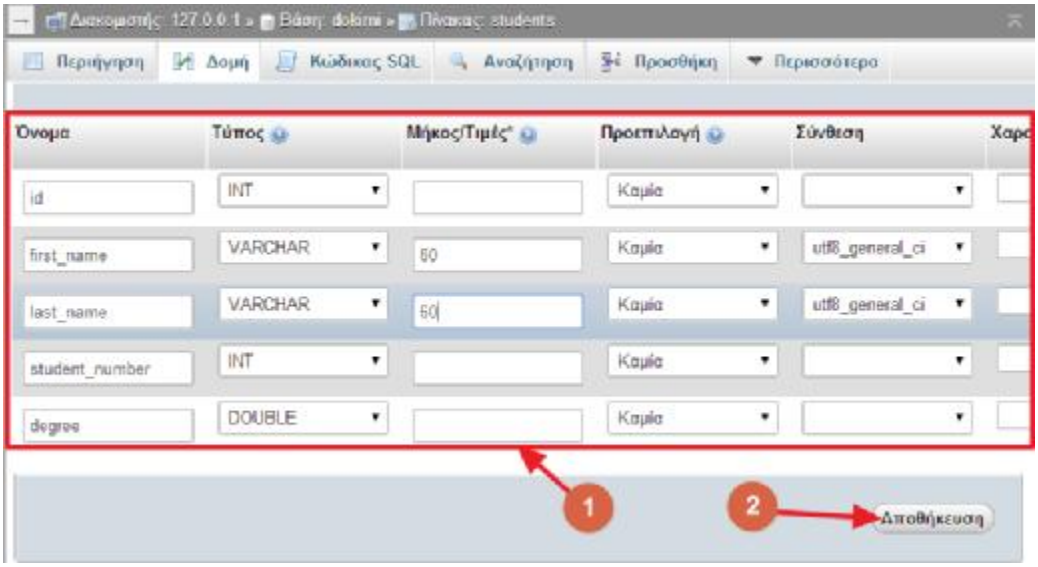
Μόλις κάνουμε το κλικ, καθότι κενή η βάση δεδομένων , μας εμφανίζεται οδηγός για δημιουργία πίνακα μέσα σε αυτήν. Εμείς θα δημιουργήσουμε έναν πίνακα «students» με 5 στήλες.



Εικόνα Α.176 Δημιουργία πίνακα σε βάση δεδομένων

Βήμα πέμπτο

Μόλις κάνουμε κλικ στο «εκτέλεση» μας ζητούνται τα στοιχεία δημιουργίας των πεδίων του πίνακα. Αφού εισάγουμε τα στοιχεία κάνουμε κλικ στο «Αποθήκευση».



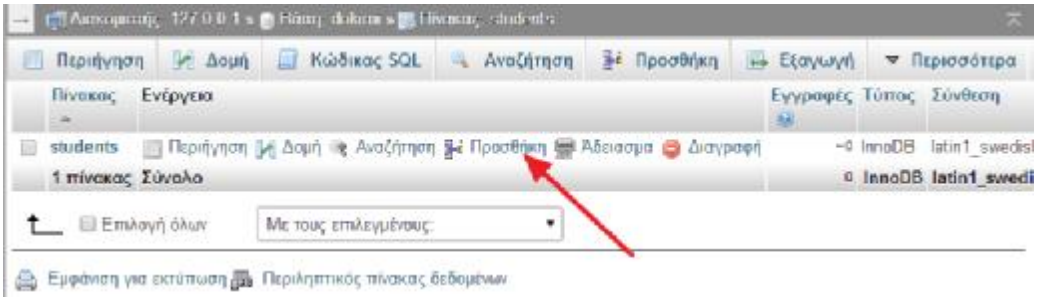
Εικόνα Α.177 Δημιουργία πεδίων σε νέο πίνακα

Όνομα	Τύπος	Μήκος	Σύνθεση	Ευρετήριο	A_I
id	INT			PRIMARY	Ναι
first_name	VARCHAR	60	utf8_general_ci	--	Όχι
last_name	VARCHAR	60	utf8_general_ci	--	Όχι
student_number	INT			--	Όχι
degree	DOUBLE			--	Όχι

Πίνακας Α.26 Τα στοιχεία των πεδίων του πίνακα students

Βήμα έκτο

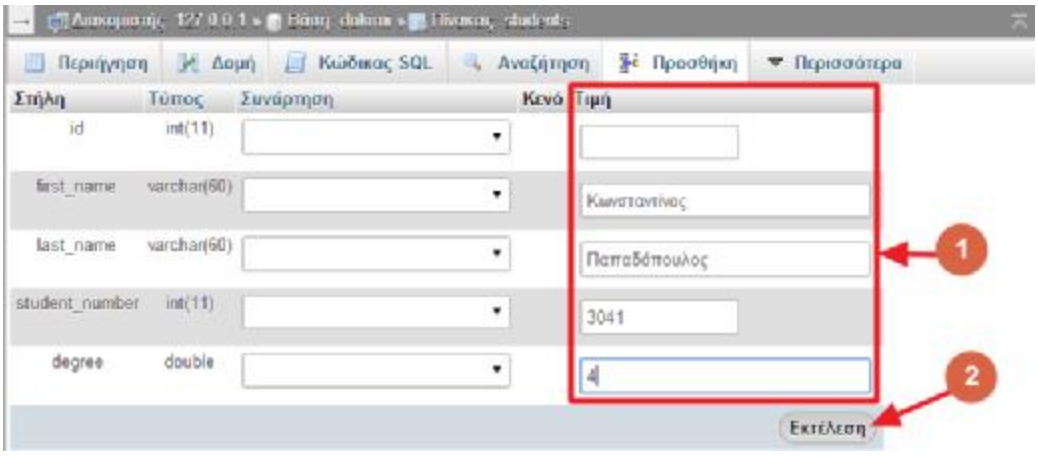
Από τον σελίδα που μας εμφανίζεται βλέπουμε όλους τους πίνακες στην βάση δεδομένων στην οποία βρισκόμαστε. Στην πρώτη του γραμμή βρίσκουμε τον τίτλο του πίνακα που δημιουργήσαμε. Στην ίδια ευθεία κάνουμε κλικ στην επιλογή «Προσθήκη» ώστε να του εισάγουμε γραμμές.



Εικόνα Α.178 Προσθήκη γραμμών σε πίνακα

Βήμα έβδομο

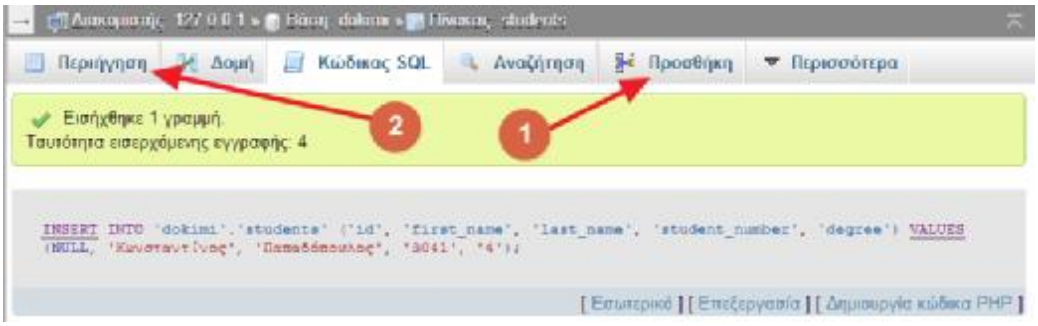
Από την φόρμα που μας φορτώνεται εισάγουμε τα στοιχεία που θέλουμε να βάλουμε ως εγγραφές. Εισάγουμε μερικά στοιχεία που να ταιριάζουν στους τύπους των πεδίων. Το πεδίο «id» είναι κενό γιατί έχουμε ενεργοποιήσει σε αυτό την ιδιότητα «A_I» δηλαδή «Auto Increment» που σημαίνει ότι θα παίρνει αυτόματα ακέραιες τιμές αυξάνοντας κάθε φορά κατά ένα. Κάνουμε κλικ στο κουμπί «εκτέλεση» αφού τελειώσουμε.



Εικόνα Α.179 Διαδικασία εισαγωγής νέων γραμμών σε πίνακα

Βήμα όγδοο

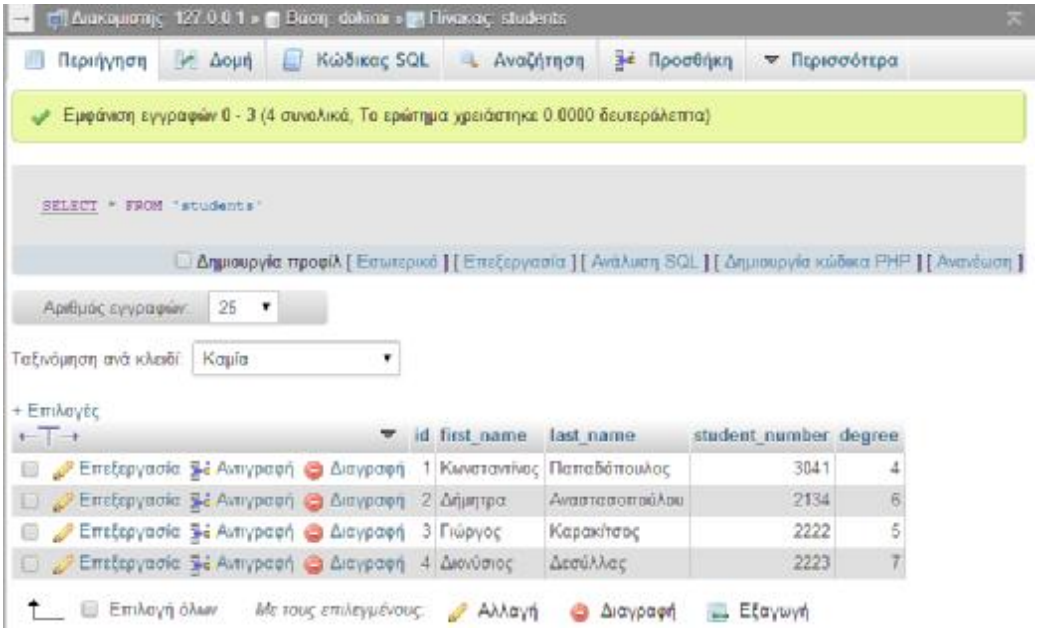
Ενημερωνόμαστε για την επιτυχή εισαγωγή των δεδομένων στον πίνακα. Ξανά κάνουμε κλικ στο «Προσθήκη» και εισάγουμε μερικές ακόμη τιμές έτσι ώστε να περιέχει κάτι ο πίνακας. Αφού εισάγουμε κάποιες γραμμές, επιλέγουμε το κουμπί «Περιήγηση».



Εικόνα Α.180 Μήνυμα επιβεβαίωσης εισαγωγής γραμμής σε πίνακα

Βήμα ένατο

Μας εμφανίζει μια λίστα με όλα αυτά που έχουμε εισάγει και που μπορούμε να επεξεργαστούμε: Με αυτά τα δεδομένα θα ασχοληθούμε αργότερα δημιουργώντας php αρχεία τα οποία θα αποκτούν πρόσβαση σε αυτά προσθέτοντας, αφαιρώντας και γενικά κάνοντας διάφορες πράξεις με τα δεδομένα.



Εικόνα Α.181 Τελική μορφή πίνακα students

A.9.3 Mysql

Ο τρόπος σύνδεσης της μέσω εντολών mysql μέσα από την php είναι ο πιο κοινός και ο παλαιότερος. Τώρα πλέον αυτή η μέθοδος τείνει να εξαφανιστεί καθότι θεωρείται ξεπερασμένη και έχει αντικατασταθεί από το mysqli ή το PDO.

Για τα παρακάτω παραδείγματα θα χρησιμοποιηθεί ο πίνακας με όνομα students η δομή του οποίου είναι όπως ο παρακάτω πίνακας με 4 πεδία (fields) και 4 εγγραφές (records).

Id	first_name	last_name	student_number	degree
1	Κωνσταντίνος	Παπαδόπουλος	3041	4
2	Δήμητρα	Αναστασοπούλου	2134	6
3	Γιώργος	Καρακίτσος	2222	5
4	Διονύσιος	Δεσύλλας	2223	7

Πίνακας Α.27 Περιεχόμενο πίνακα students

Σας παραχωρούμε και των κώδικα SQL σε περίπτωση που δεν έχετε εκτελέσει τον οδηγό χρήσης phpmyadmin.

```
CREATE TABLE IF NOT EXISTS `students` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `first_name` varchar(60) CHARACTER SET utf8 NOT NULL,
  `last_name` varchar(60) CHARACTER SET utf8 NOT NULL,
  `student_number` int(11) NOT NULL,
  `degree` double NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=5 ;

INSERT INTO `students`
 (`id`, `first_name`, `last_name`, `student_number`, `degree`)
VALUES
 (1, 'Κωνσταντίνος', 'Παπαδόπουλος', 3041, 4),
 (2, 'Δήμητρα', 'Αναστασοπούλου', 2134, 6),
 (3, 'Γιώργος', 'Καρακίτσος', 2222, 5),
 (4, 'Διονύσιος', 'Δεσύλλας', 2223, 7);
```

A.9.3.1 Σύνδεση

Για να συνδεθούμε σε μια βάση δεδομένων χρησιμοποιούμε την εντολή «mysql_connect» η οποία δέχεται διαδοχικά τα τρία προαπαιτούμενα στοιχεία hostname, username, password (όνομα εξυπηρετητή, όνομα χρήστη, κωδικός).

```
mysql_connect("localhost","root","");
```

Στο παράδειγμα που ακολουθεί, η «mysql_connect» επιχειρεί την σύνδεση σε μια βάση δεδομένων. Αν αποτύχει εμφανίζει μήνυμα αποτυχίας.

```
<?php
 $conn = mysql_connect("localhost","root","");
 if(!$conn){
  die('Αδύνατη σύνδεση: ' .mysql_error());
 }
?>
```

Στην μεταβλητή «$conn» αποθηκεύεται η ταυτότητα της σύνδεσης. Εάν δεν δημιουργηθεί κατά την διάρκεια της σύνδεσης τότε σημαίνει πως δεν πραγματοποιήθηκε. Η εντολή «die» λειτουργεί σαν συνδυασμός των «echo» και «exit». Τυπώνεται δηλαδή ένα μήνυμα και ύστερα το script τερματίζεται.

Η «mysql_error» επιστρέφει ένα κείμενο με το σφάλμα που ήλθε από την εκτέλεση ενός προηγούμενου ερωτήματος προς μια βάση δεδομένων. Στην συγκεκριμένη περίπτωση θα μας επέστρεφε κάποιο σφάλμα σχετικά με το γιατί δεν μπορούμε να συνδεθούμε στην βάση δεδομένων.

Όπως μπορούμε να ανοίξουμε μια σύνδεση παρομοίως μπορούμε να την κλείσουμε κιόλας. Όταν πλέον έχουμε τελειώσει με τις αλλαγές μας με την βάση δεδομένων είναι προτιμότερο να κλείνουμε μια σύνδεση. Αυτό πραγματοποιείται με την χρήση της εντολής «mysql_close», όπου χρειάζεται ένα όρισμα την ταυτότητα της σύνδεσης δηλαδή την μεταβλητή «$conn» του προηγούμενου παραδείγματος.

```
<?php
 $conn = mysql_connect("localhost","root","");
 if(!$conn){
  die('Αδύνατη σύνδεση:' .mysql_error());
 }

 //εντολές εδώ

 mysql_close($conn);
?>
```

A.9.3.2 Ανάγνωση

Αρχικά για να μπορέσουμε να εκτελέσουμε ένα ερώτημα θα πρέπει να επιλέξουμε την βάση δεδομένων μέσα στην οποία θα εκτελέσουμε τα αντίστοιχα ερωτήματα. Αυτό γίνεται μέσω της εντολής «mysql_select_db» η οποία παίρνει δυο ορίσματα, πρώτα το

όνομα της βάσης δεδομένων που θέλουμε να χρησιμοποιήσουμε (στην συγκεκριμένη περίπτωση «"dokimi"»), και ύστερα την ταυτότητα της σύνδεσης (στην συγκεκριμένη περίπτωση «$conn»).

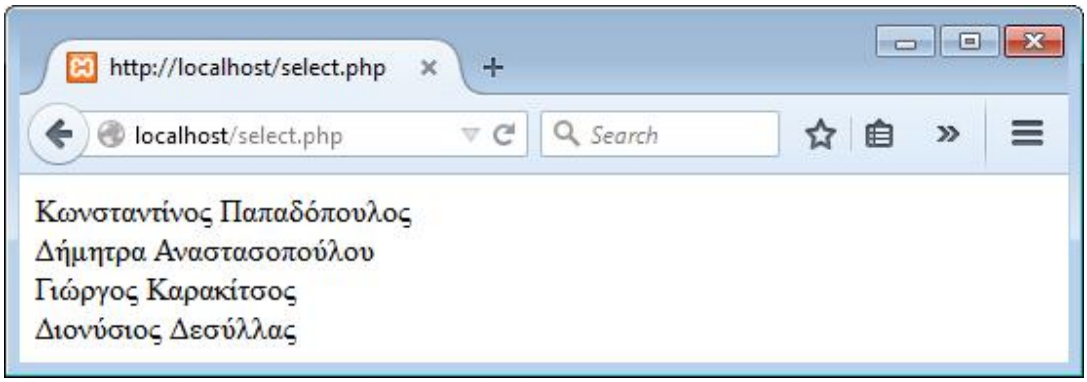
Για να εκτελέσουμε ένα ερώτημα προς μια βάση δεδομένων θα πρέπει να χρησιμοποιήσουμε την εντολή «mysql_query» η οποία δέχεται ένα όρισμα, την εντολή σε SQL που θέλουμε να εκτελέσουμε. Όταν εκτελεστεί το ερώτημα επιστρέφεται το αποτέλεσμα σε μια μεταβλητή την οποία ορίζουμε εμείς.

Για να διαβάσουμε δεδομένα από έναν πίνακα μιας ΒΔ χρησιμοποιούμε την εντολή SELECT. Αν θέλουμε να επιλέξουμε όλες τις εγγραφές από τον πίνακα «students» χρησιμοποιούμε τον χαρακτήρα «*» που σημαίνει «όλα».

Αρχείο: select.php

```php
<?php
 $conn = mysql_connect("localhost","root","rootroot");
 if(!$conn){
  die('Αδύνατη σύνδεση: ' .mysql_error());
 }

 mysql_select_db("dokimi",$conn);
 mysql_set_charset('utf8');

 $result = mysql_query("SELECT * FROM students");

 while($row=mysql_fetch_array($result)){
  echo $row['first_name']." ".$row['last_name'];
  echo "<br />";
 }

 mysql_close($conn);
?>
```

Αποτέλεσμα:



Εικόνα Α.182 Άντληση δεδομένων μέσω της php από μια βάση δεδομένων

Χρησιμοποιούμε την εντολή «mysql_set_charset('utf8');» προκειμένου οι χαρακτήρες που μεταφέρονται ως κείμενο από την βάση δεδομένων να μεταβιβάζονται σε κωδικοποίηση UTF-8 έτσι ώστε να είναι συμβατοί και αναγνώσιμοι με την κωδικοποίηση που χρησιμοποιούμε τόσο στον κώδικα μας όσο και στον φυλλομετρητή μας.

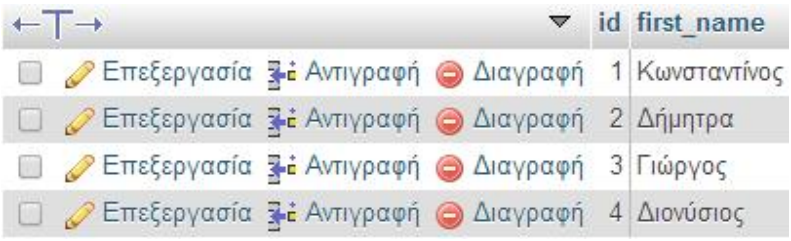
Το αποτέλεσμα του ερωτήματος αποθηκεύεται σε μορφή αντικειμένου mysql στην μεταβλητή «$result» την οποία χρησιμοποιούμε αργότερα για να μετατρέψουμε σε πίνακα προκειμένου να αντλήσουμε τα δεδομένα με κάποια δομή επανάληψης.

Αφού αντλήσουμε τα δεδομένα από την βάση δεδομένων τότε η mysql μας επιστρέφει το αποτέλεσμα σε μορφή πίνακα μέσω της εντολής «mysql_fetch_array». Για μπορέσουμε να τα τυπώσουμε θα πρέπει να «τρέξουμε» τον πίνακα αυτόν με κάποια μορφή επανάληψης (for, while, foreach).

Αν θέλαμε να επιλέξουμε συγκεκριμένα πεδία από τον πίνακα, για παράδειγμα μόνο το id και το first_name τότε θα εκτελούσαμε το SQL query κάπως έτσι:

```
SELECT id,first_name FROM students;
```

Το οποίο μας επιστρέφει αυτό το αποτέλεσμα (σε phpmyadmin):



Εικόνα Α.183 Επιλογή συγκεκριμένων πεδίων πίνακα μέσω της SQL

Α.9.3.3 Εισαγωγή

Για να εισάγουμε δεδομένα σε μια βάση δεδομένων χρησιμοποιούμε την εντολή INSERT. Υπάρχουν δύο τρόποι με τους οποίους μπορούμε να εισάγουμε δεδομένα με την INSERT. Ο πρώτος είναι να τα εισάγουμε χωρίς να αναφέρουμε τα πεδία του πίνακα που θέλουμε να τοποθετηθούν, αλλά δίνοντας τα δεδομένα στην κατάλληλη σειρά με την οποία τα πεδία έχουν δημιουργηθεί. Και ο δεύτερος είναι να εισάγουμε δεδομένα δίνοντας και την σειρά των πεδίων, αυτό μας δίνει την δυνατότητα να αγνοήσουμε κάποια πεδία και να εισάγουμε τιμές μόνο εκεί που μας ενδιαφέρει.

Ας δούμε ένα παράδειγμα SQL για την πρώτη περίπτωση:

```
INSERT INTO students VALUES (5,'Ιωάννης','Τζιτζήκας',3000,10);
```

Αν χρησιμοποιήσουμε αυτή την μέθοδο, τότε θα πρέπει να δώσουμε για όλα τα πεδία τιμές, ακόμη και για το id, που στην προκειμένη είναι αυτόματης αρίθμησης.

Αντιθέτως σε ένα διακριτό INSERT επιλέγουμε εμείς ποια πεδία θα «γεμίσουμε».

```
INSERT INTO students (`first_name`,`last_name`,`student_number`)
VALUES ('Κώστας','Γιώτολης',3000);
```

Με την παραπάνω εκτέλεση το «id» που χει οριστεί αυτόματης αρίθμησης θα πάρει αυτόματη τιμή χωρίς να του εισάγουμε εμείς χειροκίνητα, ενώ το «degree» θα πάρει εξ-ορισμού την τιμή μηδέν. Ας δούμε πως γίνεται αυτό σε php:

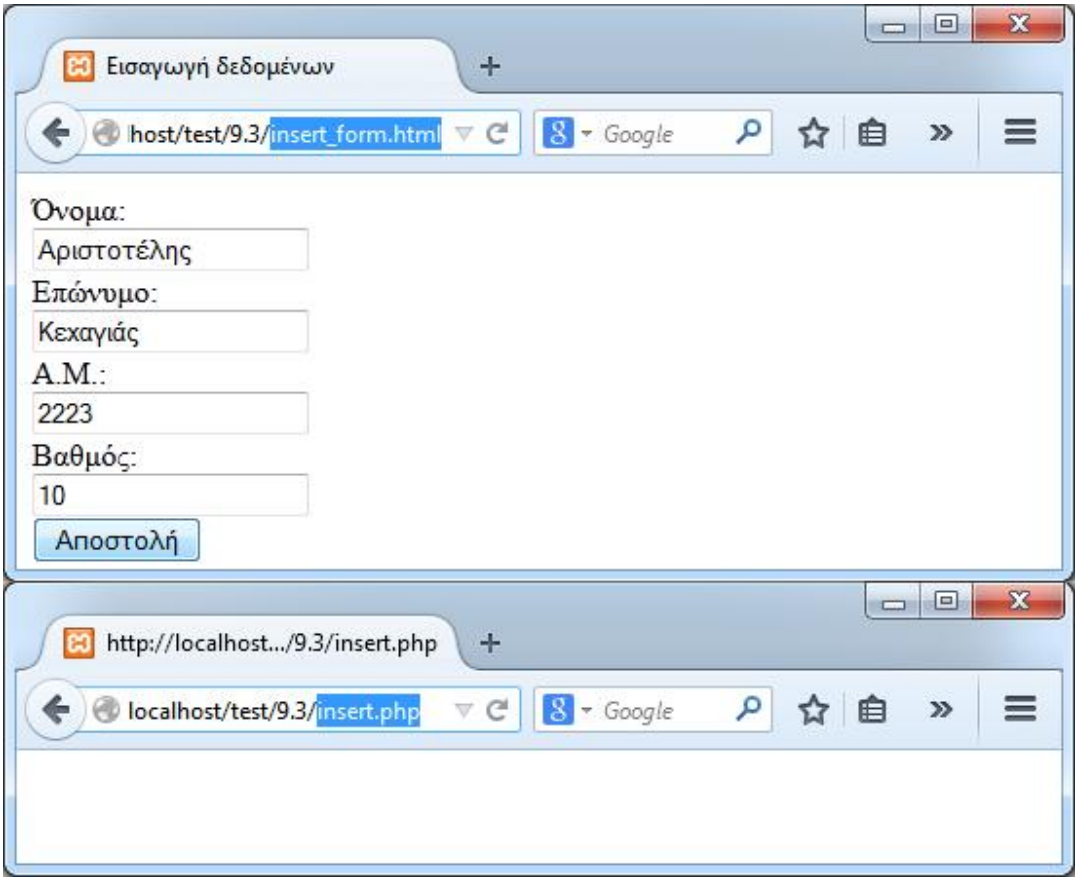
Αρχείο: insert_form.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Εισαγωγή δεδομένων</title>
 </head>
 <body>
  <form method='post' action='insert.php'>
   Όνομα:<br/>
   <input type='text' name='fname'><br/>
   Επώνυμο:<br/>
   <input type='text' name='lname'><br/>
   A.M.:<br/>
   <input type='text' name='snumber'><br/>
   Βαθμός:<br/>
   <input type='text' name='degree'><br/>
   <input type='submit' value='Αποστολή'>
  </form>
 </body>
</html>
```

Αρχείο: insert.php

```
<?php
 $conn = mysql_connect("localhost","root","");
 if(!$conn){
  die('Αδύνατη σύνδεση: ' . mysql_error());
 }

 mysql_select_db("dokimi",$conn);
 mysql_set_charset('utf8');

 $query="INSERT INTO students
  (`first_name`,`last_name`,`student_number`,`degree`)
  VALUES
   ('". $_POST['fname'] ."',
    '". $_POST['lname'] ."',
    '". $_POST['snumber'] ."',
    '". $_POST['degree'] . "'
   );";

 mysql_query("SET NAMES utf8");

 $result = mysql_query($query);

 if(!$result){
  die('Αδύνατη εγγραφή! :' . mysql_error());
 }

 mysql_close($conn);
?>
```

Ας εισάγουμε μια νέα εγγραφή με στοιχεία όνομα: Αριστοτέλης, επώνυμο: Κεχαγιάς, ΑΜ: 2223, και βαθμό 10.



Εικόνα Α.184 Εισαγωγή εγγραφής μέσω της php

Όταν κάνουμε κλικ στο «Αποστολή», τίποτε δεν εμφανίζεται στην νέα σελίδα. Αυτό συμβαίνει γιατί τίποτα δεν έχουμε ορίσει να τυπώνεται σε περίπτωση που δεν εμφανιστεί κάποιο πρόβλημα, επομένως το ερώτημα εκτελέστηκε σωστά, ας δούμε και την βάση δεδομένων μας:

	id	first_name	last_name	student_number	degree
Διαγραφή	1	Κωνσταντίνος	Παπαδόπουλος	3041	4
Διαγραφή	2	Δήμητρα	Αναστασοπούλου	2134	6
Διαγραφή	3	Γιώργος	Καρακίτσος	2222	5
Διαγραφή	4	Διονύσιος	Δεσύλλας	2223	7
Διαγραφή	5	Αριστοτέλης	Κεχαγιάς	2223	10

Εικόνα Α.185 Αποτέλεσμα εισαγωγής στο phpmyadmin

Όπως με την SELECT έτσι και με την INSERT η mysql επιστρέφει ένα αποτέλεσμα σε μια μεταβλητή της επιλογής μας. Το αποτέλεσμα αυτό εάν είναι κενό, σημαίνει ότι

το ερώτημα δεν εκτελέστηκε σωστά και έτσι υπάρχει πρόβλημα εισαγωγής δεδομένων, έτσι προκύπτει και το «if(!$result)» το οποίο αναλαμβάνει τον ρόλο του ελέγχου. Παρομοίως το «mysql_error()» όπως και στην SELECT επιστρέφει το ανάλογο πρόβλημα της περίπτωσης.

Χρησιμοποιούμε το ερώτημα «mysql_query("SET NAMES utf8");» πριν την εκτέλεση του ερωτήματος εισαγωγής έτσι ώστε οι ελληνικοί χαρακτήρες να αναγνωριστούν από την βάση δεδομένων μας.

A.9.3.4 Ενημέρωση

Παρόμοια με την εισαγωγή και την ανάγνωση υπάρχει και η ενημέρωση. Όταν σε μια υπάρχουσα εγγραφή θέλουμε να αλλάξουμε μια τιμή, π.χ. να επεξεργαστούμε κάποιο λάθος όπως αυτό που κάναμε στην εντολή INSERT εισάγαμε μια εγγραφή με αριθμό μητρώο φοιτητή το οποίο υπάρχει ήδη. Αυτό θα ήταν ένα μεγάλο σφάλμα αν το πεδίο αριθμός μητρώου ήταν πεδίο-κλειδί για τον πίνακα students. Στην δικιά μας περίπτωση δεν είναι τεχνικό σφάλμα είναι όμως νοητό.

Για να ενημερώσουμε μια εγγραφή σε έναν πίνακα θα πρέπει να χρησιμοποιήσουμε την εντολή UPDATE. Για να χρησιμοποιηθεί σωστά η UPDATE θα πρέπει να της πούμε «ΠΟΥ» θα βρει αυτό που θέλουμε να αλλάξουμε, επομένως χρησιμοποιούμε την εντολή «WHERE» σε συνδυασμό με την «UPDATE».

```
UPDATE students SET student_number = 2221
WHERE last_name = 'Κεχαγιάς';
```

Στην ουσία αυτό που λέει παραπάνω είναι:

```
ΕΝΗΜΕΡΩΣΕ ΤΟΝ ΠΙΝΑΚΑ students ΘΕΣΕ student_number = 2221
ΟΠΟΥ last_name = 'Κεχαγιάς';
```

Ας δούμε πως πραγματοποιείται το παραπάνω με χρήση της php.

Αρχείο: update.php

```php
<?php
 $conn = mysql_connect("localhost","root","");
 if(!$conn){
  die('Αδύνατη σύνδεση: ' . mysql_error());
 }

 mysql_select_db("dokimi",$conn);
 mysql_set_charset('utf8');

 mysql_query("SET NAMES utf8");

 $result = mysql_query("UPDATE students SET student_number = 2221
WHERE last_name = 'Κεχαγιάς';");

 if(!$result){
  die('Η ενημέρωση απέτυχε :' . mysql_error());
 }

 mysql_close($conn);
?>
```

Αν τρέξουμε το παραπάνω αρχείο δεν θα μας εμφανίσει τίποτα στην οθόνη, αν δούμε όμως τον πίνακα «students» στο phpmyadmin θα παρατηρήσουμε την αλλαγή εκεί ακριβώς όπου την θέλαμε.

	id	first_name	last_name	student_number	degree
Διαγραφή	1	Κωνσταντίνος	Παπαδόπουλος	3041	4
Διαγραφή	2	Δήμητρα	Αναστασοπούλου	2134	6
Διαγραφή	3	Γιώργος	Καρακίτσος	2222	5
Διαγραφή	4	Διονύσιος	Δεσύλλας	2223	7
Διαγραφή	5	Αριστοτέλης	Κεχαγιάς	2221	10

Εικόνα Α.186 Ενημέρωση γραμμής μέσω της php και της εντολής UPDATE

Α.9.3.5 Διαγραφή

Η διαγραφή μιας εγγραφής γίνεται μέσω της εντολής DELETE σε συνδυασμό με την WHERE. Για παράδειγμα:

```
DELETE FROM students WHERE student_number = 2222;
```

Η παραπάνω εντολή αν εισαχθεί σε phpmyadmin θα διαγράψει την εγγραφή με student_number 2222.

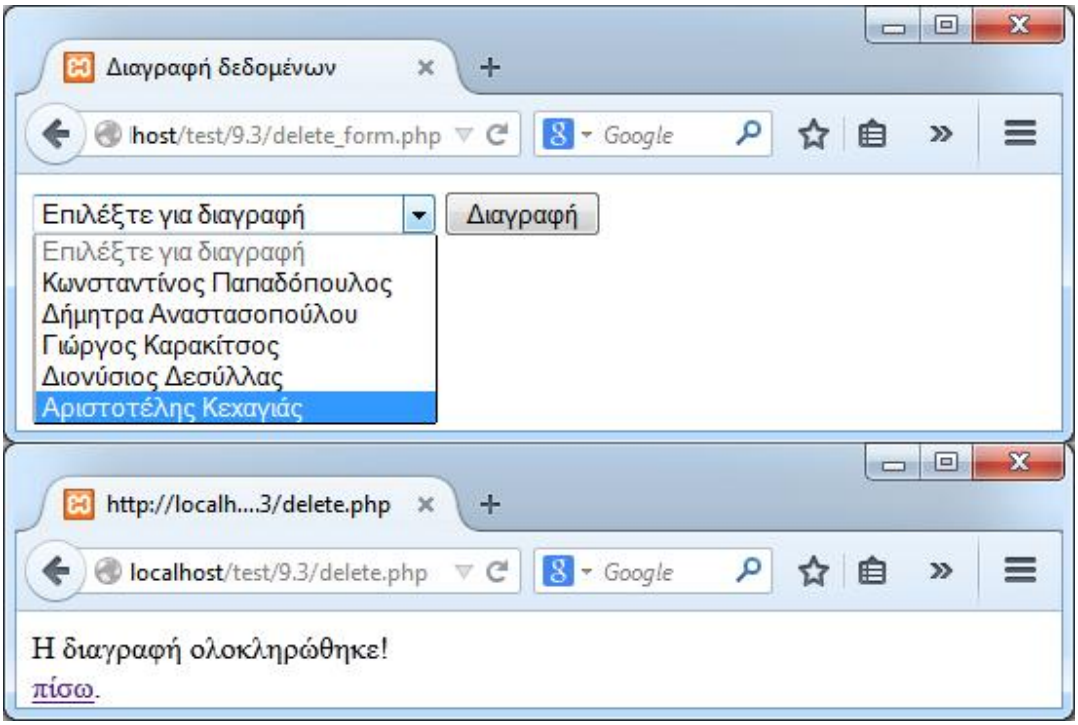
Αρχείο: delete_form.php

```php
<?php
 $conn=mysql_connect("localhost","root","rootroot");
 if(!$conn){
  die('Αδύνατη σύνδεση: ' .mysql_error());
 }

 mysql_select_db("dokimi",$conn);
 mysql_query("SET NAMES utf8");
 $result = mysql_query("SELECT * FROM students");
?>
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Διαγραφή δεδομένων</title>
 </head>
 <body>
  <form method='post' action='delete.php'>
   <select name='delid'>
    <option value="" disabled selected>Επιλέξτε για διαγραφή</option>
    <?php
     while($row=mysql_fetch_array($result)){
       echo "<option value='".$row['id']."'>".$row['first_name']."
".$row['last_name']."</option>";
     }
    ?>

   </select>
   <input type='submit' value='Διαγραφή'>
  </form>
 </body>
</html>
<?php
 mysql_close($conn);
?>
```

Αρχείο: delete.php

```php
<?php
 $conn = mysql_connect("localhost","root","");
 if(!$conn){
  die('Αδύνατη σύνδεση: ' . mysql_error());
 }

 mysql_select_db("dokimi",$conn);

 $query = "DELETE FROM students WHERE id = ".$_POST['delid'].";";

 mysql_query("SET NAMES utf8");
 $result = mysql_query($query);

 if(!$result){
  die('Η διαγραφή απέτυχε :' . mysql_error());
 }

 echo "Η διαγραφή ολοκληρώθηκε!<br/> <a
href='delete_form.php'>πίσω</a>.";
 mysql_close($conn);
?>
```

Αποτέλεσμα:



Εικόνα Α.187 Διαγραφή εγγραφής με την χρήση php

Μετά την διαγραφή ο πίνακας θα εμφανίζεται στο phpmyadmin έτσι:

	id	first_name	last_name	student_number	degree
Διαγραφή	1	Κωνσταντίνος	Παπαδόπουλος	3041	4
Διαγραφή	2	Δήμητρα	Αναστασοπούλου	2134	6
Διαγραφή	3	Γιώργος	Καρακίτσος	2222	5
Διαγραφή	4	Διονύσιος	Δεσύλλας	2223	7

Εικόνα Α.188 Επιβεβαίωση διαγραφής μέσα από το phpmyadmin

Παράρτημα Β:JAVASCRIPT

Javascript

Οδηγός εκμάθησης

ΣΠΟΥΔΑΣΤΗΣ:Κωνσταντίνος Κολέτσος

Επιβλέπων καθηγητής: Κωνσταντίνος Γιωτόπουλος

Πάτρα – Μάιος 2015

Β.1 Εισαγωγη

Η Javascript είναι μια γλώσσα ελαφριάς μορφής προγραμματισμού. Ο κώδικας της είναι άμεσα εκτελέσιμος και εισάγεται συνήθως μέσα σε ένα έγγραφο html. Είναι μια γλώσσα open scripting, έτσι μπορεί να χρησιμοποιηθεί οπουδήποτε και χωρίς άδεια ή δικαιώματα, υποστηρίζεται από όλα τα γνωστά προγράμματα περιήγησης, καθώς επίσης πλέον και από πολλά κινητά. Η javascript είναι ένα σημαντικό εργαλείο στην ανάπτυξη δυναμικών ιστοσελίδων, διαδικτυακών εφαρμογών, καθώς και εφαρμογών τεχνολογίας Ajax.

Για να μπορέσει κάποιος να μάθει javascript θα πρέπει πρώτα να γνωρίζει HTML και CSS. Σε αυτό το κεφάλαιο θα μάθουμε να δημιουργούμε μια javascript ιστοσελίδα και να τυπώνουμε διάφορων ειδών μηνύματα μέσω αυτής.

Β.1.1 Ιστοσελίδα και Javascript

Η χρήση της Javascript εξαρτάται από την ύπαρξη html εγγράφων. Χωρίς αυτά η Javascript είναι αδύνατο να εκτελεστεί. Συνεπακόλουθα ένα script θα πρέπει να εισαχθεί μέσα σε ένα html έγγραφο. Η Javascript χρησιμοποιεί την δομή και τα στοιχεία του εγγράφου hml για να δημιουργήσει δυναμικό περιεχόμενο. Οι αλλαγές που επιτυγχάνονται είναι συνήθως αλλαγές που επηρεάζουν τις ιδιότητες ενός στοιχείου, το περιεχόμενο του ή το (css) style του.

Δεν χρειαζόμαστε κάποιου είδους διαδικτυακού εξυπηρετητή για να εκτελεστεί η Javascript, αρκεί και μόνο να ανοίξουμε έστω και τοπικά το αρχείο που περιλαμβάνει τον κώδικα αυτόν και η εκτέλεση θα επέλθει κατά την φόρτωση του αρχείου από τον φυλλομετρητή.

Υπάρχουν τριών ειδών script που μπορούμε να εισάγουμε μέσα σε ένα έγγραφο html:

- Εμφωλευμένο script
  Βρίσκεται μέσα σε μια ετικέτα
- Εσωτερικό script
  Βρίσκεται μέσα σε ένα έγγραφο ανάμεσα σε ετικέτες «<script>» και «</script>».
- Εξωτερικό script

Βρίσκεται σε ξεχωριστό αρχείο με προέκταση «.js» και καλείται μέσα από το έγγραφο html.

B.1.2 Το πρώτο Javascript script

Υπάρχουν πολλοί τρόποι για να χρησιμοποιήσουμε javascript σε ένα έγγραφο html. Σε αυτή την ενότητα περιγράφονται ολοι οι πιο κοινοί τρόποι για να γίνει αυτό.

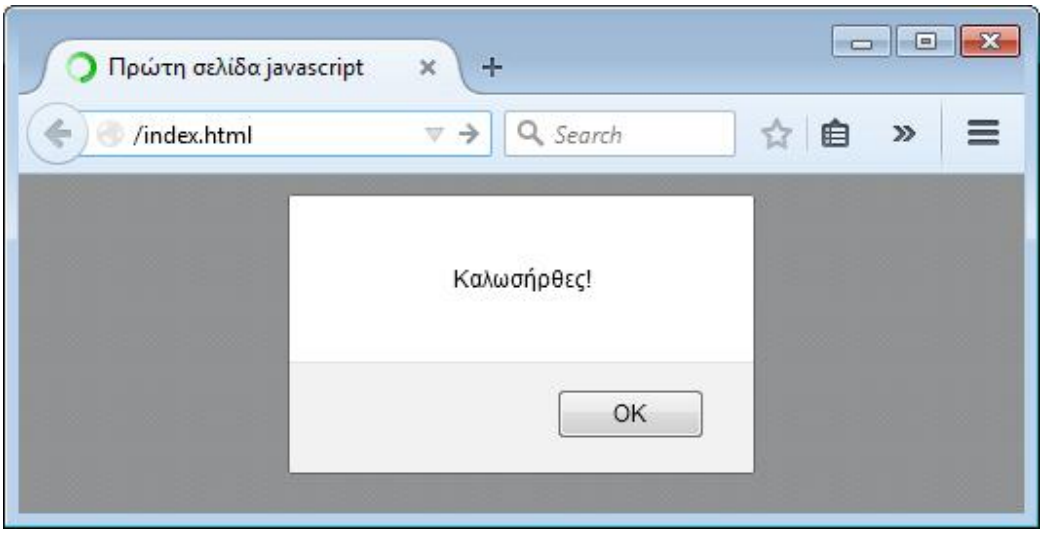
B.1.2.1 Εσωτερικό script

Για να εισάγουμε ένα ή περισσότερα script σε ένα έγγραφο html θα πρέπει να χρησιμοποιήσουμε την ετικέτα «<script>» για άνοιγμα και «</script>» για κλείσιμο. Στο συγκεκριμένο παράδειγμα το script θα εκτελεστεί μόλις η φόρτωση της σελίδας φτάσει στο συγκεκριμένο σημείο όπου υπάρχει γραμμένο το script.

Αρχείο: first_js.html

```
<html>
 <head>
  <meta charset="utf-8">
  <title>Πρώτη σελίδα javascript</title>
 </head>
 <body>
  <script>
   alert("Καλωσήρθες!");
  </script>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα B.1 Αποτέλεσμα εκτέλεσης του εσωτερικού script javascript

Η εντολή «alert» μας επιστρέφει ένα αναδυόμενο μήνυμα του οποίου το περιεχόμενο καθορίζεται από το τι έχουμε εισάγει μέσα στις παρενθέσεις της.

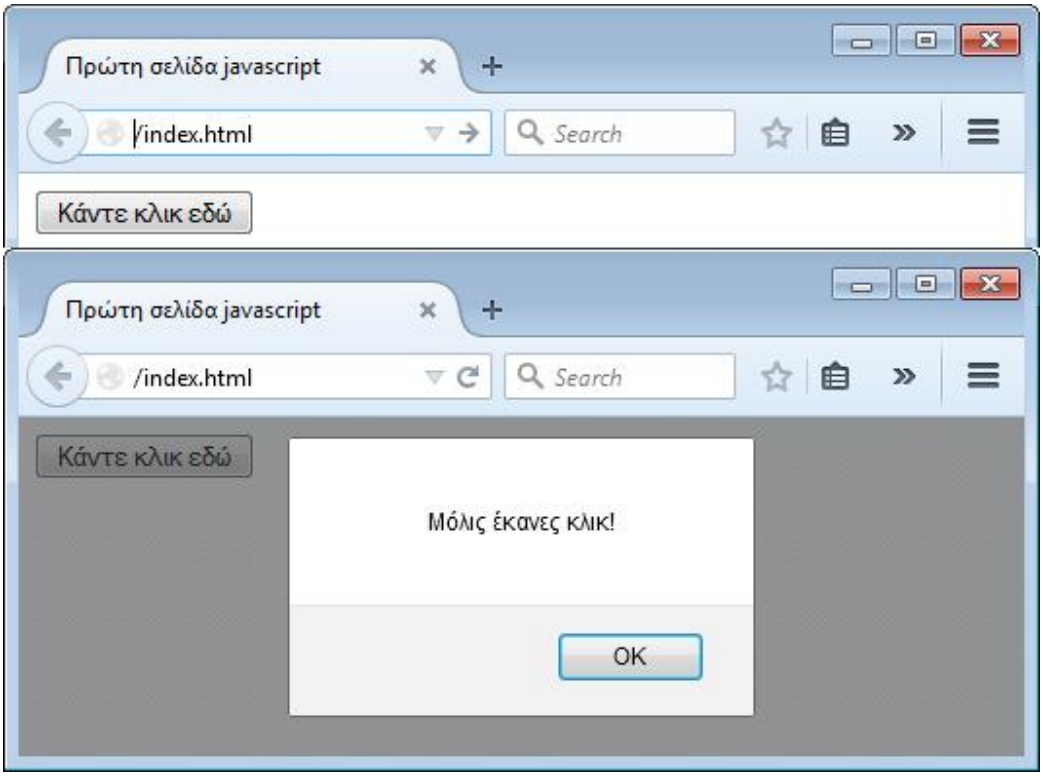
B.1.2.2 Εμφωλευμένο script

Για να εισάγουμε ένα εμφωλευμένο script μέσα σε ένα έγγραφο html, θα πρέπει να εισάγουμε τον κώδικα Javascript μέσα σε κάποια ιδιότητα-παράμετρο κάποιου στοιχείου. Έτσι δεν χρειάζεται να χρησιμοποιηθούν οι ετικέτες «<script>» και «</script>».

Αρχείο: first_button.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Πρώτη σελίδα javascript</title>
 </head>
 <body>
  <input type='button' value='Κάντε κλικ εδώ' onClick="alert('Μόλις
έκανες κλικ!')">
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.2 Αποτέλεσμα εκτέλεσης εμφωλευμένου javascript

Όταν υπάρχουν εισαγωγικά μέσα σε εισαγωγικά τότε (όπως στην παραπάνω περίπτωση) τα εξωτερικά με τα εσωτερικά εισαγωγικά θα πρέπει να είναι διαφορετικά. Στην συγκεκριμένη χρησιμοποιούμε ως εξωτερικά τα διπλά εισαγωγικά, ενώ ως εσωτερικά τα μονά. Θα μπορούσε να ήταν αντίθετα, αλλά όχι ίδια και στις δύο περιπτώσεις για να μην συγχέεται το πρόγραμμα περιήγησης.

Στην προκειμένη περίπτωση χρησιμοποιείται το event «onClick» που σημαίνει πως όταν κάποιος κάνει κλικ πάνω στο συγκεκριμένο στοιχείο html τότε θα εκτελεστεί ο κώδικας που βρίσκεται μέσα στα εισαγωγίκα «"», «"».

Β.1.2.3 Εξωτερικό script

Αν θέλουμε να καλέσουμε ένα εξωτερικό script από εξωτερική πηγή, τότε χρησιμοποιούμε την ετικέτα «script» και την παράμετρο «src».

Αρχείο: script.js

```
alert("Καλωσήρθατε!");
```

Αρχείο: first_script.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Πρώτη σελίδα javascript</title>
  <script type='text/javascript' src='first_script.js'></script>
 </head>
 <body>
 </body>
</html>
```

Επειδή το αρχείο script βρίσκεται στο ίδιο φάκελο με το αρχείο html τότε μέσα στο «src» απλά καλούμε το όνομα του. Αν βρισκόταν σε άλλο directory θα βάζαμε το full directory του.

Η ιδιότητα «type» μίας ετικέτας script μας δείχνει, τι είδους script θα εισάγουμε μέσα σε αυτή την ετικέτα. Στην προκειμένη περίπτωση για την javascript εισάγουμε ως περιεχόμενο ιδιότητας το λεκτικό «text/javascript», με αυτόν τον τρόπο ο περιηγητής γνωρίζει ότι μέσα στις ετικέτες βρίσκεται κείμενο ή και κώδικας javascript.

B.1.3 Έξοδοι αποτελεσμάτων

Για να μπορέσουμε να αλληλεπιδράσουμε με τον χρήστη θα πρέπει να μπορούμε να του παρουσιάζουμε αντίστοιχα μηνύματα αποτελεσμάτων, είτε άμεσα, είτε έμμεσα. Στην παρούσα ενότητα θα δούμε τους πιο απλούς και σύντομους τρόπους για να το πετύχουμε αυτό.

B.1.3.1 alert

Το alert είναι μια εντολή με την οποία μπορούμε να παρουσιάσουμε αναδυόμενα παράθυρα στον χρήστη με μόνο περιεχόμενο απλό κείμενο. Η δομή του είναι απλή, ένα απλό αναδυόμενο παράθυρο με ένα κουμπί OK στο κέντρο, το οποίο μόλις ο χρήστης το πιέσει το αναδυόμενο παράθυρο κλείνει. Η εκτέλεση της εντολής αυτής είναι σύγχρονη, όταν δηλαδή το script μας φτάσει σε αυτήν, περιμένει το «οκ» του χρήστη προκειμένου να συνεχίσει την εκτέλεση του υπόλοιπου script.

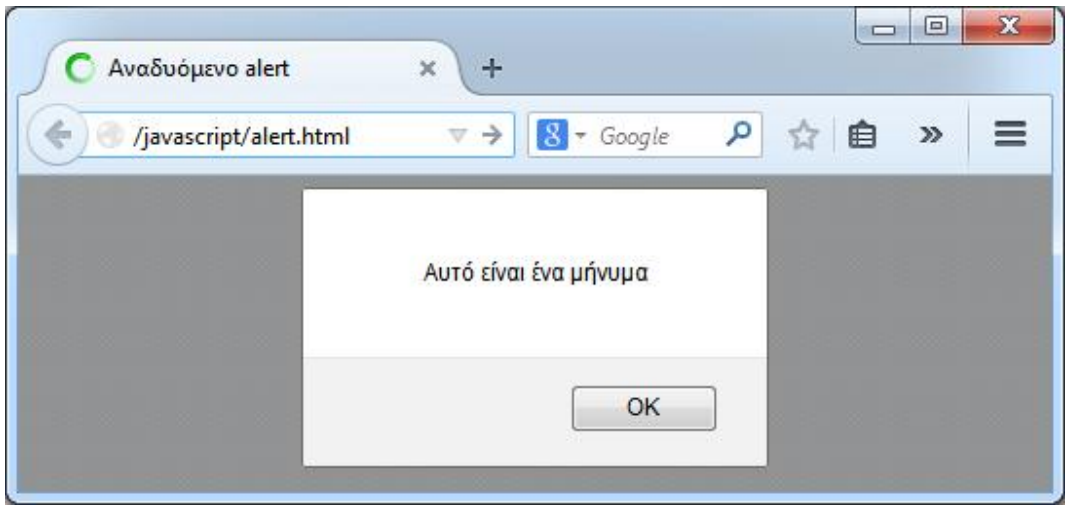
```
alert("Αυτό είναι ένα μήνυμα");
```

Όπου «Αυτό είναι ένα μήνυμα» εισάγουμε το μήνυμα μας. Επίσης η alert μπορεί να δεχθεί και μεταβλητές μαζί με αλφαριθμητικά κείμενα για να εμφανίσει σε αναδυόμενα παράθυρα πχ:

```
var num = 10;
alert("Αυτό είναι ένα μήνυμα με αριθμό: "+num);
```

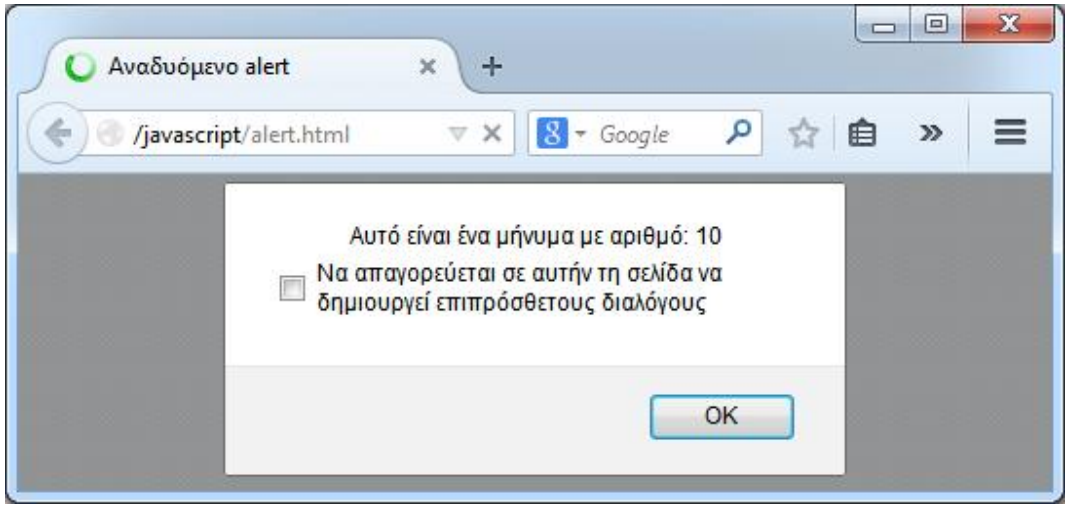
Οι παραπάνω δυο εντολές θα μας εμφανίσουν στο αναδυόμενο παράθυρο το παρακάτω αποτέλεσμα:

Το πρώτο alert:



Εικόνα Β.3 Παράδειγμα χρήσης alert σε firefox

Το δεύτερο:



Εικόνα Β.4 Αποτροπή πολλαπλών αναδυόμενων μηνυμάτων απο τον firefox

Στην Εικόνα Β.4, υπάρχει μια πρόσθετη επιλογή πέρα από το μήνυμα μας. Αυτό γίνεται γιατί η χρήση των αναδυόμενων παραθύρων είναι σύγχρονη με την εκτέλεση του κώδικα στην σελίδα, έτσι θα μπορούσε κάποιος, αν μέσα σε μια επανάληψη με

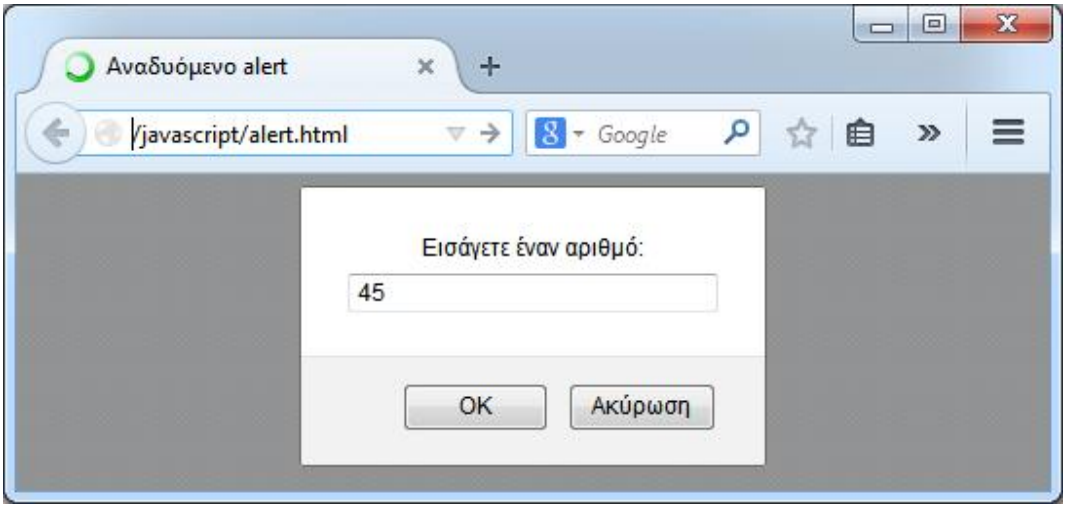
μεγάλο αριθμό επαναλήψεων υπήρχε ένα τέτοιο alert, τα αναδυόμενα παράθυρα που είχε θέσει να μην τελείωναν ποτέ. Έτσι οι φυλλομετρητές πλέον διαθέτουν τέτοιου είδους ασφάλειες όπου μπορούμε να απενεργοποιήσουμε μια σελίδα από το να μας προβάλει επιπρόσθετους διαλόγους κατά σειρά.

B.1.3.2 prompt

Το prompt είναι μια εντολή αναδυόμενου παραθύρου, λίγο πιο διαφορετική από το alert. Το prompt δεν εμφανίζει απλά ένα μήνυμα αλφαριθμητικού κειμένου, αλλά ταυτόχρονα μαζί με αυτό εμφανίζει επίσης ένα πεδίο κειμένου έτσι ώστε να μπορέσει ο χρήστης να εισάγει μία τιμή μέσα σε αυτό. Επίσης διαθέτει δυο κουμπιά διαχείρισης, ένα «OK» και ένα «Ακύρωση». Η τιμή που θα εισαχθεί μέσα στο πεδίο κειμένου αντιστοιχείται αυτόματα σε μια μεταβλητή που ορίζουμε εμείς μέσα στον κώδικα. Παρόμοια με την alert έτσι και η prompt είναι μια σύγχρονη μορφή εμφάνισης αναδυόμενου παραθύρου όπου, όταν φτάνει η εκτέλεση του script σε αυτή την εντολή τότε ο φυλλομετρητής περιμένει μέχρι ο χρήστης να πιέσει ένα από τα δυο διαθέσιμα κουμπιά για να προχωρήσει παρακάτω. Αν ο χρήστης πιέσει το «OK», η τιμή που έχει γραφτεί μέσα στο πεδίο κειμένου αμέσως αναθέτεται ως τιμή στην αντίστοιχη μεταβλητή που έχουμε ορίσει. Αν ο χρήστης πιέσει το «Ακύρωση» τότε η μεταβλητή παίρνει την τιμή «null» και το πρόγραμμα προχωράει παρακάτω. Η εντολή αυτή παίρνει δυο ορίσματα ως είσοδο για την εκτέλεση της, το πρώτο είναι το κείμενο-μήνυμα που θα εμφανίζεται ως ετικέτα για το πεδίο κειμένου, και το δεύτερο είναι μια προεπιλεγμένη τιμή μέσα στο πεδίο κειμένου:

```
var num = prompt("Εισάγετε έναν αριθμό:",45);
```

Αυτό θα μας δώσει το παρακάτω αποτέλεσμα:

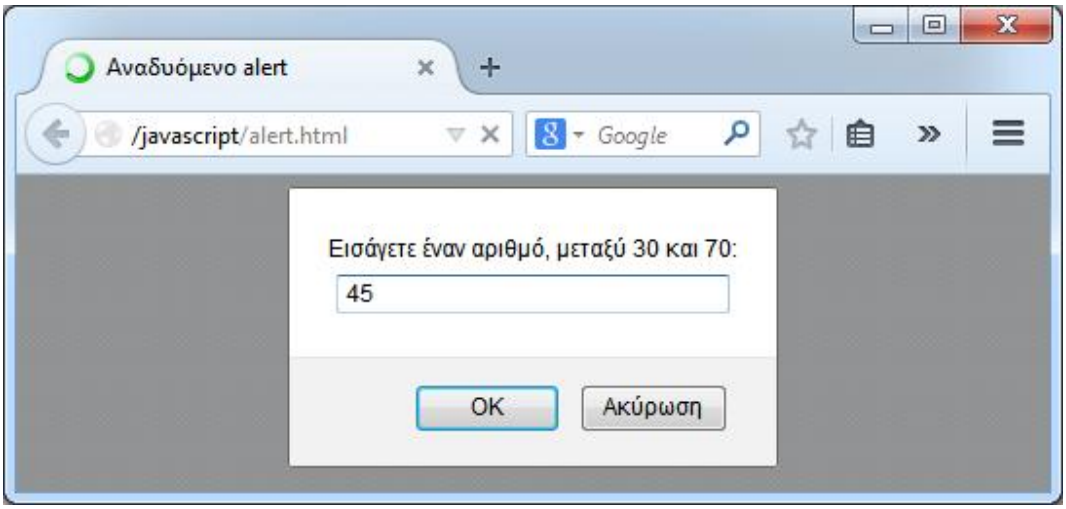


Εικόνα Β.5 Παράδειγμα προεπιλεγμένης τιμής στην prompt σε firefox

Παρόμοια με την alert έτσι κι εδώ μέσα στις παραμέτρους της εντολής μπορούμε να χρησιμοποιήσουμε μεταβλητές πχ:

```
var min = 30;
var max = 70;
var num = prompt("Εισάγετε έναν αριθμό, μεταξύ "+min+" και
"+max+":",45);
```

Αποτέλεσμα:



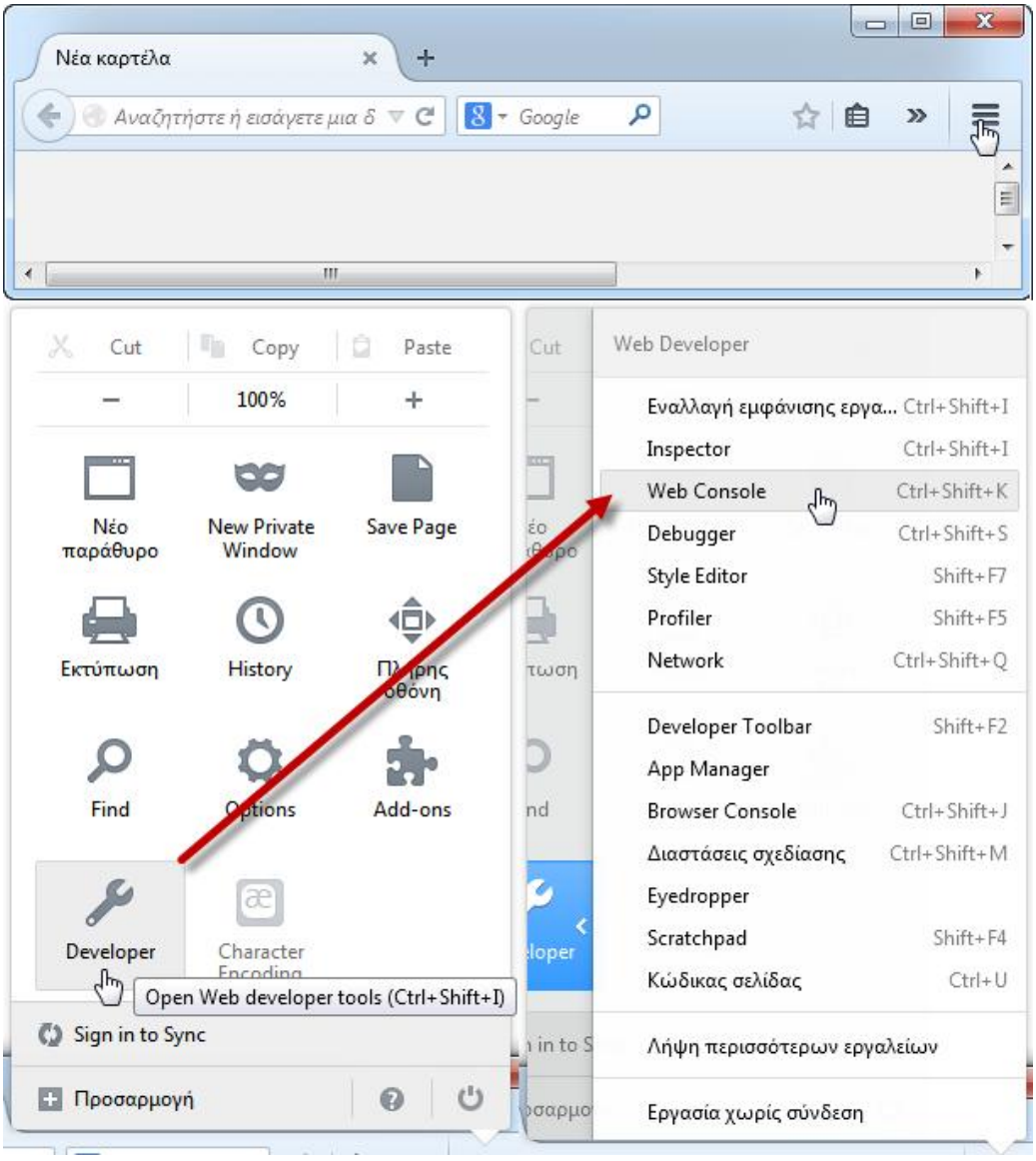
Εικόνα Β.6 Δοκιμή της prompt σε firefox

Β.1.3.3 console

Εκτός από τα αναδυόμενα παράθυρα, υπάρχει κι άλλος τρόπος εξαγωγής αποτελεσμάτων, αλλά όχι τόσο φιλική προς τον χρήστη. Για να μπορέσει να

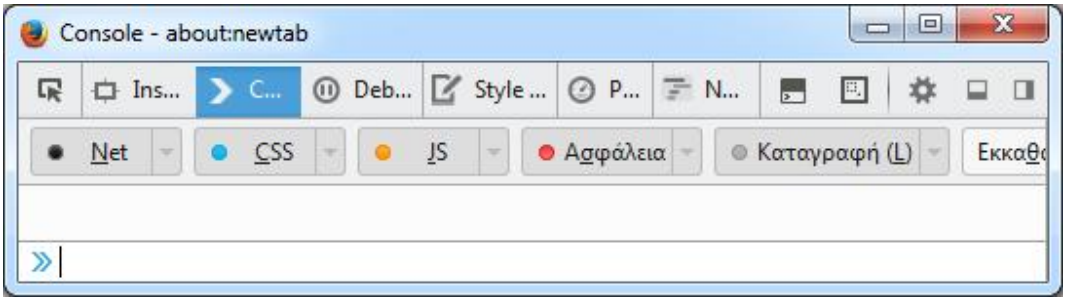
χρησιμοποιήσει κάποιος αυτή την μέθοδο εξαγωγής πληροφοριών θα πρέπει πρώτα να γνωρίζει πώς να χρησιμοποιήσει την κονσόλα ενός φυλλομετρητή μέσω του οποίου καταγράφονται όλα τα συμβάντα μια σελίδας.

Στον firefox:



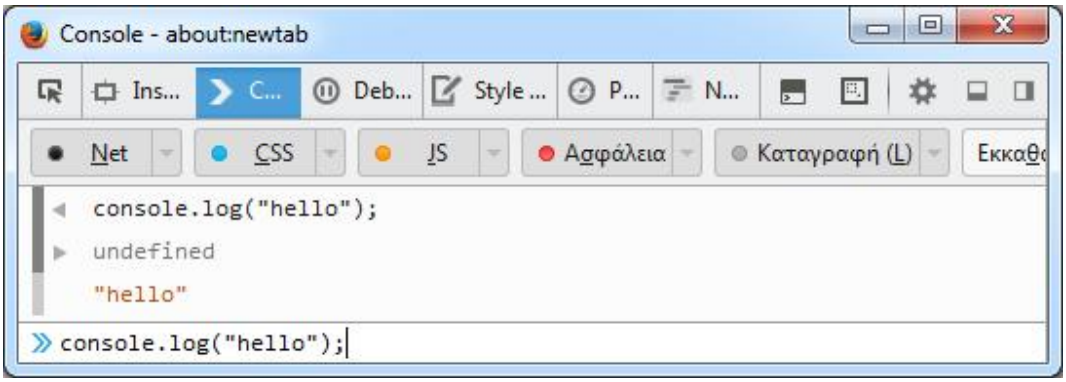
Εικόνα Β.7 Επιλογή εμφάνισης κονσόλας στον firefox

Αν ακολουθήσουμε σωστά την διαδικασία θα μας εμφανίσει το εξής παράθυρο:



Εικόνα Β.8 Η κονσόλα του firefox

Σε αυτό το παράθυρο είναι το σημείο στο οποίο θα βλέπουμε τα αποτελέσματα των εντολών μας. Κάτω αριστερά υπάρχει ένα διπλό γαλάζιο εισαγωγικό βελάκι στο οποίο εάν θέλουμε μπορούμε να γράψουμε μια εντολή javascript και πατώντας enter να την εκτελέσουμε άμεσα εκείνη την στιγμή:



Εικόνα Β.9 Άμεση εκτέλεση εντολής μέσα απο την κονσόλα του firefox

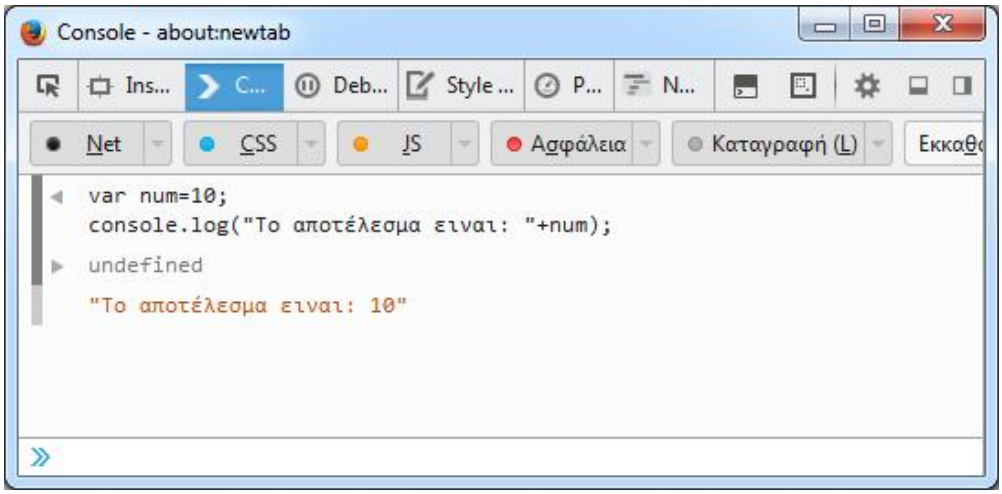
Β.1.3.3.1 console.log(μήνυμα)

Είναι η απλούστερη μορφή εμφάνισης μηνύματος στην κονσόλα. Το μόνο που χρειάζεται είναι μια παράμετρο που αντιπροσωπεύει το μήνυμα που θα πρέπει να εμφανίσει. Μπορεί να είναι οποιοδήποτε αλφαριθμητικό καθώς επίσης και συνδυασμός τους με μεταβλητές. Ο τρόπος σύνταξης είναι πολύ απλός. Υπάρχουν δύο τρόποι για να συμπεριληφθούν μεταβλητές μέσα στο μήνυμα.

1ος Συμπεριλαμβάνοντας απλά τις μεταβλητές με τον τελεστή «+» στα αλφαριθμητικά πχ:

```
var num=10;
console.log("Το αποτέλεσμα ειναι: "+num);
```

Το οποίο μας δίνει ως αποτέλεσμα:

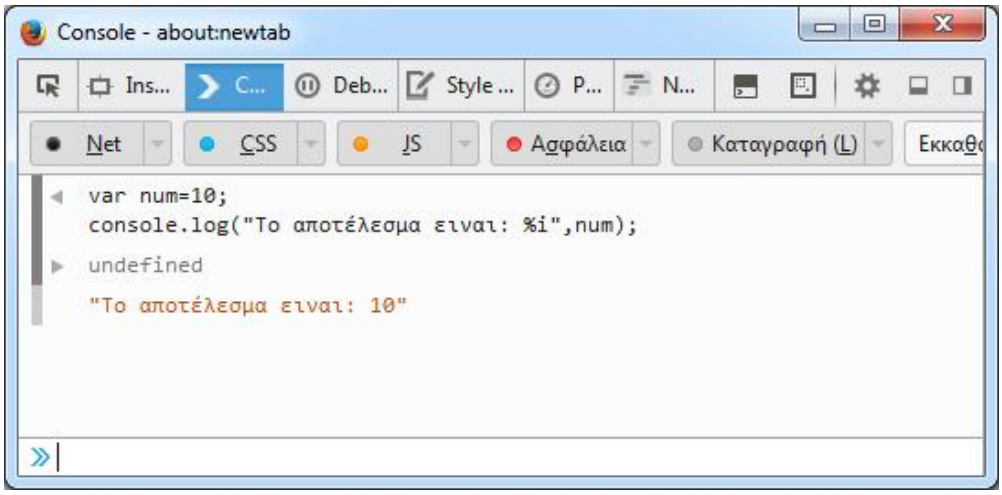


Εικόνα Β.10 Αποτέλεσμα εκτέλεσης console.log με συμπερίληψη μεταβλητών στον firefox

2ος Με την χρήση ειδικών χαρακτήρων:

```
var num=10;
console.log("Το αποτέλεσμα είναι: %i",num);
```

Αποτέλεσμα:



Εικόνα Β.11 Αποτέλεσμα εκτέλεσης console.log με χρήση ειδικών χαρακτήρων στον firefox

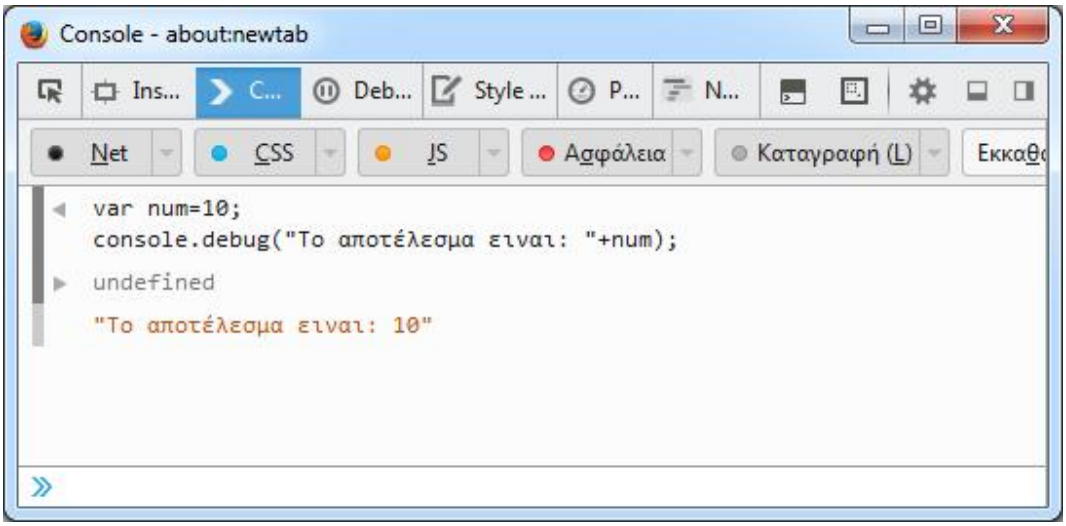
Το «%i» σημαίνει πως στο σημείο στο οποίο βρίσκεται μέσα στο μήνυμα θα τυπωθεί το περιεχόμενο της μεταβλητής num. Αν είχαμε περισσότερα «%i» τότε θα τυπώνονταν οι μεταβλητές με την σειρά στην οποία τα «%i» βρίσκονταν μέσα στο μήνυμα.

B.1.3.3.2 console.debug(μήνυμα)

Παρόμοια ακριβώς με την console.log τυπώνει ένα μήνυμα το οποίο υπάρχει πλαισιωμένο στις παρενθέσεις της εντολής. Η διαφορά με το console.log είναι ότι περιλαμβάνεται στην τύπωση του αποτελέσματος και hyperlink με το σημείο ακριβώς στον κώδικα από όπου καλέστηκε η εντολή.

```
var num=10;
console.debug("Το αποτέλεσμα ειναι: "+num);
```

Αποτέλεσμα:



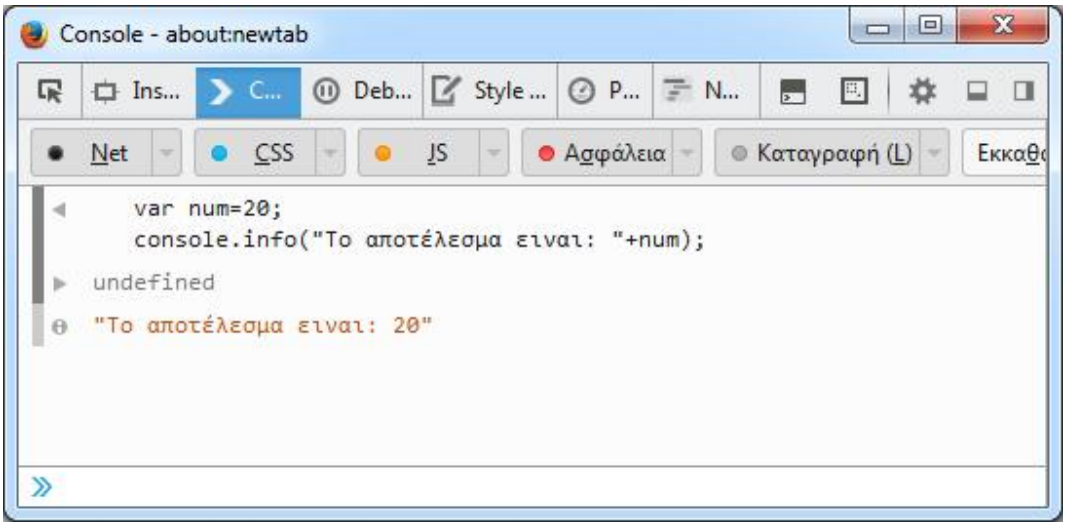
Εικόνα Β.12 Αποτέλεσμα εκτέλεσης console.debug στον firefox

B.1.3.3.3 console.info(μήνυμα)

Παρόμοια με την console.log και την console.debug μόνο που υπάρχει διαφορετικό είδος μορφοποίησης και εκτός αυτού θεωρείται το μήνυμα ως πληροφορία της εκτέλεσης του script. Διαθέτει κι αυτό hyperlink.

```
var num=30;
console.warn("Το ποσό καταχωρήθηκε: "+num);
```

Αποτέλεσμα:



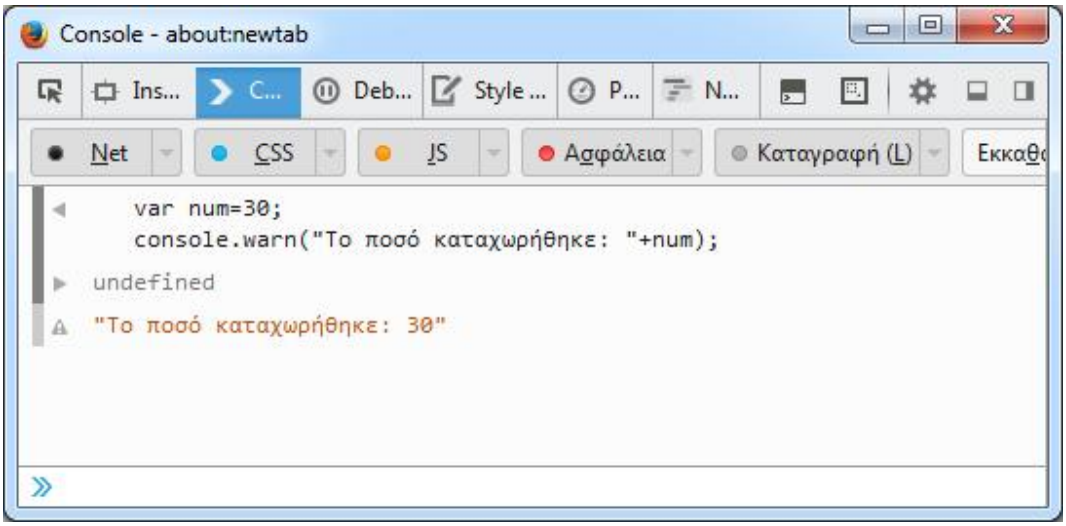
Εικόνα Β.13 Αποτέλεσμα εκτέλεσης console.info στον firefox

Β.1.3.3.4 console.warn(μήνυμα)

Παρόμοια με την console.info, διαθέτει όμως διαφορετική μορφοποίηση και διαφορετική κατηγοριοποίηση. Κατηγοριοποιείται στις προειδοποιήσεις.

```
var num=30;
console.warn("Το ποσό καταχωρήθηκε: "+num);
```

Αποτέλεσμα:



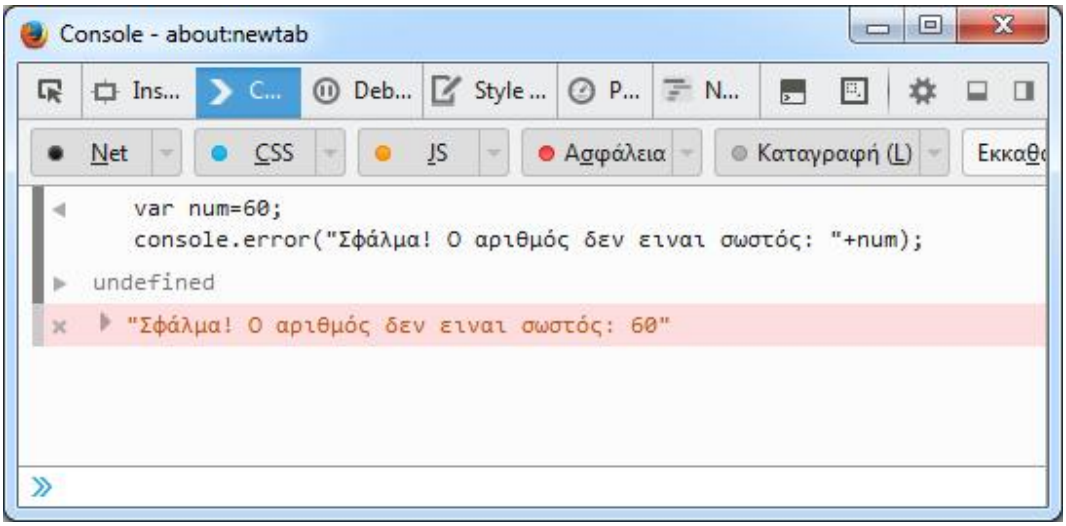
Εικόνα Β.14 Αποτέλεσμα εκτέλεσης console.warn στον firefox

Β.1.3.3.5 console.error(μήνυμα)

Παρόμοια με την console.warn, διαθέτει όμως διαφορετική μορφοποίηση και κατηγοριοποιείται στις ειδοποιήσεις σφαλμάτων.

```
var num=60;
console.error("Σφάλμα! Ο αριθμός δεν ειναι σωστός: "+num);
```

Αποτέλεσμα:



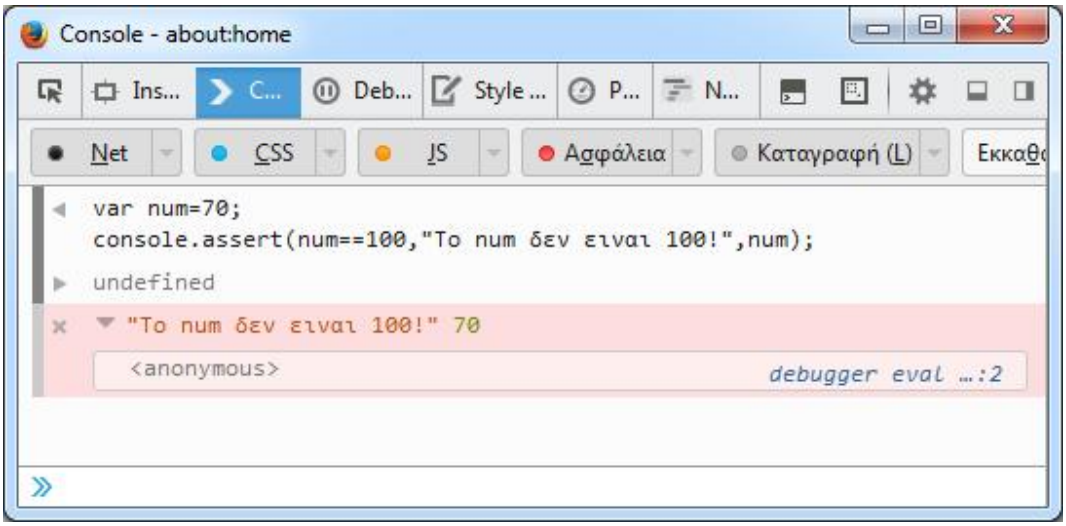
Εικόνα Β.15 Αποτέλεσμα εκτέλεσης console.error στον firefox

Β.1.3.3.6 console.assert(έλεγχος, μήνυμα, αντικείμενα)

Η assert είναι μια πιο προχωρημένη μέθοδος εκτύπωσης αποτελεσμάτων συγκριτικά με όλες τις προηγούμενες που είδαμε. Η διαφορά βρίσκεται στο ότι η συγκεκριμένη εντολή έχει διαφορετικό τρόπο λειτουργίας, δέχεται μια λογική συνθήκη ως πρώτο όρισμα, ύστερα ένα μήνυμα το οποίο εμφανίζεται μόνο όταν η συνθήκη αυτή δεν είναι αληθής, και τέλος μεταβλητές οι οποίες θα τυπωθούν μαζί με το μήνυμα.

```
var num=70;
console.assert(num==100,"Το num δεν ειναι 100!",num);
```

Αποτέλεσμα:



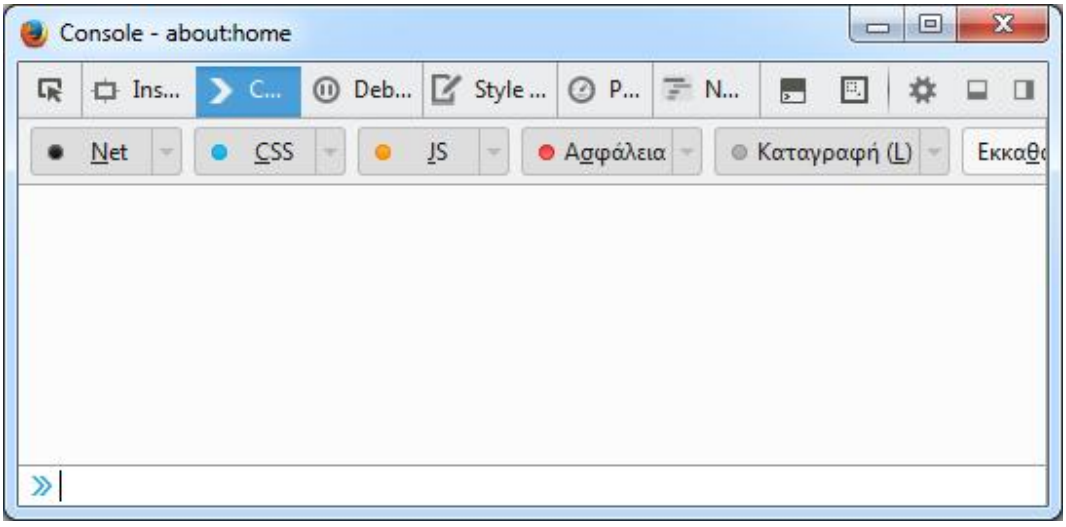
Εικόνα Β.16 Αποτέλεσμα εκτέλεσης console.assert στον firefox

Β.1.3.3.7 console.clear()

Εντολή μέσω τις οποίας «καθαρίζεται» η κονσόλα, διαγράφεται δηλαδή ότι έχει γραφτεί σε αυτήν. Δοκιμάσαμε το «console.clear()» σε firefox αλλά δεν λειτούργησε, αντ' αυτού λειτούργησε το «clear();».

```
clear();
```

Αποτέλεσμα:



Εικόνα Β.17 Αποτέλεσμα εκτέλεσης clear() στον firefox

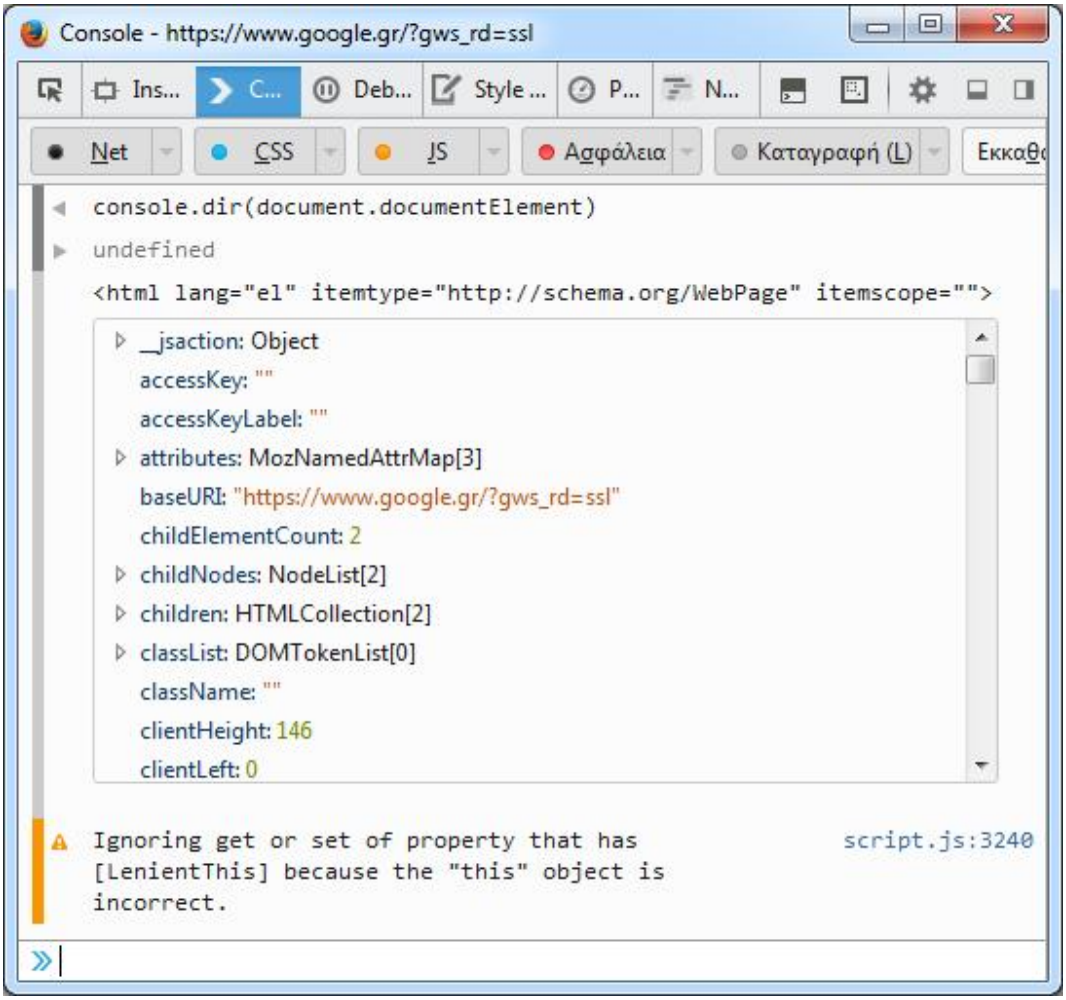
*Β.1.3.3.8 console.dir(**αντικείμενο**) , console.dirxml(**αντικείμενο**)*

Παρέχεται η τύπωση ιδιοτήτων ενός αντικειμένου το οποίο έχουμε εισάγει μέσα στις παρενθέσεις. Το document.documentElement αναφέρεται στην ίδια την ιστοσελίδα ως

αντικείμενο. Διαλέξαμε για την δοκιμή μας το site της google και εκτελέσαμε την εντολή μας μέσα από την κονσόλα:

```
console.dir(document.documentElement);
```

Αποτέλεσμα:



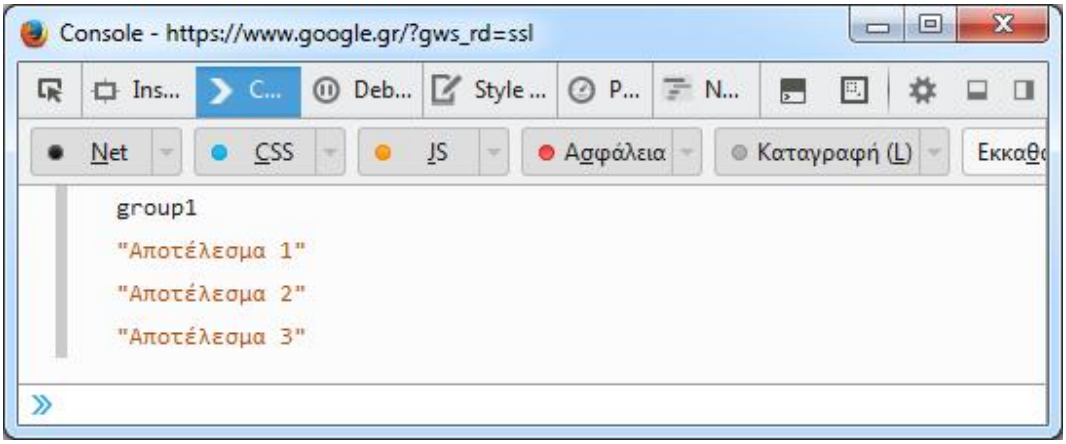
Εικόνα Β.18 Αποτέλεσμα εκτέλεσης console.dir στον firefox

*Β.1.3.3.9 console.group(**όνομα**) , console.groupCollapsed(**όνομα**)*

Πολλές φορές υπάρχει η ανάγκη για ομαδοποίηση των αποτελεσμάτων που προκύπτουν έτσι ώστε να μπορεί κάποιος που διαβάζει την κονσόλα να βγάλει εύκολα και σύντομα συμπεράσματα. Σε αυτό προσπαθούν να μας βοηθήσουν οι παραπάνω εντολές, μέσω των οποίων μπορούμε να ομαδοποιήσουμε αποτελέσματα.

```
console.group("group1");
console.log("Αποτέλεσμα 1");
console.log("Αποτέλεσμα 2");
console.log("Αποτέλεσμα 3");
console.groupEnd();
```

Αποτέλεσμα:



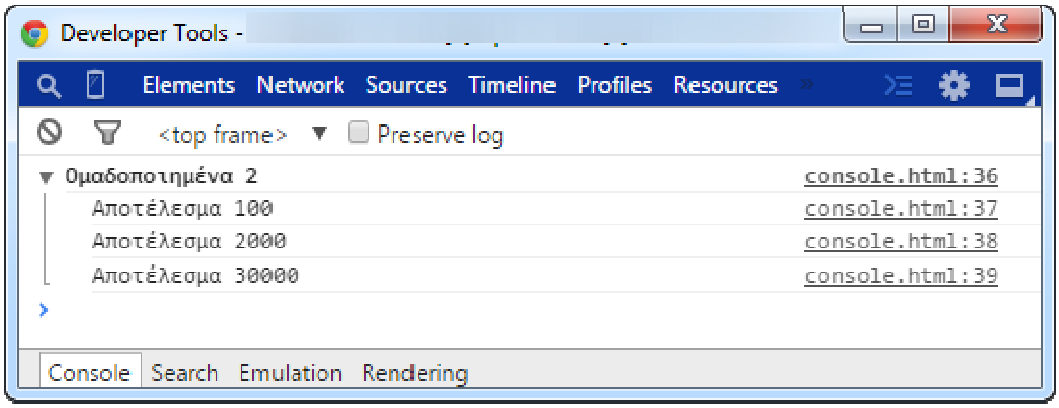
Εικόνα Β.19 Αποτέλεσμα εκτέλεσης console.group στον firefox

Γίνεται διαφορετική μορφοποίηση, καθώς επίσης και η τακτοποίηση των αποτελεσμάτων χωρικά. Το όρισμα που παίρνει το console.group είναι ο τίτλος της ομαδοποιημένης κατάστασης αποτελεσμάτων. Ύστερα χρησιμοποιούμε εντολές για να τυπώσουμε αποτελέσματα μέσα στην ομάδα, και στο τέλος κλείνουμε την ομάδα με το console.groupEnd.

Παρόμοια χρησιμοποιείται και η console.groupCollapsed μόνο που τα αποτελέσματα δεν φαίνονται άμεσα στο χρήστη αλλά είναι εκκολαπτόμενα κάνοντας κλικ πάνω στον τίτλο που θα εισάγουμε ως όρισμα στο groupCollapsed.

```
console.groupCollapsed("Ομαδοποιημένα 2");
console.log("Αποτέλεσμα 100");
console.log("Αποτέλεσμα 2000");
console.log("Αποτέλεσμα 30000");
console.groupEnd();
```

Στον firefox δεν λειτούργησε όπως θα περιμέναμε, ωστόσο μια δοκιμή στον Google Chrome μας έπεισε για την λειτουργικότητα του:



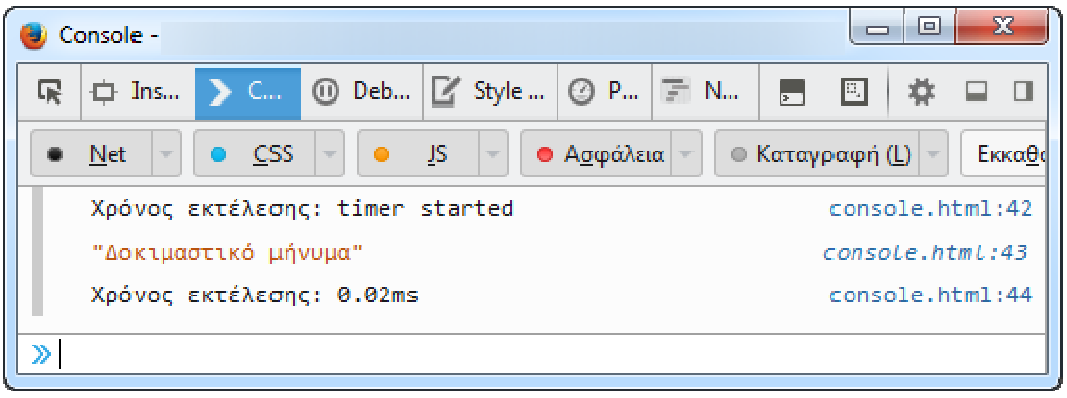
Εικόνα Β.20 Αποτέλεσμα εκτέλεσης console.groupCollapsed στον google chrome

Β.1.3.3.10 console.time , console.timeEnd

Μία χρήσιμη ακόμη εντολή εξαγωγής αποτελεσμάτων είναι και η time μέσω της οποίας μπορούμε να χρονομετρήσουμε τον χρόνο εκτέλεσης κάποιων γραμμών κώδικα καθώς αυτοί θα εκτελούνται. Η console.time() παίρνει ένα όρισμα μέσα στις παρενθέσεις της όπου είναι το όνομα του μετρητή, αυτό το όνομα θα το χρησιμοποιήσουμε αργότερα όταν θέλουμε να σταματήσουμε τον μετρητή με το console.timeEnd(). Μπορούμε να έχουμε πολλούς ταυτόχρονους χρονομετρητές στο ίδιο πρόγραμμα και να τους ξεκινάμε και να τους σταματάμε με την αντίστοιχη ετικέτα τους. Όταν ένας χρονομετρητής τελειώσει εμφανίζει την ετικέτα του και τον χρόνο εκτέλεσης σε miliseconds. Πχ:

```
console.time("Χρόνος εκτέλεσης");
console.log("Δοκιμαστικό μήνυμα");
console.timeEnd("Χρόνος εκτέλεσης");
```

Αποτέλεσμα:



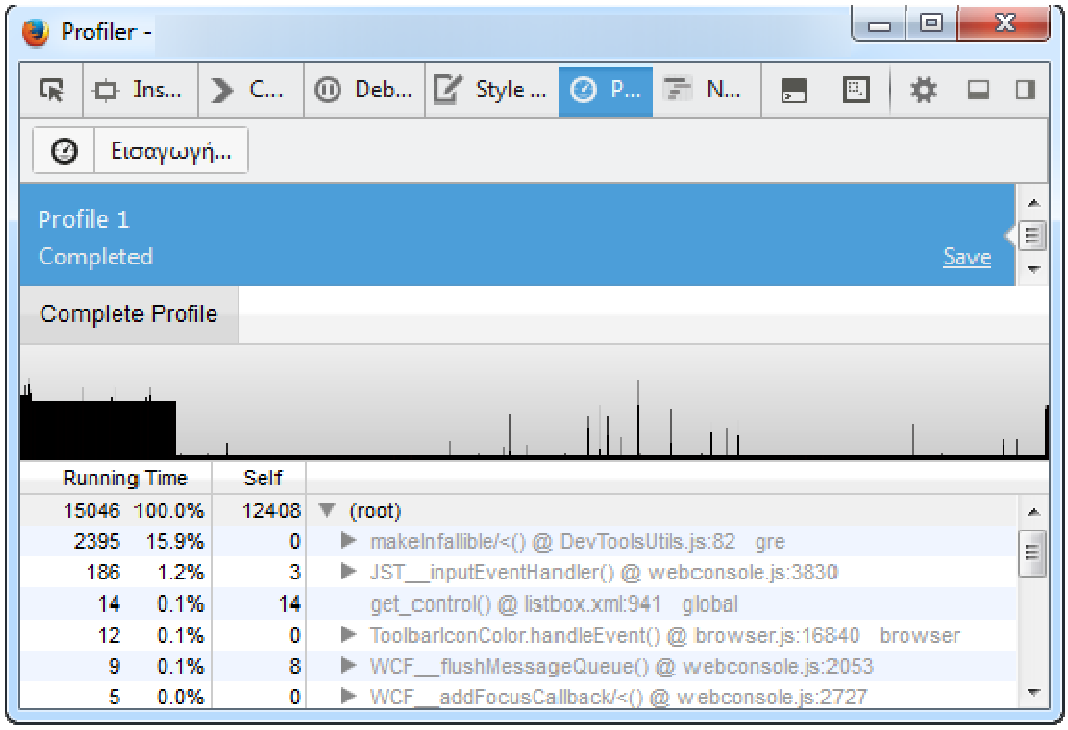
Εικόνα Β.21 Παράδειγμα αποτελέσματος εκτέλεσης console.time στον firefox

B.1.3.3.11 console.profile() , console.profileEnd()

Το profile είναι μια εντολή μέσω της οποίας μπορούμε να αντλήσουμε δεδομένα σχετικά με το τι συνέβη κατά την διάρκεια που το είχαμε ενεργό. Χρησιμοποιούμε το console.profile στο σημείο που θέλουμε να ξεκινήσει η διαδικασία καταγραφής και το console.profileEnd στο σημείο που θέλουμε να τελειώσει. Ανάμεσα σε αυτά τα δυο θα πρέπει να υπάρχει εκτελέσιμος κώδικας. Μόλις φθάσει στο profileEnd τα αποτελέσματα θα εμφανιστούν στην καρτέλα «profiler» της κονσόλας:

```
console.profile();
alert("Μήνυμα");
console.profileEnd();
```

Αποτέλεσμα:



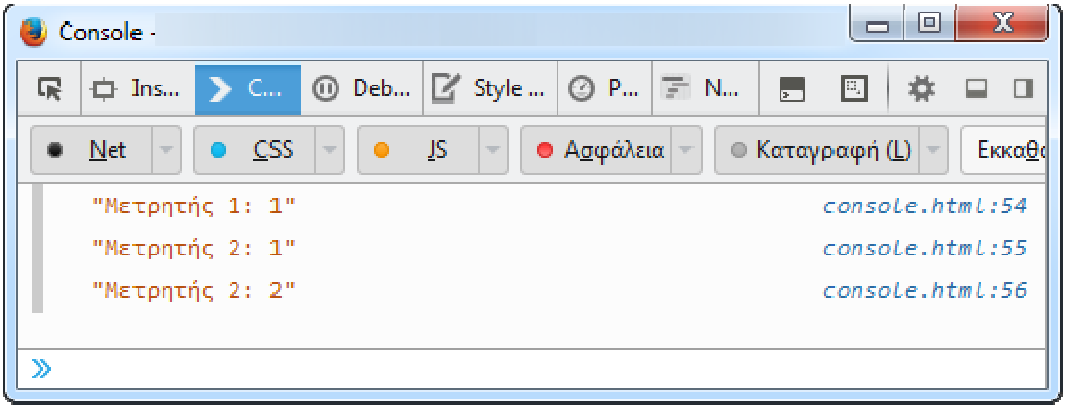
Εικόνα B.22 Αποτέλεσμα εκτέλεσης console.profile στον firefox

B.1.3.3.12 console.count

Η εντολή αυτή μας προσφέρει την δυνατότητα να μετράει όλες τις φορές που εμείς θα την καλέσουμε, είναι κάτι σαν ελεγχόμενος μετρητής μέσα στον κώδικα. Παρόμοια με άλλες εντολές της κονσόλας είναι δυνατό να δημιουργήσουμε μετρητές με αντίστοιχες ετικέτες:

```
console.count("Μετρητής 1");
console.count("Μετρητής 2");
console.count("Μετρητής 2");
```

Αποτέλεσμα:



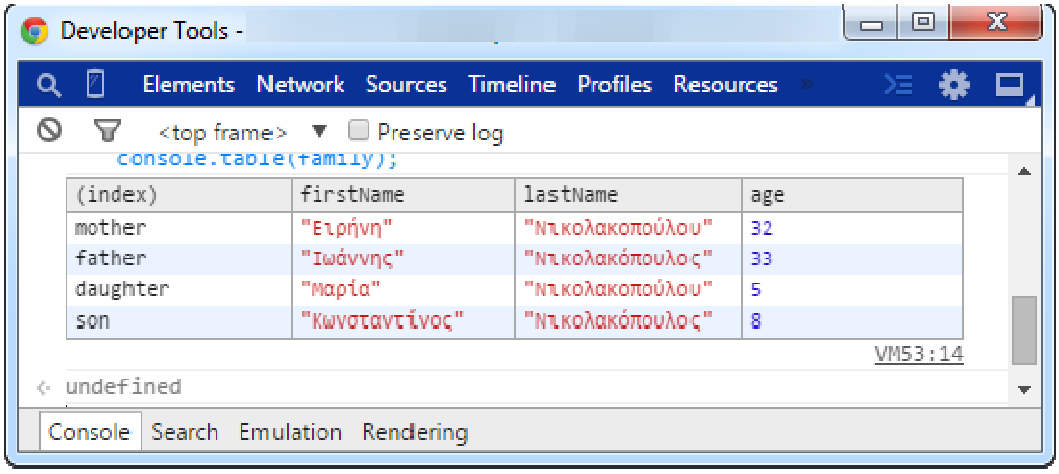
Εικόνα Β.23 Μέτρηση τυπώσεων από την console.count στον firefox

Β.1.3.3.13 console.table

Μας παρέχει την δυνατότητα να προβάλουμε αποτελέσματα σε μορφή πίνακα. Η διαδικασία είναι πολύπλοκη. Δεν λειτουργεί σε firefox γι αυτό παρακάτω το αποτέλεσμα είναι σε Google Chrome. Για να λειτουργήσει το console.table θα πρέπει να του εισάγουμε μια παράμετρο η οποία να έχει την μορφή αντικειμένου, ένα εύκολο και απλό παράδειγμα είναι το παρακάτω:

```
function Person(firstName, lastName, age){ //αντικείμενο Person
 this.firstName = firstName; //ιδιότητα firstName
 this.lastName = lastName; //ιδιότητα lastName
 this.age = age; //ιδιότητα age
}
var family = {}; //οικογένεια
 family.mother = new Person("Ειρήνη", "Νικολακοπούλου", 32);
//μητέρα
 family.father = new Person("Ιωάννης", "Νικολακόπουλος", 33);
//πατέρας
 family.daughter = new Person("Μαρία", "Νικολακοπούλου", 5);
//κόρη
 family.son = new Person("Κωνσταντίνος", "Νικολακόπουλος", 8);
//γιος
console.table(family);
```

Αποτέλεσμα:



Εικόνα B.24 Τύπωση πίνακα στην κονσόλα μέσω της console.table στον google chrome

Κάθε ιδιότητα του αντικειμένου γίνεται μια στήλη, ενώ για κάθε ρόλο που αποκτά μέσα στην οικογένεια ένα αντικείμενο ο ρόλος αναγράφεται σε κάθε γραμμή της πρώτης στήλης (index).

B.1.3.4 Document.Write

Η μέθοδος αυτή είναι μια ξεχωριστή λειτουργία σε σχέση με όλες όσες είδαμε. Η alert είναι άμεση και σύγχρονη μέθοδος εξαγωγής αποτελεσμάτων, ενώ η console.log είναι έμμεση , ασύγχρονη η οποία δεν προορίζεται για τον χρήστη. Καμία από τις παραπάνω όμως δεν επιδρά στο περιεχόμενο μιας ιστοσελίδας. Αυτό ακριβώς επιτυγχάνει η Document.write η οποία είναι μια μέθοδος εξαγωγής αποτελεσμάτων , άμεση, ασύγχρονη, μεταβάλλοντας το περιεχόμενο της τρέχουσας ιστοσελίδας.

Η document.write πρωτοεισάχθηκε στον browser Netscape 2 πριν ακόμη καλά καλά επινοηθεί το DOM, και ήταν ο μοναδικός τρόπος για να εξάγει κάποιος αποτελέσματα σε μια ιστοσελίδα από εκτέλεση javascript. Τώρα πλέον δεν χρησιμοποιείται καθώς έχει αντικατασταθεί από άλλες εντολές. Η λειτουργία της είναι να «γράφει» στο έγγραφο html ότι εισάγουμε μέσα της ως παράμετρο. Αν δεν χρησιμοποιήσουμε σωστά την εντολή αυτή είναι πιθανό να διαγραφούν τμήματα της ιστοσελίδας που χρησιμοποιούμε εκείνη την στιγμή.

Αν η write χρησιμοποιηθεί μέσα σε κώδικα javascript καθώς η σελίδα αυτή φορτώνεται τότε το κάθε σημείο στο οποίο βρίσκεται η write θα τυπωθεί και το αντίστοιχο μήνυμα που έχουμε ορίσει. Αν προσπαθήσουμε αφότου φορτωθεί η σελίδα

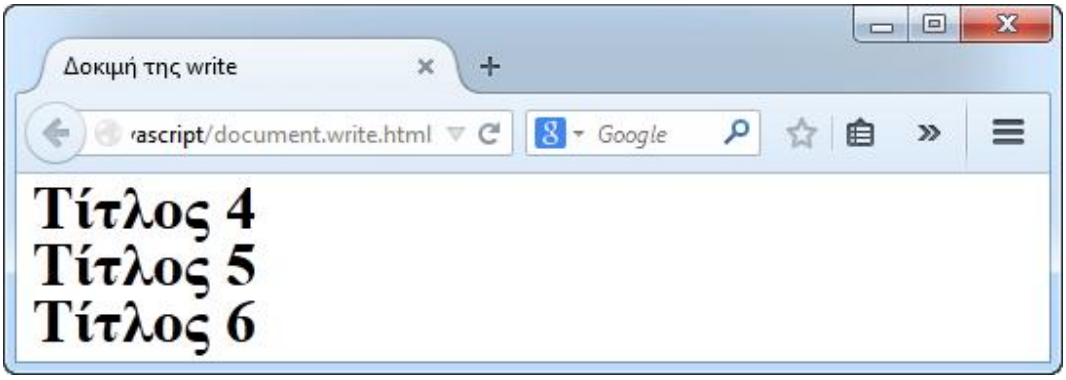
να γράψουμε κάτι μέσα σε αυτήν με το document.write τότε το πιο πιθανό είναι να διαγραφεί όλο το περιεχόμενο της σελίδας και ύστερα να εισαχθεί το νέο που έχουμε εισάγει ως παράμετρο στην document.write. Πολλοί browsers πλέον δεν επιτρέπουν αυτού του είδους αλληλεπίδραση εντολών javascript με τις ιστοσελίδες. Η write λοιπόν λειτουργεί ΜΟΝΟΝ κατά την διάρκεια φόρτωσης (διερμηνείας) της ιστοσελίδας και όχι μετά.

Συντάσσεται γράφοντας «document.write», και αμέσως μετά μέσα σε παρενθέσεις ένα αλφαριθμητικό που θέλουμε να «γράψουμε» πχ «document.write("ΚΑΛΗΜΕΡΑ");». Αυτό το τοποθετούμε μέσα στον κώδικα μας στο σημείο ακριβώς που θέλουμε να εμφανιστεί, κλεισμένο σε «script» tags. Πχ:

Αρχείο: document.write.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Δοκιμή της write</title>
  <meta charset='utf-8'/>
  <script>
   var a = 3;
  </script>
 </head>
 <body>
  <h1><script>document.write("Τίτλος "+(a+1));</script></h1>
  <h1><script>document.write("Τίτλος "+(a+2));</script></h1>
  <h1><script>document.write("Τίτλος "+(a+3));</script></h1>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.25 Εισαγωγή περιεχομένου μέσα σε μια ιστοσελίδα με την document.write

Εάν θέλετε να τυπώσετε αποτελέσματα μέσα στην ιστοσελίδα επηρεάζοντας μόνο συγκεκριμένα μέρη της, τότε δοκιμάστε μια από τις μεθόδους, που περιγράφονται στην ενότητα B.2.2. Αν πάλι απλά θέλετε να φορτώνεται το αποτέλεσμα μαζί με την σελίδα τότε η document.write είναι η κατάλληλη επιλογή.

Ωστόσο υπάρχει μια έξυπνη χρήση της document.write έτσι ώστε να εξυπηρετήσει με κάποιον τρόπο τα σύγχρονα δεδομένα. Μπορεί να χρησιμοποιηθεί έτσι ώστε να δημιουργηθεί μια εξολοκλήρου νέα σελίδα για προβολή σε αναδυόμενο παράθυρο. Ακολουθείτε μια πιο περίπλοκη διαδικασία αλλά αξίζει τον κόπο να την δούμε:

Αρχείο: write.popup.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Δοκιμή της write σε popup</title>
  <meta charset='utf-8'/>
  <script>
   function msg(){
    var name = document.getElementsByName("fname")[0].value;
    var site="<html><head><meta charset='utf-
8'/></head><body><h1>Γειά σου "+name+"</h1></body></html>";
    var popupWindow = window.open("","","menubar=0,scrollbars=0");
    popupWindow.document.write(site);
    popupWindow.document.close();
   }
  </script>
 </head>
 <body>
 Συμπλήρωσε το όνομα σου: <input type='text' name='fname'
id='fname'/> και
 <input type='button' value='κάνε κλίκ' onclick='msg();'/>
 </body>
</html>
```

Αποτέλεσμα:

Εικόνα B.26 Χρήση αναδυόμενων παραθύρων με την document.write

Για να μπορέσουμε να χρησιμοποιήσουμε αυτή την δυνατότητα θα πρέπει να συμπεριλάβουμε στον κώδικα μας την εντολή «window.open» και την εντολή «document.close()».

Η window.open ανοίγει ένα νέο παράθυρο το οποίο είναι διαχειρίσιμο αν το αποθηκεύσουμε προσωρινά σε κάποια μεταβλητή στην συγκεκριμένη περίπτωση χρησιμοποιούμε την «popupWindow», ύστερα με την «document.write» γράφουμε μέσα σε αυτό, και κλείνουμε το έγγραφο δηλαδή ότι τελειώσαμε το γράψιμο με την εντολή «document.close()». Η εντολή «document.getElementsByName» είναι μια εντολή επιλογής αντικειμένων html και επεξηγείται αναλυτικά στην ενότητα B.2.2.2.4.

B.2 HTML & JAVASCRIPT

Η javascript στηρίζει την ύπαρξη της στην ύπαρξη της HTML, καθότι πάνω στο παραγόμενο αποτέλεσμα της, επιδρά αλλάζοντας το στατικό περιεχόμενο της. Έτσι οι σελίδες HTML μπορούν πλέον να αποκτήσουν δυναμική μορφή στην εμφάνιση τους ή και στο περιεχόμενο τους. Κείμενα, εικόνες, φόρμες όλα μπορούν να αλλάξουν χάρις την Javascript.

Β.2.1 Βασικές ετικέτες χρήσης

Μια ιστοσελίδα για να μπορέσει ένας φυλλομετρητής να την ερμηνεύσει θα πρέπει να διαθέτει και τις ανάλογες ετικέτες δήλωσης της ως ιστοσελίδα html, μέσα στον κώδικα της. Για να ορίσουμε ότι το αρχείο που γράφουμε εκείνη την στιγμή είναι αρχείο το οποίο θα ερμηνευτεί ως ιστοσελίδα από τους φυλλομετρητές χρησιμοποιούμε το tag <!DOCTYPE html>, με αυτή την ετικέτα έχουμε ορίσει ότι παρακάτω ο φυλλομετρητής βρίσκοντας αντίστοιχες ετικέτες θα αρχίσει να τις ερμηνεύει ως html.

Η σωστή δομή μιας ιστοσελίδας βασιζόμενοι στα υπάρχοντα πρότυπα έχει ως εξής:

```
<!DOCTYPE html>
<html>
 <head>
 <meta charset='UTF-8'/>
  <title> </title>
 </head>
 <body>
 </body>
</html>
```

Μεταξύ του «head» εισάγουμε κομμάτια κώδικα τα οποία δεν «φαίνονται» στο φόρτωμα της σελίδας, όπως meta tags, scripts κ.α., στο μεταξύ του «<body>» και του «</body>» εισάγουμε όλες τις ετικέτες εκείνες που εμφανίζονται στο φόρτωμα της σελίδας μας, εκεί συμπεριλαμβάνονται όλα αυτά που επεξηγούνται παρακάτω και όπου θα χρησιμοποιήσουμε κατά κόρον σε όλες τις επόμενες ενότητες μας.

Β.2.1.1 Βασικές ετικέτες

Ο Πίνακας Β. περιγράφει μερικές από τις βασικότερες ετικέτες που χρησιμοποιούνται στην HTML.

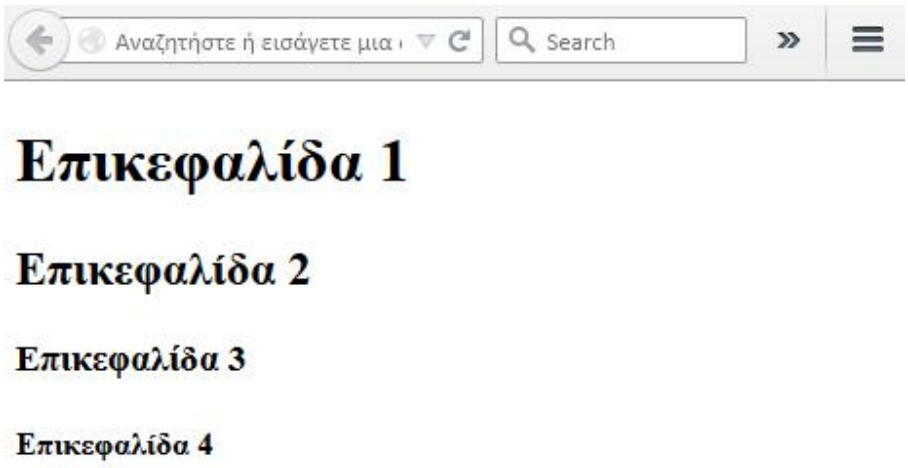
Ετικέτα	Παράδειγμα	Περιγραφή
hx	`<h1>Τίτλος</h1>`	Τίτλοι κειμένων, από 1 έως 6
p	`<p>Παράγραφος</p>`	Περιοχή κειμένου
a	`<a href="συνδεσμος">τίτλος</a>`	Χρήση συνδέσμου και τίτλου αυτού. Το url μπαίνει στο href και το εμφανιζόμενο κείμενο μέσα στο tag.

Πίνακας Β.1 Βασικές ετικέτες html

Παράδειγμα:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset='UTF-8'/>
  <title> Δοκιμή βασικών ετικετών HTML</title>
 </head>
 <body>
  <h1>Επικεφαλίδα 1 </h1>
  <h2>Επικεφαλίδα 2 </h2>
  <h3>Επικεφαλίδα 3 </h3>
  <h4>Επικεφαλίδα 4 </h4>
  <h5>Επικεφαλίδα 5 </h5>
  <h6>Επικεδαλίδα 6 </h6>
  <p>Παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου
παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου
παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου
παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου
παράγραφος κειμένου παράγραφος κειμένου παράγραφος κειμένου</p>
  <a href='http://www.google.gr/'> Σύνδεσμος κειμένου </a>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.27 Παράδειγμα χρήσης βασικών ετικετών html

Β.2.1.2 Μορφοποίηση κειμένου

Στην HTML υπάρχουν ετικέτες για την μορφοποίηση κειμένου, μερικές από αυτές περιγράφονται παρακάτω.

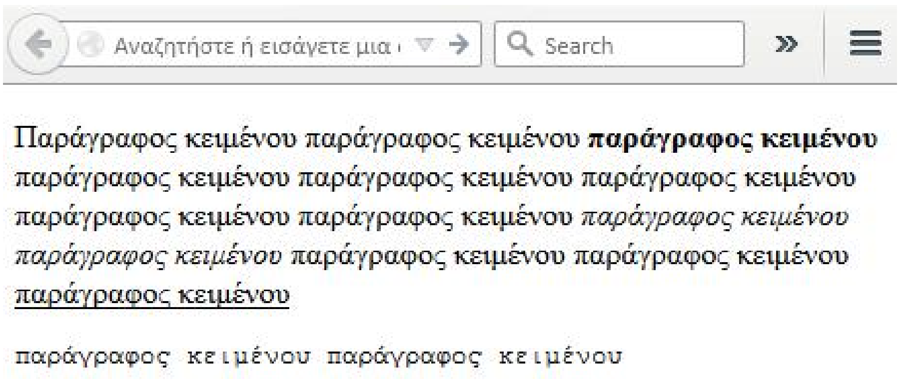
Ετικέτα	Περιγραφή	Παράδειγμα
b	Έντονο κείμενο	Κώστας
i	Πλάγιο κείμενο	Γιάννης
u	Υπογραμμισμένο κείμενο	<u>Ελένη</u>
pre	Παράγραφος σταθερού πλάτους	<pre>…κείμενο… </pre>

Πίνακας Β.2 Ετικέτες μορφοποίησης κειμένου html

Παράδειγμα:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset='UTF-8'/>
  <title> Δοκιμή μορφοποίησης κειμένου HTML</title>
 </head>
 <body>
  <p>Παράγραφος κειμένου παράγραφος κειμένου <b>παράγραφος
κειμένου</b> παράγραφος κειμένου παράγραφος κειμένου παράγραφος
κειμένου παράγραφος κειμένου παράγραφος κειμένου <i> παράγραφος
κειμένου παράγραφος κειμένου</i> παράγραφος κειμένου παράγραφος
κειμένου <u>παράγραφος κειμένου</u> <pre>παράγραφος κειμένου
παράγραφος κειμένου</pre></p>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.28 Παράδειγμα χρήσης μορφοποίησης κειμένου στην html

Β.2.1.3 Περιοχές αντικειμένων

Οι περιοχές αντικειμένων καθορίζουν τις περιοχές στις οποίες αντικείμενα μπορούν να εισαχθούν. Στα κειμενα καθορίζουν σε ποια σημεία αλλάζουν γραμμές ή που εισάγονται οριζόντιες γραμμές που χωρίζουν το περιεχόμενο.

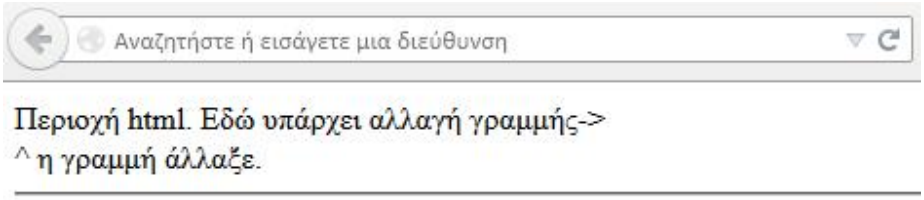
Ετικέτα	Περιγραφή	Παράδειγμα
div	Κοντέινερ που περιέχει στοιχεία html	`<div><b>ένα</b></div>`
br	Κενή γραμμή	` `
hr	Οριζόντια γραμμή	`<hr />`

Πίνακας Β.3 Ετικέτες περιοχών αντικειμένων html

Παράδειγμα:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset='UTF-8'/>
  <title> Δοκιμή περιοχών αντικειμένων HTML</title>
 </head>
 <body>
  <div>
   Περιοχή html. Εδώ υπάρχει αλλαγή γραμμής-> </br> ^ η γραμμή
άλλαξε. <hr/>
  </div>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.29 Χρήση περιοχών αντικειμένων στην html

Β.2.1.4 Εικόνα

Η εικόνα χρησιμοποιείται με την ετικέτα «img» παίρνοντας αντίστοιχα κάποιες παραμέτρους:

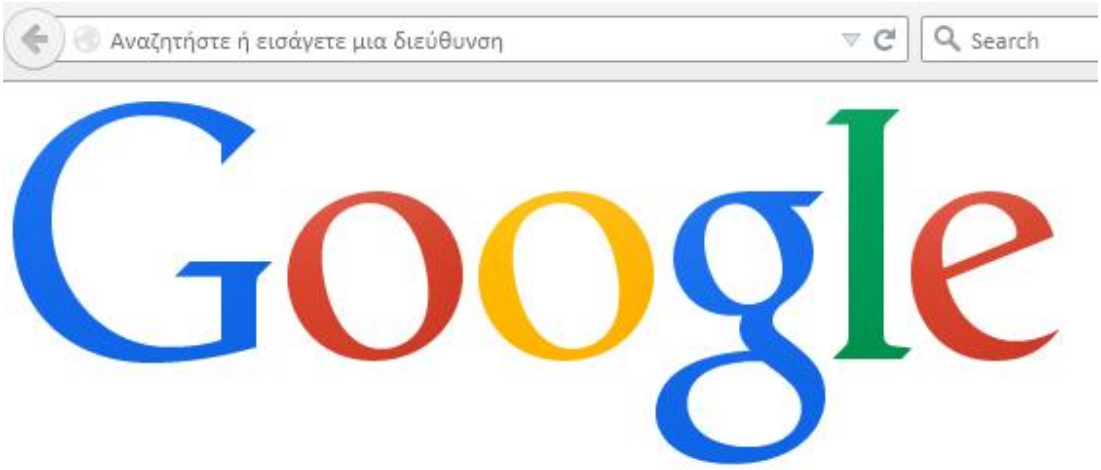
```
<img src="https://www.google.com/images/srpr/logo11w.png"
alt='google'/>
```

Στην παράμετρο «src» εισάγουμε το link όπου βρίσκεται η εικόνα, ενώ στην παράμετρο «alt» εισάγουμε εναλλακτικό κείμενο το οποίο θα εμφανίζεται αν δεν μπορεί για οποιοδήποτε λόγο να φορτωθεί η εικόνα.

Παράδειγμα:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="UTF-8"/>
  <title>Δοκιμή img</title>
 </head>
 <body>
  <img src="https://www.google.com/images/srpr/logo11w.png"
alt="google"/>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.30 Αποτέλεσμα χρήσης ετικέτας img

Β.2.1.5 Πίνακες

Οι πίνακες στην html είναι ένας τρόπο χωρικής ταξινόμησης προβαλλόμενων δεδομένων. Έτσι ότι προβάλλουμε είναι πιο εύκολα κατανοητό και ευανάγνωστο.

Ετικέτα	Περιγραφή	Παράδειγμα
table	Ετικέτα που δείχνει ότι ξεκινά και τελειώνει πίνακας	<table> <tr> <th>Τίτλος 1</th> <td>Κελί 2</td> </tr> </table>
tr	Table row = Γραμμή πίνακα	
th	Table heading = Επικεφαλίδα πίνακα	
td	Table division = Κοντέινερ πίνακα (Κελί πίνακα)	

Πίνακας Β.4 Ετικέτες πίνακα στην html

Αρχείο: table.html

```
<html>
 <body>
  <table border='1'>
   <tr><th>Όνομα</th><th>Επώνυμο</th></tr>
   <tr><td>Χρήστος</td><td>Νικολακόπουλος</td></tr>
   <tr><td>Ιωάννα</td><td>Χριστοπούλου</td></tr>
  </table>
 </body>
</html>
```

Αποτέλεσμα:

Εικόνα Β.31 Χρήση πινάκων της html

Η παράμετρος «border» καθορίζει πόσο θα είναι το μέγεθος των διαχωριστικών γραμμών που διαχωρίζουν τα κελιά μεταξύ τους και μετριέται σε pixels.

Β.2.1.6 Φόρμες

Οι φόρμες είναι ένας τρόπος για την html έτσι ώστε ένας χρήστης να μπορεί να εισάγει δικά του δεδομένα και αυτά τα δεδομένα να μπορούν να χρησιμοποιηθούν κάπως μέσα στα scripts μας. Παρακάτω θα δούμε μερικές από τις πιο σημαντικές ετικέτες στις φόρμες html.

Ετικέτα: form

Η ετικέτα form χρησιμοποιείται για να δηλώσει ότι τα περιεχόμενα αντικείμενα της θα χρησιμοποιηθούν έτσι ώστε τα στοιχεία που θα εισαχθούν σε αυτά από κάποιον χρήστη, θα σταλθούν κάπου συγκεκριμένα και με συγκεκριμένο τρόπο.

Παράδειγμα:

```
<form method='post' action='file.php'>
 …Υπόλοιπα στοιχεία φόρμας…
</form>
```

- Το χαρακτηριστικό «action» καθορίζει που θα σταλούν τα δεδομένα μετά την υποβολή της φόρμας. Μπορεί να είναι ένα απλό όνομα αρχείου αν βρίσκεται στον ίδιο κατάλογο με την φόρμα ή μπορεί να είναι και ολόκληρο url.

- Το χαρακτηριστικό «method» καθορίζει τον τρόπο με τον οποίο μπορούν να σταλθούν τα στοιχεία και υπάρχουν δυο τρόποι για να συμβεί αυτό post και get.
    - Όταν χρησιμοποιείται η μέθοδος «post» τότε τα στοιχεία μεταφέρονται από την φόρμα στο αρχείο της παραμέτρου «action» χωρίς να αποθηκεύονται στο ιστορικό του φυλλομετρητή και χωρίς να αφήνονται «ίχνη» πίσω τους. Είναι κατάλληλη έτσι για μεταφορά ευαίσθητων δεδομένων, όπως κωδικούς, προσωπικά στοιχεία κ.α.
    - Όταν χρησιμοποιείται η μέθοδος «get» τότε τα στοιχεία μεταφέρονται από την φόρμα στο αρχείο της παραμέτρου «action» μέσω του url του φυλλομετρητή. Έτσι τα στοιχεία αυτά είναι εμφανή ακόμη και μετά την ολοκλήρωση της φόρμας και την αποστολή των στοιχείων, μέσω του url με την μορφή «ονομα_πεδίου=τιμή». Αυτή η μέθοδος χρησιμοποιείται για απλή μεταφορά μη ευαίσθητων δεδομένων, διότι είναι δυνατή η αποθήκευση των στοιχείων τόσο στο ιστορικό του φυλλομετρητή καθώς και ευάλωτη σε πρόσβαση από τρίτους.

Ετικέτα: input

Η ετικέτα input είναι μια ετικέτα που πρέπει να συμπεριληφθεί μέσα στα περιεχόμενα μιας ετικέτας form. Διαθέτει κάποιες παραμέτρους που καθορίζουν τον τρόπο με τον οποίο συμπεριφέρεται η συγκεκριμένη ετικέτα.

Ο τύπος της ετικέτας είναι αυτός που καθορίζει τον τρόπο με τον οποίο θα αλληλεπιδρά ένας χρήστης με αυτήν.

Παράμετρος	Τύπος	Παράδειγμα χρήσης
text	Μικρό κείμενο	Όνομα: Κείμενο
password	Κωδικός	Κωδικός: •••••••••
checkbox	Τικ επιλογής	Διατήρηση σύνδεσης: ☑
radio	Επιλογή μόνο μίας απάντησης	Φύλλο: Άνδρας ○ Γυναίκα ◉
button	Κουμπί	Αποστολή
submit	Κουμπί υποβολής	

Πίνακας Β.5 Παράμετροι γνωρίσματος type για την ετικέτα input

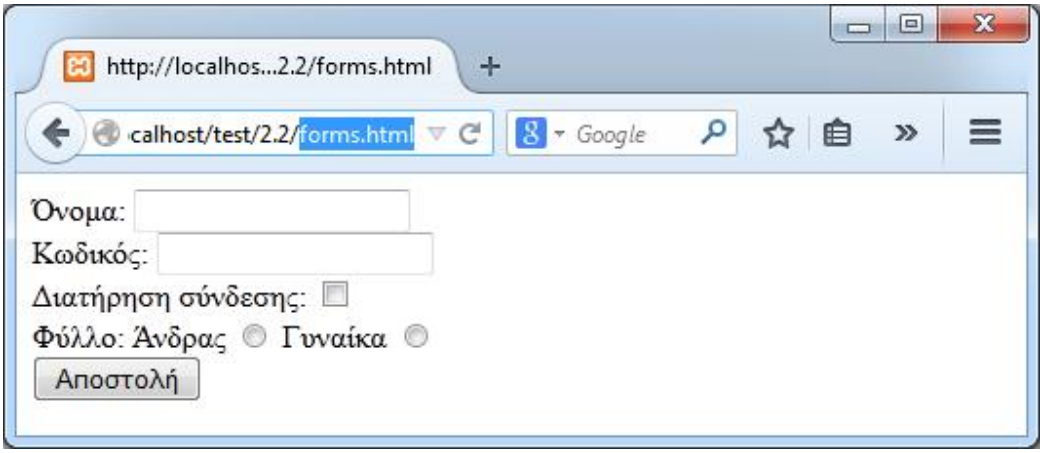
Χαρακτηριστικό	Περιγραφή
name	Όνομα μεταβλητής όταν υποβληθεί
value	Προεπιλεγμένη τιμή
size	Προβαλλόμενο μέγεθος πεδίου.
maxlength	Μέγιστος αριθμό χαρακτήρων
checked	Σε radio ή checkbox μαρκάρει κάποιο συγκεκριμένο.

Πίνακας Β.6 Χαρακτηριστικά της ετικέτας input

Αρχείο: form.html

```
<html>
 <body>
  <form method='get' action='data.php'>
   Όνομα: <input type='text' name='name'/>
   <br />
   Κωδικός: <input type='password' name='password'/>
   <br />
   Διατήρηση σύνδεσης: <input type='checkbox' name='staylogged'/>
   <br />
   Φύλλο:
   Άνδρας <input type='radio' name='sex' value='male'/>
   Γυναίκα <input type='radio' name='sex' value='female'/>
   <br />
   <input type='submit' value='Αποστολή'/>
  </form>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.32 Παράδειγμα φόρμας στην html

Β.2.2 Οι κόμβοι-στοιχεία μιας ιστοσελίδας

Μια ιστοσελίδα γραμμένη σε HTML, όπως την ερμηνεύει ένας φυλλομετρητής αποκτά κάποια συγκεκριμένα χαρακτηριστικά που μπορούν έτσι να μας βοηθήσουν να αποκτήσουμε πρόσβαση στα στοιχεία της μέσω της Javascript. Συγκεκριμένα κάθε tag

της html γίνεται ένα οπτικό αντικείμενο στο παράθυρο του φυλλομετρητή, το λεγόμενο και ως DOM object από τα Document Object Model.

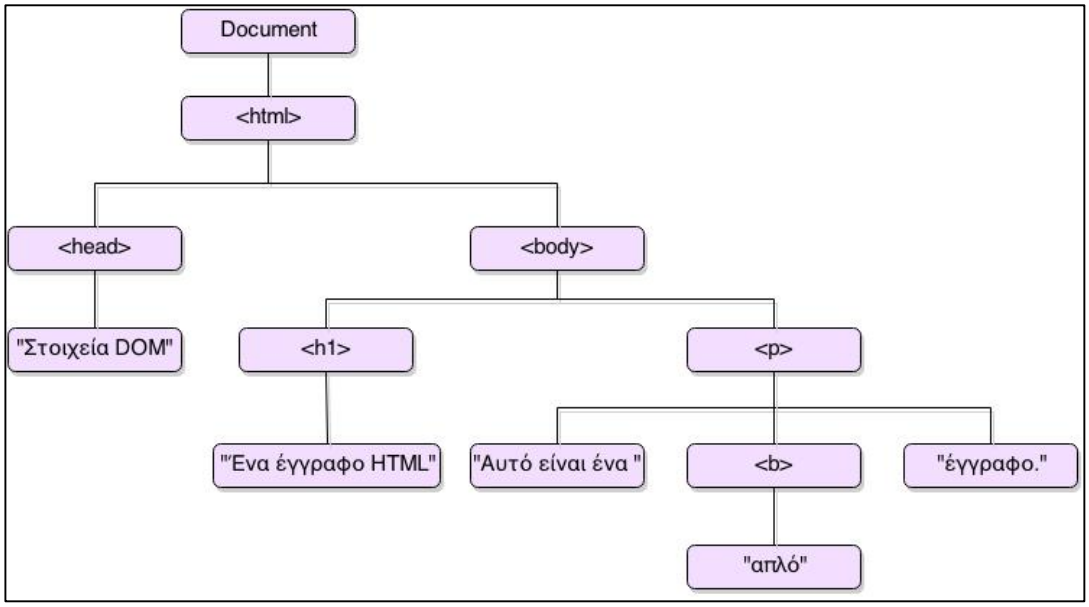
Το DOM είναι η θεμελιωμένη προτυποποίηση αναπαράστασης και χρήσης των περιεχομένων ενός HTML ή XML εγγράφου. Το πρότυπο αυτό δεν είναι περίπλοκο αλλά χρειάζεται εκτενή επεξήγηση προκειμένου να το κατανοήσετε. Αρχικά θα πρέπει να καταλάβετε τα εμφωλευμένα στοιχεία (κυρίως tags) ενός εγγράφου HTML/XML όπως αναπαριστώνται μέσω του DOM.

B.2.2.1 Επιλογή αντικειμένων με βάση την θέση του μέσα στο δέντρο

Στο DOM κάθε έγγραφο αναπαρίστανται σαν ένα δέντρο, όπου κορυφή του είναι το στοιχείο «Document» και κάτω από αυτό υπάρχουν όλα τα στοιχεία που υπάρχουν και μέσα στο κάθε έγγραφο. Κάθε tag αποτελεί και έναν κόμβο του δέντρου, το οποίο αντίστοιχα αν έχει άλλα εμφωλευμένα tags θα διαθέτει κι άλλους κόμβους κάτω από αυτό. Ένας κόμβος μπορεί να είναι ένα tag ή ένα κείμενο, ή ακόμη και τα σχόλια σε HTML. Ας δούμε ένα παράδειγμα:

Αρχείο: dom.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM</title>
 </head>
 <body>
  <h1>Ένα έγγραφο HTML</h1>
  <p>Αυτό είναι ένα <b>απλό</b> έγγραφο.</p>
 </body>
</html>
```



Σχήμα Β.1 Αναπαράσταση στοιχείων ιστοσελίδας σε σχήμα δέντρου

Εάν δεν είστε εξοικειωμένοι με τέτοια δέντρα, τότε θα πρέπει να ξέρετε πως μοιάζουν αρκετά με τα γενεαλογικά δέντρα, όπου ένας γονιός έχεις παιδιά, τα παιδιά του γονιού γίνονται γονείς για τα δικά τους παιδιά κ.ο.κ. Άρα παραπάνω το στοιχείο «<p>» είναι παιδί του «<body>» και αδερφάκι του «<h1>» καθώς επίσης έχει τα παιδιά «Αυτό είναι ένα», «<b>» και «έγγραφο». Το «απλό» είναι εγγόνι του «<p>» και παιδί του «<b>».

Με βάση όλα τα παραπάνω ισχύουν κάποιες ετικέτες για κάθε κόμβο με βάση την θέση του μέσα στο δέντρο. Πιο συγκεκριμένα:

- parentNode

  Είναι ο κόμβος-πατέρας στον κόμβο τον οποίο βρισκόμαστε την εκάστοτε στιγμή. Αν βρισκόμαστε σε ένα στοιχείο που δεν έχει πατέρα όπως το Document τότε το parentNode είναι null.
- childNodes

  Ένας πίνακας μόνο προς ανάγνωση με όλα τα αντικείμενα τα οποία είναι παιδία του εκάστοτε κόμβου.
- firstChild, lastChild

Το πρώτο ή το τελευταίο παιδί ενός κόμβου που βρισκόμαστε εκείνη την στιγμή. Αν κόμβος δεν έχει παιδιά εμφανίζεται η τιμή null.

- nextSibling, previousSibling

  Ο επόμενος ή ο προηγούμενος αδερφός ενός κόμβου. Δυο ή και παραπάνω παιδία ενός και μόνο κόμβου είναι αδέρφια.

- nodeType

  Ο τύπος του κόμβου που εξετάζουμε εκείνη την στιγμή. Οι κόμβοι εγγράφου έχουν την τιμή 9. Οι κόμβοι στοιχείων έχουν την τιμή 1. Οι κόμβοι κειμένου έχουν την τιμή 3. Οι κόμβοι σχόλιων έχουν την τιμή 8 και οι κόμβοι τεμαχισμένου εγγράφου την τιμή 11.

- nodeValue

  Το περιεχόμενο ενός κόμβου κειμένου ή σχόλιου.

- nodeName

  Το όνομα του tag ενός στοιχείου που αποτελεί τον κόμβο σε κεφαλαία.

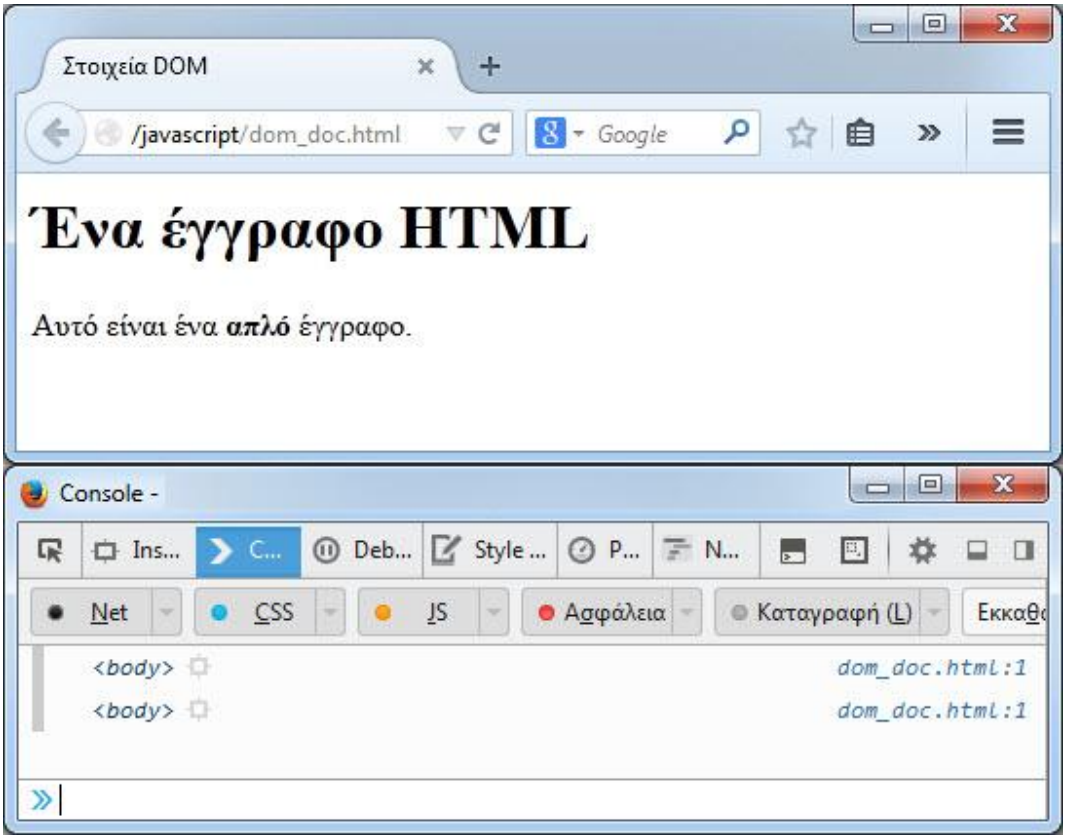
Για να χρησιμοποιήσουμε μία από τις παραπάνω ιδιότητες θα πρέπει να βρισκόμαστε ήδη σε ένα συγκεκριμένο κόμβο-στοιχείο. Το πιο κοινό και βασικό κόμβος-στοιχείο είναι το «document» (στοιχείο ρίζα) από όπου ξεκινούν και όλα.

Ας δούμε ένα παράδειγμα χρησιμοποιώντας το document ως κόμβο-στοιχείο που θα βασιστούμε, εδώ θα πρέπει να σημειώσουμε ότι αν ο κώδικας html διαθέτει τις περιττές αλλαγές γραμμών και τα περριτά κενά για να είναι πιο ευανάγνωστος ο κώδικας, τότε είναι πιθανό οι κόμβοι να είναι κατά πολύ περισσότεροι λόγω του ότι κάθε κενό στην αλλαγή γραμμής θεωρείται διαφορετικός κόμβος.

Αρχείο: dom_doc.html

```
<!DOCTYPE HTML><html><head><title>Στοιχεία DOM</title><meta
charset='utf-8'/></head><body><h1>Ενα έγγραφο HTML</h1><p>Αυτό είναι
ένα <b>απλό</b> έγγραφο.</p></body><script>
console.log(document.childNodes[0].childNodes[1]);
console.log(document.firstChild.firstChild.nextSibling);
</script></html>
```

Όπως βλέπετε ο κώδικας δεν είναι ευανάγνωστος αλλά μας παρέχει μια συμπιεσμένη μορφή του κώδικα, αποφεύγοντας έτσι τα περιττά κενά, τις περιττές αλλαγές γραμμών, και άλλα σχετικά. Το αποτέλεσμα έχει ως εξής:



Εικόνα Β.33 Προσπέλαση κόμβων ιστοσελίδας με εντολές javascript

Επιλέγεται και στις δυο περιπτώσεις το στοιχείο «<body>» το οποίο είναι το δεύτερο παιδί του «<html>» το οποίο html είναι το πρώτο παιδί του «document». Προσεγγίζουμε το ίδιον κόμβο με διαφορετική χρήση ετικετών.

Για να αντιμετωπίσουμε το πρόβλημα των περιττών κενών και των αλλαγών γραμμών διατηρώντας ταυτόχρονα την μορφή του κώδικα όπως έχει, αντί να προσπελάσουμε τα στοιχεία ως κόμβους, θα τα προσπελάσουμε ως DOM Elements (αντικείμενα DOM). Είναι ακριβώς το ίδιο πράγμα με προηγουμένως, μόνο που απορρίπτονται όλα τα στοιχεία απλού κειμένου και επιλέγονται μόνο τα tags. Για να το καταφέρουμε αυτό θα χρησιμοποιήσουμε κάποιες από τις παρακάτω ιδιότητες:

- firstElementChild, lastElementChild
  Επιστρέφει το πρώτο ή το τελευταίο DOM Αντικείμενο-παιδί ενός κόμβου που έχουμε επιλεγμένο εκείνη την στιγμή.
- nextElementSibling, previousElementSibling

Επιστρέφει το επόμενο ή το προηγούμενο DOM Αντικείμενο-συγγενή ενός κόμβου που έχουμε επιλεγμένο εκείνη την στιγμή.

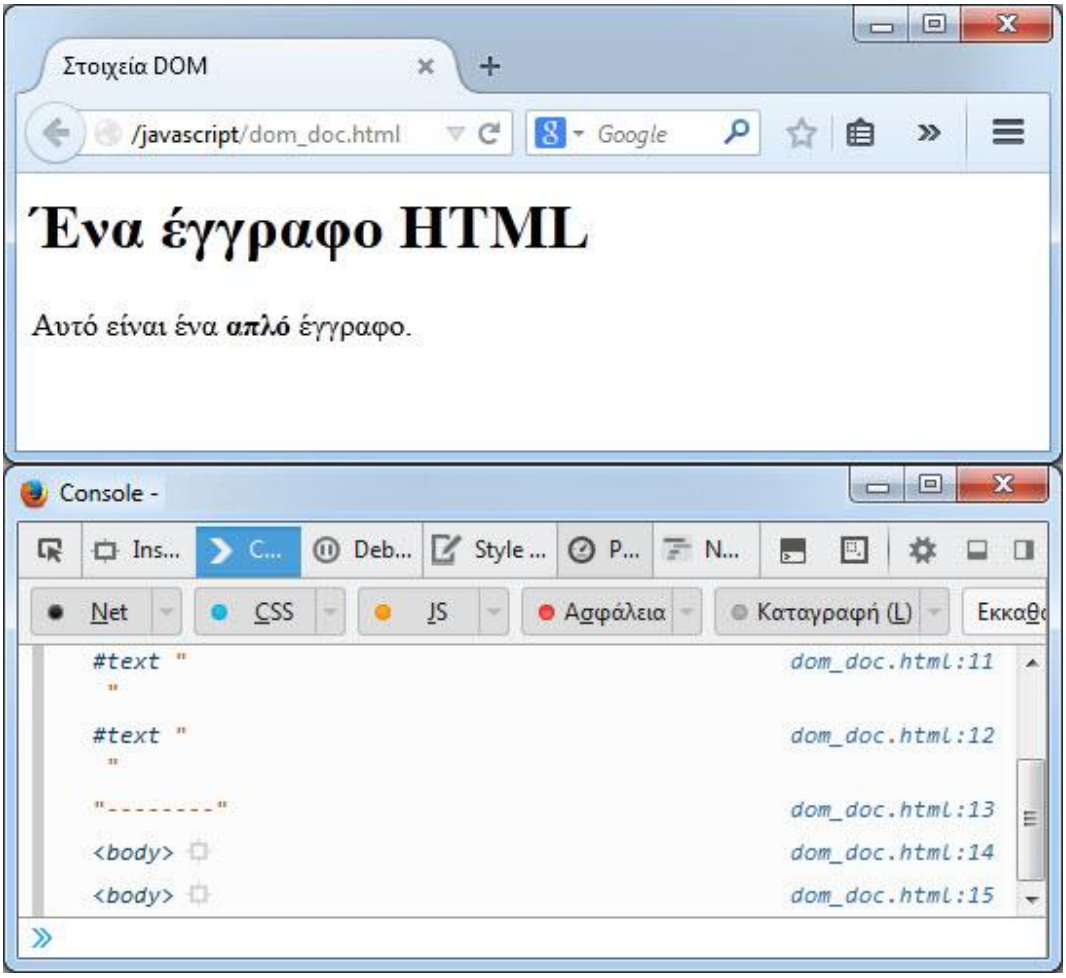
- childElementCount

  Επιστρέφεται το πλήθος των DOM Αντικειμένων-παιδιών ενός κόμβου που έχουμε επιλεγμένο εκείνη την στιγμή.

Ας δούμε το ίδιο παράδειγμα με πριν με μορφοποιημένο κώδικα. Στις δυο πρώτες γραμμές της κονσόλας θα ζητήσουμε τα παιδιά-κόμβους, ενώ στις δυο επόμενους τα παιδία-DOM αντικείμενα.

Αρχείο: dom_doc2.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM</title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ένα έγγραφο HTML</h1>
  <p>Αυτό είναι ένα <b>απλό</b> έγγραφο.</p>
 </body>
 <script>
  console.log(document.childNodes[0].childNodes[1]);
  console.log(document.firstChild.firstChild.nextSibling);
  console.log("--------");
  console.log(document.firstElementChild.lastElementChild);

console.log(document.firstElementChild.firstElementChild.nextElementS
ibling);
 </script>
</html>
```



Εικόνα Β.34 Προσπέλαση στοιχείων DOM με εντολές javascript

Β.2.2.2 Επιλογή αντικειμένων με βάση τα γνωρίσματα τους

Εκτός από την δεδομένη ιεραρχία των αντικειμένων DOM μέσα σε ένα αρχείο HTML, υπάρχουν και διαφορετικές προσεγγίσεις μέσω των οποίων μπορεί η Javascript να αποκτήσει πρόσβαση σε ένα αντικείμενο μέσα στην ιστοσελίδα, όπως με βάση το όνομα του tag του, μια συγκεκριμένη ιδιότητα-γνώρισμα ή οτιδήποτε άλλο. Μερικοί από του τρόπους περιγράφονται σε αυτή την υποενότητα.

Β.2.2.2.1 Με βάση το tag

Είναι δυνατό στην Javascript να μπορέσουμε να επιλέξουμε ένα DOM-αντικείμενο με βάση το tag του. Αν πρόκειται για χρήση της συγκεκριμένης ετικέτας μόνο σε ένα σημείο τότε αυτό που θα επιστραφεί θα είναι μόνο το συγκεκριμένο DOM-αντικείμενο. Αν όμως πρόκειται για μια συγκεκριμένη ετικέτα μέσα σε ένα πλήθος όμοιων ετικετών στην ίδια ιστοσελίδα, τότε επιστρέφεται ένας πίνακας με όλα τα

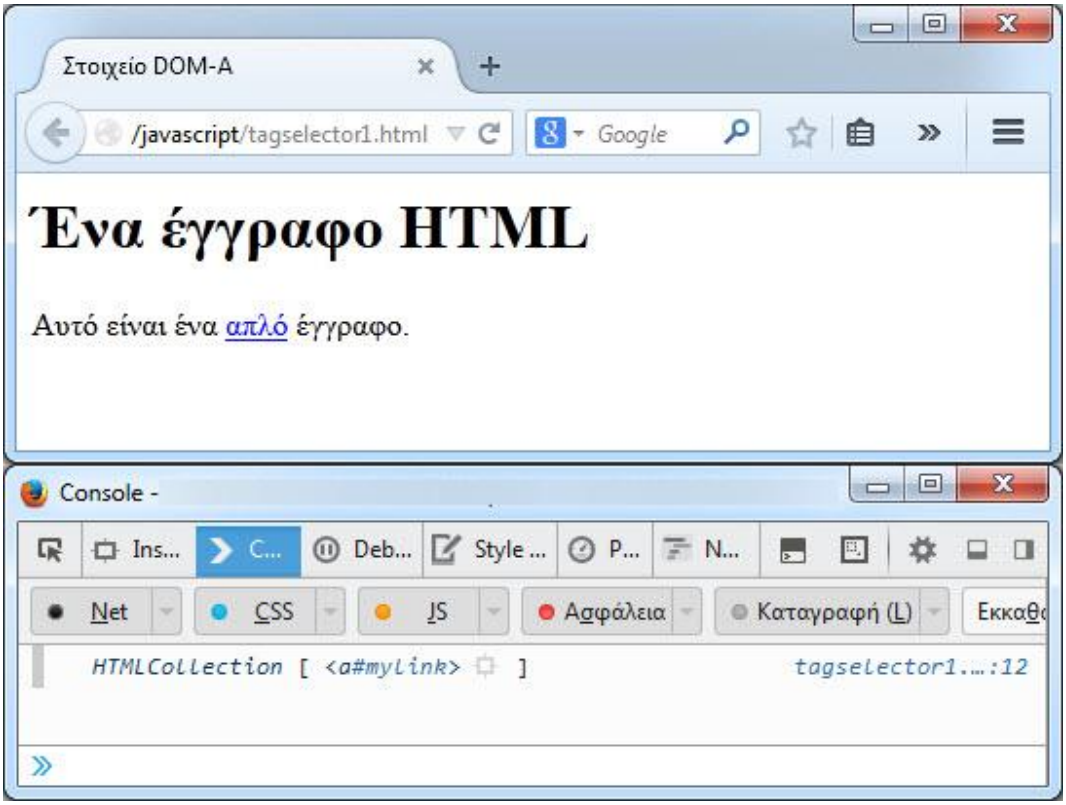
DOM-αντικείμενα, και εμείς θα πρέπει να «ψάξουμε» μέσα σε αυτόν το συγκεκριμένο πίνακα το αντικείμενο που μας ενδιαφέρει.

Ας δούμε πως μπορούμε να αποκτήσουμε πρόσβαση σε ένα DOM-Α έχοντας μοναδική ύπαρξη μέσα στην ιστοσελίδα ως tag:

Αρχείο: tagselector1.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχείο DOM-Α </title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ενα έγγραφο HTML</h1>
  <p>Αυτό είναι ένα <a id='mylink'
href='http://www.google.gr/'>απλό</a> έγγραφο.</p>
 <script>
  var aobject = document.getElementsByTagName("a");
  console.log(aobject);
 </script>
 </body>
</html>
```

Αποτέλεσμα:



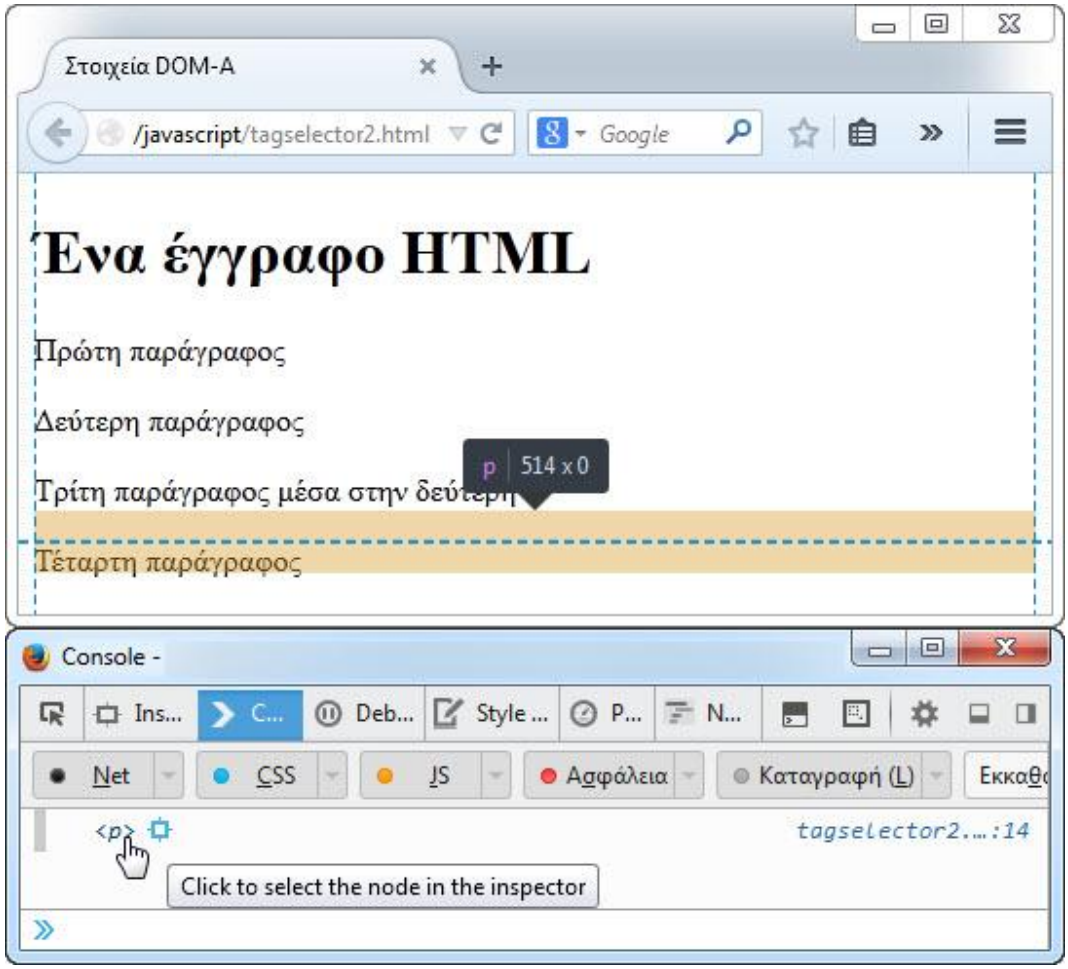
Εικόνα Β.35 Παράδειγμα πρόσβασης σε αντικείμενο html με βάση το όνομα του tag του

Προκειμένου να διαλέξουμε DOM-A μέσα σε μια ιστοσελίδα βασιζόμενοι στο όνομα του tag χρησιμοποιούμε την ιδιότητα «getElementsByTagName("όνομα ετικέτας")», όπου μέσα στην παρένθεση μπαίνει το όνομα της ετικέτας που θέλουμε να επιλέξουμε. Επίσης είναι σημαντικό να χρησιμοποιούμε συντεταγμένες θέσεων προκειμένου να χρησιμοποιήσουμε αυτό που αναζητάμε, διαφορετικά δεν θα μπορέσουμε να αποκτήσουμε πρόσβαση σε αυτό. Στην παραπάνω περίπτωση μπορούμε να χρησιμοποιήσουμε το «[0]» ως συντεταγμένη, δείτε παρακάτω πως χρησιμοποιείται. Επιλογή αντικειμένου μέσα από πλήθος όμοιων tags:

Αρχείο: tagselector2.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM-Α </title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ενα έγγραφο HTML</h1>
  <p>Πρώτη παράγραφος</p>
  <p>Δεύτερη παράγραφος <p>Τρίτη παράγραφος μέσα στην δεύτερη</p></p>
  <p>Τέταρτη παράγραφος</p>
 <script>
  var aobject = document.getElementsByTagName("p")[3];
  console.log(aobject);
 </script>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.36 Παράδειγμα πρόσβασης σε αντικείμενο html μέσω όμοιων αντικειμένων

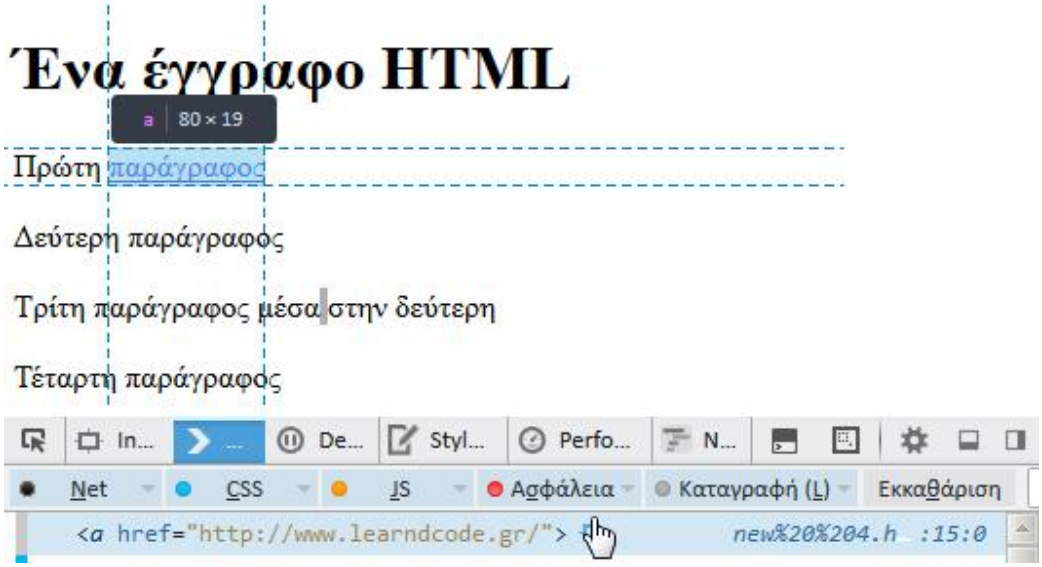
Γίνεται η χρήση συντεταγμένης θέσης όπως θα γινόταν και σε έναν πίνακα. Έτσι το πρώτο «<p>» έχει την συντεταγμένη «0», το δεύτερο «1» και λοιπά. Εμείς θέλαμε την τέταρτη παράγραφο επομένως «3». Εδώ θα πρέπει να σημειώσουμε ότι δεν έχει σημασία το tag πως θα είναι γραμμένο, αν είναι δηλαδή με μικρά γράμματα γραμμένο ή με κεφαλαία ή και μικτά. Όταν καλούμε την συγκεκριμένη ιδιότητα του εγγράφου τότε η αναζήτηση γίνεται σε όλο το έγγραφο και επιστρέφονται όλα τα στοιχεία που ταιριάζουν στην αναζήτηση όποιο τρόπο γραφής και να έχουν.

Η ιδιότητα αυτή μπορεί να χρησιμοποιηθεί και στα στοιχεία DOM-Α εκτός από το στοιχείο-ρίζα του document. Η διαφορά είναι ότι πλέον όταν επιλέγεται να γίνει αναζήτηση μέσα σε ένα DOM-Α η αναζήτηση γίνεται από αυτό το στοιχείο και κάτω, δηλαδή στα παιδιά του τα εγγόνια του τα δισέγγονα του κτλ. Σαν να γίνεται αυτό το στοιχείο για την συγκεκριμένη αναζήτηση το στοιχείο-ρίζα. Ένα σύντομο παράδειγμα με βάση το αρχείο «tagselector1.html» είναι:

Αρχείο: tagselector2.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM-Α </title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ενα έγγραφο HTML</h1>
  <p>Πρώτη <a href='http://www.learncode.gr/'>παράγραφος</a></p>
  <p>Δεύτερη παράγραφος <p>Τρίτη παράγραφος μέσα στην δεύτερη</p></p>
  <p>Τέταρτη παράγραφος</p>
 <script>
  var object1 = document.getElementsByTagName("p")[0];
  var object2 = object1.getElementsByTagName("a")[0];
  console.log(object2);
 </script>
 </body>
</html>
```

Το οποίο θα μας δώσει ως αποτέλεσμα:



Εικόνα Β.37 Πρόσβαση στα αντικείμενα DOM μέσω συντεταγμένων πίνακα

Β.2.2.2.2 Με βάση το id

Αν θέλουμε να προσπελάσουμε αντικείμενα ένα προς ένα, τότε στηριζόμαστε σε κάποιο γνώρισμα του αντικειμένου που θέλουμε να εξετάσουμε και που το κάνει να διαφέρει από τα άλλα αντικείμενα, όπως για παράδειγμα ένα μοναδικό id, μια μοναδική class ή γενικώς ένα attribute με μοναδικό περιεχόμενο. Ένα απλό παράδειγμα id σε ένα html αντικείμενο:

```
<a id='mylink' href='http://www.google.gr/'>απλό</a>
```

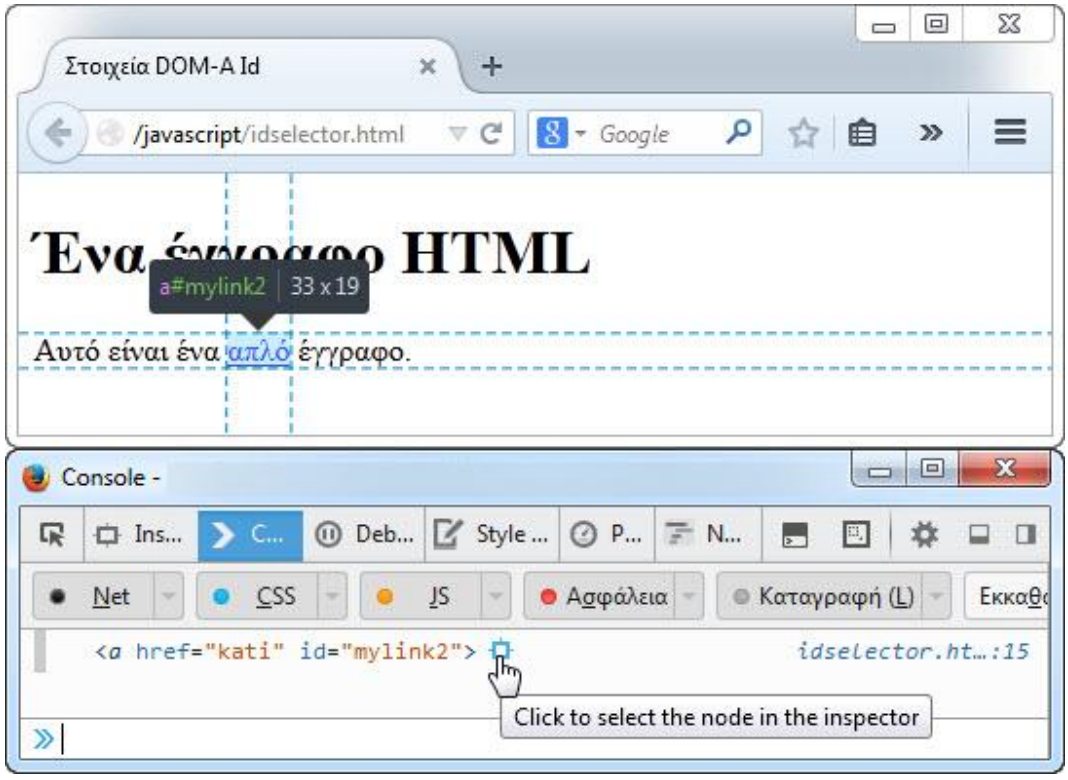
Το αντικείμενο μας στην προκειμένη περίπτωση είναι το «a», όπου ως γνωρίσματα διαθέτει το «id» με τιμή «mylink», και το «href», με τιμή «http://www.google.gr/».

Για να επιλέξουμε το συγκεκριμένο αντικείμενο μέσα από μια HTML σελίδα χρησιμοποιώντας ιδιότητες Javascript θα κάνουμε χρήση της εντολής «getElementById("mylink")», μέσα στην παρένθεση μπαίνει το όνομα του id που θέλουμε να διαλέξουμε, στην προκειμένη «mylink». Επιλέγοντας το συγκεκριμένο αντικείμενο μπορούμε να προσπελάσουμε τα γνωρίσματα του σαν ιδιότητες μεταβλητής, πχ το «href» του «<a>» στην σελίδα, μπορεί να γίνει «link.href» ιδιότητα μιας μεταβλητής με όνομα «link». Ας δούμε ένα παράδειγμα:

Αρχείο: idselector.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM-A Id</title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ένα έγγραφο HTML</h1>
  <p>Αυτό είναι ένα <a id='mylink' href='kati'>απλό</a> έγγραφο.</p>
 </body>
 <script>
  var link = document.getElementById("mylink");
  var url = link.href;
  link.id='mylink2';
  console.log(link);
 </script>
</html>
```

Αποτέλεσμα:



Εικόνα Β.38 Προσπέλαση αντικειμένων html με βάση κάποιο διακριτό γνώρισμα

Ο κώδικας javascript μας, επέλεξε το συγκεκριμένο αντικείμενο με id «mylink2» δημιουργώντας την μεταβλητή «link» που σε αυτήν καταχωρήθηκε το αντικείμενο, και ύστερα προσπελάστηκε από αυτήν την μεταβλητή το γνώρισμα «id» ως ιδιότητα αλλάζοντας του τιμή.

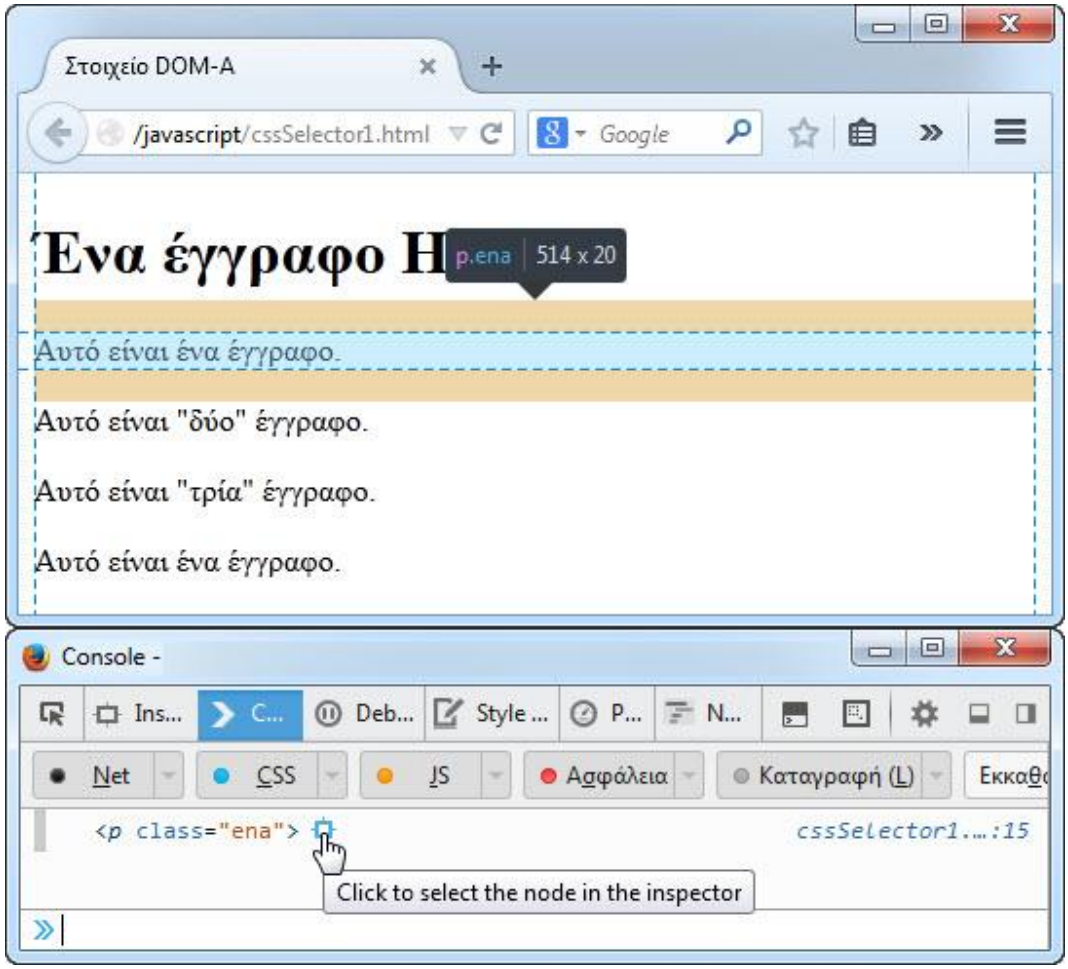
B.2.2.2.3 Με βάση την css κλάση

Το να επιλέξει κάποιος ένα DOM-A με βάση την CSS κλάση του είναι πολύ εύκολο, αρκεί να γνωρίζει την αντίστοιχη κλάση. Εννοείται πως εάν υπάρχει μόνο ένα DOM-A θα επιστραφεί μόνο αυτό, ενώ αν υπάρχουν πολλά με την ίδια κλάση ανεξαρτήτως αν έχουν ίδιο tag ή όχι θα επιστραφούν όλα σε μορφή πίνακα. Η κλάση ενός DOM-A tag αναφέρεται στο γνώρισμα «class» που έχουν τα tags στην HTML και μπορούν να εκμεταλλευτούν οι μορφοποιήσεις CSS προκειμένου να μορφοποιήσουν κατάλληλα το στοιχείο με αυτό το «class». Για να επιτευχθεί η επιλογή χρησιμοποιείται η ιδιότητα «getElementsByClassName("όνομα κλάσης")» όπου μέσα στις παρενθέσεις εισάγεται το όνομα της κλάσης.

Αρχείο: cssSelector1.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχείο DOM-A </title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ενα έγγραφο HTML</h1>
  <p class='ena'>Αυτό είναι ένα έγγραφο.</p>
  <p class='duo'>Αυτό είναι "δύο" έγγραφο.</p>
  <p class='tria'>Αυτό είναι "τρία" έγγραφο.</p>
  <p class='ena'>Αυτό είναι ένα έγγραφο.</p>
 <script>
  var aobject = document.getElementsByClassName("ena")[0];
  console.log(aobject);
 </script>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.39 Παράδειγμα χρήσης css κλάσεων σε αντικείμενα html

Η επιλογή μέσα σε πολλές ίδιες κλάσεις, συγκεκριμένα δυο ίδιες για το πρώτο και το τελευταίο «<p>», γίνεται μέσω της χρήσης συντεταγμένης θέσης μέσα σε πίνακα, ενώ αν είχαμε μόνο ένα στοιχείο με αυτήν την κλάση τότε θα χρειαζόταν να χρησιμοποιήσουμε την συντεταγμένη θέσης «[0]», διαφορετικά θα χρησιμοποιήσαμε τον πίνακα και όχι το ίδιο το αντικείμενο που θέλουμε.

Στην javascript γενικώς υπάρχει ένα πλήθος τρόπων με τους οποίους μπορεί κάποιος να διαλέξει αντικείμενα με βάση οτιδήποτε υπάρχει ως γνώρισμα σε ένα DOM-A. Παρακάτω παραθέτουμε μερικούς τρόπους με τους οποίους γίνεται αυτή η επιλογή μέσα από την CSS.

Επιλογέας	Αντιστοίχηση
Απλές επιλογές	

#nav	
div	
.warning	
Επιλογές με βάση κάποιο γνώρισμα	
p[lang="gr"]	<p lang='gr'>
*[name="x"]	
Σύνθετες επιλογές	
span.fatal.error	
span.[lang='fr'].warning	
Επιλογές απογόνων	
#log span	
#log>span	
body>h1:first-child	<body> </body>
Διάφοροι άλλοι	
div,#log	

Πίνακας Β.7 Αντιστοίχηση css με html στην επιλογή μέσω javascript

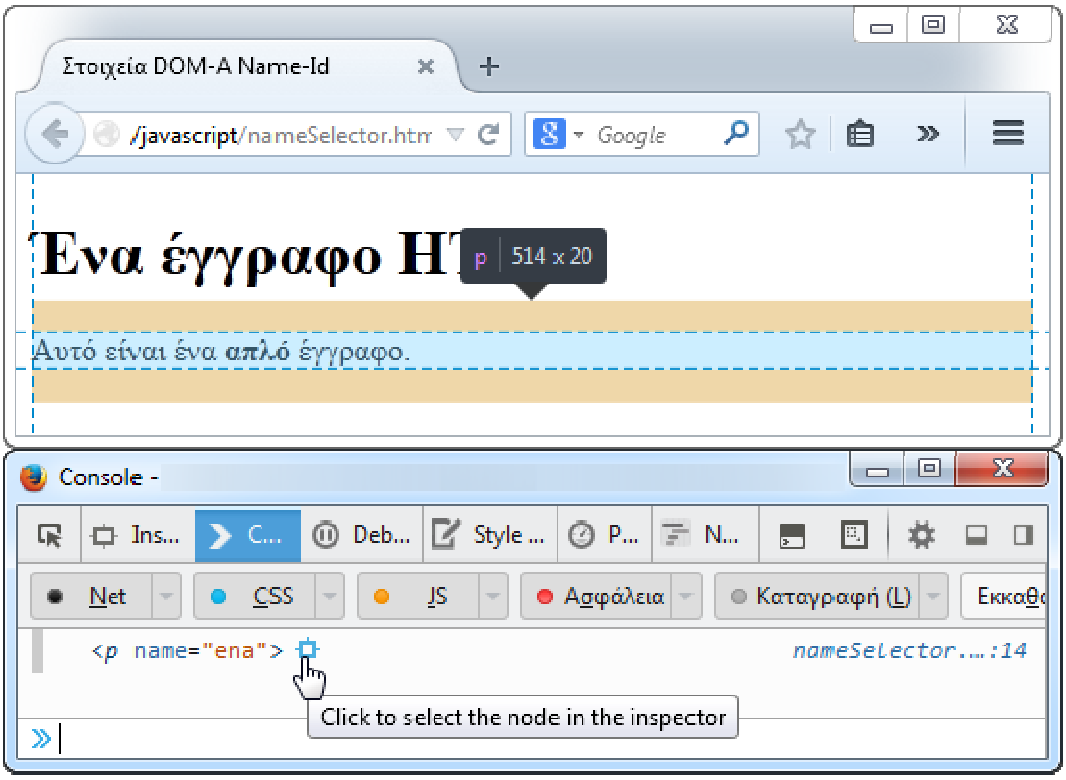
Β.2.2.2.4 Με βάση το όνομα

Εκτός από ένα σωρό γνωρίσματα, ιδιότητες, φόρμες και λοιπά, υπάρχει και ο πιο απλούστερος τρόπος, ο οποίος είναι να βρούμε ένα στοιχείο DOM-A με βάση το όνομα του. Αυτό γίνεται αφού στο αντίστοιχο DOM-A υπάρχει γνώρισμα «name».

Αρχείο: nameSelector.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM-A Name-Id</title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1 name='ena'>Ενα έγγραφο HTML</h1>
  <p name='ena'>Αυτό είναι ένα <b name='duo'>απλό</b> έγγραφο.</p>
 </body>
 <script>
  var objects=document.getElementsByName("ena");
  var object=objects[1];
  console.log(object);
 </script>
</html>
```

Αποτέλεσμα:



Εικόνα Β.40 Χρήση ονόματος στοιχείου μέσω javascript

Β.2.2.2.5 Επιλογές φόρμας εισαγωγής και εικόνων

Για λόγους ευκολίας τα HTML έγγραφα διαθέτουν κάποιες συντομεύσεις προσπέλασης των αντικειμένων τους βάση του ονόματος τους. Παραπάνω είδαμε πως μπορούμε να καλέσουμε ένα οποιοδήποτε στοιχείο με ένα συγκεκριμένο όνομα. Εδώ θα δούμε πως μπορούμε να καλέσουμε φόρμες καθώς και τα στοιχεία που τις περιβάλλουν όπως επίσης ξεχωριστά και εικόνες μέσα από ένα τέτοιο έγγραφο.

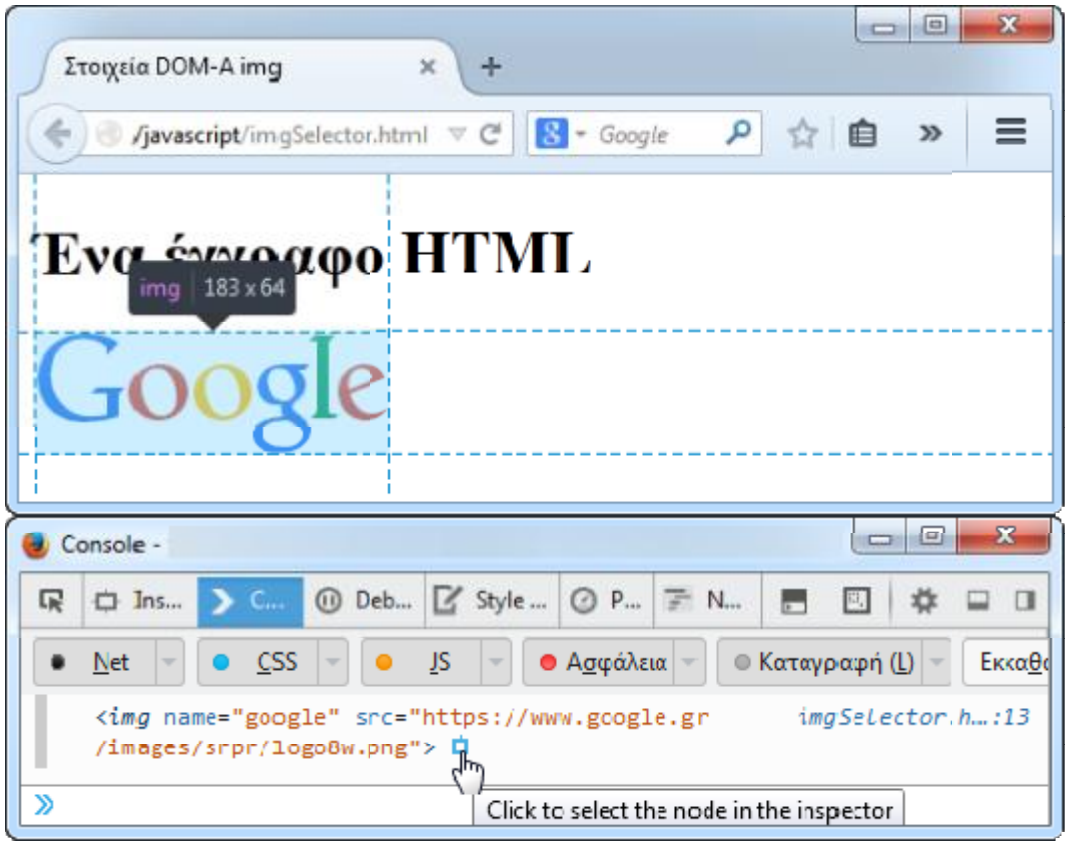
Εικόνες

Για να προσπελάσουμε μια εικόνα αρκεί μόνο να χρησιμοποιήσουμε το στοιχείο-ρίζα «document», μία τελεία και το όνομα της εικόνας που δόθηκε στο γνώρισμα «name».

Αρχείο: imgSelector.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM-A img</title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ένα έγγραφο HTML</h1>
  <img src='https://www.google.gr/images/srpr/logo8w.png'
name='google'/>
 </body>
 <script>
  var image=document.google;
  console.log(image);
 </script>
</html>
```

Αποτέλεσμα:



Εικόνα Β.41 Χρήση εικόνων μέσα από το DOM

Δεν χρειάζεται να χρησιμοποιήσουμε κανένα είδος ιδιότητας αναζήτησης αλλά αρκεί απλά να φωνάξουμε το στοιχείο με το όνομα του. Αν τύχαινε να είχαμε δυο εικόνες με το ίδιο όνομα τότε θα μας επιστρεφόταν όχι ένα σκέτο αντικείμενο αλλά ένας πίνακας

NodeList με τα διαθέσιμα imgs τα οποία θα μπορούσαμε να προσπελάσουμε μέσω ενός δείκτη-συντεταγμένης θέσης.

Φόρμες

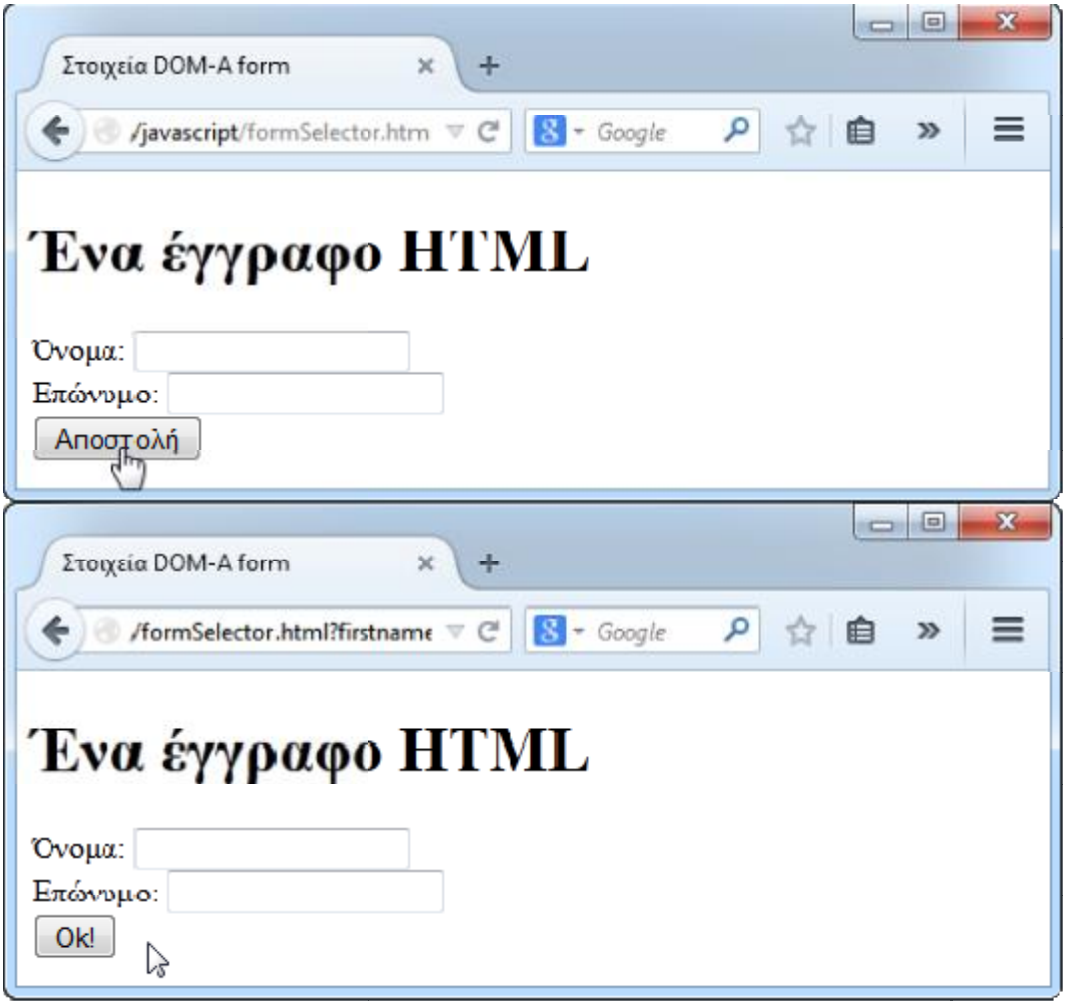
Στις φόρμες έχουμε πολύ παραπάνω δυνατότητες από απλά να αντλήσουμε τα DOM-A τους. Μπορούμε να έχουμε πρόσβαση σε όλα τα στοιχεία τους, αλλάζοντας έτσι τα δεδομένα τους και κάνοντας δυναμικό το περιεχόμενο τους.

Για να αποκτήσουμε πρόσβαση σε μια φόρμα αρκεί να χρησιμοποιήσουμε το στοιχείο ρίζα «document», μία τελεία και το όνομα της φόρμας. Ύστερα για να επιλέξουμε ένα στοιχείο μέσα στην φόρμα αρκεί να συνεχίσουμε μετα το όνομα της φόρμας με μια τελεία και το όνομα του στοιχείου που θέλουμε να αντλήσουμε. Τα ονόματα αυτά προέρχονται από το γνώρισμα «name» του κάθε στοιχείου, είναι απαραίτητο δηλαδή να υπάρχει προκειμένου να μπορέσουμε να χρησιμοποιήσουμε αυτές τις συντομεύσεις.

Αρχείο: formSelector.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM-A form</title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ένα έγγραφο HTML</h1>
  <form name='person' method='get' action=''>
   Όνομα: <input type='text' name='firstname'/><br/>
   Επώνυμο: <input type='text' name='surname' /><br/>
   <input type='submit' value='Αποστολή' name='submitbutton'
onclick='button.value="Ok!"'/>
  </form>
 </body>
 <script>
  var form=document.person;
  var button = form.submitbutton;
 </script>
</html>
```

Αποτέλεσμα:



Εικόνα Β.42 Αλληλεπίδραση με τις ιδιότητες αντικειμένου φόρμας

Κάνοντας κλικ στο κουμπί «Αποστολή» το οποίο έχει γνώρισμα «onclick='button.value="Ok!"'» τότε η ιδιότητα «value» του αντικειμένου «'button» θα αλλάξει στην τιμή «"Ok!"». Το αντικείμενο αυτό δημιουργείται όταν φορτώνεται η σελίδα στον κώδικα που υπάρχει στα tags «<script>», μετά το «</body>».

Β.2.2.3 Επιλογή όλων

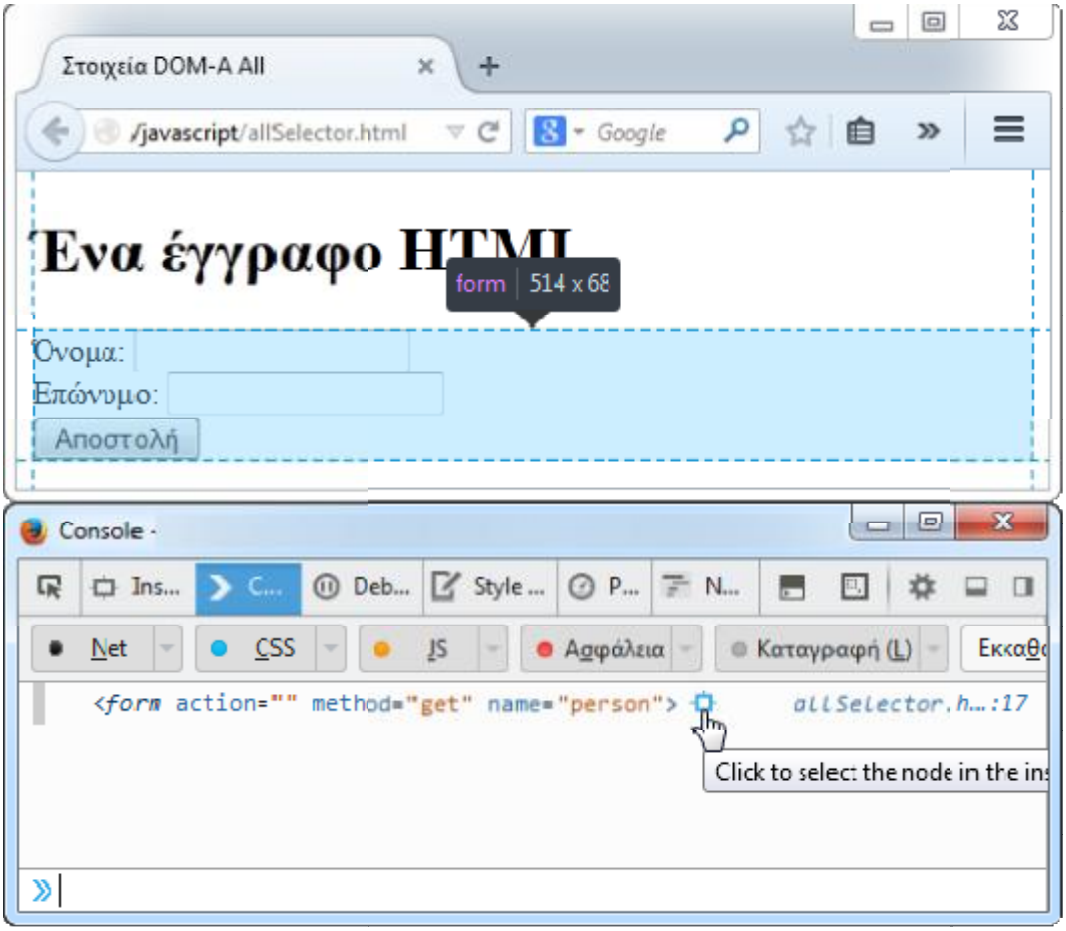
Πριν υπάρξει το DOM και να σταθεροποιηθεί η χρήση του, υπήρχε το «document.all», το οποίο αναπαριστούσε όλα τα αντικείμενα (αλλά όχι τα texts) μέσα σε ένα έγγραφο. Τώρα πλέον έχει αντικατασταθεί με όλα τα παραπάνω που επεξηγήσαμε, αλλά θα ήταν καλό να επεξηγήσουμε πως λειτουργεί και αυτό. Σε όλες τις προηγούμενες επιλογές στοιχείων DOM-A είχαμε μια συγκεκριμένη δομή συντεταγμένων μέσα στους πίνακες που περιηγούμασταν. Στην συγκεκριμένη περίπτωση όλα τα

αντικείμενα παρουσιάζονται σε έναν μονοδιάστατο πίνακα με σειρά όπως βρίσκονται τα αντικείμενα γραμμένα στον κώδικα, αυτό σημαίνει πως το στοιχείο «φόρμα» δεν θα έχει συντεταγμένες «[0][1][1]» αλλά θα βρίσκεται σε μια μονή συντεταγμένη «[6]» με βάση την σειρά που βρίσκεται στον κώδικα.

Αρχείο: allSelector.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Στοιχεία DOM-A All</title>
  <meta charset='utf-8'/>
 </head>
 <body>
  <h1>Ενα έγγραφο HTML</h1>
  <form name='person' method='get' action=''>
   Όνομα: <input type='text' name='firstname'/><br/>
   Επώνυμο: <input type='text' name='surname' /><br/>
   <input type='submit' value='Αποστολή' name='submitbutton'/>
  </form>
 </body>
 <script>
  var form = document.all[6];
  console.log(form);
 </script>
</html>
```

Αποτέλεσμα:



Εικόνα Β.43 Παράδειγμα χρήσης document.all

Β.2.3 Μεταβλητές και πράξεις

Οι μεταβλητές όπως και σε κάθε γλώσσα προγραμματισμού έτσι και στην Javascript είναι ένα μέσο προσωρινής αποθήκευσης και καταχώρησης στην μνήμη. Μια μεταβλητή μπορεί να περιέχει αριθμητικές τιμές, κείμενο, τιμές Boolean (true ή false) τιμές null και undefined ή ακόμη και πίνακες ή αντικείμενα.

Ο τρόπος με τον οποίο αλληλεπιδρούν οι μεταβλητές αυτές μεταξύ τους είναι μέσω πράξεων και τελεστών. Σε αυτό το κεφάλαιο θα μελετήσουμε αναλυτικά την χρήση τόσο των μεταβλητών, όσο και των πράξεων μεταξύ τους.

Β.3 ΜΕΤΑΒΛΗΤΕΣ

Στην Javascript δεν γίνεται διαχωρισμός των τύπων των μεταβλητών, αλλά χρησιμοποιείται από κοινού η δέσμευση χώρου στην μνήμη ανεξαρτήτου τύπου

περιεχομένου που θα χρησιμοποιήσουμε, μπορούμε δηλαδή να δηλώσουμε μια αόριστη μεταβλητή στην αρχή να της καταχωρήσουμε ακέραιο αριθμό, ύστερα να την χρησιμοποιήσουμε σαν String ή Boolean και αργότερα αλλιώς.

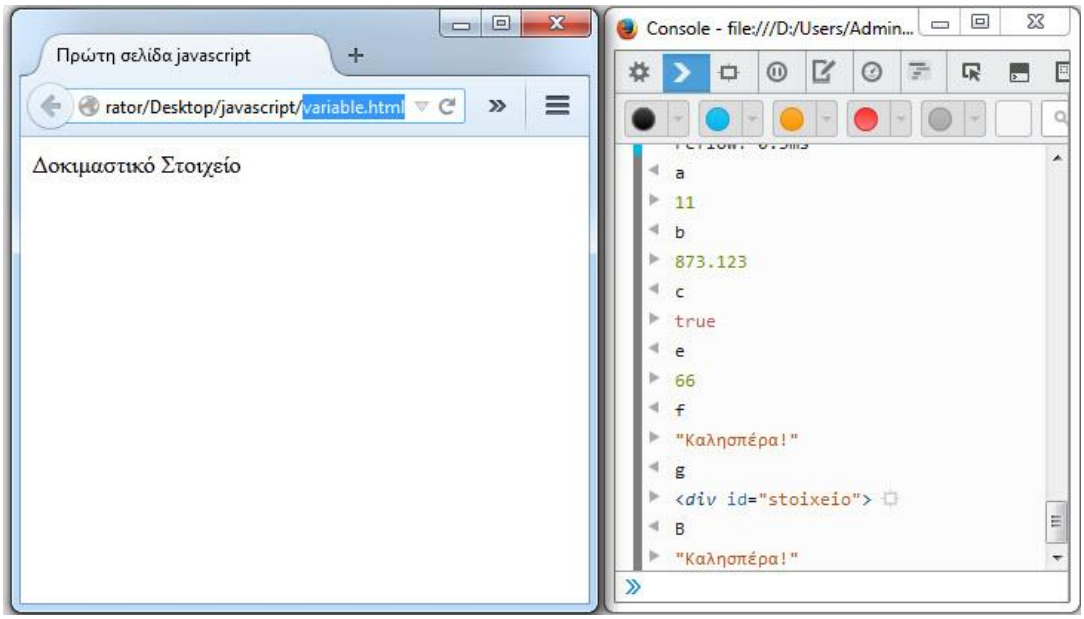
Μία μεταβλητή δηλώνεται με την εντολή «var» και ένα όνομα μεταβλητής το οποίο θα πρέπει αναγκαστικά να είναι σε λατινικούς χαρακτήρες. Μια μεταβλητή πρέπει να χρησιμοποιείται ακριβώς όπως έχει δηλωθεί, έτσι το «var y» δεν είναι ίδιο με το «var Y».

Το «var» προέρχεται από το «variable» που σε ελεύθερη μετάφραση σημαίνει «μεταβλητή».

Αρχείο: variable.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Πρώτη σελίδα javascript</title>
 </head>
 <body>
  <div id='stoixeio'>Δοκιμαστικό Στοιχείο</div>
  <script>
   var a,b,c;
   var B;

   a=11;
   b=873.123;
   B="Καλησπέρα!";
   c=true;

   var e=66;
   var f="Καλησπέρα!";
   var g = document.getElementById("stoixeio");
  </script>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.44 Παράδειγμα χρήσεων μεταβλητών

Κάθε μεταβλητή έχει ημερομηνία λήξεως και γεννιέται όταν την δηλώνουμε. Αν μια μεταβλητή δηλωθεί μέσα σε μια μέθοδο τότε «ζει» για όσο εκτελείται η μέθοδος, μόλις ολοκληρωθεί, η μεταβλητή διαγράφεται από την μνήμη. Στην περίπτωση οι μεταβλητές αυτές λέγονται local (τοπικές), ενώ αν μια μεταβλητή είναι δηλωμένη εκτός των μεθόδων τότε λέγεται global (παγκόσμιες) διότι αυτού του είδους οι μεταβλητές «ζουν» για όσο είναι φορτωμένη η ιστοσελίδα.

Β.3.1 Τελεστές πράξεων και σύγκρισης

Η χρησιμότητα των μεταβλητών και άλλων στοιχείων που δημιουργούνται μέσω μιας γλώσσας προγραμματισμού, γίνεται μέσα από την μεταξύ τους αλληλεπίδραση μέσω των πράξεων που μπορούν να γίνουν. Έστω πως έχουμε μια μεταβλητή x και θέλουμε να την πολλαπλασιάσουμε με την μεταβλητή y. Αυτό που θέλουμε να κάνουμε είναι να καταχωρήσουμε το αποτέλεσμα της αλληλεπίδρασης των μεταξύ μεταβλητών x και y που πραγματοποιείται με τον τελεστή του πολλαπλασιασμού.

Υπάρχουν τριών ειδών τελεστές:

- Αριθμητικοί τελεστές : χρησιμοποιούνται με την πράξη μεταξύ δυο μεταβλητών.

- Συγκριτικοί τελεστές : χρησιμοποιούνται με την μεταξύ σύγκριση δυο μεταβλητών η τιμών. Όταν η σύγκριση είναι αληθής επιστρέφεται η τιμή true ενώ όταν δεν είναι επιστρέφεται η τιμή false.
- Λογικοί τελεστές : χρησιμοποιούνται για την σύγκριση δυο λογικών προτάσεων. Παρομοίως το αποτέλεσμα θα είναι true ή false.

Σε αυτή την ενότητα θα μελετήσουμε και τα τρία είδη τελεστών με αναλυτικά παραδείγματα και επεξήγηση πάνω στην λειτουργία τους.

B.3.1.1 Αριθμητικοί τελεστές

Οι αριθμητικοί τελεστές χρησιμοποιούνται για την αλληλεπίδραση μεταξύ δυο μελών είτε είναι μεταβλητών, είτε οποιαδήποτε άλλου τύπου δεδομένων. Είναι το κατώτερο επίπεδο αλληλεπίδρασης μεταξύ στοιχείων. Παρακάτω θα μελετήσουμε τους πιο σημαντικούς.

- **Πρόσθεση (+)**

  Από τα μαθηματικά γνωρίζουμε πως αν έχουμε να κάνουμε με δυο αριθμούς, τότε από αυτούς θα προκύψει το άθροισμα τους. Εάν όμως έχουμε να κάνουμε με άλλο τύπο για παράδειγμα αν και τα δυο μέλη είναι κείμενο, τότε το ένα κείμενο θα προστεθεί στο τέλος του άλλου, αν το ένα μέλος είναι αριθμός και το άλλο κείμενο, τότε ο αριθμός θα θεωρηθεί ως κείμενο. Γενικώς όταν ένας αριθμός βρίσκεται μέσα σε εισαγωγικά θεωρείται κείμενο.

  Παράδειγμα:

```
var a=5,b=6;
console.log((a+b));
var c="Γεια", d=" σας!";
console.log(c+d);
console.log(a+c);
```

  Αποτέλεσμα:

```
11
Γεια σας!
5Γειά
```

- **Αφαίρεση (-)**

  Αφαιρεί δυο αριθμούς. Ο τελεστής χρησιμοποιείται αποκλειστικά για αριθμητικές πράξεις ακόμη κι αν ο αριθμός βρίσκεται σε εισαγωγικά.

  Παράδειγμα:

```
var a=5,b=6;
console.log((a-b));
var c="20", d="30";
console.log(c-d);
```

Αποτέλεσμα:

```
-1
-10
```

- **Πολλαπλασιασμός (*)**

  Χρησιμοποιείται αποκλειστικά για τον πολλαπλασιασμό αριθμών.

  Παράδειγμα:

```
var a=5,b=6;
console.log((a*b));
var c="20", d="30";
console.log(c*d);
```

Αποτέλεσμα:

```
30
600
```

- **Διαίρεση (/)**

  Χρησιμοποιείται αποκλειστικά για την διαίρεση αριθμών το αποτέλεσμα μπορεί να είναι ακέραιος αριθμός ή δεκαδικός.

  Παράδειγμα:

```
var a=5,b=6;
console.log((a/b));
var c="20", d="30";
console.log(c/d);
```

Αποτέλεσμα:

```
0.8333333333333334
0.6666666666666666
```

- **Υπόλοιπο (%)**

  Χρησιμοποιείται αποκλειστικά για το υπόλοιπο διαίρεσης μεταξύ δυο αριθμών.

  Παράδειγμα:

```
var a=5,b=6;
console.log((a%b));
var c="20", d="30";
console.log(c%d);
```

Αποτέλεσμα:

```
5
20
```

- **Μείωση(--) και Αύξηση (++)**

Χρησιμοποιούνται αποκλειστικά σε αριθμούς. Αυξάνουν ή μειώνουν αντίστοιχα την τιμή ενός αριθμού κατά ένα. Μπορεί να βρίσκεται αριστερά της μεταβλητής ή δεξιά αυτής. Όταν βρίσκεται αριστερά τότε πρώτα αυξάνεται κατά ένα και ύστερα δίνεται το αποτέλεσμα, αν βρίσκεται δεξιά της μεταβλητής τότε πρώτα δίνεται το αποτέλεσμα με την υπάρχουσα τιμή και ύστερα αυξάνεται η τιμή της κατά ένα.

Παράδειγμα:

```
var a=1;
console.log("a++: "+(a++));
console.log("++a: "+(++a));
console.log("a: "+a);
console.log("--a: "+(--a));
console.log("a--: "+(a--));
console.log("a: "+a);
```

Αποτέλεσμα:

```
a++: 1
++a: 3
a: 3
--a: 2
a--: 2
a: 1
```

B.3.1.2 Συγκριτικοί τελεστές

Οι συγκριτικοί τελεστές χρησιμοποιούνται για να ελέγχουν συγκριτικά αν κάτι ισχύει ή δεν ισχύει. Κάθε τελεστής αποτελείται από αριστερό και δεξιό μέλος τα οποία μεταξύ τους συγκρίνονται με βάση των εκάστοτε τελεστή. Αν η σύγκριση βγει αληθής τότε επιστρέφεται τιμή «true» αν δεν είναι, τότε επιστρέφεται τιμή «false». Παρακάτω δίδεται σχετικός πίνακας με όλους τους συγκριτικούς τελεστές και ένα απλό παράδειγμα χρήσης τους.

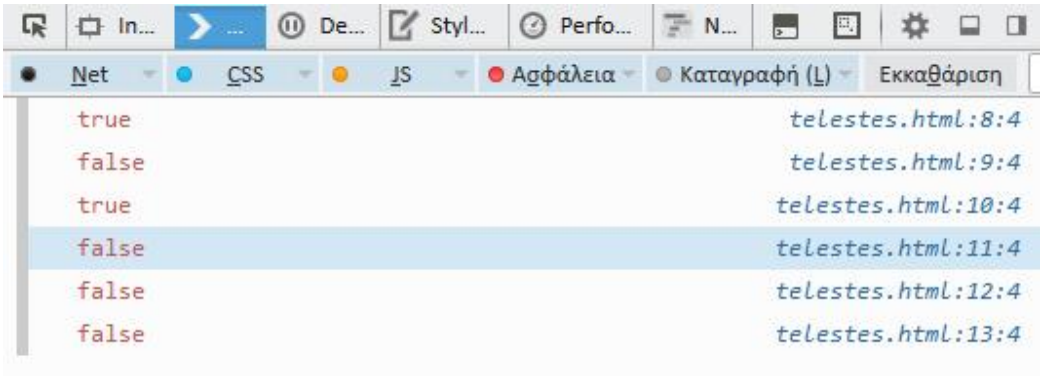
Σύμβολο	Επεξήγηση
>	Μεγαλύτερο
<	Μικρότερο
>=	Μεγαλύτερο ή ίσο
<=	Μικρότερο ή ίσο
==	Ίσον
===	Ίσον (με την ίδιο τύπο μεταβλητής)

Πίνακας B.8 Λίστα συγκριτικών τελεστών

Αρχείο: telestes.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Συντελεστές</title>
  <meta charset='utf-8'/>
  <script>
    var a=5,b=2;
    console.log(a>b);
    console.log(a<b);
    console.log(a>=b);
    console.log(a<=b);
    console.log(a==b);
    console.log(a===b);
  </script>
 </head>
 <body>
  <h1>Μια αμέριμνη ιστοσελίδα.</h1>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.45 Χρήση συγκριτικών τελεστών στην Javascript

Β.3.1.3 Λογικοί τελεστές

Οι λογικοί τελεστές χρησιμοποιούνται σε συνδυασμό με τους συγκριτικούς όπου μπορούν να συνδυαστούν μεταξύ τους περίπλοκοι και συνδυαστικοί έλεγχοι.

- **Λογικό ΚΑΙ (&&)**

  Όταν και τα δυο μέλη είναι true τότε το αποτέλεσμα είναι true αλλιώς είναι false.

```
var a=5,b=6,c=2,d=2;
console.log((a<b && c==d));
```

Το αποτέλεσμα που μας επιστρέφεται στην κονσόλα από την εκτέλεση του παραπάνω κώδικα είναι:

```
true
```

- **Λογικό Ή (||)**

  Όταν τουλάχιστον ένα από τα δυο μέλη είναι true τότε το αποτέλεσμα είναι true διαφορετικά το αποτέλεσμα είναι false.

```
var a=5,b=6,c=2,d=2;
console.log((a<b || c>d));
```

  Το αποτέλεσμα που μας επιστρέφεται στην κονσόλα από την εκτέλεση του παραπάνω κώδικα είναι:

```
true
```

- **Λογικό ΟΧΙ (!)**

  Όταν εφαρμόζεται σε ένα μέλος αλλάζει την λογική τιμή του. Πχ όταν είναι true γίνεται false και όταν είναι false γίνεται true.

```
var a=5,b=6,c=2,d=2;
console.log(!(a<b && c==d));
```

  Το αποτέλεσμα που μας επιστρέφεται στην κονσόλα από την εκτέλεση του παραπάνω κώδικα είναι:

```
false
```

Β.3.1.4 Μαθηματικές πράξεις

Εκτός από τις βασικές πράξεις μεταξύ αριθμών (πρόσθεση, αφαίρεση, πολλαπλασιασμός, διαίρεση, υπόλοιπο) υπάρχουν και κάποιες επιπλέον πράξεις οι οποίες αντιπροσωπεύουν μαθηματικές πράξεις όπως ύψωση σε δύναμη, ρίζα κτλ.

Στην javascript για να χρησιμοποιήσουμε αυτού του είδους τις μεταβλητές θα πρέπει να χρησιμοποιήσουμε την βιβλιοθήκη «Math» όπου περιέχει μέσα τις ένα σύνολο από μεθόδους που κάνουν τέτοιου είδους μαθηματικές πράξεις. Για να χρησιμοποιήσουμε μια από τις παρακάτω δυνατότητες θα πρέπει να έχουμε εισάγει αριστερά αυτών το πρόθεμα «Math.» και ύστερα το όνομα της μεθόδου και τις παραμέτρους τις.

Πράξη	Επεξήγηση
`pow()`	Ύψωση σε δύναμη.
`Round()`	Στρογγυλοποίηση.
`Ceil()`	Στρογγυλοποίηση προς τα πάνω στον κοντινότερο ακέραιο.
`Floor()`	Στρογγυλοποίηση προς τα κάτω στον κοντινότερο ακέραιο.
`Abs()`	Απόλυτη τιμή.
`Max()`	Επιστρέφει την μεγαλύτερη τιμή.

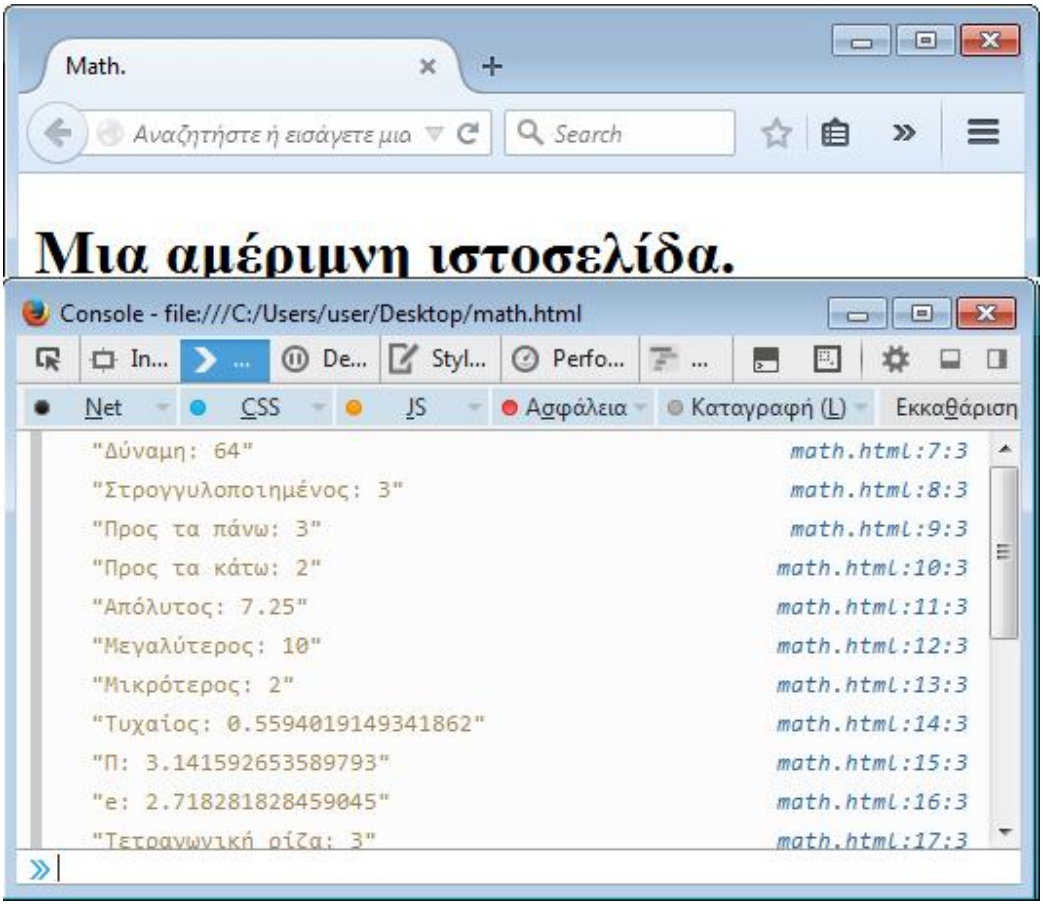
Min()	Επιστρέφει την μικρότερη τιμή.
Random()	Επιστρέφει έναν ψευδό-τυχαίο αριθμό μεταξύ 0 και 1.
PI	Επιστρέφει την τιμή του αριθμού π.
E	Επιστρέφει την τιμή του αριθμού e.
Sqrt()	Επιστρέφει την τετραγωνική ρίζα ενός αριθμού μέσα στις παρενθέσεις.
Sin()	Ημίτονο
Cos()	Συνημίτονο
Atan()	Τόξο εφαπτομένης
Tan()	Εφαπτομένη
Log()	Ο φυσικός λογάριθμος του αριθμού μέσα στις παρενθέσεις
log()/Math.LN10	Λογάριθμος με βάση το 10.
Log()/Math.LN2	Λογάριθμος με βάση το 2.
Exp(3)	Η ύψωση κυβικής δύναμης της τιμής του e.

Πίνακας Β.9 Λίστα μαθηματικών πράξεων στην Javascript

Αρχείο: Math.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Math.</title>
  <meta charset='utf-8'/>
  <script>
   console.log("Δύναμη: "+Math.pow(4,3));
   console.log("Στρογγυλοποιημένος: "+Math.round(2.5));
   console.log("Προς τα πάνω: "+Math.ceil(2.2));
   console.log("Προς τα κάτω: "+Math.floor(2.8));
   console.log("Απόλυτος: "+Math.abs(-7.25));
   console.log("Μεγαλύτερος: "+Math.max(2,5,10,6,7));
   console.log("Μικρότερος: "+Math.min(2,5,10,6,7));
   console.log("Τυχαίος: "+Math.random());
   console.log("Π: "+Math.PI);
   console.log("e: "+Math.E);
   console.log("Τετραγωνική ρίζα: "+Math.sqrt(9));
   console.log("Ημίτονο: "+Math.sin(3));
   console.log("Συνημίτονο: "+Math.cos(3));
   console.log("Τόξο εφαπτομένης: "+Math.atan(2));
   console.log("Εφαπτομένη: "+Math.tan(9));
   console.log("Λογάριθμος: "+Math.log(2));
   console.log("Λογάριθμος βάση 10: "+Math.log(25)/Math.LN10);
   console.log("Λογάριθμος βάση 2: "+Math.log(30)/Math.LN2);
   console.log("e υψωμένο: "+Math.exp(3));
  </script>
 </head>
 <body>
  <h1>Μια αμέριμνη ιστοσελίδα.</h1>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.46 Παράδειγμα χρήσης μαθηματικών συναρτήσεων

Β.3.1.5 Συν άπειρο , πλήν άπειρο

Η javascript δεν εμφανίζει σφάλματα σε περίπτωση υπερχείλισης ή διαίρεσης με το μηδέν. Όταν ένα αποτέλεσμα μιας αριθμητικής/μαθηματικής πράξης είναι μεγαλύτερο από τα άνω όρια που μπορεί να αναπαρασταθεί, τότε το αποτέλεσμα που επιστρέφει η javascript είναι το λεκτικό «infinity». Παρομοίως όταν το αρνητικό αποτέλεσμα μιας τιμής ξεπεράσει τα κάτω όρια που μπορεί να αναπαρασταθεί, τότε το αποτέλεσμα που επιστρέφει η javascript είναι «-infinity». Οι τιμές αυτές μπορούν να χρησιμοποιηθούν όπως χρησιμοποιούνται και στην καθημερινότητα, να τις προσθέσουμε, να τις διαιρέσουμε, να τις πολλαπλασιάσουμε κ.α.

Όταν μια τιμή που προκύπτει βρίσκεται κοντά στο μηδέν αλλά μακριά από την μικρότερη δυνατή αναπαράσταση αριθμού σε αυτό το εύρος, τότε η javascript επιστρέφει μηδέν. Αν μια τιμή που προκύπτει βρίσκεται κοντά στο μηδέν από αρνητικές τιμές , αλλά μακριά από την μεγαλύτερη δυνατή αναπαράσταση αριθμού σε

αυτό το εύρος, τότε η javascript επιστρέφει το λεγόμενο αρνητικό μηδέν. Αυτή η περίπτωση είναι τελείως δυσδιάκριτη από ένα απλό μηδενικό και συνήθως δεν χρησιμοποιείται.

Η διαίρεση με το μηδέν δεν αποτελεί σφάλμα στην javascript, έτσι επιστρέφεται συνήθως άπειρο ή πλην άπειρο. Παρόλα αυτά υπάρχει μια περίπτωση στην οποία η javascript δεν επιστρέφει τιμή, κι αυτή είναι όταν διαιρούμε μηδέν με το μηδέν. Ως αποτέλεσμα επιστρέφεται η τιμή «NaN» (Not a Number), στην προκειμένη δεν μπορούμε να κάνουμε πράξεις με αυτή την τιμή όπως στην περίπτωση του άπειρου κι έτσι ένα script μπορεί να προκύψει προβληματικό λόγω αυτού του λάθους.

Η τιμή «NaN» δεν μπορεί να ελεγχθεί με τελεστή, για παράδειγμα δεν μπορούμε να γράψουμε «x == NaN» για να προσδιορίσουμε αν μια μεταβλητή x είναι NaN. Αντιθέτως θα μπορούσαμε να χρησιμοποιήσουμε την έκφραση «x !=x» το οποίο θα επιστρέφει true εάν η έκφραση είναι NaN. Παρομοίως υπάρχει η μέθοδος isNaN() η οποία είναι σχετική με το παραπάνω. Επιστρέφει true εάν η τιμή είναι NaN ή αν η τιμή περιέχει κείμενο η αντικείμενο. Παρομοίως η μέθοδος isFinite() επιστρέφει true αν μια τιμή είναι NaN, Infinity ή –Infinity.

B.4 ΜΕΘΟΔΟΙ

Συχνά στον προγραμματισμό παρατηρείται η ανάγκη για χρήση κώδικα επαναλαμβανόμενα. Έτσι αντί κάποιος να γράφει τον ίδιο κώδικα συνέχεια σε όποια σημεία τον χρειάζεται, μπορεί απλά να γράψει ένα πακέτο κώδικα το οποίο με το όνομα του θα μπορεί να το «φωνάξει» και να το χρησιμοποιήσει όποτε το χρειαστεί, κάνοντας πάντα αυτό για το οποίο έχει δημιουργηθεί, εκτελώντας δηλαδή τον κώδικα που περιέχει. Αυτό ακριβώς είναι μια μέθοδος, είναι ένα «σύνολο κώδικα» το οποίο τρέχει όποτε εμείς το χρειαζόμαστε, αλλά είναι αυτόνομο σε σχέση με την βασική ροή του προγράμματος μας.

Φανταστείτε πως έχουμε μια συσκευή η οποία μας φτιάχνει ψωμί με αυτόματο τρόπο. Εμείς το μόνο που κάνουμε είναι να εισάγουμε τα απαραίτητα υλικά… πχ Αλεύρι, μαγιά, ζάχαρη, αλάτι, νερό, και η συσκευή αναλαμβάνει από μόνη της να ανακατέψει τα υλικά μεταξύ τους, να φτιάξει την ζύμη και ύστερα να την ψήσει. Ας την

ονομάσουμε «Αρτοπαρασκευαστή». Ο Αρτοπαρασκευαστής μας δέχεται ως είσοδο τα υλικά: Αλεύρι, μαγιά, ζάχαρη, αλάτι, νερό και επιστρέφει ως αποτέλεσμα έτοιμο προς κατανάλωση ψωμί.

Έτσι λειτουργούν και οι μέθοδοι. Συνήθως δέχονται κάποια είσοδο η οποία ονομάζεται «παράμετροι», εκτελούν την λειτουργία για την οποία έχουν οριστεί, και αν τους έχουμε ορίσει να επιστρέφουν αποτέλεσμα τότε το επιστρέφουν. Στην Javascript οι μέθοδοι είναι αντικείμενα και μπορούν να οριστούν ακόμη και μέσα σε μια μεταβλητή. Έτσι μπορούμε να ορίσουμε ιδιότητες γι αυτές , να καλέσουμε άλλες μεθόδους κτλ.

Σε αυτό το κεφάλαιο θα δούμε αναλυτικά πως χρησιμοποιούμε μεθόδους και αντικείμενα.

Β.4.1 Μέθοδος

Μία μέθοδος για να δηλωθεί στην javascript χρειάζεται ένα όνομα, ορίσματα, τιμή επιστροφής και ένα περιεχόμενο κώδικα (μια λειτουργία). Η σύνταξη της έχει ως εξής:

1. Ξεκινάμε με την δήλωση «function» που σημαίνει ότι αυτό που ακολουθεί είναι μια μέθοδος.
2. Ύστερα γράφουμε το όνομα της με λατινικούς χαρακτήρες όπως θα βαφτίζαμε και μια μεταβλητή. Το όνομα της μεθόδου καλείται επίσης και προσδιοριστής μεθόδου και το πιο σωστό είναι τις βαπτίζουμε με ονόματα που αντικατοπτρίζουν την λειτουργία τους. Συνήθως είναι ρήματα ή φράσεις που ξεκινούν από ρήματα. Όταν ένα όνομα διαθέτει πολλές λέξεις τότε τις χωρίζουμε μεταξύ τους με την κάτω παύλα (πχ: like_this();), αλλιώς μπορούμε να γράψουμε όλες τις λέξεις κολλητά, αφήνοντας την πρώτη με μικρά γράμματα αλλά από την δεύτερη λέξη και ύστερα το πρώτο γράμμα πάντα κεφαλαίο (πχ likeThis();). Σε μεθόδους που πρόκειται να παραμείνουν κρυφές και εσωτερικές στον κώδικα συνήθως δίνουν μια ονομασία που ξεκινά με κάτω παύλα (πχ: _likeThis();)
3. Μετά από το όνομα ακολουθεί ένα ζεύγος παρενθέσεων όπου προαιρετικά εάν θέλουμε μπορούμε να συμπεριλάβουμε μέσα σε αυτό παραμέτρους όπου θα

δέχεται η μέθοδος ως είσοδος. Οι συγκεκριμένοι παράμετροι είναι μεταβλητές οι οποίες «ζουν» και «υπάρχουν» για όσο τρέχει η μέθοδος, επομένως μέσα σε αυτές τις παρενθέσεις «δημιουργούμε» μεταβλητές που θα χρησιμοποιηθούν ως το πέρας της εκτέλεσης της μεθόδου, ύστερα καταστρέφονται.

4. Τέλος, ένα ζεύγος αγκύλων όπου μέσα μπορεί να περιέχει κώδικά ή και όχι, ανάλογα με τις προτιμήσεις που έχουμε και θέλουμε η συγκεκριμένη μέθοδος να εκτελεί. Επίσης εάν θέλουμε να επιστρέφεται κάποιο είδος τιμής στο τέλος της εκτέλεσης της μεθόδου χρησιμοποιούμε μέσα στον κώδικα της την εντολή «return». Αν πάλι δεν θέλουμε τότε δεν χρειάζεται καν να την χρησιμοποιήσουμε.

```
function όνομα(πρώτο όρισμα,δεύτερο όρισμα){
 ...Εντολές - λειτουργίες...
 return τιμή ή μεταβλητή;
}
```

- Όπου «όνομα» γράφουμε ένα όνομα για την μέθοδο ακολουθώντας τους παραπάνω κανόνες.
- Όπου «πρώτο όρισμα», «δεύτερο όρισμα» είναι τα σημεία στα οποία τοποθετούμε ονόματα μεταβλητών που αντιπροσωπεύουν τις παραμέτρους εισόδου όπως πχ height, width.
- Όπου «...Εντολές - λειτουργίες...» τοποθετούμε τον κώδικα μας. Συνήθως ο κώδικας αυτός λαμβάνει υπόψη τις εισαγόμενες τιμές, κάνει κάποιες πράξεις με αυτές και επιστρέφει ένα αποτέλεσμα.
- Όπου «τιμή ή μεταβλητή» τοποθετούμε την τιμή που θα επιστραφεί ή την μεταβλητή που περιέχει την τιμή.

Ας δούμε ένα παράδειγμα κώδικα για να εκτελέσουμε μια λειτουργία πρόσθεσης:

```
function addition(num1,num2){
 var sum = num1+num2;
 return sum;
}
```

Όνομα μεθόδου: addition
Ορίσματα εισόδου: num1,num2
Εντολές εκτέλεσης: var sum = num1+num2
Επιστροφή: sum

Στο συγκεκριμένο παράδειγμα θέλουμε να εισάγουμε δυο αριθμούς και να επιστρέφεται το αποτέλεσμα της πρόσθεσης τους. Το ρόλο των δύο εισαγόμενων αριθμών τον τελούν η μεταβλητή num1 και η num2. Όταν κάποιος καλέσει την συγκεκριμένη μέθοδο με το όνομα της θα πρέπει να τις δώσει δυο τιμές ως ορίσματα όπου θα αντιστοιχηθούν στις συγκεκριμένες μεταβλητές num1, num2. Ύστερα θα εκτελεστούν οι εντολές – σώμα της μεθόδου και θα επιστραφεί ως αποτέλεσμα ότι υπάρχει ακριβώς δίπλα από το return στην συγκεκριμένη περίπτωση το sum. Ας δούμε ένα ολοκληρωμένο παράδειγμα:

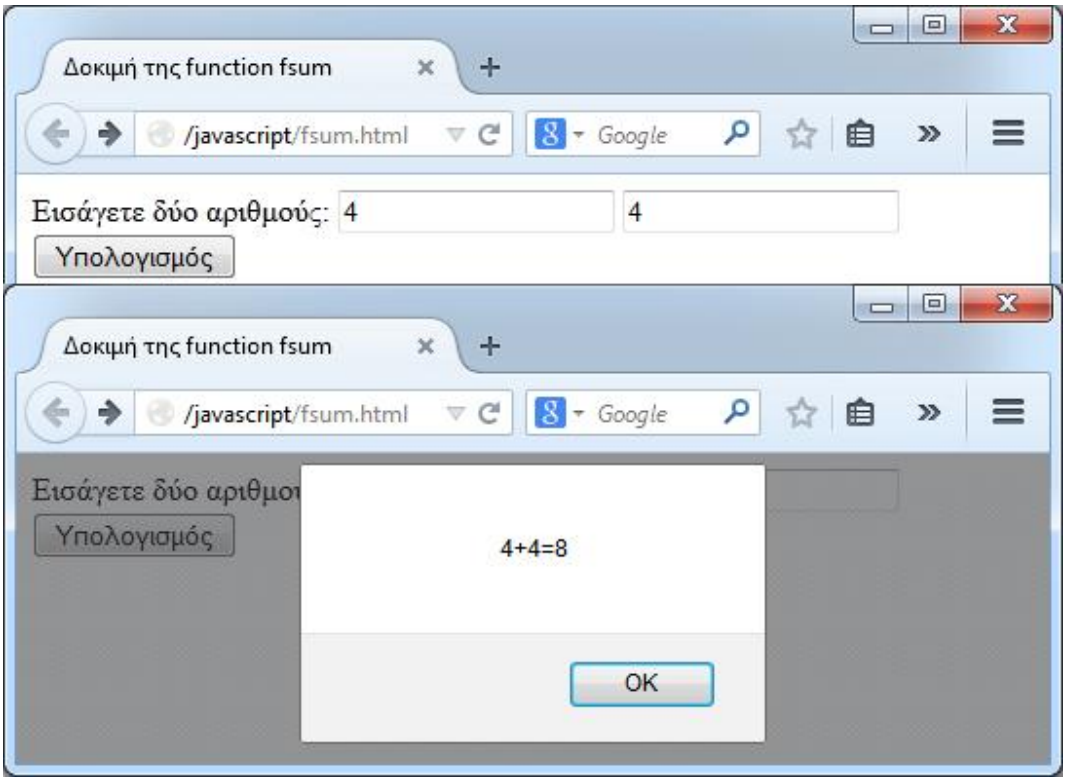
Το παρακάτω πλαίσιο κώδικα κάνει χρήση των στοιχείων DOM μιας σελίδας html έτσι ώστε να μπορέσει να αντλήσει δεδομένα από τα πεδία κειμένου. Επίσης χρησιμοποιείται η εμφωλευμένη μέθοδος Number που μετατρέπει ένα αλφαριθμητικό σε αριθμό.

Παίρνοντας το παραπάνω παράδειγμα πρόσθεσης το μεταφράζουμε σε html-javascript έτσι ώστε να μπορεί να γίνει εκτελέσιμο:

Αρχείο: fsum.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της function fsum</title>
  <script type='text/javascript'>
   function fsum(num1,num2){
    var result = Number(num1)+Number(num2);
    alert(num1+"+"+num2+"="+result);
   }
  </script>
 </head>
 <body>
  Εισάγετε δύο αριθμούς:
  <input type='text' name='num1' id='num1'/>
  <input type='text' name='num2' id='num2' />
  <input type='button' value='Υπολογισμός' onclick='fsum(
   document.getElementById("num1").value,
   document.getElementById("num2").value);' />
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.47 Πρόσθεση με την χρήση μεθόδου

Β.4.2 Οι διάφοροι τύποι μεθόδων

Οι μέθοδοι ως λειτουργίες διακρίνονται σε διάφορους τύπους με βάση τον τρόπο με τον οποίο αλληλεπιδρούν μεταξύ άλλων μεθόδων. Σε αυτή την ενότητα θα δούμε ποίοι είναι οι τύποι αλληλεπίδρασης τους, και πως αυτοί χρησιμοποιούνται.

Β.4.2.1 Αλυσιδωτές μέθοδοι

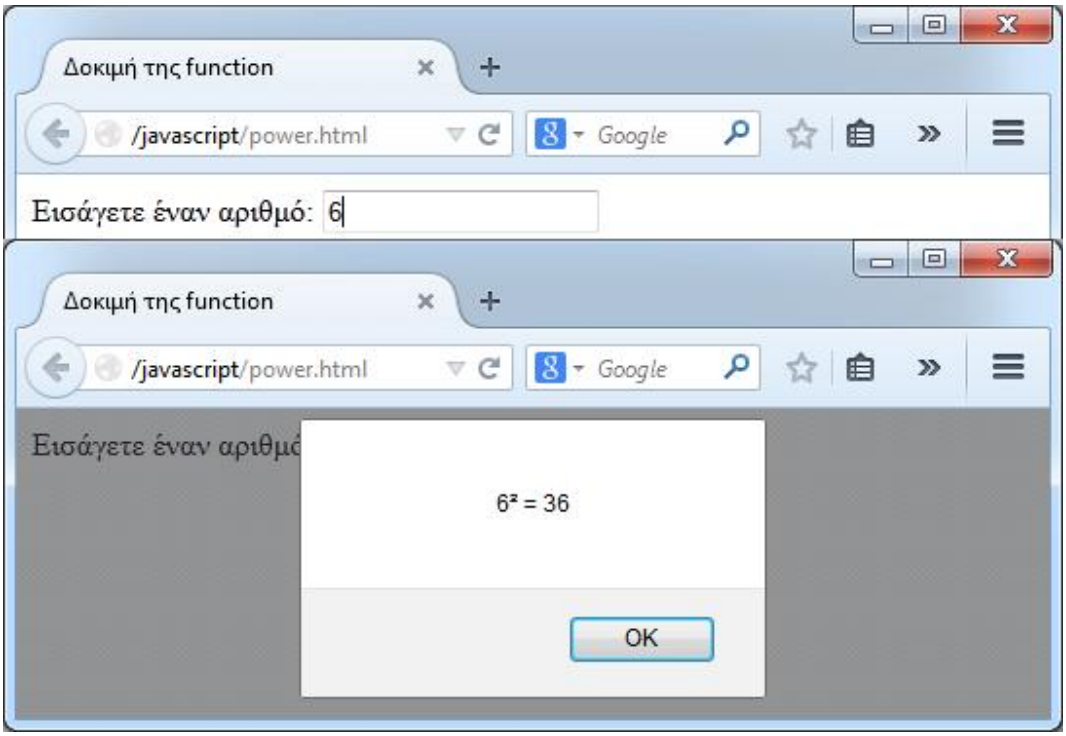
Οι αλυσιδωτές μέθοδοι χρησιμοποιούνται μέσα σε άλλες μεθόδους και έτσι αλυσιδωτά ένα πρόγραμμα μπορεί να εκτελεστεί από μια σειρά μεθόδων όπου κάθε μια έχει αναλάβει ένα ξεχωριστό κομμάτι εκτέλεσης. Αυτό που γίνεται στην ουσία είναι κατά την διάρκεια της εκτέλεσης του κώδικα μιας μεθόδου, υπάρχει μέσα σε αυτόν ένα «κάλεσμα» μιας άλλης μεθόδου.

Ας υποθέσουμε πως θέλουμε να δημιουργήσουμε ένα script το οποίο θα μας υπολογίζει την δύναμη ενός αριθμού που εισάγουμε σε ένα πεδίο κειμένου.

Αρχείο: power.html

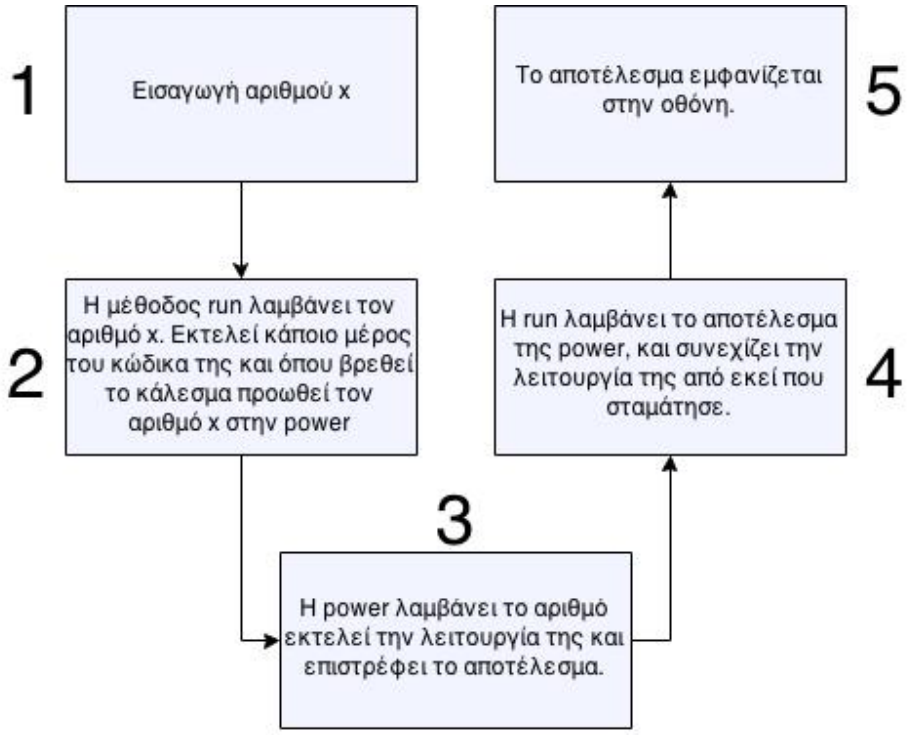
```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της function</title>
  <script type='text/javascript'>
   function power(num){
    var result = num*num;
    return result;
   }

   function run(num1){
    alert(+num1+"² = "+power(num1));
   }
  </script>
 </head>
 <body>
  Εισάγετε έναν αριθμό:
  <input type='text' name='num' id='num' onchange='run(this.value);'>
 </body>
</html>
```

Ο παραπάνω κώδικας μας επιστρέφει τον αριθμό που θα του εισάγουμε, πολλαπλασιασμένο με τον εαυτό του δηλαδή αριθμός^2. Αν εισάγουμε έναν αριθμό στο κουτάκι και πιέσουμε το πλήκτρο enter θα μας εμφανίσει το αποτέλεσμα:



Εικόνα Β.48 Παράδειγμα χρήσης αλυσιδωτών μεθόδων

Στο παράδειγμα με την εφαρμογή της πράξης του υπολογισμού δύναμης ενός αριθμού, δεν υπάρχει μια μόνο μέθοδος αλλά δύο μέσα στο ίδιο script , και η εκτέλεση της λειτουργίας (power) που υπολογίζει το αποτέλεσμα του τετραγώνου του εκάστοτε αριθμού δεν γίνεται άμεσα, αλλά έμμεσα αφού πρώτα εκτελεστεί η μέθοδος run.



Εικόνα Β.49 Τρόπος λειτουργίας αλυσιδωτών μεθόδων

Αν προσέξετε καλά την δεύτερη γραμμή στον κώδικα της run θα παρατηρήσετε ότι μέσα στην εντολή alert υπάρχει το όνομα της power. Στο σημείο στο οποίο βρίσκεται το όνομα της θα γίνει η κλήση προς το μέρος της με παράμετρο το num1.

```
function run(num1){
 alert(+num1+"² = "+power(num1));
}
```

Κάθε μέθοδος που περιέχει μέσα τις λειτουργίες της, είναι κάτι σαν εργάτες που εκτελούν το έργο κατασκευής ενός αμαξιού μέσα σε ένα εργοστάσιο. Για να μπορέσει ο εργάτης Α ,που ως αντικείμενο του είναι να συναρμολογήσει το αμάξι, να ολοκληρώσει το αυτοκίνητο θα πρέπει να περιμένει οι εργάτες Β,Γ,Δ,Ε κ.ο.κ να έχουν τελειώσει την κατασκευή των υπόλοιπων κομματιών του αυτοκινήτου όπως ο κινητήρας, τα καθίσματα, τα φρένα, οι ρόδες κτλ.

B.4.2.2 Εμφωλευμένες μέθοδοι

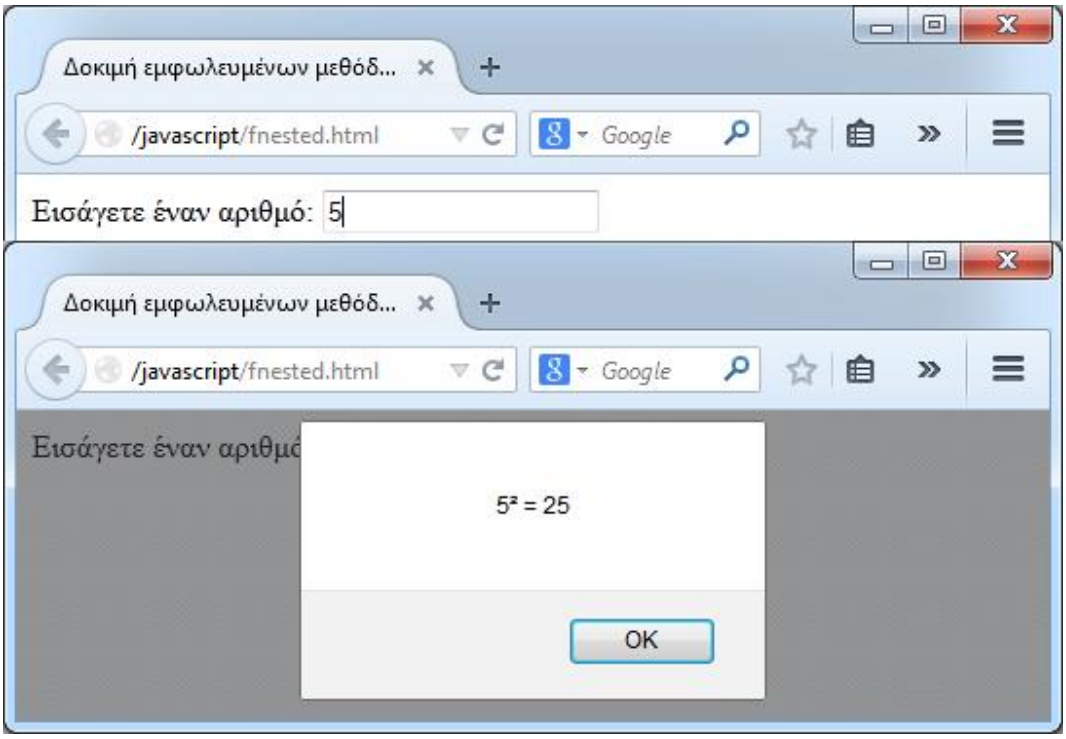
Οι εμφωλευμένες μέθοδοι είναι μια ιδιαίτερη περίπτωση χρήσης μεθόδων όπου μία μέθοδος είναι γραμμένη μέσα σε μια άλλη μέθοδο, και όχι απλά καλείται μέσα από αυτήν.

Το ξεχωριστό όλης αυτής της υπόθεσης είναι ότι η εμφωλευμένη μέθοδος μπορεί να χρησιμοποιηθεί μόνο μέσα στον περιεχόμενο κώδικα της μεθόδου που την εμφωλεύει καθώς επίσης η εμφωλευμένη μπορεί να χρησιμοποιήσει κανονικά τις παραμέτρους της μεθόδου που την εμφωλεύει. Ας δούμε το παράδειγμα των αλυσιδωτών μεθόδων με την χρήση εμφωλευμένων:

Αρχείο: fnested.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή εμφωλευμένων μεθόδων</title>
  <script type='text/javascript'>
   function run(number){
    function power(number){
     var result = number * number;
     return result;
    }
    alert(+number+"² = "+power(number));
   }
  </script>
 </head>
 <body>
  Εισάγετε έναν αριθμό:
  <input type='text' name='number' id='number'
onchange='run(this.value);'/>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.50 Παράδειγμα εκτέλεσης εμφωλευμένων μεθόδων

Το αποτέλεσμα είναι ίδιο, αλλά βέβαια χρησιμοποιώντας τελείως διαφορετικό τρόπο. Με τον τρόπο των αλυσιδωτών μεθόδων τόσο η power όσο και η run μπορεί να χρησιμοποιηθεί οπουδήποτε αλλού την χρειαστούμε. Με τις εμφωλευμένες μεθόδους η power είναι προσπελάσιμη μόνο μέσα στην run, επομένως ΧΡΕΙΑΖΕΤΑΙ να τρέξει η run προκειμένου να τρέξει και η power. Εδώ να σημειώσουμε πως είναι απαραίτητη η κλήση της μεθόδου ή με τον έναν ή με τον άλλο τρόπο.

Β.4.3 Αντικείμενα

Ένα αντικείμενο στην καθημερινή μας ζωή είναι κάτι το οποίο έχει μια υλική υπόσταση, με κάποια συγκεκριμένα χαρακτηριστικά αλλά επίσης και κάποιες λειτουργίες. Ένα αυτοκίνητο για παράδειγμα ως χαρακτηριστικά έχει το μοντέλο του, το έτος παρασκευής του, την μέγιστη ταχύτητα του, τα κυβικά εκατοστά της μηχανής του, και ως λειτουργίες έχει την εκκίνηση του, την χρήση του κινητήρα, την αλλαγή ταχύτητας κ.α.

Κάπως έτσι χρησιμοποιούνται και τα αντικείμενα σε κάθε γλώσσα προγραμματισμού. Παρόμοια λογική ακολουθεί και η javascript. Τα αντικείμενα είναι μεταβλητές στις

οποίες μέσα σε αυτές έχει οριστεί ότι είναι απαραίτητο για την λειτουργία του εκάστοτε αντικειμένου. Το συντακτικό για την δημιουργία αντικειμένου έχει ως εξής:

```
var όνομα_αντικειμένου = {
 ιδιότητα_1: τιμή,
 ιδιότητα_2: τιμή,
 λειτουργία_1: function(παράμετρος_1, παράμετρος_2){
  ...κώδικας που εκτελείται...
  return τιμή;
 },
 λειτουργία_2: function(){
  ...κώδικας που εκτελείται...
  return τιμή;
 }
}
```

1. Χρησιμοποιούμε την λέξη «var» για να ορίσουμε το αντικείμενο μας.
2. Ακολουθεί η ονομασία του τηρώντας τους κανόνες ονομασίας μεταβλητών (λατινικοί χαρακτήρες κλπ).
3. Ύστερα αρχικοποιούμε το αντικείμενο με «=» ίσον.
4. Στην συνέχεια ένα ζεύγος αγκύλων περικλείει τις ιδιότητες (άλλοτε και χαρακτηριστικά) και τις λειτουργίες του αντικειμένου μας.
5. Κάθε χαρακτηριστικό ή λειτουργία αναγράφεται με λατινικούς χαρακτήρες και ύστερα ακολουθούν άνω κάτω τελείες. Μετά τις τελείες ακολουθεί ένας ορισμός τιμής ή λειτουργίας
6. Συνήθως αναγράφονται πρώτα οι ιδιότητες, οπότε και τους δίνουμε τιμές. Κάθε ιδιότητα μετά από το πέρας της δήλωσης τους, τις χωρίζουμε με κόμμα έτσι ώστε να προχωρήσουμε στην επόμενη.
7. Όταν φθάσουμε σε λειτουργίες, ακολουθούμε την ίδια λογική ονομασίας αλλά αντί για κάποια τιμή αναγράφουμε την λέξη «function» ύστερα ένα ζεύγος παρενθέσεων – αν θέλουμε παραμέτρους τις ορίζουμε – και ύστερα ένα ζεύγος αγκύλων όπου θα πλαισιωθεί ο κώδικας της λειτουργίας – μεθόδου του αντικειμένου.
8. Τέλος κλείνοντας την αγκύλη μιας λειτουργίας, πάντα συμπεριλαμβάνουμε και κόμμα αν έχουμε σκοπό να προσθέσουμε και άλλα στοιχεία αλλιώς αν πρόκειται για το τελευταίο στοιχείο, τότε απλά κλείνουμε με αγκύλη.

Ας δούμε ένα παράδειγμα αντικειμένου με βάση την παραπάνω περιγραφή του αυτοκινήτου:

```
var car = {
 model: 10,
 maxspeed: 300,
 gear:0,
 cc:1000,
 started:false,
 startEngine: function(){
  this.started = true;
 },
 changeGear: function(g){
  this.gear=g;
 }
}
```

Ας δούμε την αντιστοίχηση:

Ιδιότητα Μοντέλο: model
Ιδιότητα Μέγιστη ταχύτητα: maxspeed
Ιδιότητα Τρέχουσα ταχύτητα: gear
Ιδιότητα Κυβικά εκατοστά: cc
Ιδιότητα Ενεργό: started
Λειτουργία Εκκίνηση κινητήρα: startEngine
Λειτουργία Αλλαγή ταχύτητας: changeGear

Με όλο τον παραπάνω κώδικα έχουμε κατασκευάσει ένα αντικείμενο με όνομα car και ιδιότητες-λειτουργίες όλες τις παραπάνω. Πως όμως τις χρησιμοποιούμε στον κώδικα μας ; Πολύ απλά χρησιμοποιούμε το όνομα του αντικειμένου ως προσδιοριστή και ύστερα όποια ιδιότητα η λειτουργία θέλουμε χωρισμένη με τελεία από το όνομα του αντικειμένου, πχ:

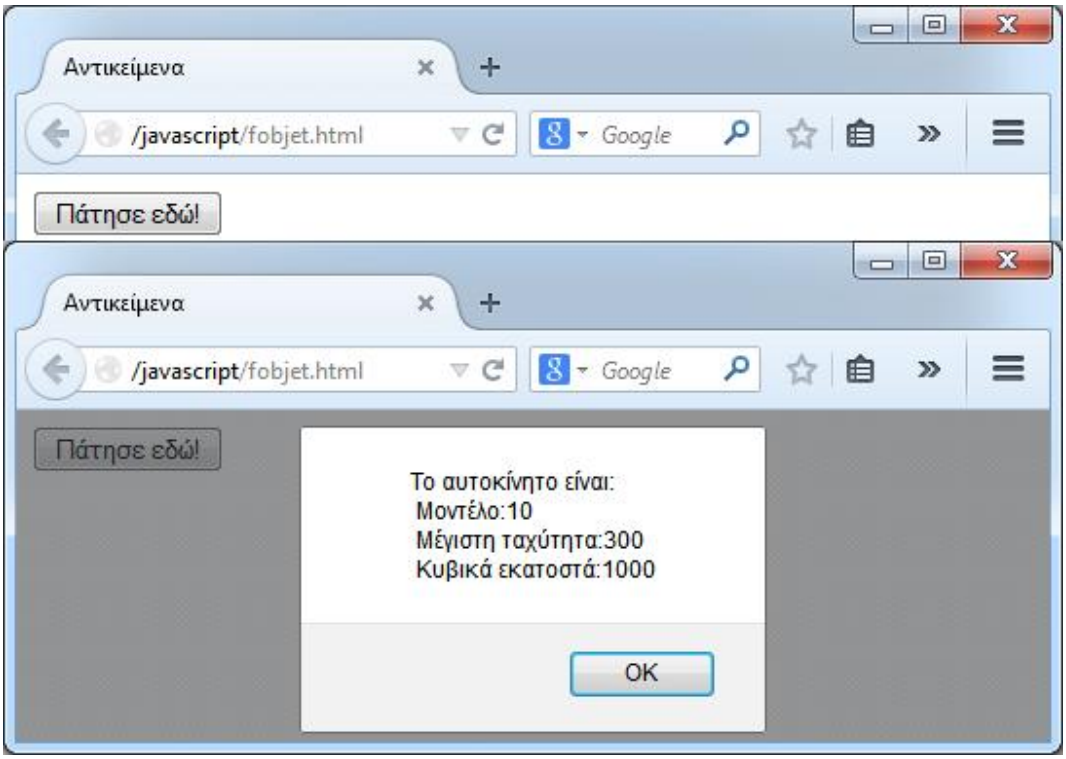
Ιδιότητα-λειτουργία	Τρόπος κλήσης
`model`	`car.model`
`maxspeed`	`car.maxspeed`
`startEngine`	`car.startEngine()`
`changeGear`	`car.changeGear(4)`

Εικόνα Β.51 Τρόπος χρήσης ιδιοτήτων και λειτουργιών αντικειμένου

Αρχείο: fcar.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Αυτοκίνητο</title>
  <script type='text/javascript'>
   var car = {
    model: 10,
    maxspeed: 300,
    gear:0,
    cc:1000,
    started:false,
    startEngine: function(){
     this.started = true;
    },
    changeGear: function(g){
     this.gear=g;
    }
   }
  </script>
 </head>
 <body>
  <input type='button' value='Πάτησε εδώ!' onclick='alert("Το
αυτοκίνητο είναι:\n Μοντέλο:"+car.model+"\n Μέγιστη
ταχύτητα:"+car.maxspeed+"\n Κυβικά εκατοστά:"+car.cc);'/>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.52 Παράδειγμα χρήσης αντικειμένου

Β.5 ΑΠΟΦΑΣΕΙΣ

Στην ζωή μας καθημερινά παίρνουμε κάποιες αποφάσεις με βάση κάποια κριτήρια. Αν έξω βρέχει θα πάρω ομπρέλα, αν το αμάξι δεν έχει βενζίνη θα πάρω λεωφορείο, κ.α. Κάπως έτσι και στον προγραμματισμό μπορούμε να πάρουμε αποφάσεις διακλαδώνοντας των κώδικα μας σε περιπτώσεις με βάση διακριτά μέρη, είτε αριθμητικά είτε και αλφαριθμητικά και άλλα πολλά. Σε αυτό το κεφάλαιο θα δούμε τις βασικότερες εντολές αποφάσεων που υπάρχουν στην javascript καθώς και παραδείγματα χρήσεων τους.

Β.5.1 Εντολή if – εάν

Η εντολή «if» είναι δομή απόφασης η οποία απαντά στο αν ισχύει ή δεν ισχύει μια συνθήκη μέσα σε αυτήν, έτσι μπορούμε να ελέγχουμε την ροή ενός προγράμματος.

Ας θεωρήσουμε μια ηλιόλουστη ημέρα του χειμώνα πως θέλουμε να βγούμε έξω για βόλτα. Το κριτήριο απόφασης για το αν θα πάρουμε μαζί μας μπουφάν είναι αν η θερμοκρασία είναι κάτω από 20 βαθμούς Κελσίου. Η λέξη «εάν» είναι η λέξη κλειδί

στην χρήση της «if». Εάν ισχύει κάτι, τότε κάνε κάτι γι αυτό. Πχ… αν ισχύει ότι ο βαθμός του μαθήματος είναι μεγαλύτερος η ίσος του 5 τότε ο φοιτητής πέρασε το μάθημα. Η σύνταξη της έχει ως εξής:

```
if(συνθήκη){
 ...εντολές...
}
```

Όπου «συνθήκη» μπορεί να εισαχθεί κάποιος έλεγχος μέσω των συντελεστών σύγκρισης ή κάποια μεταβλητή Boolean.

Στο παρακάτω παράδειγμα χρησιμοποιείται εμφωλευμένο script στο συμβάν «onchange» δηλαδή όταν η τιμή του πεδίου αλλάξει.

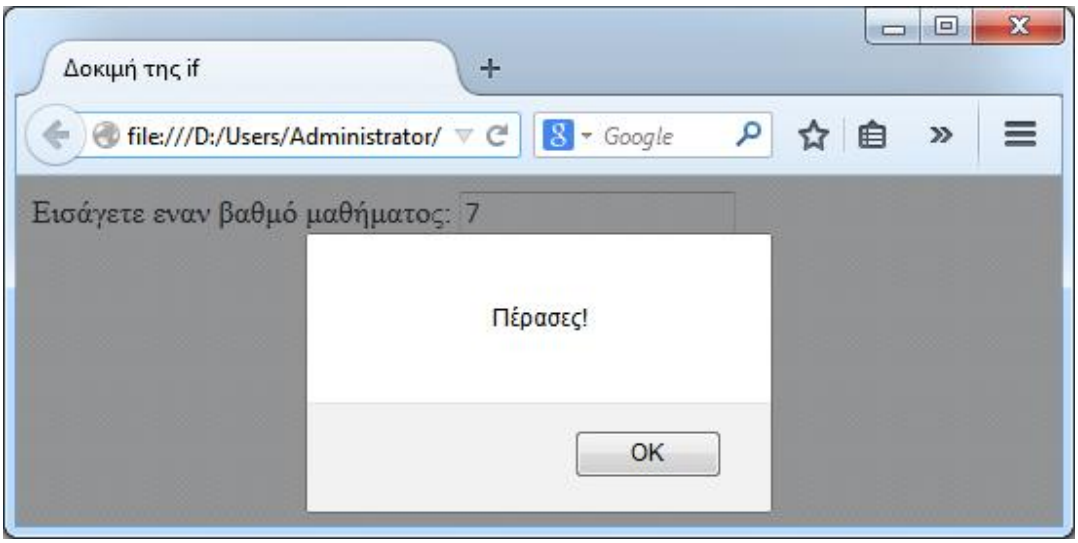
Αρχείο: if.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της if</title>
 </head>
 <body>
  Εισάγετε εναν βαθμό μαθήματος: <input type='text' name='temp'
id='temp' onchange='if(this.value>=5){alert("Πέρασες!");}'>
 </body>
</html>
```

Το παραπάνω με την χρήση μεθόδων, μπορεί να γραφτεί και ως:

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της if</title>
  <script type='text/javascript'>
   function check(x){
    if(x>=5){
     alert("Πέρασες!");
    }
   }
  </script>
 </head>
 <body>
  Εισάγετε εναν βαθμό μαθήματος: <input type='text' name='temp'
id='temp' onchange='check(this.value);'>
 </body>
</html>
```

Όταν εισάγουμε μια τιμή από 5 και πάνω στο πεδίο κειμένου τότε θα μας εμφανιστεί το παρακάτω μήνυμα:



Εικόνα Β.53 Εκτέλεση συνθήκης if

Β.5.2 Εντολή else και else if – αλλιώς

Το «else» συνήθως χρησιμοποιείται όταν κάποιος θέλει να λάβει υπόψη του όλες τις άλλες περιπτώσεις εκτός από αυτή που ελέγχει εκείνη την στιγμή με το «if». Εκτελείται το μέρος του κώδικα μέσα στην «else» αν η συνθήκη μέσα στην «if» είναι ψευδής Δηλαδή στο παράδειγμα της ηλιόλουστης ημέρας του χειμώνα, θα μπορούσε να συμπεριληφθεί ένα αποτέλεσμα στην περίπτωση που η θερμοκρασία ήταν σε οποιοδήποτε άλλο επίπεδο θερμοκρασίας, πχ.

Αρχείο: ifelse_jacket.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της if-else</title>
 </head>
 <body>
  Εισάγετε την παρούσα θερμοκρασία:
  <input type='text' name='temp' id='temp'
onchange='if(this.value<20){alert("Πρέπει να πάρεις
μπουφάν!");}else{alert("Δεν χρειάζεται να πάρεις μπουφάν!");}'>
 </body>
</html>
```

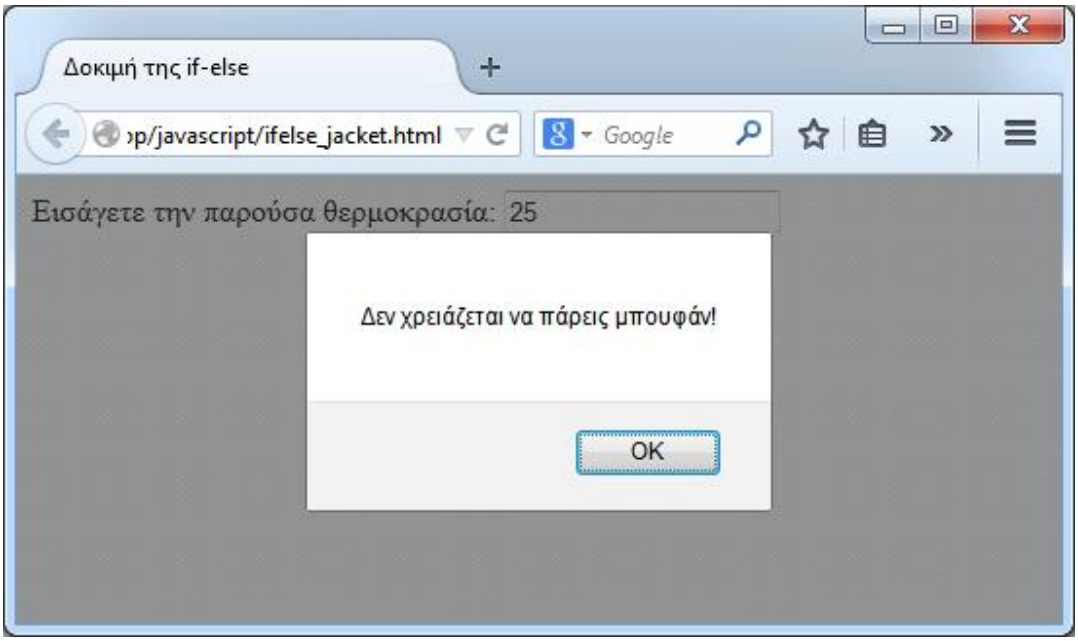
Στο εμφωλευμένο script ο κώδικας ακολουθεί την συνοχή της ίδιας γραμμής, χωρίς να τονίζονται οι διαφορές στα επίπεδα στα οποία μπαίνουμε με αλλαγή γραμμής ή στοίχισης. Αυτό γίνεται καθότι βρισκόμαστε μέσα σε μια ιδιότητα ενός αντικειμένου html και έτσι δεν μπορούμε να χρησιμοποιήσουμε «enter» μέσα σε αυτήν. Το παραπάνω μπορεί να γραφτεί και με την χρήση μεθόδων ως εξής:

Αρχείο: ifelse_jacket_2.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της if-else</title>
  <script type='text/javascript'>
   function check_temp(x){
    if(x<20){ //η θερμοκρασία είναι κάτω από 20 ;
     alert("Πρέπει να πάρεις μπουφάν!"); //αν ναι τυπώνεται
    } else { //αν όχι πηγαίνουμε εδώ
     alert("Δεν χρειάζεται να πάρεις μπουφάν!"); // και τυπώνεται
αυτό
    }
   }
  </script>
 </head>
 <body>
  Εισάγετε την παρούσα θερμοκρασία:
  <input type='text' name='temp' id='temp'
onchange='check_temp(this.value);'>
 </body>
</html>
```

Με το «else» ο κώδικας μας έγινε λίγο πιο πλήρης και έτσι ότι τιμή και να εισάγουμε μέσα στο πεδίο κειμένου θα μας επιστραφεί μια απάντηση.

Αποτέλεσμα:



Εικόνα Β.54 Παράδειγμα εκτέλεσης if-else

Με το παραπάνω παράδειγμα καλύπτουμε την περίπτωση που η θερμοκρασία μπορεί να είναι κάτω από 20 βαθμούς Κελσίου, και επομένως θα πρέπει να πάρουμε μπουφάν, αλλά επίσης καλύπτουμε όλες τις υπόλοιπες περιπτώσεις όπου η θερμοκρασία θα μπορούσε να είναι 20 ή και πάνω από 20 βαθμούς Κελσίου και στην προκειμένη δεν θα χρειαζόταν να πάρουμε μπουφάν.

Συμπεριλαμβάνοντας έτσι το «else», βοηθάμε διευκρινίζοντας το τι γίνεται σε όλες τις άλλες περιπτώσεις, από αυτή που το πρόγραμμα μας θα εκτελούσε ένα ευτυχές σενάριο. Αν το πρόγραμμα μας ήταν να μας υποδεικνύει πότε να πάρουμε μπουφάν με βάση την θερμοκρασία του περιβάλλοντος τότε το να συμπεριλάβουμε το πότε ΔΕΝ χρειάζεται να πάρουμε μπουφάν, είναι προφανώς κάτι που απαιτείται, καθώς ένας απλός χρήστης του προγράμματος δεν θα καταλάβει ότι το πρόγραμμα «λειτουργεί» αν δεν του εμφανίσει τίποτα ως αποτέλεσμα.

Τι γίνεται όμως όταν θέλουμε να διακρίνουμε αυτή την διαφορά σε περισσότερα μέρη ;

Ας θεωρήσουμε πως μπουφάν φοράμε από τους 15 βαθμούς Κελσίου και κάτω. Από τους 15 έως τους 22 παίρνουμε ζακέτα, και από 22 και πάνω δεν παίρνουμε τίποτα.

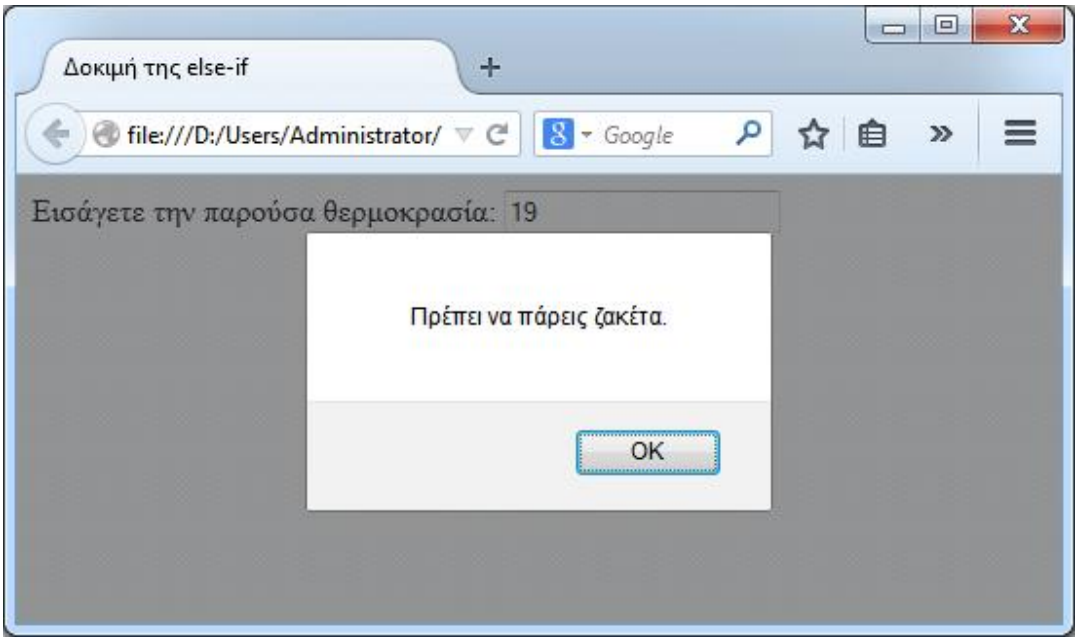
Αρχείο: elseif_jacket.html (με εμφωλευμένο script)

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της else-if</title>
 </head>
 <body>
  Εισάγετε την παρούσα θερμοκρασία:
  <input type='text' name='temp' id='temp'
onchange='if(this.value<=15){alert("Πρέπει να πάρεις μπουφάν!");}else
if(this.value>15 && this.value<22){alert("Πρέπει να πάρεις
ζακέτα.");}else if(this.value>=22){alert("Δεν χρειάζεται να πάρεις
μπουφάν η ζακέτα.");}'>
 </body>
</html>
```

Αρχείο: elseif_jacket2.html (ξεχωριστό script)

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της else-if</title>
  <script type='text/javascript'>
   function check_temp(x){
    if(x<=15){
     alert("Πρέπει να πάρεις μπουφάν!");
    }else if(x>15 && x<22){
     alert("Πρέπει να πάρεις ζακέτα.");
    }else if(x>=22){
     alert("Δεν χρειάζεται να πάρεις μπουφάν η ζακέτα.");
    }
   }
  </script>
 </head>
 <body>
  Εισάγετε την παρούσα θερμοκρασία:
  <input type='text' name='temp' id='temp'
onchange='check_temp(this.value);'>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.55 Διακλάδωση περιπτώσεων μέσα σε πολλαπλά if-else

Β.5.3 Εντολή switch και case

Η εντολή «switch» είναι μια εντολή απόφασης που προσπαθεί να συντομεύσει την χρήση πολλαπλών εντολών «if». Με την χρήση της «switch» αποφεύγουμε την χρήση πολλαπλών «if» τα οποία θα έκαναν πολύ δύσκολο τον έλεγχο του προγράμματος από την οπτική γωνία του προγραμματιστή, και συντακτικά θα ήταν δύσκολο να βρεθούν συγκεκριμένες περιπτώσεις. Ο τρόπος σύνταξης της «switch» έχει συγκεκριμένο τρόπο και γίνεται ως εξής:

```
switch(μεταβλητή){
 case περίπτωση πρώτη:
  ...εντολές για πρώτη περίπτωση ...
 break;
 case περίπτωση δεύτερη:
  ...εντολές για δεύτερη περίπτωση ...
 break;
 default:
  ...εντολές για καμία από τις παραπάνω περιπτώσεις ...
 break;
}
```

- Όπου «μεταβλητή» εισάγουμε το όνομα της μεταβλητής που θέλουμε να μελετήσουμε.
- Όπου «περίπτωση πρώτη» εισάγουμε τον έλεγχο που θέλουμε να κάνουμε σε αυτή την περίπτωση, ότι θα βάζαμε δηλαδή στις παρενθέσεις της «if».

- Παρομοίως για την «περίπτωση δεύτερη», «περίπτωση τρίτη» κ.ο.κ.
- Η εντολή «break» σημαίνει ότι θα σταματήσει η javascript την εκτέλεση του κώδικα της «switch» στο σημείο όπου θα βρεθεί, διαφορετικά η javascript θα εκτελούσε όλες τις εντολές από το σημείο που βρήκε την «case» που ταιριάζει με την περίπτωση. Όλες οι εντολές για κάθε περίπτωση ξεκινούν από την άνω και κάτω τελεία του «case» και τελειώνουν στο «break».
- Το «default» χρησιμοποιείται για όλες τις άλλες περιπτώσεις, είναι δηλαδή το «else» της «switch».

Σε αντίθεση με άλλες γλώσσες προγραμματισμού και scripting η javascript δεν επιτρέπει τις συνθήκες μέσα σε περιπτώσεις της «switch». Έτσι δεν μπορεί να ελεγχθεί μια περίπτωση της οποίας η θερμοκρασία για παράδειγμα είναι μικρότερη ή ίση του 15, αλλά μόνο αν είναι 15. Δεν μπορούμε έτσι να καθορίσουμε περιοχές τιμών μέσα σε μια «case» της «switch». Την χρησιμοποιούμε διακριτά για τιμές που γνωρίζουμε από πριν.

Ας υποθέσουμε πως δίνοντας έναν αριθμό από το 1 έως το 7 θέλουμε να μας επιστρέφεται η μέρα της εβδομάδας στην οποία αντιστοιχεί ξεκινώντας από την Δευτέρα. Αν χρησιμοποιούσαμε «if» , το αποτέλεσμα θα είχε ως εξής:

Αρχείο: if_switch.html

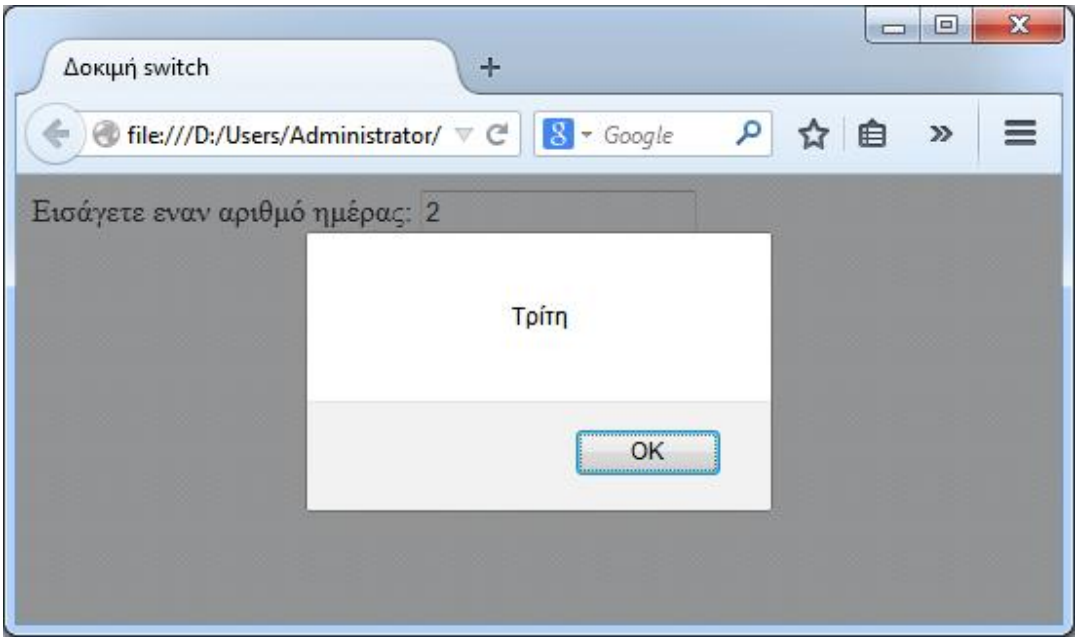
```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Πολλαπλά if, else-if</title>
  <script type='text/javascript'>
   function check_day(x){
    if(x==1){
     alert("Δευτέρα");
    }else if(x==2){
     alert("Τρίτη");
    }else if(x==3){
     alert("Τετάρτη");
    }else if(x==4){
     alert("Πέμπτη");
    }else if(x==5){
     alert("Παρασκευή");
    }else if(x==6){
     alert("Σαββάτο");
    }else if(x==7){
     alert("Κυριακή");
    }else{
     alert("Δεν ισάχθηκε σωστός αριθμός!");
    }
   }
  </script>
 </head>
 <body>
  Εισάγετε εναν αριθμό ημέρας:
  <input type='text' name='temp' id='temp'
onchange='check_day(this.value);'>
 </body>
</html>
```

Ας δούμε τώρα το παραπάνω πως μπορεί να υλοποιηθεί σε switch.

Αρχείο: switch.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή switch</title>
  <script type='text/javascript'>
   function check_day(x){
    switch(Number(x)){
     case 1:
      alert("Δευτέρα");
     break;
     case 2:
      alert("Τρίτη");
     break;
     case 3:
      alert("Τετάρτη");
     break;
     case 4:
      alert("Πέμπτη");
     break;
     case 5:
      alert("Παρασκευή");
     break;
     case 6:
      alert("Σαββάτο");
     break;
     case 7:
      alert("Κυριακή");
     break;
     default:
      alert("Δεν εισάχθηκε σωστός αριθμός");
     break;
    }
   }
  </script>
 </head>
 <body>
  Εισάγετε εναν αριθμό ημέρας:
  <input type='text' name='temp' id='temp'
onchange='check_day(this.value);'>
 </body>
</html>
```

Το αποτέλεσμα και των δυο είναι το ίδιο:



Εικόνα Β.56 Αποτέλεσμα εκτέλεσης εντολής switch

Παρατηρώντας την σύνταξη της «switch», βλέπουμε πως μας δίνεται μια πιο συγκεκριμένη δομή στον τρόπο με τον οποίο «εξετάζονται» οι περιπτώσεις. Έτσι είναι ευκολότερη η επεξεργασία και η προσθήκη γραμμών κώδικα στο μέλλον.

Β.6 Βρογχοι

Στην ζωή μας καθημερινά εμφανίζεται μια φυσική διαδικασία ή μια υποχρέωση η οποία είναι επαναλαμβανόμενη . Ένα φυσικό παράδειγμα είναι η έμμηνος ρύση, ή οι εποχές της γης. Ένα παράδειγμα υποχρέωσης θα μπορούσε να ήταν η χρήση ενός χαπιού σε καθημερινή βάση.

Στον προγραμματισμό το πότε ολοκληρώνεται ένας κύκλος επαναλήψεων το ορίζει μια συνθήκη. Η συνθήκη αυτή ελέγχεται σε κάθε τέλος ή αρχή του κάθε γύρου, κι αν ισχύει τότε ο βρόγχος συνεχίζεται ώσπου η συνθήκη πλέον να μην ισχύει. Συνήθως μέσα σε κάθε βρόγχο υπάρχει και ένας μετρητής όπου θα καθορίζει για πόσο θα επαναλαμβάνεται η διαδικασία. Ας υποθέσουμε πως έχουμε μια μεταβλητή που συμβολίζει τις ημέρες τις εβδομάδας, μηδέν για Κυριακή, ένα για Δευτέρα κτλ. Αν θέλαμε να κάνουμε κάτι επαναλαμβανόμενο με βάση αυτό τον τρόπο λειτουργίας θα ορίζαμε την συνθήκη, για όσο η μεταβλητή είναι μικρότερη ή ίση του 6 (=Σαββάτο).

Έτσι όταν ο μετρητής φθάσει την τιμή 7 που δεν αντιστοιχεί σε κάποια μέρα της εβδομάδας θα σταματούσε ο βρόγχος γιατί δεν ισχύει πλέον η συνθήκη.

Σε αυτό το κεφάλαιο θα δούμε αναλυτικά τα διαφορετικά είδη βρόγχων που υπάρχουν στην javascript και πως χρησιμοποιούνται.

B.6.1 while

Ο πιο απλούστερος βρόγχος στην javascript είναι η βρόγχος «while». Η σύνταξη του βασίζεται σε μια συνθήκη, παρόμοια με την εντολή «if». Η διαφορά τους είναι ότι η «if» εκτελεί μόνο μια φορά το τμήμα κώδικα που την αφορά, ενώ η «while» για όσο είναι αληθής η συνθήκη της.

Γενικώς η «while» χρησιμοποιείται σε περιπτώσεις που δεν γνωρίζουμε για πόσες επαναλήψεις θα απαιτηθούν για να γίνει η συνθήκη ψευδής. Δεν ξεχνάμε να ορίσουμε την μεταβλητή που αντιπροσωπεύει τον μετρητή, πριν από το «while», όπως επίσης δεν ξεχνάμε να αυξήσουμε η μειώσουμε τον μετρητή μέσα στο «while», αλλιώς θα καταλήξουμε σε ατέρμονο βρόγχο.

Συνήθως σε έναν βρόγχο δεν θέλουμε να κάνουμε πράξεις με την ίδια συνεχόμενα τιμή, επομένως χρησιμοποιούμε τον μετρητή, άλλες φορές ως έμμεσο συμμετέχοντα πχ ως δείκτη θέσεως κάποιου πίνακα, είτε άλλες το χρησιμοποιούμε άμεσα για χρήση του μέσα σε πράξεις. Έτσι κάθε επανάληψη του βρόγχου θα μας δίνει διαφορετικό αποτέλεσμα και δεν θα κάνουμε κάτι συνεχόμενα ίδιο κατ' επανάληψη. Ο τρόπος σύνταξης έχει ως εξής:

```
var μετρητής = τιμή;
while(συνθήκη){
 ...εντολές προς επανάληψη...
 μετρητής++;
}
```

- Όπου «var μετρητής = τιμή» πηγαίνει το όνομα του μετρητή μας, και μια αρχική τιμή γι αυτόν.
- Όπου «συνθήκη», η συνθήκη που θέλουμε για όσο ισχύει να επαναλαμβάνονται οι «εντολές προς επανάληψη».

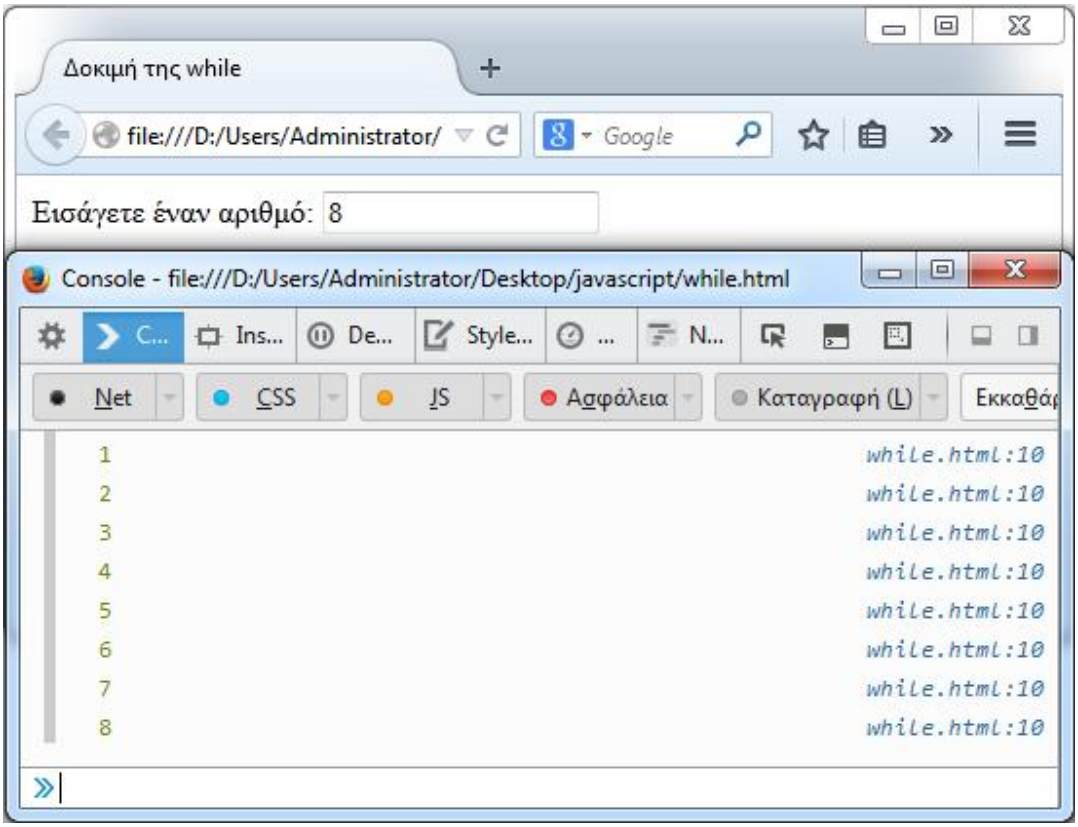
- Όπου «μετρητής++» ο μετρητής μας αυξάνεται κατά μια μονάδα, θα μπορούσε επίσης να μειώνεται ή να γίνεται οποιαδήποτε πράξη ανάθεσης τιμής.

Ας υποθέσουμε πως εισάγουμε έναν αριθμό και θέλουμε από το 1 έως και τον αριθμό που θα εισάγουμε, να τυπώσει (στην κονσόλα) κατά σειρά όλους τους αριθμούς.

Αρχείο: while.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της while</title>
  <script type='text/javascript'>
   function nums_ins(x){
    var i = 1;
    while(i<=x){
     console.log(i);
     i++;
    }
   }
  </script>
 </head>
 <body>
  Εισάγετε έναν αριθμό:
  <input type='text' name='temp' id='temp'
onchange='nums_ins(this.value);'>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.57 Αποτέλεσμα εκτέλεσης while

Όπως μπορείτε να δείτε στην Εικόνα Β.57 η μεταβλητή «i» που είναι ο μετρητής μας στην προκειμένη περίπτωση αρχικοποιείται στην τιμή 1 πριν από την εκκίνηση του βρόγχου. Μέσα στον βρόγχο ο μετρητής μας αυξάνεται κατά μια μονάδα. Όταν ο βρόγχος εκτελεστεί όσες φορές του εισάγουμε, η συνθήκη παύει να ισχύει γιατί ο μετρητής είναι μεγαλύτερος από όσες επαναλήψεις του ζητήσαμε. Εφόσον η συνθήκη δεν ισχύει ο βρόγχος σπάει και ο κώδικας συνεχίζεται αμέσως μετά το κλείσιμο του while.

Συχνά θα παρατηρείτε ονόματα μετρητών σε διάφορα παραδείγματα στο διαδίκτυο όπως «count», «counter», όπως επίσης όμως χρησιμοποιούνται μεταβλητές με ονόματα όπως «i» , «j» και «k». Όσο πιο περιγραφικά ονόματα χρησιμοποιείτε στον κώδικα σας τόσο ευκολότερο θα είναι να τον επεξεργαστείτε.

B.6.2 do while

Το «do-while» είναι ένα είδος βρόγχου το οποίο εκτελείται τουλάχιστον μία φορά πριν αρχίσει να επαναλαμβάνεται ελέγχοντας αν ισχύει η συνθήκη του. Αυτό που συμβαίνει είναι ότι ο έλεγχος της συνθήκης γίνεται στο τέλος και όχι στην αρχή κάθε επανάληψης, επομένως εκτελείται τουλάχιστον μία φορά. Παρόμοια με την while έτσι κι εδώ ο μετρητής δηλώνεται έξω από τον βρόγχο, και ο μετρητής αλλάζει (αυξάνεται ή μειώνεται) μέσα σε αυτόν. Ο τρόπος σύνταξης έχει ως εξής:

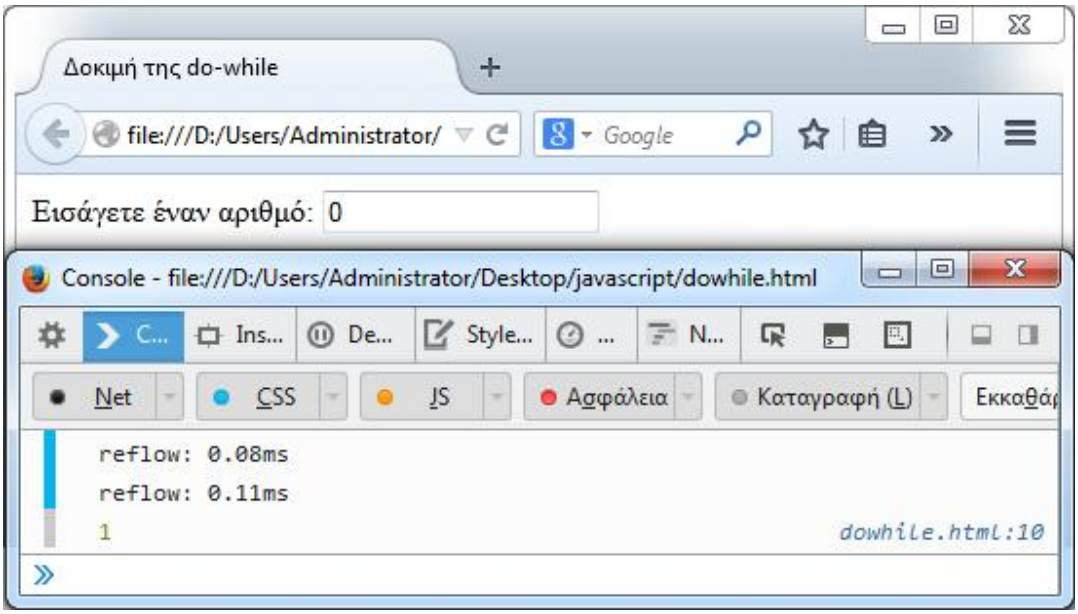
```
var μετρητής=τιμή;
do{
 ... εντολές για επανάληψη ...
 μετρητής++;
}while(συνθήκη);
```

- Όπου «var μετρητής = τιμή» πηγαίνει το όνομα του μετρητή και η αρχική τιμή του.
- Όπου «... εντολές για επανάληψη ...» πηγαίνουν οι εντολές πού θέλουμε να εκτελεστούν τουλάχιστον μία φορά και ύστερα κατ' επανάληψη
- Όπου «μετρητής++» αυξάνεται ο μετρητής κατά μια μονάδα, θα μπορούσε κάλλιστα να μειώνεται η γενικώς να γίνεται οποιαδήποτε πράξη αλλαγής του.
- Όπου «συνθήκη» πηγαίνει η συνθήκη που θέλουμε να ελέγχουμε στο τέλος κάθε επανάληψης.

Αρχείο:dowhile.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της do-while</title>
  <script type='text/javascript'>
   function nums_ins(x){
    var i=1;
    do{
     console.log(i);
     i++;
    }while(i<=x);
   }
  </script>
 </head>
 <body>
  Εισάγετε έναν αριθμό:
  <input type='text' name='temp' id='temp'
onchange='nums_ins(this.value);'>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.58 Αποτέλεσμα εκτέλεσης βρόγχου do-while

Στο παραπάνω παράδειγμα, παρόλο που εισάγαμε 0 για αριθμό βάση του οποίου θα ξεκινήσει η μέτρηση, τυπώθηκε ο αριθμός «1» που σημαίνει ότι εκτελέσθηκε ο βρόγχος τουλάχιστον μία φορά χωρίς να λάβει υπόψη την συνθήκη.

B.6.3 for

Το «for» είναι μια πιο οργανωμένη προσέγγιση βρόγχου επανάληψης όπου το πόσες φορές θα εκτελεστεί ο βρόγχος δηλώνεται στην αρχή, μαζί με τον το όνομα της εντολής for. Συνήθως οι πιο πολλοί βρόγχοι όπως είδαμε παραπάνω, ζητούν έναν ξεχωριστό μετρητή τον οποίο θα πρέπει να δηλώνουμε έξω από τον βρόγχο και τον οποίο εμείς θα πρέπει να αλλάζουμε μέσα σε αυτόν. Στην for η δήλωση του μετρητή και η αύξηση του γίνεται μαζί με την δήλωση της εντολής του βρόγχου. Επομένως μέσα στο εκτελεστικό κώδικα για κάθε επανάληψη δεν χρειάζεται να συμπεριλάβουμε κανένα είδος αύξησης ή μείωσης μετρητή καθώς επίσης, δεν χρειάζεται να δηλώσουμε τον μετρητή μας έξω από τον βρόγχο. Όλα αυτά γίνονται αυτόματα με την δήλωση της εντολής.

Η for μαζί με το όνομα της ως εντολή δέχεται και τρεις ξεχωριστές παραμέτρους χωρισμένες με Ελληνικό ερωτηματικό:

1. Δήλωση του μετρητή μαζί με την αρχική τιμή του.
2. Συνθήκη του βρόγχου (συνήθως εδώ χρησιμοποιείται ο μετρητής).
3. Αύξηση ή μείωση της τιμής του μετρητή.

```
for( Αρχικοποίηση ; Συνθήκη ; Αλλαγή){
 ..Εντολές εκτέλεσης...
}
```

1. Όπου «Αρχικοποίηση» μπαίνει η δήλωση του ονόματος του μετρητή μαζί με την αρχική τιμή του.
2. Όπου «Συνθήκη» μπαίνει η συνθήκη μέσω της οποίας καθορίζεται το πότε θα τελειώσει ο βρόγχος.
3. Όπου «Αλλαγή» μπαίνει η εντολή μέσω της οποία ο μετρητής θα πάρει διαφορετική τιμή. Μπορεί να αυξάνεται κατά μια μονάδα, να μειώνεται κατά μία μονάδα ή να χρησιμοποιηθεί οποιαδήποτε είδος ορθής ανάθεσης τιμής.

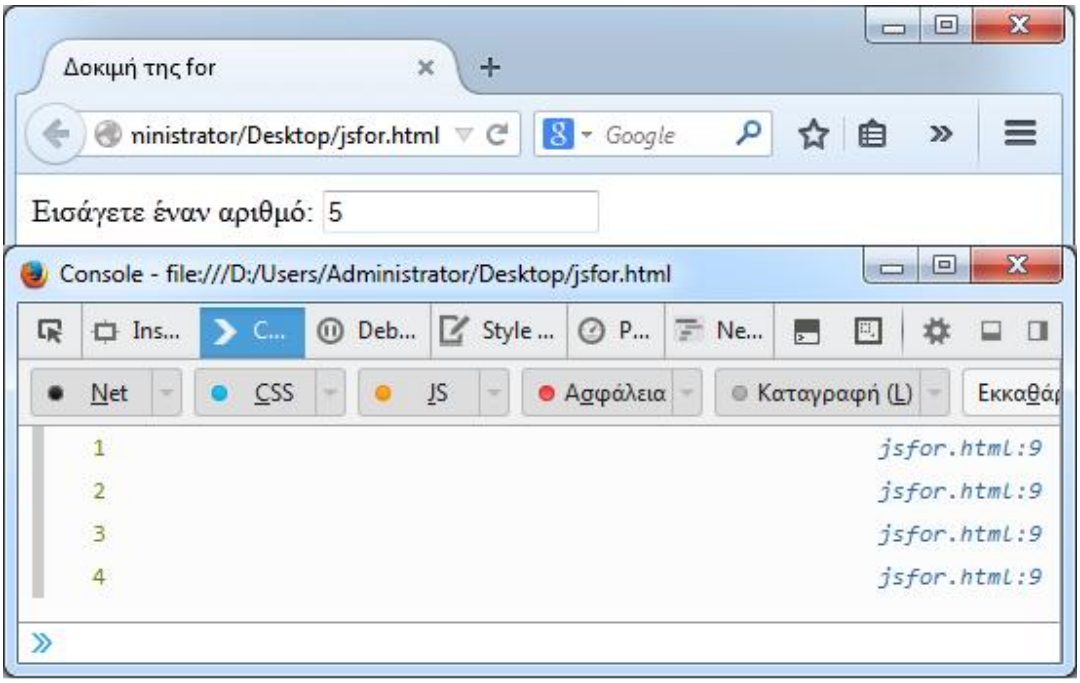
Οι βρόγχοι μπορούν να γίνουν πολύ περίπλοκοι, ειδικά όταν πρόκειται για εμφωλευμένους. Πολλοί βρόγχοι ο ένας μέσα στον άλλο, και παντού μετρητές, δηλώσεις και αλλαγές τιμών των μετρητών. Δηλώνοντας όλα αυτά τα στοιχεία στην αρχή του είναι πιο εύκολο να καταλάβει κανείς πως λειτουργεί ένας βρόγχος, αφού

μέσα σε αυτά περιλαμβάνονται στοιχεία σχετικά με το όνομα του μετρητή , την συνθήκη και την αλλαγή τιμής του μετρητή. Εδώ θα πρέπει να σημειώσουμε ότι ο μετρητής γεννιέται και υπάρχει για όσο τρέχει και υπάρχει ο βρόγχος, μετά το τέλος του βρόγχου η μεταβλητή του μετρητή καταστρέφεται και αποδεσμεύεται ο χώρος στην μνήμη. Έτσι αν προσπαθήσουμε μετά από τον βρόγχο να αποκτήσουμε πρόσβαση στην μεταβλητή του μετρητή δεν θα μπορέσουμε διότι η μεταβλητή δεν θα υπάρχει.

Αρχείο: jsfor.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της for</title>
  <script type='text/javascript'>
   function nums_ins(x){
    for(var i = 1;i<x;i++){
     console.log(i);
    }
   }
  </script>
 </head>
 <body>
  Εισάγετε έναν αριθμό:
  <input type='text' name='temp' id='temp'
onchange='nums_ins(this.value);'>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.59 Αποτέλεσμα εκτέλεσης βρόγχου for

Προσέξτε πως με την χρήση for ο κώδικας μικραίνει σε μέγεθος και έτσι είναι πιο εύκολα διαχειρίσιμο το κομμάτι κώδικα που περιλαμβάνει. Δείτε την διαφορά σε σχέση με την while και την do while:

For	Do-While	While

`for(var i = 1;i<x;i++){`
`  console.log(i);`
`}`
	`var i=1;`
	`do{`
	`  console.log(i);`
	`  i++;`
	`}while(i<=x);`
		`var i = 1;`
		`while(i<=x){`
		`  console.log(i);`
		`  i++;`
		`}`

Πίνακας Β.10 Διαφορά στην χρήση διαφορετικών βρόγχων

Με τον παραπάνω πίνακα (Πίνακας Β.10) βλέπουμε πόσο συνοπτικά το for χρησιμοποιεί την κατάλληλη πληροφορία που χρειάζεται σε σχέση με άλλες εντολές επανάληψης. Το for χρειάζεται μόλις 3 γραμμές , ενώ το do-while και το while χρειάζονται το ελάχιστο 5 γραμμές.

B.6.4 break και continue

Το break και το continue είναι ένας τρόπος να αποκτήσουμε μεγαλύτερο έλεγχο πάνω σε ότι γίνεται μέσα στον κώδικα μας, και κυρίως στους βρόγχους. Σε αυτή την ενότητα θα εξετάσουμε αναλυτικά την λειτουργία καθεμιάς από τις παραπάνω λειτουργίες.

B.6.4.1 break

Το break είναι μια ειδική εντολή η οποία χρησιμοποιείται μέσα σε βρόγχους καθώς και στην εντολή switch έτσι ώστε να τερματιστεί ο βρόγχος ή να βγει η εκτέλεση του προγράμματος έξω από το switch. Στην περίπτωση των βρόγχων χρησιμοποιείται συνήθως μετά από την ικανοποίηση κάποιας συνθήκης. Η σύνταξη του είναι απλή:

```
break;
```

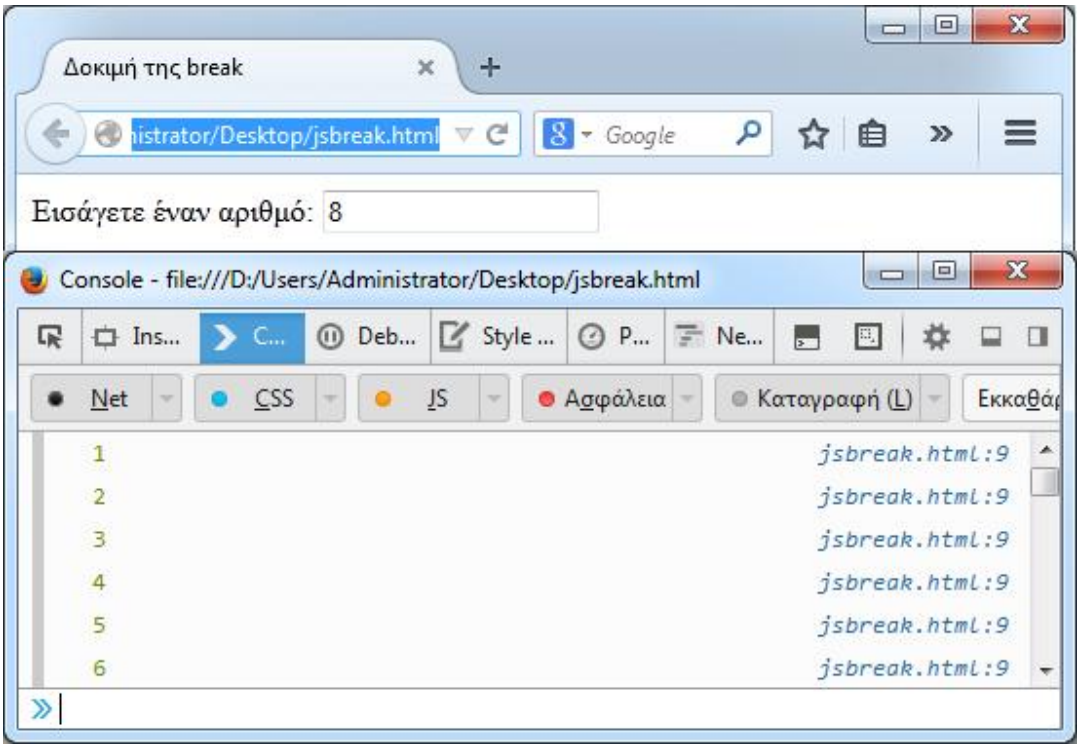
Το συμπεριλαμβάνουμε μέσα σε βρόγχους όπως while, do-while και for:

For	Do-while	While

`for(var i = 1;i<x;i++){`
`  console.log(i);`
`  if(i>x-3){`
`  break;`
`  }`
`}`
	`var i=1;`
	`do{`
	`  console.log(i);`
	`  if(i>x-3){`
	`  break;`
	`  }`
	`  i++;`
	`}while(i<=x);`
		`var i = 1;`
		`while(i<=x){`
		`  console.log(i);`
		`  if(i>x-3){`
		`  break;`
		`  }`
		`  i++;`
		`}`

Πίνακας Β.11 Χρήση break σε βρόγχους

Όλα τα παραπάνω δίνουν ακριβώς το ίδιο αποτέλεσμα:



Εικόνα Β.60 Αποτέλεσμα εκτέλεσης βρόγχου με break

Ο κώδικας που εκτελέστηκε για το παραπάνω παράδειγμα ήταν:

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της break</title>
  <script type='text/javascript'>
   function nums_ins(x){
    for(var i = 1;i<x;i++){
     console.log(i);
     if(i>x-3){
      break;
     }
    }
   }
  </script>
 </head>
 <body>
  Εισάγετε έναν αριθμό:
  <input type='text' name='temp' id='temp'
onchange='nums_ins(this.value);'>
 </body>
</html>
```

Το break βρίσκεται μέσα σε κάποιο είδος ελέγχου συνθήκης, διότι εάν το τοποθετούσαμε απλά μέσα στο for, τότε άμεσα ο βρόγχος θα έσπαγε γιατί στην πρώτη επανάληψη θα έβρισκε την εντολή break. Επομένως η ορθή χρήση του break, γίνεται μέσω κάποιας συνθήκης όταν αναφερόμαστε σε βρόγχους. Όταν αυτοί πληρούνται, πχ ολοκληρώθηκε η αναζήτηση ενός αντικειμένου, τότε είναι καλό να εξοικονομούμε χρόνο σπάζοντας τον βρόγχο.

Για την χρήση του break στην εντολή switch θα ανατρέξετε την αντίστοιχη ενότητα Switch και Case.

B.6.4.2 continue

Η εντολή continue είναι παρόμοια με την break. Μόνο που η continue διακόπτει την τρέχουσα επανάληψη εκείνης της στιγμής και δεν διακόπτει ολοκληρωτικά τον βρόγχο. Η σύνταξη της είναι απλή όπως αυτή της break:

```
continue;
```

Το continue μπορεί να χρησιμοποιηθεί μόνο μέσα στον κώδικα κάποιου βρόγχου. Αν γράψετε καταλάθος ένα continue σε οποιαδήποτε άλλο μέρος του κώδικα το οποίο δεν είναι βρόγχος, τότε θα εμφανιστεί συντακτικό σφάλμα.

Ανάλογα το είδος βρόγχου που χρησιμοποιούμε το continue συμπεριφέρεται ανάλογα:

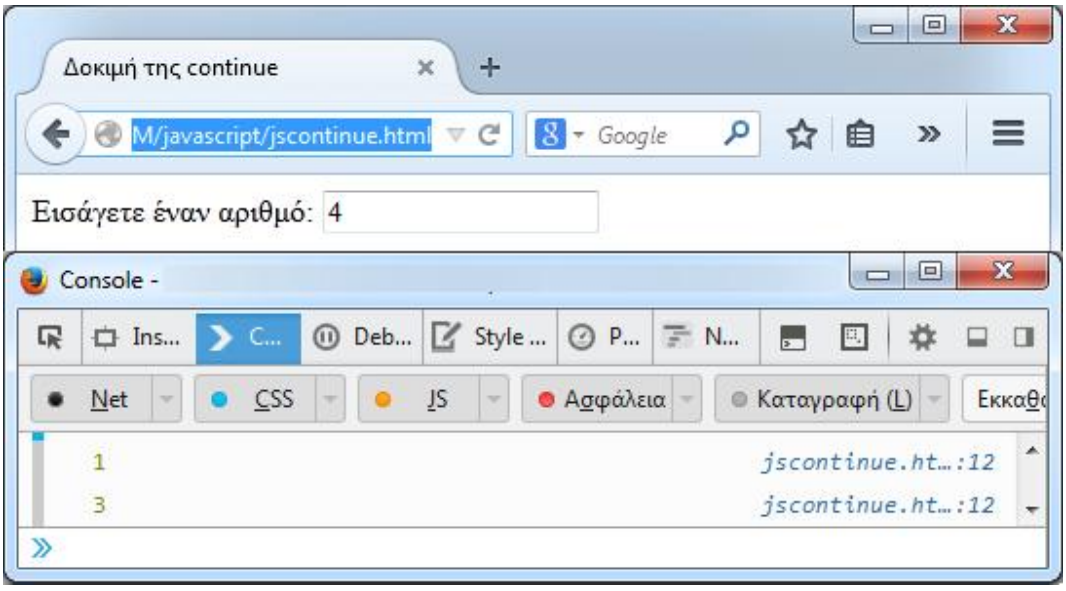
- Στο **while**, όταν εκτελεστεί το continue η συνθήκη του βρόγχου ελέγχεται ξανά και αν ισχύει συνεχίζεται η εκτέλεση του βρόγχου από την αρχή της επόμενης επανάληψης.
- Στο **do-while**, όταν εκτελεστεί το continue η εκτέλεση των υπόλοιπων γραμμών κώδικα μέσα στον βρόγχο παρακάμπτεται και το continue ελέγχει την συνθήκη του do-while που βρίσκεται στο τέλος, αν ισχύει τότε συνεχίζεται η επόμενη επανάληψη από την κορυφή του βρόγχου.
- Στο **for**, όταν εκτελεστεί το continue, αμέσως παρακάμπτονται οι υπόλοιπες γραμμές κώδικα που απομένουν μέχρι το τέλος της τρέχουσας επανάληψης. Αλλάζει ο μετρητής, ελέγχεται η συνθήκη κι αν ισχύει συνεχίζεται η επόμενη επανάληψη.
- Στο **for-in**, όταν εκτελεστεί το continue, άμεσα παρακάμπτονται ολες οι επόμενες εντολές μέχρι το τέλος της τρέχουσας επανάληψης, και η επόμενη επανάληψη ξεκινά στην επόμενη ακριβώς θέση από την τρέχουσα, στο αντικείμενο που υπάρχει μεσα στο for-in.

Σημειώστε την διαφορά συμπεριφοράς στα while, do-while και for, for-in. Στα πρώτα όλα παρακάμπτονται και αμέσως ελέγχεται η συνθήκη, ενώ στα δεύτερα, πρώτα αλλάζει ο μετρητής και μετά ελέγχεται η συνθήκη.

Αρχείο: jscontinue.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <meta charset="utf-8">
  <title>Δοκιμή της continue</title>
  <script type='text/javascript'>
   function nums_ins(x){
    for(var i = 1;i<x;i++){
     if(i==Math.round(x/2)){
      continue;
     }
     console.log(i);
    }
   }

  </script>
 </head>
 <body>
  Εισάγετε έναν αριθμό:
  <input type='text' name='temp' id='temp'
onchange='nums_ins(this.value);'>
 </body>
</html>
```

Η εντολή «Math.round» μας επιστρέφει την στρογγυλοποιημένη μορφή της τιμής «x/2», όπου αυτή είναι πιο κοντά σε κάποιον ακέραιο… πχ για 1,4 θα δώσει 1 ενώ για 1,8 θα δώσει 2.



Εικόνα Β.61 Χρήση της continue μέσα σε βρόγχο

Στο συγκεκριμένο παράδειγμα εισάγοντας 4 το 4/2 προκύπτει 2 επομένως όταν το i είναι ίσο του 2 παρακάμπτει την επανάληψη αυτή κι έτσι δεν τυπώνεται ο αριθμός, και συνεχίζει στην επόμενη, εξού και το 3.

B.7 ΠΙΝΑΚΕΣ

Όταν κάποιος σκεφτεί την λέξη «πίνακας» το μυαλό του θα πάει σε έναν πίνακα με κελιά, όπου μέσα σε αυτά υπάρχει κάποια πληροφορία χωρικά ταξινομημένη, έτσι ώστε να μπορούμε να βγάζουμε κάποια συμπεράσματα, που αν τα εξηγούσαμε με λόγια θα μας έπαιρνε περισσότερο χρόνο και «χώρο». Στην javascript όμως δεν μιλάμε για τέτοιους πίνακες.

Ένας πίνακας είναι μια διατεταγμένη σειρά από τιμές. Κάθε τιμή καλείται στοιχείο του πίνακα και κάθε στοιχείο έχει μία συγκεκριμένη συντεταγμένη που προσδιορίζει την θέση του. Οι πίνακες δεν έχουν συγκεκριμένο τύπο στην javascript, έτσι ένα στοιχείο μπορεί να είναι αλφαριθμητικό, και ένα άλλο να είναι αριθμός, αντικείμενο ή ακόμα και άλλος πίνακας στον ίδιο πίνακα. Με βάση όλα τα παραπάνω χαρακτηριστικά μπορούμε να δημιουργήσουμε σειρές από πολύπλοκα δεδομένα τα οποία θα ομαδοποιούνται όπως εμείς επιθυμούμε.

Το πρώτο στοιχείο του κάθε πίνακα ξεκινά από την συντεταγμένη [0], και λόγω του ότι οι πίνακες αυτοί χρησιμοποιούν 32bit φάσμα συντεταγμένων, η μέγιστη τιμή συντεταγμένης μπορεί είναι η [4294967294], έτσι ένας πίνακας μπορεί να χωρέσει το μέγιστο 4.294.967.295 στοιχεία. Οι πίνακες ωστόσο είναι δυναμικοί , μεγαλώνουν και μικραίνουν ανάλογα με το περιεχόμενο, δεν χρειάζεται δηλαδή εξαρχής να δηλώσουμε το μέγεθος του πίνακα, όλα γίνονται αυτόματα. Επίσης οι πίνακες μπορεί να είναι αραιοί, να υπάρχουν κάποια δεδομένα σε μια σειρά από συντεταγμένες, ύστερα να υπάρχουν κενά και ύστερα ξανά νέα δεδομένα, δεν χρειάζεται η ροή των δεδομένων να είναι συνεχής.

Κάθε πίνακας έχει ένα χαρακτηριστικό «πλάτους» το length. Για έναν πίνακα όπου όλα τα στοιχεία είναι συνεχόμενα από την αρχή του έως το τέλος του το length μας δίνει το σύνολο των στοιχείων που βρίσκονται μέσα σε αυτόν. Για έναν πίνακα όπου

τα στοιχεία είναι διάσπαρτα σε διάφορες θέσεις του πίνακα, τότε το length μας δίνει την μεγαλύτερη συντεταγμένη για την οποία ο πίνακας διαθέτει στοιχείο. Δεν μας δίνει δηλαδή το πλήθος των στοιχείων.

Για να δημιουργήσουμε έναν πίνακα στην javascript αρκεί να χρησιμοποιήσουμε την λέξη «var» ένα όνομα για τον πίνακα μας, και ύστερα να τον αρχικοποιήσουμε στην τιμή «[]» όπου σημαίνει ότι η συγκεκριμένη «μεταβλητή» θα είναι πίνακας:

```
var empty=[]; //Κενός πίνακας
var numbers = [22,10,3,41,11]; //Πίνακας με ακέραια στοιχεία
var misc = [5.2,"keimeno",false]; //Πίνακας με διάφορα στοιχεία
```

Ο πίνακας «empty» είναι ένας άδειος πίνακας. Ο πίνακας «numbers» διαθέτει ως στοιχεία απλούς ακέραιους αριθμούς. Ο πίνακας «misc» είναι ένας πίνακας όπου τα στοιχεία του είναι μικτού περιεχομένου, το πρώτο είναι δεκαδικός, το δεύτερο κείμενο, το τρίτο Boolean. Ο διαχωρισμός των στοιχείων γίνεται με κόμμα «,» η έναρξή και η λήξη του πίνακα σηματοδοτείται μέσω των άγκιστρων.

Μπορούμε εκτός από τους καθορισμένους τύπους περιεχομένων να εισάγουμε στην θέση τους ακόμη και πράξεις, να αφήσουμε κενή την θέση, ή να εισάγουμε αντικείμενα:

```
var space = [1,,3,,,6];  //Πίνακας με κενές θέσεις
var acts = [(1+2),(a+32)]; //Πίνακας με στοιχεία πράξεων
var multi = [[5],[{x:1}]]; //Πίνακας με υπό-πίνακες και αντικείμενα
```

Ο πίνακας «space» είναι ένας πίνακας όπου ανάμεσα στα στοιχεία του υπάρχουν κενά στοιχεία (δηλαδή δεν έχει γραφτεί τίποτα ανάμεσα στα κόμματα). Ο πίνακας «acts» είναι ένας πίνακας όπου μέσα σε αυτόν μπορούμε να ορίζουμε τιμές για τα στοιχεία βασιζόμενοι σε πράξεις με αριθμούς οι άλλες μεταβλητές. Ο πίνακας «multi» είναι ένας πίνακας όπου αρχικά έχουμε άλλους πίνακες μέσα σε αυτόν. Όταν χρησιμοποιούμε τα άγκιστρα μέσα σε στοιχεία πίνακα, αυτόματα έχουμε εμφωλευμένο πίνακα. Επίσης στον δεύτερο εμφωλευμένο πίνακα έχουμε τοποθετήσει ένα «αντικείμενο» με μόνη ιδιότητα το x = 1.

Υπάρχει και εναλλακτικός τρόπος για να δημιουργήσουμε έναν πίνακα αντί να χρησιμοποιήσουμε τα άγκιστρα, με την χρήση της μεθόδου Array();

1. Δημιουργία κενού πίνακα:

```
var a = new Array();
```

Με αυτόν τον τρόπο δημιουργείται ένας πίνακας χωρίς κανένα στοιχείο και είναι ισοδύναμο με τον παραπάνω τρόπο που περιγράψαμε.

2. Δημιουργία πίνακα με μία παράμετρο που καθορίζει το μήκος του:

```
var b = new Array(10);
```

Με αυτόν τον τρόπο δημιουργείται ένας πίνακας καθορισμένου μήκους. Όλα τα στοιχεία είναι κενά μέχρι να τους αποδοθούν τιμές.

3. Δημιουργία πίνακα με στοιχεία διαφόρων τύπων:

```
var c = new Array(10,20,30,"test,test");
```

Με αυτόν τον τρόπο τα στοιχεία που αναγράφονται μέσα στην παρένθεση (αν είναι παραπάνω από ένα) γίνονται στοιχεία του πίνακα στις ανάλογες θέσεις ξεκινώντας από το 0.

Β.7.1 Προσπέλαση πινάκων

Για να μπορέσουμε να αποκτήσουμε πρόσβαση στα στοιχεία ενός πίνακα χρησιμοποιούμε τις συντεταγμένες του εκάστοτε στοιχείου. Συντεταγμένη ενός στοιχείου μέσα σε ένα πίνακα, είναι η θέση στην οποία βρίσκεται μέσα σε αυτόν. Στους πίνακες δεν έχουμε ποτέ αρνητικές θέσεις παρά μόνο θετικές. Έτσι αν γνωρίζουμε ότι ένα στοιχείο βρίσκεται στην θέση 1 μπορούμε να χρησιμοποιήσουμε το περιεχόμενο του καλώντας το όνομα του πίνακα και την θέση-συντεταγμένη μέσα σε άγκιστρα:

```
var pinakas = ["Στοιχείο 1", "Στοιχείο 2"];
var timi = pinakas[0];
```

Στο προηγούμενο παράδειγμα χρησιμοποιούμε την πρώτη τιμή του πίνακα «pinakas» για να την εισάγουμε σε μια νέα μεταβλητή με όνομα «timi». Αυτό που θα εισαχθεί ως τιμή στην μεταβλητή αυτή είναι το αλφαριθμητικό «Στοιχείο 1». Με παρόμοια λογική μπορούμε να αλλάξουμε τιμές στα στοιχεία ενός πίνακα ή ακόμη και να προσθέσουμε νέα στοιχεία:

```
var pinakas = ["Στοιχείο 1", "Στοιχείο 2"];
pinakas[1] = "Stoixeio 2";
i = 2;
pinakas[i] = "Stoixeio 3";
pinakas[i+1] = "Stoixeio 4";
```

Το ιδιαίτερο με τους πίνακες στην javascript είναι πως ανάλογα με το πόσα στοιχεία έχουμε μια δεδομένη στιγμή, αν προσθέσουμε καινούργια αυτόματα το μήκος (length) του πίνακα αυτού θα προσαρμοστεί στο νέο πλήθος.

Β.7.2 Πράξεις με πίνακες

Οι πράξεις μεταξύ συνόλων είναι κάτι που υπάρχει και στα μαθηματικά. Η Javascript φροντίζει αυτές τις πράξεις να τις κάνει στην πραγματικότητα μέσω της δομής των πινάκων που διαθέτει. Παρακάτω θα δούμε μερικές από τις πιο βασικές πράξεις μεταξύ πινάκων.

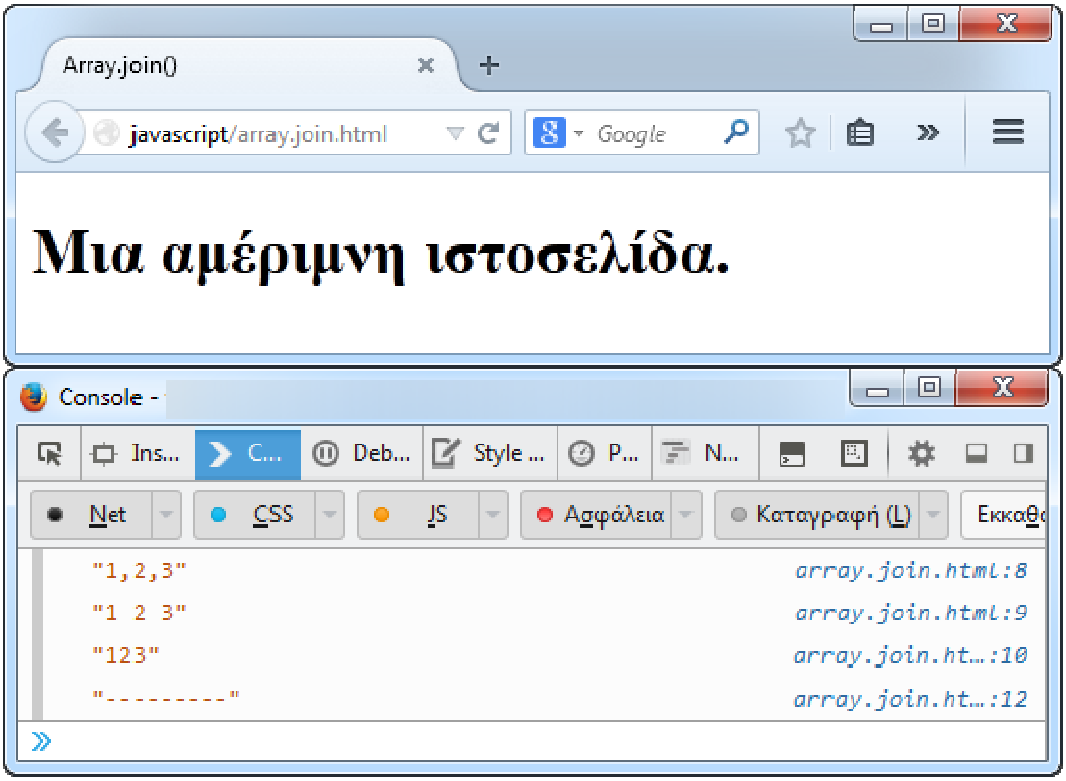
Β.7.2.1 Ένωση: join()

Η «join» είναι μια εντολή μέσω της οποία μπορούμε να ενώσουμε έναν πίνακα με έναν άλλο πίνακα ή έναν πίνακα με ένα συγκεκριμένο όρισμα, και ύστερα να μας επιστραφεί το αποτέλεσμα σε αλφαριθμητικό string. Προσοχή! Δεν αντικαθίστανται το περιεχόμενο του ήδη υπάρχον πίνακα, αλλά επιστρέφεται ως νέο αποτέλεσμα το παραγόμενο. Αυτό σημαίνει ότι όταν καλεστεί η μέθοδος join θα χρησιμοποιήσει τον πίνακα για να φτιάξει το αποτέλεσμα της, και ύστερα θα τον αφήσει όπως ήταν. Ας δούμε μερικά παραδείγματα:

Αρχείο: array.join.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Array.join()</title>
  <meta charset='utf-8'/>
  <script>
   var pinakas = [1,2,3];
   console.log(pinakas.join());
   console.log(pinakas.join(" "));
   console.log(pinakas.join(""));
   var pinakas = new Array(10);
   console.log(pinakas.join('-'));
  </script>
 </head>
 <body>
  <h1>Μια αμέριμνη σελίδα.</h1>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.62 Αποτέλεσμα ένωσης πίνακα με την join

Β.7.2.2 Αντιστροφή: reverse()

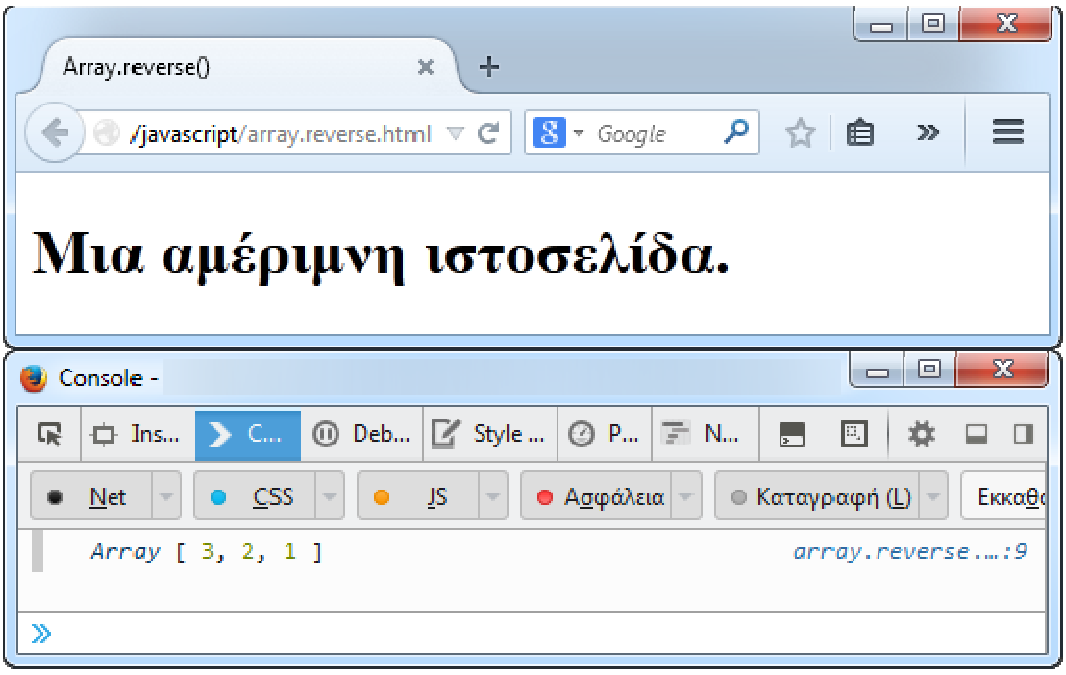
Η μέθοδος της αντιστροφής κάνει αυτό για το οποίο είναι ονομασμένη. Αντιστρέφει την σειρά των στοιχείων ενός πίνακα. Στην συγκεκριμένη περίπτωση δεν δημιουργεί νέο επιστρεφόμενο πίνακα στον οποίο βρίσκεται το αποτέλεσμα, αλλά μετατοπίζει το

περιεχόμενο του πίνακα για τον οποίο την καλούμε. Άρα ένας πίνακας που αρχικά είχε την μορφή «[1,2,3]», μετά την εκτέλεση της μεθόδου θα έχει την μορφή «[3,2,1]».

Αρχείο: array.reverse.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Array.reverse()</title>
  <meta charset='utf-8'/>
  <script>
   var pinakas = [1,2,3];
   pinakas.reverse();
   console.log(pinakas);
  </script>
 </head>
 <body>
  <h1>Μια αμέριμνη ιστοσελίδα.</h1>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.63 Αποτέλεσμα εκτέλεσης αντιστροφής σε πίνακα

Β.7.2.3 Πρόσθεση: concat()

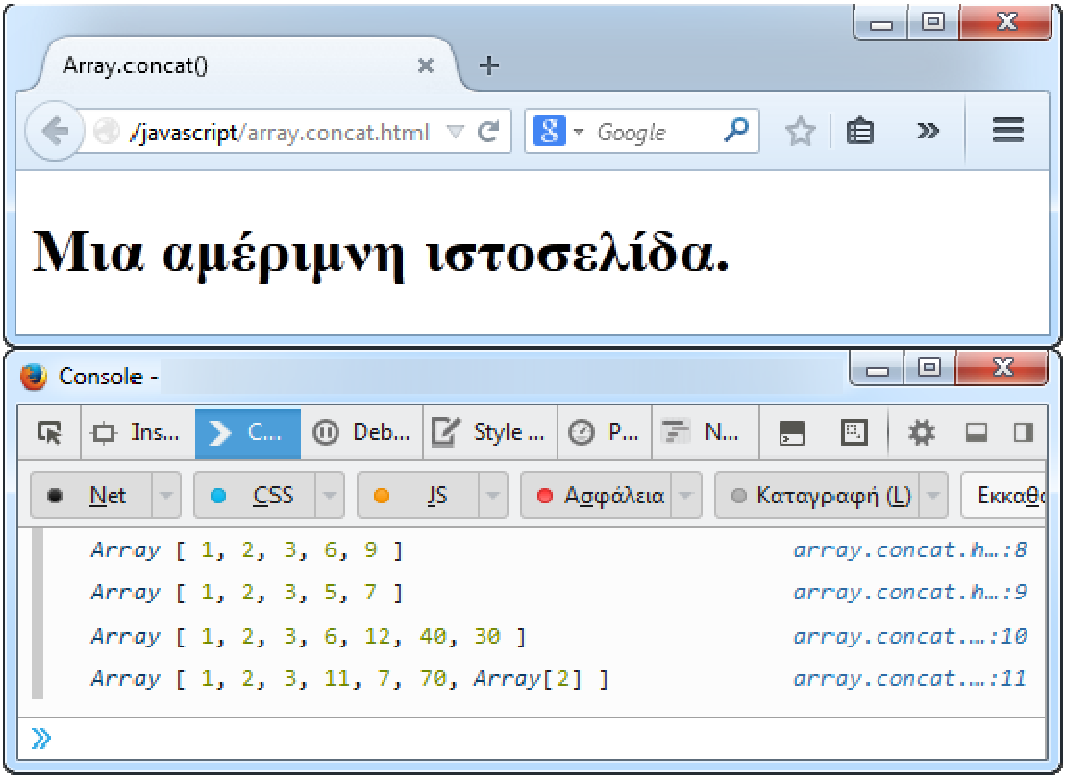
Η πρόσθεση είναι μια μέθοδος με την οποία μπορούμε να προσθέσουμε νέα στοιχεία σε έναν ήδη υπάρχουν πίνακα. Τα στοιχεία αυτά τα εισάγουμε ως ορίσματα μέσα στις παρενθέσεις της concat και αυτά προστίθενται στο τέλος του πίνακα. Η μέθοδος αυτή

ΔΕΝ επηρεάζει τον πίνακα για τον οποίο την καλούμε, παίρνει μόνο τα στοιχεία του, ύστερα προσθέτει τα νέα που θέλουμε και ΕΠΙΣΤΡΕΦΕΙ ΣΕ ΚΑΙΝΟΥΡΓΙΟ ΠΙΝΑΚΑ το αποτέλεσμα. Αν δεν δεσμεύσουμε κάποια μεταβλητή για το αποτέλεσμα τότε αυτό χάνεται. Ας δούμε ένα παράδειγμα:

Αρχείο: array.concat.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Array.concat()</title>
  <meta charset='utf-8'/>
  <script>
   var pinakas = [1,2,3];
   console.log(pinakas.concat(6,9));
   console.log(pinakas.concat([5,7]));
   console.log(pinakas.concat([6,12],[40,30]));
   console.log(pinakas.concat([11,7],[70,[40,30]]));
  </script>
 </head>
 <body>
  <h1>Μια αμέριμνη ιστοσελίδα.</h1>
 </body>
</html>
```

Αποτέλεσμα:



Εικόνα Β.64 Πρόσθεση δυο πινάκων με την concat