ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Διαδικτυακό Μεσιτικό Γραφείο με χρήση των API της GOOGLE

Του φοιτητή

Παπασωτηρίου Σωτήρη

Αρ. Μητρώου: 03/2186

Επιβλέπων καθηγητής

Κεραμόπουλος Ευκλείδης

Θεσσαλονίκη 2013

Περιεχόμενα

Ευρετήριο Εικόνων

Πίνακες

ΠΕΡΙΛΗΨΗ

Αντικείμενο της πτυχιακής εργασίας είναι η ανάπτυξη ενός Διαδικτυακού μεσιτικού γραφείου στο οποίο θα υλοποιούνται λειτουργίες που συναντώνται σε τέτοιου είδους ιστοσελίδες

Για την υλοποίηση της ιστοσελίδας χρησιμοποιήθηκε ανοιχτό λογισμικό όπως το Joomla σαν σύστημα διαχείρισης περιεχομένου (CMS) η MySQL για την διαχείριση της βάσης δεδομένων και το API των Google Maps για την απεικόνιση πληροφοριών σε χάρτη (τοποθεσία, οδηγίες). Βασικές γλώσσες προγραμματισμού που χρησιμοποιούνται είναι οι PHP, HTML και JavaScript

ABSTRACT

The objective of this thesis is the development of an online real estate office with functions that are commonly found in these types of web pages.

Open software was used for the development of the web page. Joomla as a Content Management System (CMS), MySQL for the Data Base management and Google Maps API for providing info on a map (location, directions). PHP, HTML and JavaScript are the main programming languages that are used.

Εισαγωγή

Στο διαδίκτυο σήμερα υπάρχουν εκατομμύρια ιστοσελίδες οι οποίες έχουν την λειτουργικότητα τους και το κοινό τους η κάθε μια. Όλες αυτές οι ιστοσελίδες έχουν δημιουργηθεί χρησιμοποιώντας οι περισσότερες τις ίδιες γλώσσες προγραμματισμού και τις ίδιες τεχνικές. Πλέον έχουμε ξεπεράσει προ πολλού την ανάπτυξη μιας ιστοσελίδας “από το μηδέν”. Υπάρχουν πανίσχυρα εργαλεία, πολλά από τα οποία είναι ανοιχτού κώδικα, και τα οποία μας βοηθούν στο να δημιουργήσουμε καλύτερες ιστοσελίδες, πιο φιλικές για τον χρήστη και πιο εύκολες στην κατασκευή τους και την συντήρηση τους. Στόχος αυτής της εργασίας είναι η κατασκευή ενός σύγχρονου site (http://www.real-estate-thesis.gr/) το οποίο περιέχει λειτουργίες και υπηρεσίες ενός μεσιτικού γραφείου. Στο θεωρητικό κομμάτι της εργασίας θα αναλυθούν στα επόμενα κεφάλαια:

- 1ο Κεφάλαιο: Σχεδίαση της εφαρμογής. Στο πρώτο κεφάλαιο θα αναφερθούν οι απαιτήσεις της εφαρμογής και αναλυτικά οι τεχνολογίες και τα εργαλεία τα οποία χρησιμοποιήθηκαν για την κατασκευή της. Γίνεται αναφορά σε προγραμματιστικές γλώσσες και σε τι μας εξυπηρετεί η κάθε μία. Ακόμα θα αναλυθεί το σύστημα διαχείρισης περιεχομένου Joomla με το οποίο έγινε η ιστοσελίδα, ο τρόπος λειτουργίας του και τα συστατικά από τα οποία αποτελείται.
- 2^{ο} Κεφάλαιο: Google Maps API. Στο δεύτερο κεφάλαιο θα γίνει εκτενής αναφορά στο API της εφαρμογής Google Maps. Αφού δοθούν μερικές γενικές πληροφορίες για τους χάρτες της Google θα εξετάσουμε το προγραμματιστικό κομμάτι και το πώς μπορούμε να εκμεταλλευτούμε τις δυνατότητες που μας προσφέρονται μέσω του API. Κύριο ενδιαφέρον στις λειτουργίες του API έχουν οι υπηρεσίες, αντικείμενα και μέθοδοι που θα χρησιμοποιηθούν στην ιστοσελίδα όπως για παράδειγμα η υπηρεσία Directions.
- 3^{ο} Κεφάλαιο: Παρουσίαση της ιστοσελίδας. Στο τρίτο και τελευταίο κεφάλαιο της εργασίας θα παρουσιαστεί η ιστοσελίδα αναλυτικά. Θα γίνει αναφορά και επεξήγηση κάθε λειτουργίας της και θα αναλυθεί η λειτουργία αποθήκευσης αναζητήσεων χρήστη και το πώς υλοποιήθηκε. Τέλος θα δούμε σε εφαρμογή τις υπηρεσίες Directions και Geocoder του 2^{ου} κεφαλαίου.

Κεφάλαιο 1

Σχεδίαση της Εφαρμογής

Εισαγωγή

Στο κεφάλαιο αυτό θα αναφερθούν τα εργαλεία και οι τεχνολογίες (προγραμματιστικές γλώσσες, CMS, τεχνικές) που χρησιμοποιήθηκαν για την δημιουργία της ιστοσελίδας του μεσιτικού γραφείου. Αναφέρονται λεπτομέρειες για τις γλώσσες προγραμματισμού που χρειάστηκαν στην ανάπτυξη του έργου όπως η PHP και η JavaScript. Το κεφάλαιο επικεντρώνεται στο σύστημα διαχείρισης περιεχομένου Joomla!, με το οποίο έγινε το μεγαλύτερο μέρος της σελίδας, και στον τρόπο λειτουργίας του ώστε να γίνει κατανοητή η όλη διαδικασία κατασκευής ενός σύγχρονου δυναμικού site.

1.1 Στόχοι – Απαιτήσεις της εφαρμογής

Ο κύριος στόχος της πτυχιακής είναι η δημιουργία ενός διαδικτυακού Μεσιτικού γραφείου στο οποίο ο χρήστης θα μπορεί να βρει εύκολα και γρήγορα τις πληροφορίες που αναζητά. Μια τέτοια ιστοσελίδα δεν θα μπορούσε φυσικά να δημιουργηθεί κάνοντας μόνο χρήση της HTML αλλά με έναν συνδυασμό διαφορετικών γλωσσών προγραμματισμού. Γενικότερα τα κύρια γνωρίσματα της ιστοσελίδας θα πρέπει να είναι:

- Εμφάνιση αναλυτικών πληροφοριών για τα ακίνητα όπως τιμή, εμβαδό, διεύθυνση, περιοχή στον χάρτη, τύπος θέρμανσης.
- Επιλογή δημιουργίας λογαριασμού χρήστη για αποστολή σχολίων η και ερωτήσεων για τα ακίνητα.
- Εύκολη και γρήγορη αναζήτηση ακινήτων ανάλογα με τα κριτήρια που ορίζει ο χρήστης
- Αποθήκευση προηγούμενων αναζητήσεων χρήστη και δυνατότητα νέας αναζήτησης με τα ίδια κριτήρια
- Απλή και κομψή εμφάνιση του site, ώστε να μην κουράζει στην χρήση του, με χρήση ενός ενιαίου προτύπου εμφάνισης.

1.2 PHP

Η PHP είναι μια server-side scripting γλώσσα η οποία χρησιμοποιείται για την δημιουργία δυναμικών websites. Το ότι ορίζεται ως server side γλώσσα, σημαίνει ότι τα αρχεία php εκτελούνται στον διακομιστή και όχι στον υπολογιστή του χρήστη. Συνήθως χρησιμοποιείται για δημιουργία HTML αρχείων δυναμικά, τα οποία έπειτα αποστέλλονται στον client, αλλά αποτελεί και ένα είδος συνδέσμου μεταξύ της βάσης δεδομένων του site μας και του client (αποστολή – αποθήκευση - εμφάνιση δεδομένων). Η PHP δημιουργήθηκε το 1994, ήταν έργο του Rasmus Lerdorf και η ονομασία της προερχόταν από το Personal Home Page. Στα επόμενα χρόνια άλλαξε ονομασία σε PHP Hypertext Preprocessor και εξαιτίας του ανοιχτού πηγαίου κώδικα της υπήρξαν πολλές αλλαγές και βελτιώσεις για να φτάσουμε στην σημερινή της μορφή.

Ουσιαστικά ο καθένας μπορεί να αλλάξει τον κώδικα ανάλογα με τις ανάγκες του και να βελτιώσει την γλώσσα για την υπόλοιπη κοινότητα. Σύμφωνα με την επίσημη ιστοσελίδα, 244 εκατομμύρια ιστοσελίδες χρησιμοποιούν την PHP (Ιούνιος 2013)

Ο κώδικας PHP δεν έχει περιορισμούς στην τοποθέτησή του μέσα σε ένα αρχείο HTML (άλλωστε το αρχείο αποκτά την επέκταση .php με την εισαγωγή κώδικα). Μπορεί να εισαχθεί οπουδήποτε θέλουμε είτε για την ανάκτηση τιμών από μια βάση δεδομένων είτε για την αποστολή τους σε αυτήν είτε ακόμα και για την δυναμική δημιουργία ενός ολοκληρωμένου αρχείου HTML το οποίο αργότερα θα σταλεί στον client. Ο κώδικας PHP διαχωρίζεται από τον υπόλοιπο κώδικα με τις σημάνσεις <php? όταν ξεκινά το script μας και ?> όταν τελειώνει.

1.3 JavaScript

Η JavaScript, όπως και η PHP, είναι μια scripting γλώσσα. Η σημαντικότερη διαφορά τους είναι ότι η JavaScript είναι μια client side γλώσσα προγραμματισμού. Αυτό σημαίνει ότι, αντίθετα με την PHP, ο κώδικας εκτελείται στον προσωπικό υπολογιστή του χρήστη. Δημιουργήθηκε από την Netscape Communications το 1995 και η πρώτη της ονομασία ήταν LiveScript. Αργότερα πήρε την σημερινή της ονομασία ίσως και για να εκμεταλλευτεί την εξάπλωση της Java εκείνη την

περίοδο. Έχει σίγουρα επηρεαστεί από γλώσσες όπως η Java και η C++ (εντολές επανάληψης, συνθήκες εκτέλεσης) αλλά σε αντίθεση με τις παραπάνω δεν είναι αυστηρά ορισμένη (μερικά λάθη παραβλέπονται, οι μεταβλητές δεν είναι συγκεκριμένου τύπου).

Η JavaScript έχει τις χρήσεις και τους περιορισμούς της. Ο ρόλος της σε ένα δυναμικό website μπορεί να κατανοηθεί από τις παρακάτω λειτουργίες

- Δίνει την δυνατότητα στον χρήστη να μπορεί να αλληλεπιδρά με τα στοιχεία της σελίδας όπως κουμπιά, φόρμες εισαγωγής κειμένου κ.α.
- Μπορεί να επεξεργαστεί δεδομένα τα οποία εισάγουμε στην σελίδα πριν αυτά αποσταλούν στον διακομιστή
- Μπορεί να στείλει αιτήσεις για λήψη δεδομένων από τον διακομιστή
- Μπορεί να αλλάξει το περιεχόμενο ή την εμφάνιση μιας ιστοσελίδας ανάλογα με τις ενέργειες του χρήστη

Μπορούμε να εισάγουμε κώδικα JavaScript σε ένα HTML αρχείο με παραπάνω από έναν τρόπους. Έχουμε την δυνατότητα να συνθέσουμε μικρά κομμάτια κώδικα για άπλες λειτουργιές, απευθείας μέσα σε tags της HTML (inline script), ή μεγαλύτερα κομμάτια κώδικα για πιο πολύπλοκες με δύο τρόπους: είτε ανάμεσα σε δύο <script> tags στο ίδιο αρχείο HTML είτε σε εξωτερικό αρχείο χρησιμοποιώντας την ιδιότητα *src* του <script> tag.

1.4 Cascading Style Sheets (CSS)

Η PHP και η HTML μπορούν να ορίσουν την δυναμική εμφάνιση των δεδομένων και την δομή με την οποία εμφανίζονται στο παράθυρο του browser μας δεν έχουν όμως την δυνατότητα να επέμβουν στην εμφάνιση της σελίδας. Αυτό γίνεται με την χρήση κανόνων CSS με τους οποίους μπορούμε να αλλάξουμε την εμφάνιση της ιστοσελίδας χωρίς να επηρεαστεί το περιεχόμενο της.

Αρχικά για να μπορέσουμε να αλλάξουμε την εμφάνιση των tags της HTML έπρεπε να τροποποιήσουμε τις ιδιότητες του ίδιου του tag. Υπάρχουν ιδιότητες όπως οι Font, bgcolor, color και άλλες που καθορίζουν την εμφάνιση μόνο του συγκεκριμένου tag στο οποίο τις συμπεριλαμβάνουμε. Το μεγάλο μειονέκτημα

αυτής της μεθόδου είναι η δυσκολία στην εφαρμογή της και το ότι παράγονται ιστοσελίδες οι οποίες αλλάζουν εμφάνιση πολύ δύσκολα (πρέπει να επεξεργαστούμε όλες τις ιδιότητες όλων των tags ξεχωριστά). Για να αντιμετωπιστεί αυτό το πρόβλημα προστέθηκε η έννοια των Cascading style sheets, κανόνων μορφοποίησης που θα αποτελούν ξεχωριστό κομμάτι του κώδικα (συνήθως ξεχωριστό αρχείο). Ειδικότερα τα πλεονεκτήματα του CSS έναντι του inline κώδικα (κώδικας μέσα σε κάθε tag) είναι τα εξής:

- Διαχωρισμός της δομής της σελίδας μας από την μορφοποίηση της
- Οι κανόνες που ορίζουμε μπορούν να χρησιμοποιηθούν σε ολόκληρο το site μας και όχι μόνο σε μια σελίδα
- Δημιουργείται μικρότερος όγκος κώδικα και πιο εύκολα διαχειρίσιμος
- Οι αλλαγές στην εμφάνιση γίνονται πολύ πιο γρήγορα και εύκολα. Αντί να αλλάζουμε τον κώδικα σε κάθε tag αλλάζουμε μόνο τον κώδικα στο CSS αρχείο

Η σύνταξη ενός απλού κανόνα είναι παρόμοια με την παρακάτω:

```
selector {
property: value,
another_property: another_value

}
```

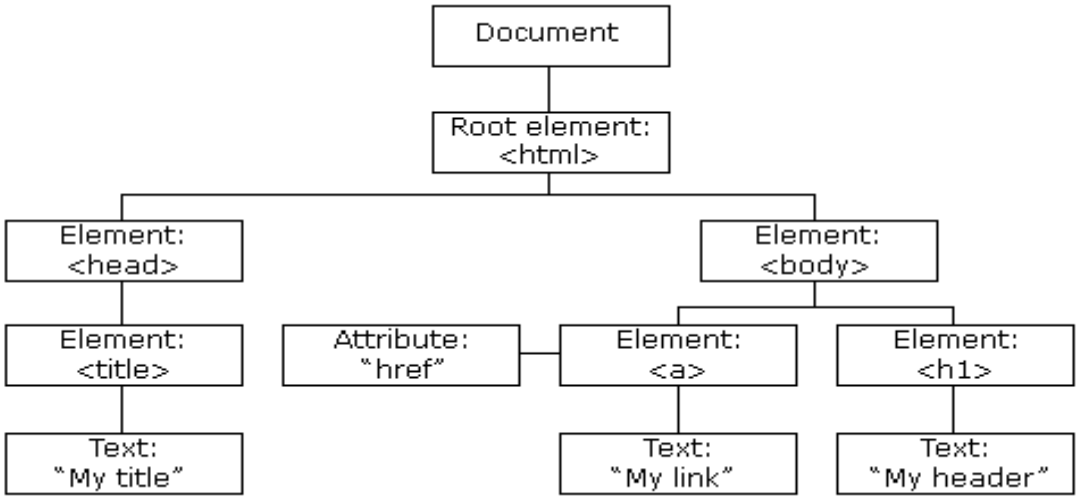
Ο επιλογέας (selector) είναι το στοιχείο πάνω στο οποίο θέλουμε να ισχύουν οι κανόνες μορφοποίησης. Μπορεί να είναι μια ομάδα από HTML tags (πχ όλοι οι υπερσυνδεσμοι της σελίδας), μια ομάδα από στοιχεία με κοινή την ιδιότητα class η συγκεκριμένα στοιχεία τα οποία ορίζουμε μέσω του id τους.

1.5 Document Object Model

Σύμφωνα με την W3Schools το DOM είναι ένα είδος διεπαφής το οποίο μας επιτρέπει να έχουμε πρόσβαση σε όλα τα στοιχεία και το περιεχόμενο ενός HTML

αρχείου χρησιμοποιώντας συνήθως μια γλώσσα προγραμματισμού (όπως η JavaScript)

Η δυνατότητα να μπορούμε να επεξεργαστούμε αυτά τα στοιχεία είναι ένα σημαντικό βήμα για την δημιουργία μιας σελίδας που μπορεί να αλληλεπιδρά με τον χρήστη.



Εικόνα 1-1: Παράδειγμα της δομής του DOM σύμφωνα με το W3Schools

1.6 MySQL

Για να δημιουργήσουμε μια ιστοσελίδα η οποία θα μπορεί να αποθηκεύει δεδομένα του χρήστη η να παρουσιάζει δεδομένα σε αυτόν χρειαζόμαστε μια βάση δεδομένων. Η βάση μας θα βρίσκεται στον διακομιστή και θα πρέπει να υποστηρίζει την πρόσβαση στα δεδομένα της από πολλούς χρήστες ταυτόχρονα. Στην περίπτωση μας, η διαχείριση της βάσης δεδομένων που θα χρησιμοποιηθεί για την αποθήκευση των δεδομένων του μεσιτικού γραφείου (λογαριασμοί χρήστη, πληροφορίες ακινήτων) γίνεται με την MySQL. Πρόκειται για ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMS) ανοιχτού κώδικα, το πιο διαδεδομένο αυτή την στιγμή παγκοσμίως. Μερικοί από τους λόγους που βοήθησαν στην εξάπλωση της MySQL είναι οι εξής:

- Η MySQL διατίθεται δωρεάν
- Είναι αρκετά σταθερή που σημαίνει ότι δεν είναι επιρρεπής σε αποτυχίες και λάθη ακόμα και με μεγάλο φόρτο δεδομένων

- Δεν έχει μεγάλες απαιτήσεις συστήματος για να λειτουργήσει
- Είναι γρήγορη και εύκολη στην χρήση της
- Προσφέρει όλα τα εργαλεία και επιλογές που χρειαζόμαστε για ανάπτυξη μιας διαδικτυακής εφαρμογής

1.7 WAMP server

Για την δημιουργία της ιστοσελίδας και τις οποιεσδήποτε δοκιμές χρειάστηκε να γίνουν, χρησιμοποιήθηκε λογισμικό το οποίο εξομοιώνει έναν Apache server στον υπολογιστή μας. Τα αρχικά WAMP αντιστοιχούν στα τέσσερα κύρια συστατικά για να μπορεί να λειτουργήσει η ιστοσελίδα μας. Windows, Apache, MySQL και PHP. Με την εγκατάσταση του λογισμικού στον προσωπικό μας υπολογιστή το site λειτουργεί τοπικά, όπως θα λειτουργούσε και αν ήταν αποθηκευμένο σε έναν κανονικό διακομιστή.

1.8 Τι είναι ένα CMS

Τα αρχικά CMS προέρχονται από το Content Management System που μεταφράζεται σε Σύστημα Διαχείρισης Περιεχομένου. Ως CMS ορίζουμε ένα οποιοδήποτε σύστημα που υποστηρίζει την συλλογή, επεξεργασία και δημοσιοποίηση περιεχομένου (δεδομένων) μέσω μιας διεπαφής. Τα δεδομένα αυτά μπορεί να είναι κείμενα, φωτογραφίες, βίντεο κ.α.

Ο όρος CMS χρησιμοποιείται γενικότερα για τέτοιου είδους συστήματα λογισμικού. Όταν όμως ένα τέτοιο σύστημα χρησιμοποιείται για την διαχείριση περιεχομένου στο διαδίκτυο ονομάζεται Web CMS (WCMS). Μας δίνει την δυνατότητα να κάνουμε όσα μας προσφέρει ένα CMS, σε μια ιστοσελίδα.

Μερικές από τις επιλογές/λειτουργίες που μας προσφέρει ένα WCMS είναι οι εξής:

- Διαχείριση των δεδομένων χωρίς αναγκαστικά να έχουμε γνώση HTML, PHP κ.α.
- Δυνατότητες ορισμού μιας ενιαίας εμφάνισης του website μας με την χρήση προτύπων εμφάνισης (templates) και την επεξεργασία αυτής.

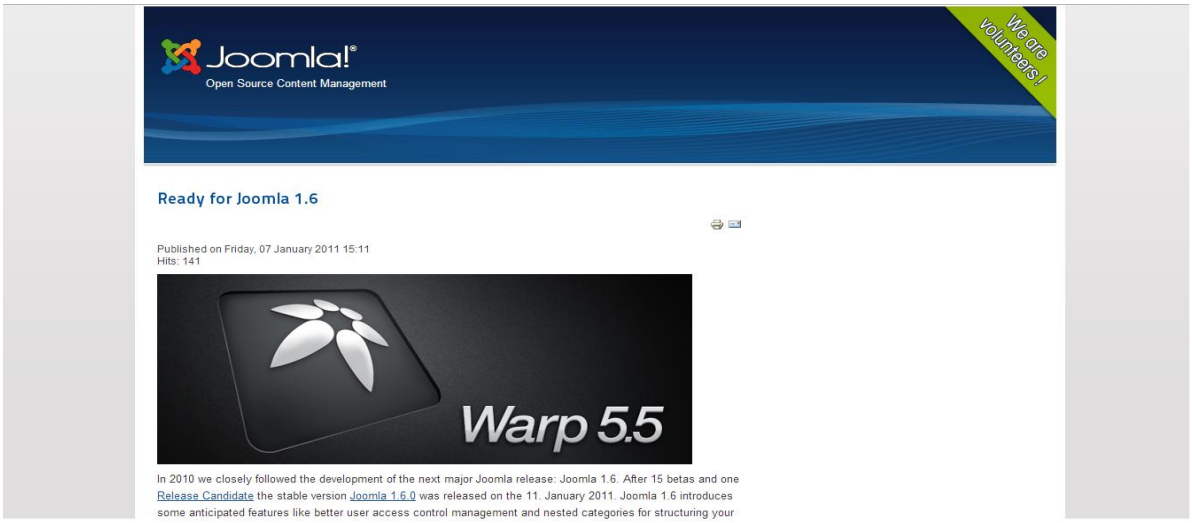
- Διαχείριση των χρηστών του site μας με επιλογές για διαφορετικές ομάδες χρηστών με διαφορετικά δικαιώματα πρόσβασης κ.α.
- Επέκταση του site μας χρησιμοποιώντας πρόσθετα συστατικά κώδικα (components, plug-ins) ανάλογα με τις ανάγκες μας

1.9 Το Joomla! CMS

Το Joomla! Είναι ένα από τα πιο γνωστά και διαδεδομένα CMS στον κόσμο αυτή τη στιγμή (μαζί με Drupal και WordPress). Πρόκειται για ένα fork του CMS Mambo. Με τον όρο Fork αναφερόμαστε σε ένα project όπου παίρνουμε ένα νόμιμο αντίγραφο λογισμικού και ξεκινάμε μια ανεξάρτητη επέκταση πάνω σε αυτό (ένα πολύ καλό παράδειγμα είναι οι δεκάδες διαφορετικές εκδόσεις του Linux). Είναι γραμμένο σε PHP και χρησιμοποιεί την MySQL ως Βάση Δεδομένων. Η ιστορία του ξεκινάει στα μέσα του 2005 όταν η Miro International, η εταιρία η οποία κατείχε τα δικαιώματα του Mambo, αποφάσισε την ανεξάρτητη ανάπτυξη του. Οι όροι τους οποίους επέβαλλε η εταιρία για την ανάπτυξη του νέου συστήματος προκάλεσαν αντιδράσεις στην ομάδα ανάπτυξης η οποία δημιούργησε το site OpenSourceMatters.org για να ενημερώσει την κοινότητα των χρηστών του Mambo για την κατάσταση του project. Όλα αυτά οδήγησαν στην απόφαση για δημιουργία ενός open source συστήματος το οποίο θα ακολουθούσε πιστά την νοοτροπία και τις αρχές του ανοιχτού κώδικα. Το όνομα Joomla προέρχεται από την γλώσσα Σουαχιλι και σημαίνει “όλοι μαζί” η “σύνολο”. Στις 16 Σεπτεμβρίου 2005 λανσάρεται η πρώτη επίσημη έκδοση του Joomla! Πρόκειται για ένα βελτιωμένο, σε μερικά σημεία, αντίγραφο της έκδοσης 4.5.2.3 του Mambo (διόρθωση bugs και κενών ασφαλείας). Από το 2005 μέχρι σήμερα έχουν κυκλοφορήσει αρκετές εκδόσεις του, με σημαντικότερες τις 1.5, 2.5 και την πιο πρόσφατη 3.0

1.9.1 Το Front-end του Joomla!

Δυο είναι τα βασικά τμήματα του Joomla!, το back-end και το front-end. Το front-end είναι το τμήμα της εφαρμογής μας το οποίο βλέπει ο χρήστης, η ιστοσελίδα μας.

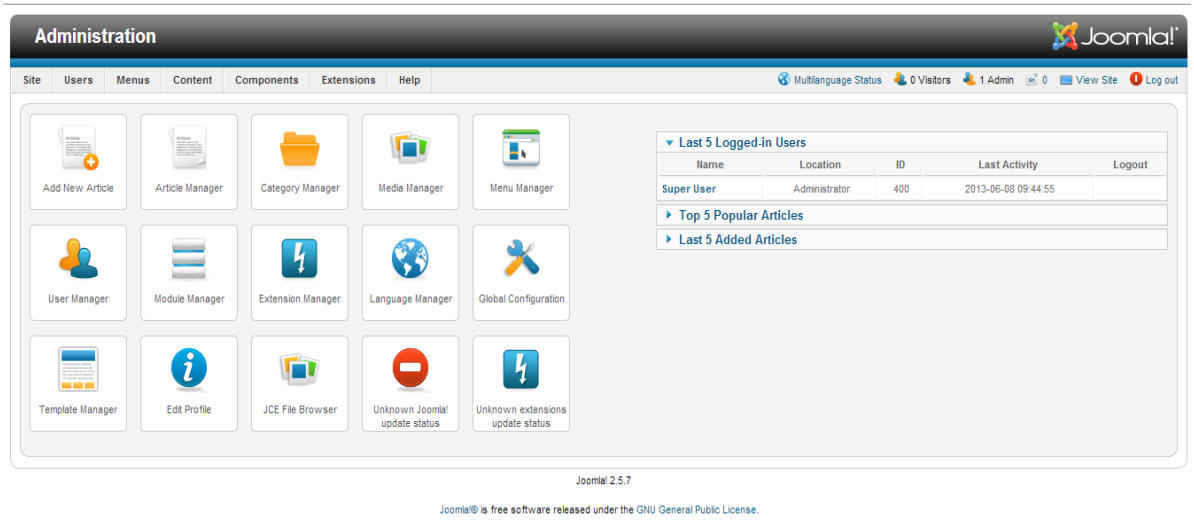


Εικόνα 1-2: Το Front-end του Joomla αμέσως μετά την εγκατάστασή του

Στην παραπάνω εικόνα βλέπουμε την μορφή της ιστοσελίδας μετά την εγκατάσταση του Joomla!. Χρησιμοποιεί ένα από τα απλά, προεγκατεστημένα πρότυπα εμφάνισης.

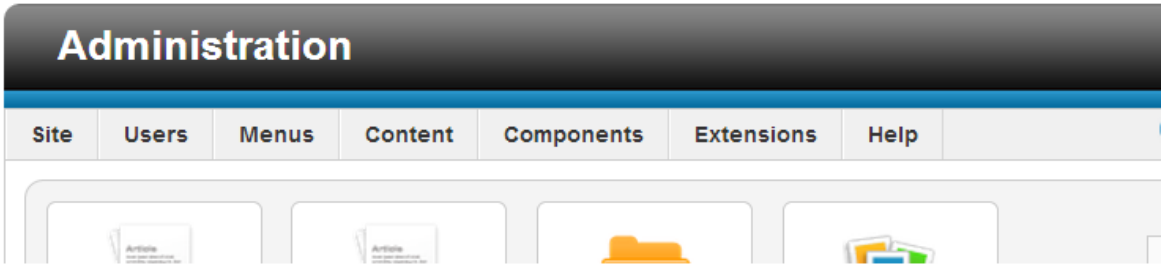
1.9.2 Το Back-end του Joomla!

Το Back-end είναι ουσιαστικά το administration site της σελίδας μας. Πρόκειται για ένα εξίσου σημαντικό κομμάτι του Joomla, ίσως και πιο σημαντικό του Front-end αφού μας παρέχει τα εργαλεία για να επεξεργαστούμε την μορφή και το περιεχόμενο του site μας. Για να αποκτήσουμε πρόσβαση στο Back-end προσθέτουμε στο url του site μας το κομμάτι /administrator. Θα μεταφερθούμε στην οθόνη εισαγωγής διαχειριστή οπού αφού εισάγουμε το όνομα χρήστη και τον κωδικό μας θα εμφανιστούν οι παρακάτω επιλογές.



Εικόνα 1-3: Η αρχική σελίδα του back-end του Joomla!

Το κύριο κομμάτι στα αριστερά αποτελείται από επιλογές που συνήθως χρησιμοποιούνται συχνά στην επεξεργασία ενός site από το Joomla. Στο δεξί τμήμα της σελίδας υπάρχουν δεδομένα για τους τελευταίους χρήστες που συνδέθηκαν στο site, τα πιο πρόσφατα άρθρα (η γενικά περιεχόμενο) που προστεθήκαν και ποια από τα άρθρα είναι τα πιο δημοφιλή. Στο πάνω τμήμα έχουμε το μενού για να πλοηγηθούμε σε όλες τις λειτουργιές και τα εργαλεία του Back-end



Εικόνα 1-4: Το μενού επιλογών του Back-end

Οι επιλογές που έχουμε στο μενού είναι οι εξής:

- **Site:** επιλογές για ολόκληρο το site, διαχείριση προφίλ administrator, πληροφορίες για τον διακομιστή και τις εκδόσεις των PHP και MySQL
- **Users:** επιλογές για τους χρήστες της ιστοσελίδας όπως δικαιώματα πρόσβασης, ομάδες χρηστών, αποστολή mail σε χρήστες κ.α.
- **Menus:** από εδώ μπορούμε να ορίσουμε πόσα και ποια θα είναι τα μενού που θα έχει η ιστοσελίδα μας καθώς και περισσότερες λεπτομέρειες όπως η ιεραρχία τους, η σειρά τους, το που θα μας ανακατευθύνει η κάθε επιλογή και πολλά άλλα
- **Content:** επιλογές για εισαγωγή περιεχομένου. Το περιεχόμενο είναι συνήθως σε μορφή άρθρου στο οποίο μπορούν να εισαχθούν εικόνες, βίντεο, υπερσύνδεσμοι κ.α. Συνήθως χρησιμοποιείται ένας WYSIWYG (What you see is what you get) editor για την συμπλήρωση του περιεχομένου
- **Components:** εδώ βρίσκουμε τις επιλογές για συγκεκριμένα συστατικά κώδικα που έχουμε εισάγει στο site μας
- **Extensions:** η επιλογή extensions μας δίνει την δυνατότητα να διαχειριστούμε όλα τα συστατικά κώδικα και τις ξεχωριστές εφαρμογές που έχουμε εισάγει στο site καθώς και τα πρότυπα εμφάνισης η τα πακέτα γλωσσών του site. Επίσης μας παρέχει τον τρόπο για να εισάγουμε νέα συστατικά-εφαρμογές στο site
- **Help:** σύνδεσμοι για αρχεία βοήθειας του Joomla! (επίσημο site, forums, documentation)

1.9.3 Επεκτάσεις του Joomla! (Extensions)

Οι επεκτάσεις του Joomla! είναι ολοκληρωμένες εφαρμογές ή πρότυπα εμφάνισης που προστίθενται στο site ανάλογα με τις ανάγκες μας.

Χωρίζονται στις εξής κατηγορίες:

- Components
- Modules
- Plugins
- Templates
- Languages

1.9.4 Components

Τα components αποτελούν επεκτάσεις οι οποίες επηρεάζουν ολόκληρο το περιεχόμενο του site μας. Πρόκειται για ολοκληρωμένες εφαρμογές οι οποίες προσφέρουν λύσεις ανάλογα με τις ανάγκες μας. Προσθέτουν νέες λειτουργιές και μπορούν να επεξεργαστούν την βάση δεδομένων του site προσθέτοντας πίνακες ή αποθηκεύοντας/ανακτώντας δεδομένα από αυτήν. Συνήθως ένα component αποτελεί το κύριο μέρος της ιστοσελίδας μας. Ειδικά, στην ιστοσελίδα του Μεσιτικού γραφείου έχει χρησιμοποιηθεί το component Joomla Estate Agency το οποίο διαχειρίζεται ακίνητα και τα παρουσιάζει στον χρήστη. Τα περισσότερα components αποτελούνται από δυο κομμάτια (όμοια δομή με το ίδιο το Joomla!) το κομμάτι του site, που βλέπει ο χρήστης και το κομμάτι του διαχειριστή της σελίδας.

1.9.5 Modules

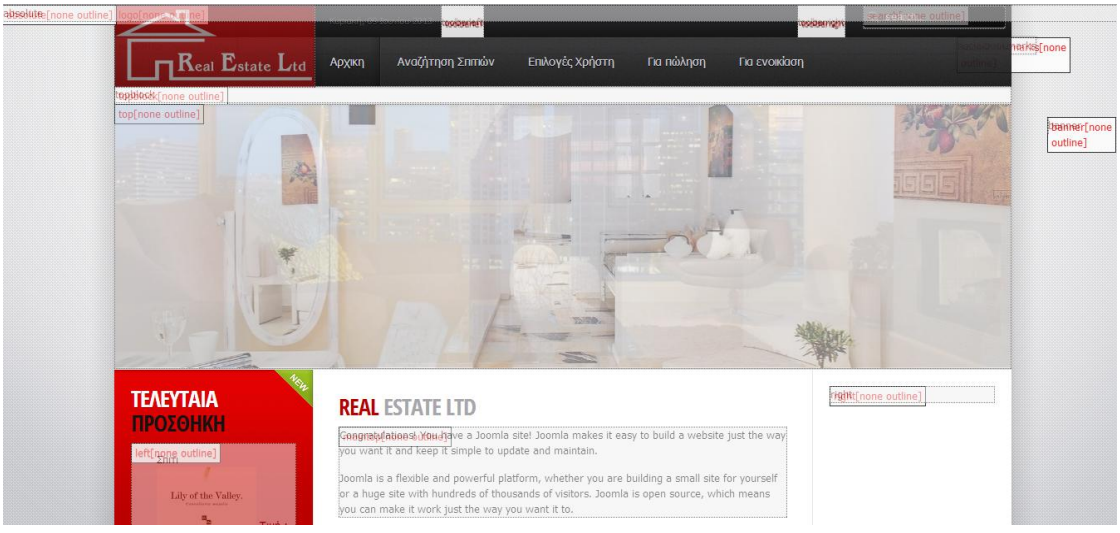
Τα modules είναι πρόσθετες μικρές λειτουργιές του site που βελτιώνουν την εμπειρία του χρήστη ή επεκτείνουν την χρηστικότητα ενός component. Αποτελούν δευτερεύον κομμάτι της σελίδας και πολλές φορές δεν διαχειρίζονται δεδομένα από και προς την βάση αν και αυτό δεν ισχύει πάντα. Για παράδειγμα η εναλλαγή εικόνων στην αρχική σελίδα του site, είναι ένα module στο οποίο απλά εισάγουμε εικόνες και επιλεγούμε το πώς θα εμφανίζονται-εναλλάσσονται. Δεν έχει κάποια σημαντική χρήση, αλλά βελτιώνει αρκετά την εμφάνιση του site.

1.9.6 Plugins

Τα Plugins αποτελούν επεκτάσεις που ο χρήστης ίσως δεν παρατηρήσει ποτέ γιατί δουλεύουν στο παρασκήνιο. Μια αρκετά καλή περιγραφή τους είναι ότι αποτελούν event handlers για άλλες λειτουργιές του site. Για παράδειγμα σε περίπτωση που κάποιος χρήστης θέλει να αναρτήσει ένα κείμενο σε ένα forum, με το που πατάει το κουμπί αποστολή, ενεργοποιείται ένα Plugin για τον έλεγχο και το φιλτράρισμα λέξεων που δεν επιτρέπονται.

1.9.7 Templates

Τα templates αποτελούν ουσιαστικά, την εμφάνιση του site μας. Ορίζουν το μεγαλύτερο κομμάτι των CSS κανόνων που διέπουν το site μας. Τα περισσότερα templates έχουν αρκετές επιλογές για την εμφάνιση του περιεχομένου, την τοποθέτηση του στο site ή και για την εμφάνιση διαφόρων γενικών λειτουργιών (διάφορα είδη μενού κ.α.). Μάλιστα, έχουμε την επιλογή να δούμε τα “κομμάτια” που συνθέτουν το site σύμφωνα με το template απλά εισάγοντας τους χαρακτήρες ***?tp=1*** στο τέλος του url



Εικόνα 1-3: Οι θέσεις εισαγωγής περιεχομένου στο template

1.9.8 Language

Μια πολύ βασική επέκταση είναι οι γλώσσες του site. Αυτές οι επεκτάσεις μπορούν να εισαχθούν είτε σε επίπεδο κάποιας άλλης επέκτασης (για παράδειγμα πακέτο ελληνικής γλώσσας για ένα component) είτε σε επίπεδο site. Τα πακέτα γλωσσών μπορούν να επηρεάσουν το Front αλλά και το Back-end του site μας.

Επίλογος

Η δημιουργία ενός ολοκληρωμένου site έχει γίνει πλέον πολύ πιο εύκολη σε σχέση με το παρελθόν. Χρησιμοποιώντας ένα οποιοδήποτε CMS μπορούμε πλέον να δημιουργήσουμε καλαίσθητα site υψηλού επίπεδου που προσφέρουν πληθώρα υπηρεσιών στους χρήστες. Ειδικότερα για τα CMS που δεν είναι εμπορικές εφαρμογές αλλά ανοιχτά προς το κοινό, μπορούμε να δημιουργήσουμε τις δικές μας εφαρμογές-επεκτάσεις που να καλύπτουν τις ανάγκες μας και να τις ενσωματώσουμε στο Σύστημα Διαχείρισης. Σε αυτό το κεφάλαιο είδαμε όλες τις τεχνικές, γλώσσες και εφαρμογές που χρησιμοποιήθηκαν για την δημιουργία μιας ιστοσελίδας μεσιτικού γραφείου. Ένα πολύ μεγάλο κομμάτι όμως στην αγορά ακινήτων είναι η τοποθεσία τους. Ο τρόπος με τον οποίο μπορούμε να εμφανίσουμε στο site μας πληροφορίες για τοποθεσία των ακινήτων με έναν πιο διαδραστικό τρόπο από μια απλή διεύθυνση είναι το κύριο θέμα του επόμενου κεφαλαίου οπού αναλύεται η εφαρμογή Google Maps

Κεφάλαιο 2

Google maps API

Εισαγωγή

Το Google Maps είναι η υπηρεσία χαρτογράφησης της εταιρίας Google. Περιέχει χάρτες για όλο τον πλανήτη και μας δίνει την δυνατότητα να πλοηγηθούμε σε αυτόν μόνο με την χρήση του ποντικιού μας. Είναι μια από τις πιο γνωστές web based εφαρμογές στο διαδίκτυο και ουσιαστικά αποτελείται από έναν συνδυασμό HTML, JavaScript και CSS. Λειτούργει με AJAX κλήσεις στον διακομιστή δεδομένων που σημαίνει ότι η εμπειρία του χρήστη δεν διακόπτεται από συνεχείς ανανεώσεις της σελίδας για νέα δεδομένα αφού αυτές οι ενέργειες γίνονται στο παρασκήνιο. Η Google μας δίνει την δυνατότητα να εισάγουμε την εφαρμογή της σε μια δική μας ιστοσελίδα και να επεξεργαστούμε τις πληροφορίες που προβάλλουμε στους χάρτες, ακόμα και να αλλάξουμε τον χάρτη ανάλογα με τις ανάγκες μας. Ο τρόπος με τον οποίο γίνονται όλα αυτά είναι το θέμα αυτού του κεφαλαίου.

2.1 Ιστορία του Google Maps

Το Google Maps ξεκίνησε σαν μια απλή εφαρμογή χαρτογράφησης γραμμένη σε C++ από τους αδελφούς Lars και Jens Rasmussen ιδρυτές της εταιρίας Where2 στο Σύδνεϋ. Η αρχική ιδέα ήταν η εφαρμογή να χρειάζεται εγκατάσταση στον υπολογιστή του χρήστη αλλά η ιδέα αυτή απορρίφθηκε για χάρη μιας εφαρμογής που θα λειτουργεί αποκλειστικά σε browser.

Τον Οκτώβριο του 2004 η Where2 εξαγοράστηκε από την Google και το Google Maps πήρε την μορφή με την οποία το γνωρίζουμε σήμερα δηλαδή μια web-based εφαρμογή χαρτογράφησης. Η κύρια διαφορά του ήταν η διαδραστικότητα που προσφέρει στον χρήστη. Ενώ εκείνη την περίοδο υπήρχαν και άλλες εφαρμογές για απεικόνιση χαρτών καμία δεν έδινε την δυνατότητα στον χρήστη να περιηγηθεί στον χάρτη με την χρήση του ποντικιού.

Τον Ιούνιο του 2005 η Google δημοσιοποίησε την πρώτη έκδοση του API της για το Google maps δίνοντας την δυνατότητα στο κοινό να μπορεί να τροποποιήσει τους χάρτες (τοποθεσίες, επίπεδο zoom και πολλά άλλα)

Η πρώτη ιστοσελίδα που έκανε εκτενή χρήση των χαρτών ήταν η www.Housingmaps.com η οποία συνδύασε το Google maps με πληροφορίες για

ακίνητα από την σελίδα craigslist.org (μάλιστα το Housingmaps προσέφερε αυτή την υπηρεσία πριν ακόμα δημοσιοποιηθεί το API, αφού υπήρχαν τρόποι για να παρακαμφθεί η ασφάλεια της εφαρμογής και να χρησιμοποιηθούν οι συναρτήσεις και οι ιδιότητες της).

Στα επόμενα χρόνια μέχρι σήμερα η εφαρμογή βελτιώθηκε σε μεγάλο βαθμό προσφέροντας περισσότερες υπηρεσίες και σε περισσότερες συσκευές (κινητά τηλέφωνα, tablets)

Μερικές από τις σημαντικότερες στιγμές στην ανάπτυξη της εφαρμογής μέχρι σήμερα είναι οι εξής:

Οκτώβριος 2005: το Google Maps κυκλοφορεί για κινητά τηλέφωνα

Απρίλιος 2006: Το API αναβαθμίζεται στην 2η έκδοση του. η εφαρμογή χρησιμοποιεί δορυφορικές εικόνες από το Google Earth

Οκτώβριος 2007: η υπηρεσία Google Transit (πληροφορίες για δρομολόγια μέσων μαζικής μεταφοράς) ενσωματώνεται στους χάρτες

Ιούλιος 2008: Προστίθεται η δυνατότητα λήψης οδηγιών για πεζούς

Οκτώβριος 2008: Το Google maps κυκλοφορεί για κινητά με λειτουργικό σύστημα Android

Μάιος 2009: Ανακοινώνεται από την Google η beta έκδοση του API V3 των Maps

Μάρτιος 2010: Προστίθεται η δυνατότητα λήψης οδηγιών για ποδήλατα

Μάιος 2010 Η 3η έκδοση του API κυκλοφορεί επίσημα

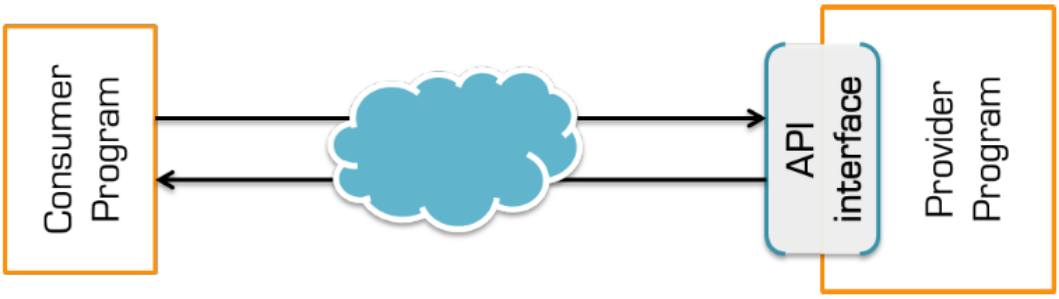
2.2 Τι είναι το API

Πριν αναφερθούμε αναλυτικότερα στο API του Google Maps θα πρέπει να εξηγήσουμε τι ακριβώς είναι ένα API.

Το API προέρχεται από το Application Programming Interface (διεπαφή προγραμματισμού εφαρμογής) και ουσιαστικά είναι ένα πρωτόκολλο σύμφωνα με το οποίο μια εφαρμογή/υπηρεσία μπορεί να δώσει την δυνατότητα σε άλλες

εφαρμογές να της στέλνουν αιτήσεις ή και ακόμα να ανταλλάζουν δεδομένα μεταξύ τους. Ο τρόπος με τον οποίο η πρωταρχική εφαρμογή χειρίζεται αυτές τις αιτήσεις και τα δεδομένα, παραμένει κρυμμένος από τις εφαρμογές που επικοινωνούν μαζί της και τους χρήστες τους.

Το API αποτελείται από ένα σύνολο κανόνων/εργαλείων για την επικοινωνία με την εφαρμογή. Κανόνες οι οποίοι έχουν την μορφή συναρτήσεων, δομών δεδομένων, αντικειμένων και μεταβλητών/ιδιοτήτων.



Εικόνα 2-1 Μια απλή αναπαράσταση της επικοινωνίας μιας εφαρμογής με ένα API

2.3 Mashups με χρήση API

Τα τελευταία χρόνια υπάρχει η τάση στο διαδίκτυο να χρησιμοποιούνται εφαρμογές σε συνδυασμό με βάσεις δεδομένων η και άλλες εφαρμογές, με σκοπό τον εμπλουτισμό των πληροφοριών που παρέχονται στους χρήστες. Ένας τέτοιου είδους συνδυασμός λέγεται mashup. Ουσιαστικά η ιστοσελίδα ενός μεσιτικού γραφείου αποτελεί ένα απλό mashup αφού χρησιμοποιεί το API του Google Maps σε συνδυασμό με μια βάση δεδομένων για την απεικόνιση των ακινήτων πάνω στον χάρτη.

2.4 Google Maps API

Το Google maps API πρόκειται για μια διεπαφή γραμμένη σε JavaScript η οποία μας δίνει την δυνατότητα μέσω κλήσεων AJAX να χειριστούμε τον χάρτη (επίπεδο zoom, συντεταγμένες) άλλα και τα δεδομένα που υπάρχουν σε αυτόν (οδηγίες, δρομολόγια) ανάλογα με τις ανάγκες μας.

Αυτή την χρονική περίοδο χρησιμοποιείται η 3η έκδοση του API η οποία κυκλοφόρησε επίσημα το 2010. Προσφέρει έναν μεγάλο αριθμό συναρτήσεων και αντικειμένων τα οποία μας δίνουν την δυνατότητα να εισάγουμε τους χάρτες σε οποιαδήποτε ιστοσελίδα θέλουμε. Πρόκειται μάλιστα για το πιο δημοφιλές API αφού κατέχει το 38% των mashups που υπάρχουν αυτή τη στιγμή στο διαδίκτυο (2460 mashups)

2.5 Εισαγωγή του API στην σελίδα μας

Το API είναι ουσιαστικά ένα JavaScript αρχείο το οποίο βρίσκεται στους διακομιστείς της Google. Για να μπορέσουμε να χρησιμοποιήσουμε τις δυνατότητες που μας προσφέρει πρέπει πρώτα να ορίσουμε το μονοπάτι για αυτό το αρχείο στον HTML κώδικα μας.

Αυτό γίνεται με την ετικέτα <script> της HTML και ορίζοντας σαν πηγή (source) ένα url το οποίο δείχνει την διαδρομή στην οποία βρίσκεται ουσιαστικά η υλοποίηση της εφαρμογής.

Η πλήρης μορφή της ετικέτας <script> σύμφωνα με την Google είναι η παρακάτω:

```
<script type="text/javascript"
src="https://maps.googleapis.com/maps/api/js?key=YOUR_API_KEY&sens
or=SET_TO_TRUE_OR_FALSE">
 </script>
```

Παρατηρούμε δυο πράγματα στο παραπάνω κομμάτι κώδικα και συγκεκριμένα στο url της ιδιότητας source: για να μπορέσουμε να επικοινωνήσουμε με το API χρειαζόμαστε έναν κλειδάριθμο (key) ο οποίος μπαίνει στην θέση του πεδίου YOUR_API_KEY. Στην σημερινή έκδοση του API ο κλειδάριθμος είναι αναγκαίος μόνο σε σελίδες με μεγάλη κίνηση (περισσότερες από 25000 κλήσεις δεδομένων την ημέρα). Ο σκοπός του είναι η παρακολούθηση και καταγραφή των δεδομένων που χρησιμοποιούνται από την σελίδα μας.

Εκτός από το API key παρατηρούμε και το πεδίο sensor το οποίο αναφέρεται σε αισθητήρες εντοπισμού μέσω δορυφόρου και μπορεί να πάρει δυο τιμές, true η false. Συνήθως χρησιμοποιείται η τιμή false εκτός και αν η εφαρμογή μας είναι προσβάσιμη από συσκευές οι οποίες έχουν αισθητήρα gps (για παράδειγμα κινητά τηλέφωνα)

Υπάρχουν ακόμα τρία προαιρετικά πεδία στο url τα οποία έχουν να κάνουν με την γλώσσα, την τοποθεσία του χάρτη και την έκδοση του API που θέλουμε να χρησιμοποιήσουμε. Όπως και στο παραπάνω παράδειγμα όλα τα πεδία χωρίζονται με τον χαρακτήρα &.

```
<script type="text/JavaScript
src="https://maps.googleapis.com/maps/api/js?v=3.exp&key=YOUR_API_
KEY&sensor=SET_TO_TRUE_OR_FALSE&region=GR&language=GR">
</script>
```

Το πεδίο v μπορεί να πάρει τρεις τιμές και ορίζει την ακριβή έκδοση του API που χρησιμοποιούμε (πάντα μέσα στα πλαίσια της 3^{ης} έκδοσης). Για παράδειγμα:

- V=3 η v=3.12 η πιο πρόσφατη επίσημη έκδοση για το κοινό
- V=3.exp πειραματική έκδοση όπου οι εκάστοτε αλλαγές δεν έχουν δοκιμαστεί πλήρως
- V=3.10 (για παράδειγμα) οποιαδήποτε προηγούμενη αριθμημένη έκδοση

Τα πεδία language και region έχουν να κάνουν με την γλώσσα με την οποία εμφανίζονται οι ονομασίες στον χάρτη και με την περιοχή η οποία μας ενδιαφέρει αντίστοιχα. Ειδικότερα, το πεδίο region επιλύει το πρόβλημα περιοχών (πόλεων/οδών) που έχουν το ίδιο όνομα, εμφανίζοντας μας την περιοχή ανάλογα με την επιλογή που έχουμε κάνει στο url.

Αφού έχουμε εισάγει αυτές τις 2-3 γραμμές κώδικα στην σελίδα μας είμαστε έτοιμοι να χρησιμοποιήσουμε το API και να χειριστούμε/τροποποιήσουμε όλα τα δεδομένα χαρτογράφησης που προσφέρει το Google Maps

2.6 Ο χάρτης

Πριν προχωρήσουμε στο προγραμματιστικό κομμάτι της εισαγωγής ενός χάρτη στο site μας είναι καλό να αναφέρουμε μερικές πληροφορίες για το πώς η Google έχει χαρτογραφήσει τον πλανήτη.

Ο χάρτης είναι σχεδιασμένος χρησιμοποιώντας την *μερκατορική* προβολή. Ο κόσμος χαρτογραφείται σαν κύλινδρος και όχι σαν σφαίρα με αποτέλεσμα περιοχές κοντά στους πόλους να μην αναπαριστώνται στο σωστό μέγεθος αλλά σε μεγαλύτερο.

Υπάρχουν 20 διαφορετικά πεδία zoom με το 20 να είναι το μεγαλύτερο και το 1 να απεικονίζει όλο τον πλανήτη. Ακόμα, μας δίνεται η δυνατότητα να έχουμε 4 διαφορετικά ειδή χαρτών τα όποια είναι:

- Roadmap: οδικός χάρτης
- Satellite: δορυφορικές εικόνες (ίδιες με του Google Earth)
- Hybrid: Συνδυασμός των 2 παραπάνω
- Terrain: Γεωμορφολογικός χάρτης

Αρχικά η Google έκανε χρήση εικόνων σε ανάλυση 256x256 pixels για την χαρτογράφηση. Οι εικόνες αυτές λαμβάνονταν από την συσκευή του χρήστη δυναμικά, ανάλογα με το σημείο στο οποίο είχε πλοηγηθεί στον χάρτη και το επίπεδο zoom που χρησιμοποιούσε. Αυτή η τεχνική δημιούργησε μια τεράστια βάση δεδομένων αφού η Google για να καλύψει όλο τον πλανήτη σε 20 διαφορετικά επίπεδα zoom χρειαζόταν περίπου 360 δισεκατομμύρια εικόνες. Η εφαρμογή πλέον δεν λειτούργει με αυτόν τον τρόπο για όλες τις συσκευές άλλα χρησιμοποιώντας vector graphics (για κινητά τηλέφωνα και tablets), μια τεχνική που είναι παρόμοια με τον τρόπο με τον οποίο ένας browser σχεδιάζει μια σελίδα από ένα html αρχείο.

2.7 Εισαγωγή του χάρτη στην σελίδα μας

Η βάση για οποιαδήποτε σελίδα στο διαδίκτυο είναι η γλώσσα σήμανσης HTML (Hyper-Text Markup Language) η οποία ορίζει πως θα εμφανιστούν τα δεδομένα στον browser του χρήστη. Ο κώδικας που θα γράψουμε για την εμφάνιση του χάρτη θα πρέπει να συμπεριλαμβάνεται σε ένα αρχείο HTML το οποίο θα μοιάζει με το παρακάτω:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
      <head>
            <meta content="text/html; charset=utf-8" http-
            equiv="Content-Type" />
            <title>Untitled 1</title>
      </head>
      <body>
            <div id=map-canvas> </div>
      </body>
</html>
```

Ανάμεσα από τα tags <head> θα μπει ο κώδικας για τον χάρτη γραμμένος σε JavaScript καθώς και το μονοπάτι (path) για το API όπως αναφέραμε στο προηγούμενο υποκεφάλαιο (σε 2 διαφορετικά script tags). Στο body παρατηρούμε ένα div με το id “map-canvas” μέσα στο οποίο θα εμφανιστεί ο χάρτης μας. Αφού έχουμε εισάγει το path, πλέον μπορούμε να δημιουργήσουμε τον πρώτο μας χάρτη.

Ένα απλό παράδειγμα είναι το παρακάτω:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
```

```
<meta content="text/html; charset=utf-8" http-equiv="Content-Type"
/>
<title>Map Example</title>
<script
src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"
></script>
<script type="text/javascript">
      function initialize() {
        var mapOptions = {
          center: new google.maps.LatLng(-34.397, 150.644),
          zoom: 8,
          mapTypeId: google.maps.MapTypeId.ROADMAP
        };
        var map = new
google.maps.Map(document.getElementById("map-canvas"),
            mapOptions);
      }
      google.maps.event.addDomListener(window, 'load',
initialize);
</script>
</head>
<body>
            <div id=map-canvas> </div>
</body>
</html>
```

Εισάγοντας αυτό το κομμάτι κώδικα στο HTML αρχείο μας θα έχουμε μια σελίδα η οποία θα εμφανίζει έναν απλό χάρτη. Το script μας αποτελείται από δυο βασικά κομμάτια: μια συνάρτηση με το όνομα initialize() μαζί με την υλοποίηση της και έναν event listener ο οποίος θα ενεργοποιήσει την συνάρτηση.
Η συνάρτηση initialize() περιέχει:

Το αντικείμενο map, το οποίο ουσιαστικά αποτελεί τον χάρτη μας και ένα αντικείμενο με όνομα mapOptions το οποίο περιέχει τις επιλογές για την προβολή του χάρτη.

2.7.1 Η κλάση MAP

Η κλάση Map περιέχει μεθόδους, ιδιότητες events και έναν δομητή που δημιουργεί αντικείμενα τύπου Map. Ένα τέτοιο αντικείμενο μπορεί να πάρει δυο ορίσματα:

- Το πρώτο όρισμα είναι οι επιλογές του χάρτη (MapOptions)
- το δεύτερο είναι το σημείο στο οποίο θα εμφανιστεί ο χάρτης στην σελίδα μας (div με id map-canvas).

Η μορφή ενός στιγμιότυπου του αντικείμενου map είναι η εξής:

Δομητής	Περιγραφή
Map(mapDiv:node, opts?MapOptions)	Το αντικείμενο map το οποίο δέχεται δυο ορίσματα: ένα αντικείμενο MapOptions και ένα αντικείμενο του HTML έγγραφου όπως ορίζεται από το DOM

Πίνακας 2-1: Το αντικείμενο Map

2.7.2 Το αντικείμενο MapOptions

Το αντικείμενο mapOptions ουσιαστικά είναι υπεύθυνο για ολόκληρη την εμφάνιση του χάρτη μας. Περιέχει πολλές ιδιότητες οι οποίες καθορίζουν σημαντικές λεπτομέρειες του χάρτη όπως οι αρχικές συντεταγμένες (το κέντρο του χάρτη), ο τύπος του, το αν θα εμφανίζονται τα κουμπιά πλοήγησης/έλεγχου, αν θα μπορεί ο χρήστης να πλοηγηθεί με το ποντίκι ή όχι και πολλά άλλα. Στον παρακάτω πινάκα αναφέρονται οι ιδιότητες που μπορεί να έχει ένα στιγμιότυπο του αντικειμένου με μια σύντομη περιγραφή για την χρησιμότητα τους

Ιδιότητα	Τύπος	Περιγραφή
backgroundColor	string	χρώμα για το φόντο του χάρτη. Το

Ιδιότητα	Τύπος	Περιγραφή
		βλέπουμε μέχρι να εμφανιστούν τα εικονίδια του χάρτη
center	Latlng	το κέντρο του χάρτη
disableDefaultUI	boolean	απενεργοποίηση του προκαθορισμένου user interface
disableDoubleClickZoom	boolean	απενεργοποίηση της λειτουργιάς ζουμ με διπλό κλικ
draggable	boolean	αν είναι false δεν μπορούμε να μετακινηθούμε στον χάρτη
draggableCursor	string	εικονίδιο του κέρσορα όταν ο χάρτης είναι draggable
draggingCursor	string	εικονίδιο του κέρσορα όταν μετακινούμαστε στον χάρτη
heading	number	επιλογή γωνίας διαδρομής για πλοήγηση σε σφαιρικό χάρτη
keyboardShortcuts	boolean	αν είναι false δεν λειτουργούν οι συντόμευσης πληκτρολογίου
mapMaker	boolean	θέτουμε ως true αν χρησιμοποιούνται mapmaker εικονίδια στον χάρτη
mapTypeControl	boolean	ενεργοποίηση/απενεργοποίηση του mapTypeControl κουμπιού
mapTypeControlOptions	MapTypeControlOptions	επιλογές για την εμφάνιση του mapTypeControl
mapTypeID	MapTypeID	ο τύπος του χάρτη
maxZoom	number	μέγιστο επίπεδο zoom
minZoom	number	ελάχιστο επίπεδο zoom
noClear	boolean	αν είναι true δεν χάνονται τα περιεχόμενα του map-div
overviewMapControl	boolean	ενεργοποίηση/απενεργοποίηση του overviewMapControl
overviewMapControlOptions	overviewMapControlOptions	επιλογές για την εμφάνιση του overviewMapControl
panControl	boolean	ενεργοποίηση/απενεργοποίηση του panControl κουμπιού
panControlOptions	PanControlOptions	επιλογές για την εμφάνιση του panControl
rotateControl	boolean	ενεργοποίηση/απενεργοποίηση

Ιδιότητα	Τύπος	Περιγραφή
		του rotateControl κουμπιού
rotateControlOptions	RotateControlOptions	επιλογές για την εμφάνιση του rotateControl
scaleControl	boolean	ενεργοποίηση/απενεργοποίηση του scaleControl κουμπιού
scaleControlOptions	ScaleControlOptions	επιλογές για την εμφάνιση του scaleControl
scrollWheel	boolean	ενεργοποίηση/απενεργοποίηση του zoom με scrolling
streetView	StreetViewPanorama	ορισμός ενός πανοράματος StreetView
streetViewControl	boolean	ενεργοποίηση/απενεργοποίηση του streetView κουμπιού
streetViewControlOptions	StreetViewControlOptions	επιλογές για την εμφάνιση του streetViewControl
styles	πίνακας <MapTypeStyle>	πίνακας με στοιχεία εμφάνισης που μπορούν να εφαρμοστούν σε χάρτες
tilt	number	γωνία θέασης του χάρτη
zoom	number	επίπεδο zoom
zoomControl	boolean	ενεργοποίηση/απενεργοποίηση της μπάρας zoom
zoomControlOptions	ZoomControlOptions	επιλογές για την εμφάνιση της μπάρας zoom

Πίνακας 2-2: Ιδιότητες του αντικειμένου MapOptions

Από τις ιδιότητες που αναφέρθηκαν στον πίνακα τρεις πρέπει να υπάρχουν υποχρεωτικά σε κάθε αντικείμενο. Αυτές είναι οι εξής:

- Center: το σημείο το οποίο αποτελεί το κέντρο του χάρτη. χρησιμοποιεί ένα αντικείμενο τύπου latlng για να ορίσει τις συντεταγμένες
- Zoom: το αρχικό επίπεδο μεγέθυνσης όταν εμφανιστεί ο χάρτης και
- MapTypeId: ο τύπος του χάρτη με 4 επιλογής. Roadmap, satellite hybrid και terrain

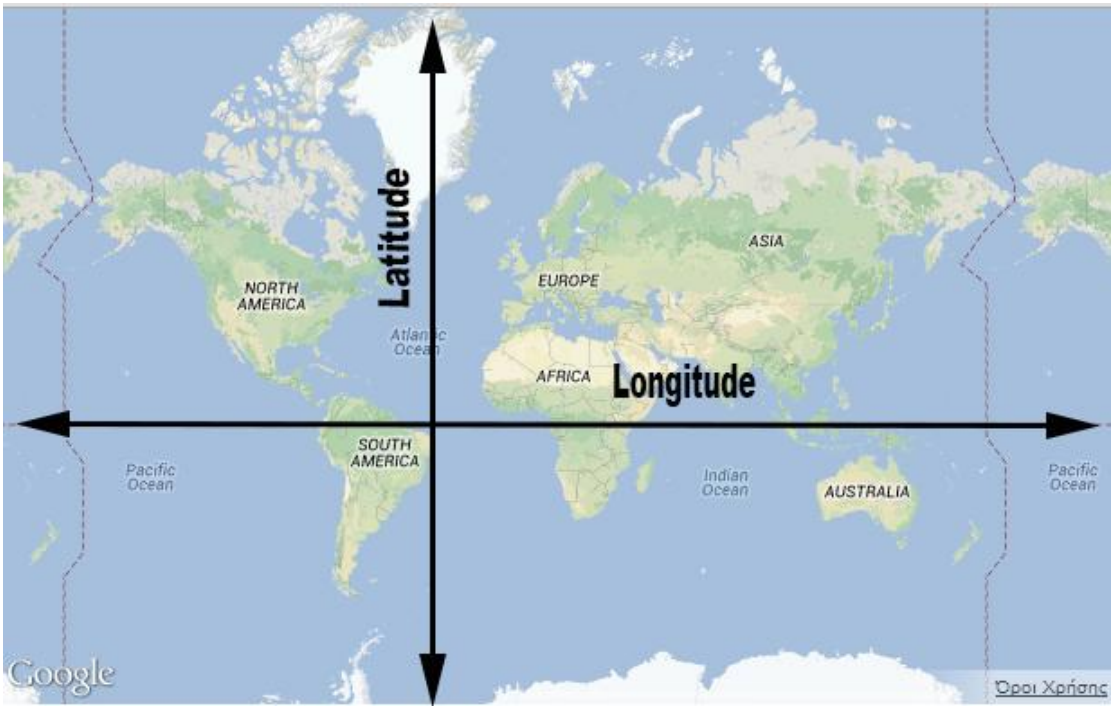
2.7.3 Η κλάση LatLng

Η κλάση latlng όπως και η Map περιέχει μεθόδους, ιδιότητες, events και έναν δομητή που δημιουργεί στιγμιότυπα της. Χρησιμοποιούμε το αντικείμενο Latlng για να ορίζουμε συντεταγμένες διαφόρων σημείων στον χάρτη όπως για παράδειγμα το κέντρο του χάρτη η την τοποθεσία ενός marker.

Δομητής	Περιγραφή
LatLng(lat:number, lng:number, noWrap?:boolean)	Δημιουργία ενός Latlng αντικειμένου το οποίο αναπαριστά ένα σημείο στον χάρτη. Οι μεταβλητές lat και lng είναι οι συντεταγμένες. Το πεδίο noWrap μας επιτρέπει να ορίσουμε συντεταγμένες εκτός των ορίων

Πίνακας 2-3: Το αντικείμενο LatLng

Η μεταβλητή latitude αναπαριστά το γεωγραφικό πλάτος και παίρνει τιμές από -90 έως 90. Η μεταβλητή longitude αναπαριστά το γεωγραφικό μήκος και παίρνει τιμές από -180 έως 180. Στην παρακάτω εικόνα φαίνεται ο παγκόσμιος χάρτης χωρισμένος σε τέσσερα κομμάτια ανάλογα με τις τιμές των συντεταγμένων. Ο ισημερινός χωρίζει τον χάρτη στο σημείο 0 για το γεωγραφικό πλάτος και ο μεσημβρινός του Greenwich στο σημείο 0 για το γεωγραφικό μήκος. Πάνω από τον ισημερινό οι τιμές για το γεωγραφικό πλάτος είναι πάντα θετικές (αρνητικές από κάτω) και αντίστοιχα δεξιά από τον μεσημβρινό του Greenwich (αριστερά του Greenwich οι τιμές είναι αρνητικές)



Εικόνα 2-2: Οι συντεταγμένες σε έναν χάρτη του Google Maps

2.8 Markers

Το API μας δίνει την δυνατότητα να “μαρκάρουμε” σημεία του χάρτη τα οποία παρουσιάζουν κάποιο ενδιαφέρον για τους χρήστες της εφαρμογής μας. Αυτά τα σημεία λέγονται markers και είναι ουσιαστικά γραφικά στοιχεία τα οποία μπαίνουν πάνω στον χάρτη μας.

Εικόνα 2-3: Ένα απλό Marker το οποίο βρίσκεται στο ΤΕΙ Θεσσαλονίκης

Το marker που βλέπουμε στην παραπάνω εικόνα ορίζεται από ένα αντικείμενο με το ίδιο όνομα.

Δομητής	Περιγραφή
Marker(opts?: MarkerOptions)	Ο δομητής δημιουργεί ένα αντικείμενο τύπου marker για την εισαγωγή του στον χάρτη.

Πίνακας 2-4: Το αντικείμενο Marker

Όπως παρατηρούμε ο δομητής παίρνει σαν όρισμα ένα αντικείμενο τύπου markerOptions. Με τις ιδιότητες αυτού του αντικείμενου μπορούμε να επεξεργαστούμε όλες τις πληροφορίες που θα παρέχει το marker καθώς και την συμπεριφορά του σε διάφορα events. Ο παρακάτω πίνακας αναφέρει όλες τις ιδιότητες του MarkerOptions:

Ιδιότητα	Τύπος	Περιγραφή
anchorPoint	Point	απόσταση του marker από το InfoWindow του αν υπάρχει
animation	Animation	εφέ κίνησης του marker
clickable	boolean	αν είναι true το marker δέχεται mouse events
cursor	string	εικονίδιο κέρσορα όταν είναι πάνω από το marker
draggable	boolean	ενεργοποίηση/απενεργοποίηση για την μετακίνηση του marker
flat	boolean	ενεργοποίηση/απενεργοποίηση σκιάς εικονιδίου
icon	string | icon | symbol	το εικονίδιο του marker
map	map | streetViewPanorama	ο χάρτης στον οποίο εισάγεται το marker
optimized	boolean	επιλογή για την σχεδίαση των markers σαν ένα ενιαίο στατικό συστατικό του χάρτη
position	LatLng	η θέση του marker
raiseOnDrag	boolean	ενεργοποίηση/απενεργοποίηση του animation ανύψωσης του marker κατά την μετακίνηση του
crossOnDrag	boolean	ενεργοποίηση/απενεργοποίηση του

Ιδιότητα	Τύπος	Περιγραφή
		σταυρού κάτω από το marker όταν μετακινείται
shadow	string | icon | symbol	το εικονίδιο της σκιάς του marker
shape	MarkerShape	το σχήμα ενός marker
title	string	τίτλος που εμφανίζεται όταν ο κέρσορας είναι πάνω από το marker
visible	boolean	ορατό/μη ορατό marker
zIndex	number	σειρά προτεραιότητας εμφάνισης σε περίπτωση που έχουμε πολλά markers σε μια περιοχή

Πίνακας 2-5: Ιδιότητες του αντικειμένου MarkerOptions

Η μοναδική υποχρεωτική ιδιότητα του MarkerOptions είναι η τοποθεσία του marker (position). Μπορεί να φαίνεται οξύμωρο ότι, κατά την δημιουργία ενός marker δεν χρειάζεται να οριστεί ο χάρτης πάνω στον οποίο θα προβληθεί, άλλα αυτό ουσιαστικά μας δίνει την ελευθερία να δημιουργούμε markers και να τα εμφανίζουμε οποιαδήποτε στιγμή θέλουμε και σε όποιο χάρτη θέλουμε.

Ένα απλό παράδειγμα κώδικα για την δημιουργία ενός marker στον χάρτη είναι το παρακάτω:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta content="text/html; charset=utf-8" http-equiv="Content-Type"
/>
<title>Marker Example</title>
<script
src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"
></script>
<script type="text/javascript">
      function initialize() {
```

```
        var myLatlng = new google.maps.LatLng(-
25.363882,131.044922);
        var mapOptions = {
            zoom: 4,
            center: myLatlng,
            mapTypeId: google.maps.MapTypeId.ROADMAP
        }
        var map = new
google.maps.Map(document.getElementById("map-canvas"),
mapOptions);

        var marker = new google.maps.Marker({
            position: myLatlng,
            map: map,
            title:"Hello World!"
        });

    }
    google.maps.event.addDomListener(window, 'load',
initialize);
</script>
</head>
<body>
            <div id=map-canvas> </div>
</body>
</html>
```

Η δημιουργία του marker όπως βλέπουμε είναι ξεχωριστό κομμάτι από την δημιουργία του συγκεκριμένου χάρτη. Απλά στις επιλογές του marker ορίζουμε τον χάρτη μας για την εμφάνιση του marker πάνω του. Η ιδιότητα position ορίζει τις συντεταγμένες στις οποίες θα εμφανιστεί (είναι οι ίδιες με το κέντρο του χάρτη). Τέλος ορίζουμε ένα κείμενο το όποιο εμφανίζεται όταν έχουμε τον κέρσορα μας πάνω από το marker.

2.8.1 Το αντικείμενο InfoWindow

Σε μερικές περιπτώσεις θέλουμε να παρουσιάσουμε περισσότερες πληροφορίες από ένα απλό σημείο στον χάρτη. Την λύση μας την δίνει το αντικείμενο InfoWindow το οποίο εμφανίζει ένα νέο παράθυρο πληροφοριών πάνω στον χάρτη.

Δομητής	Περιγραφή
InfoWindow(opts?:InfoWindowOptions)	ο δομητής δημιουργεί ένα αντικείμενο τύπου InfoWindow το οποίο παίρνει σαν παραμέτρους ένα αντικείμενο τύπου InfoWindowOptions

Πίνακας 2-6: Το αντικείμενο InfoWindow

Οι επιλογές σύμφωνα με τις οποίες εμφανίζεται ένα InfoWindow δίνονται μέσω του αντικειμένου InfoWindowOptions. Σε αυτό περιλαμβάνονται οι εξής ιδιότητες:

- **content:** το περιεχόμενο του InfoWindow. Μπορεί να είναι ένα απλό κείμενο η και κώδικας HTML.
- **disableAutoPan:** θέτοντας αυτή την επιλογή ως false (προκαθορισμένη τιμή) ο χάρτης μετακινείται ώστε να εμφανίζεται ολόκληρο το InfoWindow. Με την τιμή true ο χάρτης παραμένει στο ίδιο σημείο αγνοώντας το μέγεθος η την θέση του InfoWindow.
- **maxWidth:** το μέγιστο πλάτος του InfoWindow, άσχετα από το περιεχόμενο του
- **pixelOffset:** η διαφορά θέσης σε pixel του InfoWindow από το σημείο πάνω στο οποίο είναι δεσμευμένο (συνήθως ένα marker).
- **position:** η ιδιότητα αυτή χρησιμοποιείται για να ορίσουμε διαφορετικό σημείο εμφάνισης του InfoWindow από το σημείο στο οποίο είναι δεσμευμένο.
- **zIndex:** αριθμός προτεραιότητας για την εμφάνιση πολλών InfoWindows. Μεγαλύτερη τιμή σημαίνει και προτεραιότητα στην εμφάνιση (το InfoWindow καλύπτει ένα άλλο με μικρότερη zIndex τιμή)

για να κατανοήσουμε καλυτέρα τον συνδυασμό χάρτη, marker και InfoWindow ας δούμε το παρακάτω παράδειγμα:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta content="text/html; charset=utf-8" http-equiv="Content-Type"
/>
<title>Info Window Example</title>
<script
src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"
></script>
<script type="text/javascript">
      function initialize() {
        var myLatlng = new google.maps.LatLng(-45,100);
        var mapOptions = {
            zoom: 5,
            center: myLatlng,
            mapTypeId: google.maps.MapTypeId.ROADMAP
        }

        var map = new
google.maps.Map(document.getElementById("map-canvas"),
mapOptions);

        var infoContent = "A simple Info Window";

        var infowindow = new google.maps.InfoWindow({
            content: infoContent
        });

        var marker = new google.maps.Marker({
            position: myLatlng,
            map: map,
            title:"Dummy Title"
```

```
        });

        google.maps.event.addListener(marker, 'click', function()
{
            infowindow.open(map,marker);
        });
      }
      google.maps.event.addDomListener(window, 'load',
initialize);
</script>
</head>
<body>
            <div id=map-canvas> </div>
</body>
</html>
```

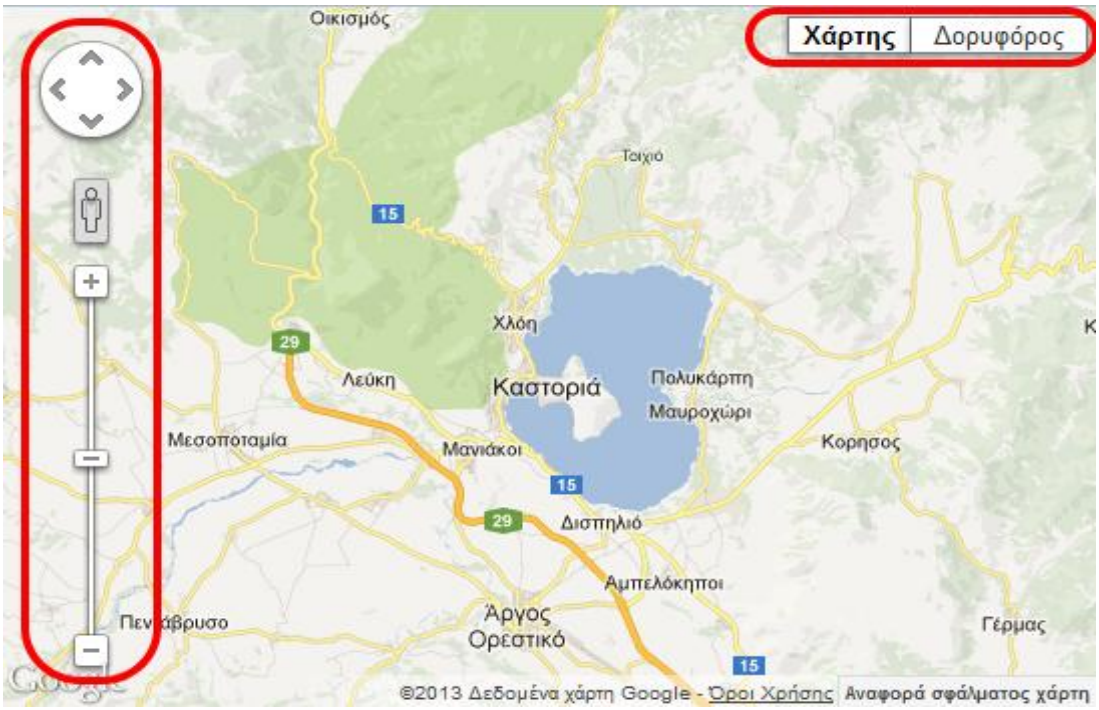
Η δημιουργία του χάρτη και του marker είναι γνωστή και δεν έχει καμία διαφορά από τα προηγούμενα παραδείγματα. Το αντικείμενο InfoWindow δημιουργείται καλώντας τον δομητή του με παραμέτρους τα options του. Στο παράδειγμα έχουμε μόνο μια επιλογή, το περιεχόμενο του παραθύρου στο οποίο έχουμε εισάγει το κείμενο “A simple Info Window”. Το πιο σημαντικό κομμάτι είναι ο event listener που δημιουργούμε ώστε το παράθυρο να εμφανίζεται με ένα κλικ στο marker. Μόλις ανιχνεύει το event (click) στο αντικείμενο το οποίο αφόρα (marker) καλείται μια μέθοδος που με την σειρά της καλεί την μέθοδο open() του αντικειμένου InfoWindow. Η open() δέχεται δυο ορίσματα: **1)** τον χάρτη πάνω στον οποίο θέλουμε να εμφανιστεί το InfoWindow μας και **2)** το στοιχείο με το οποίο δεσμεύεται (anchor).

2.9 Το User Interface του Google Maps

Εκτός από την πλοήγηση μόνο με τον κέρσορα του ποντικιού ο χάρτης περιέχει κουμπιά ελέγχου (controls από εδώ και στο εξής) τα οποία μας βοηθούν να πλοηγηθούμε σε αυτόν αλλά και να αλλάξουμε τον τύπο του ανάλογα με τις

ανάγκες μας. Μπορούμε να επεξεργαστούμε τα κουμπιά έλεγχου, να τους αλλάξουμε θέση και ακόμα να απενεργοποιήσουμε μερικά από αυτά η και όλα.
Τα controls που υπάρχουν στο Google Maps είναι τα εξής:

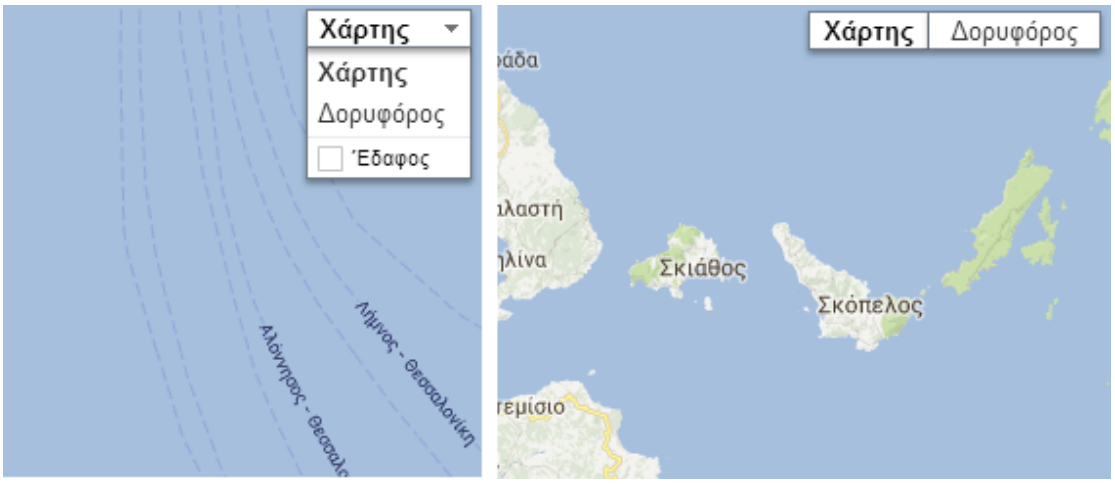
- **zoom control:** μια μπάρα ολίσθησης (slider) για μεγαλύτερους χάρτες ή δυο κουμπιά με τα σύμβολα + και – για μικρότερους, για τον έλεγχο του επίπεδου zoom. Η προκαθορισμένη του θέση είναι στα δεξιά του χάρτη.
- **pan control:** control σε σχήμα κύκλου για την μετακίνηση μας στον χάρτη. Η προκαθορισμένη του θέση είναι στο πάνω δεξί τμήμα
- **scale control:** ένδειξη της κλίμακας του χάρτη
- **Map Type Control:** control για την επιλογή του τύπου χάρτη που θα εμφανίζεται. Η προκαθορισμένη του θέση είναι στο πάνω αριστερό τμήμα
- **Street View Control:** control για την μετάβαση (σύροντας τον “pegman” στον χάρτη) σε Street View.
- **Rotate Control:** control το οποίο μας δίνει την δυνατότητα να αλλάξουμε την οπτική γωνία του χάρτη (όπου αυτό υποστηρίζεται)
- **OverView Map Control:** μικρογραφία του χάρτη της ευρύτερης περιοχής από το σημείο στο οποίο βρισκόμαστε



Εικόνα 2-4: Τα κουμπιά ελέγχου του Google Maps

Τα προκαθορισμένα controls ενός χάρτη (το default UI του) περιλαμβάνουν τα zoom pan και Map Type controls. Η εμφάνιση τους εξαρτάται από την συσκευή από την οποία βλέπουμε τον χάρτη και το μέγεθος της οθόνης. Για παράδειγμα το zoom control εμφανίζεται σαν slider σε μεγάλες οθόνες και σαν δυο απλά κουμπιά με τα σύμβολα πρόσθεσης και αφαίρεσης σε μικρότερες.

Όλες αυτές οι επιλογές μπορούμε να τις επεξεργαστούμε μέσω του αντικειμένου MapOptions που είδαμε σε προηγούμενο υποκεφάλαιο. Πολλές από αυτές αφορούν την εμφάνιση η απόκρυψη των controls. Μερικά από τα controls έχουν δικές τους επιλογές οι οποίες ορίζουν την εμφάνιση τους. Για παράδειγμα μπορούμε να αλλάξουμε την εμφάνιση του Map Type control από dropdown εικονίδια σε μια ενιαία μπάρα με τις επιλογές για την εμφάνιση του χάρτη



Εικόνα 2-5: Δυο διαφορετικοί τρόποι εμφάνισης του MapType control

2.9.1 Δημιουργία νέων controls και η θέση τους στον χάρτη

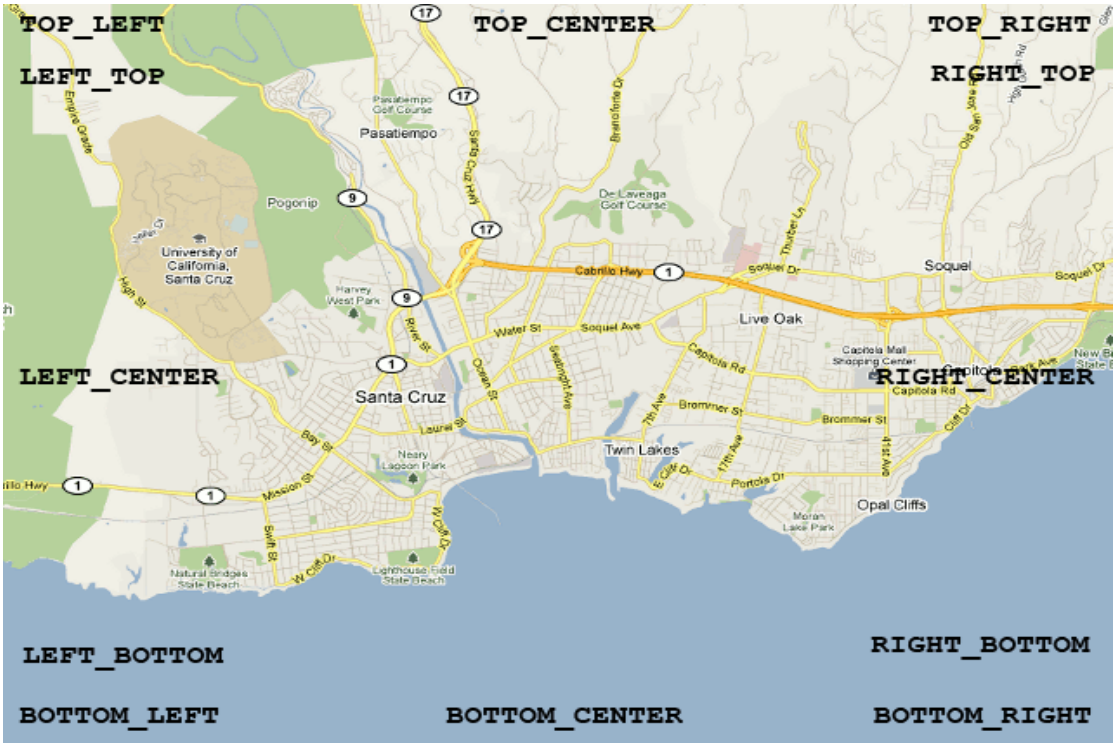
Τα ήδη υπάρχοντα controls του Google Maps μπορεί να μην καλύπτουν τις ανάγκες μας για μια πιο διαδραστική εφαρμογή. Έχουμε όμως την δυνατότητα να δημιουργήσουμε νέα controls τα οποία θα εμφανίζονται πάνω στον χάρτη και θα εξυπηρετούν τις ανάγκες μας. Τα αναγκαία βήματα για να δημιουργηθεί το δικό μας control είναι τα εξής:

- Ορισμός της εμφάνισης του control χρησιμοποιώντας CSS κανόνες
- Δέσμευση του control με έναν event handler ανάλογα με το πώς θέλουμε να ενεργοποιείται το control μας
- Δημιουργία ενός div-υποδοχέα για το control και προσθήκη του στα controls του χάρτη

Μετά την δημιουργία του control, σημαντικό θέμα είναι θέση του στον χάρτη. Μπορούμε να αλλάξουμε την θέση χρησιμοποιώντας την ιδιότητα position που έχουν οι επιλογές για το εκάστοτε control όπως φαίνεται στο παρακάτω παράδειγμα:

```
scaleControl: true,
scaleControlOptions: {
    position: google.maps.ControlPosition.TOP_LEFT
},
```

Οι επιλογές μας και η ακριβής θέση τους, είναι οι εξής όπως φαίνονται στην παρακάτω εικόνα:



Εικόνα 2-6: Θέσεις των controls στον χάρτη

Σε περίπτωση που έχουμε περισσότερα του ενός controls στο ίδιο σημείο το Google Maps θα τροποποιήσει την ακριβή τους θέση ώστε να είναι ευδιάκριτα. Σαν τελευταία λύση σε αυτό το πρόβλημα υπάρχει η μεταβλητή index η οποία καθορίζει ποιο control θα έχει προτεραιότητα στην εμφάνιση.

2.10 Υπηρεσία αναζήτησης οδηγιών (directions)

Το Google Maps προσφέρει την υπηρεσία directions η οποία μπορεί να μας δώσει ακριβείς οδηγίες για το πώς μπορούμε να μετακινηθούμε από ένα σημείο του χάρτη σε ένα άλλο. Η λειτουργιά αυτή γίνεται με ασύγχρονες κλήσεις (όπως δηλαδή και στους χάρτες) άλλα σε διαφορετικό διακομιστή από αυτόν που έχει τους χάρτες.

Για να χρησιμοποιήσουμε την υπηρεσία Directions χρειαζόμαστε:

- ένα αντικείμενο τύπου *DirectionService* το οποίο θα στέλνει τις αιτήσεις μας στον διακομιστή και θα δέχεται τα αποτελέσματα
- ένα αντικείμενο τύπου *DirectionsRequest* το οποίο θα περιέχει όλες τις πληροφορίες για την διαδρομή για την οποία θέλουμε οδηγίες
- ένα αντικείμενο τύπου *DirectionsRenderer* για να μπορέσουμε να σχεδιάσουμε την διαδρομή μας πάνω στον επιλεγμένο χάρτη

2.10.1 Το αντικείμενο DirectionsService

Το directionsService είναι ουσιαστικά ο σύνδεσμος μας με τον διακομιστή. Με την μέθοδο route(), η οποία είναι και η μοναδική του, στέλνει αιτήσεις για οδηγίες στον διακομιστή και δέχεται δεδομένα ελέγχου για την πορεία της αίτησης και τα δεδομένα προς χρήση.

Μέθοδος	Περιγραφή
route(request: directionsRequest, callback:function(DirectionsResult, DirectionsStatus))	Αίτηση προς τον διακομιστή για μια συγκεκριμένη διαδρομή. Επιστρέφονται μέσω συνάρτησης τα αντικείμενα DirectionsResult και DirectionsStatus

Πίνακας 2-7: Η μέθοδος Route

Το αντικείμενο directionsResult είναι προφανώς τα αποτελέσματα οδηγιών. Το directionStatus όμως έχει έναν άλλον ρόλο. Αυτός είναι να μας ενημερώνει για την πορεία της αίτησης και για το αν όλα πήγαν η αν προέκυψε κάποιο λάθος και ποιο ήταν αυτό.

Οι τιμές οι οποίες μπορεί να επιστραφούν από το directionStatus είναι οι εξής

- **OK:** έχουμε σαν αποτέλεσμα ένα έγκυρο DirectionsResult.
- **NOT_FOUND:** τουλάχιστον μια από τις τοποθεσίες που έχουμε ορίσει (αφετηρία, προορισμός, ενδιάμεσο σημείο) δεν μπορεί να βρεθεί.
- **ZERO_RESULTS:** δεν υπάρχει καμία διαδρομή ανάμεσα στη συγκεκριμένη αφετηρία και προορισμό.

- **MAX_WAYPOINTS_EXCEEDED:** έχουμε περισσότερα directionWaypoints από το επιτρεπτό όριο. Αυτό είναι 10 waypoints (αφετηρία κ προορισμός περιλαμβάνονται σε αυτά) ενώ ο αριθμός αυτός ανεβαίνει στα 25 για το business API του Google Maps.
- **INVALID_REQUEST:** η directionRequest δεν ήταν έγκυρη. Αυτό το μήνυμα λάθους οφείλεται συνήθως σε παράλειψη πεδίων όπως η αφετηρία η ο προορισμός η ακόμα και σε αιτήσεις με waypoints οι οποίες όμως αφορούν μέσα μαζικής μεταφοράς στα οποία δεν γίνεται να ορίσουμε εμείς προτεινόμενη διαδρομή.
- **OVER_QUERY_LIMIT:** η ιστοσελίδα μας έχει στείλει πολλές αιτήσεις σε μικρό χρονικό διάστημα
- **REQUEST_DENIED:** η ιστοσελίδα μας δεν επιτρέπεται να χρησιμοποιήσει την υπηρεσία Directions
- **UNKNOWN_ERROR:** η αίτηση μας αντιμετώπισε ένα άγνωστο πρόβλημα. Πιθανότατα πρόκειται για πρόβλημα του διακομιστή

2.10.2 Το αντικείμενο DirectionsRequest

Το αντικείμενο DirectionsRequest περιέχει όλες τις πληροφορίες τις οποίες χρειαζόμαστε για να κάνουμε μια αίτηση για οδηγίες πάνω στον χάρτη. Εισάγεται σαν πρώτη παράμετρος στην μέθοδο route() του αντικειμένου directionsService. Η πλήρης γενική μορφή του αντικειμένου είναι η εξής:

```
request = {
origin: Latlng | String
destination: Latlng | String
travelMode: TravelMode
transitOptions: TransitOptions
unitSystem: UnitSystem,
durationInTraffic: Boolean,
```

```
waypoints[]: DirectionsWaypoint,
optimizeWaypoints: Boolean,
provideRouteAlternatives: Boolean,
avoidHighways: Boolean,
avoidTolls: Boolean,
region: String
}
```

Τα δυο πρώτα πεδία είναι υποχρεωτικά και έχουν να κάνουν με την αφετηρία και το τέλος της διαδρομής μας (origin και destination αντίστοιχα) και είναι τύπου latlng.

Το πεδίο travelMode είναι επίσης υποχρεωτικό και ορίζει το μέσο με το οποίο ταξιδεύουμε αφού ανάμεσα σε δυο σημεία υπάρχουν διαφορετικές διαδρομές ανάλογα με το μέσο που έχουμε επιλέξει. Μπορεί να πάρει τις εξής 4 τιμές

- DRIVING (προκαθορισμένη τιμή) οδηγίες για μετακίνηση με αυτοκίνητο
- TRANSIT οδηγίες για μετακίνηση με μέσα μαζικής μεταφοράς
- BICYCLING οδηγίες για μετακίνηση με ποδήλατο
- WALKING οδηγίες για μετακίνηση πεζών

Τα υπόλοιπα πεδία είναι προαιρετικά και αφορούν άλλες επιλογές που μπορεί να έχουμε για την αίτηση μας.

Αναλυτικά:

***transitOptions*:** επιλογές που αφορούν την διαδρομή μας σε περίπτωση που έχουμε επιλέξει μέσα μαζικής μεταφοράς (transit) σαν travelMode.

***unitSystem*:** επιλογή ανάμεσα σε μετρικό και αγγλικό σύστημα μέτρησης αποστάσεων

***durationInTraffic*:** καθορίζει αν τα αποτελέσματα του directionsLeg θα περιέχουν πληροφορίες για την κίνηση στον δρόμο. Η συγκεκριμένη επιλογή δεν είναι ελεύθερη προς όλους τους χρήστες (χρήστες του Maps for Business μόνο) και επιστρέφει αποτελέσματα μόνο αν υπάρχουν πληροφορίες για την κίνηση στην συγκεκριμένη περιοχή.

***waypoints[]*:** το πεδίο αυτό μας επιτρέπει να ορίσουμε μια σειρά από σημεία στον χάρτη (waypoints) από τα οποία θέλουμε να παίρνει η διαδρομή μας.

***optimizeWaypoints*:** βελτίωση μιας διαδρομής με waypoints ώστε να έχουμε το καλύτερο αποτέλεσμα (μικρότερη διαδρομή)

***provideRouteAlternatives*:** καθορίζει το αν θα μπορούμε να λάβουμε σαν απάντηση παραπάνω από μια διαδρομές ανάμεσα σε δυο σημεία.

***avoidHighways*:** αν είναι true η διαδρομή μας δεν περιέχει εθνικές οδούς (όπου αυτό είναι δυνατό)

***avoidTolls*:** αν είναι true η διαδρομή μας δεν περιέχει σταθμούς διοδίων (όπου αυτό είναι δυνατό)

***region*:** κωδικός της περιοχής που μας ενδιαφέρει (για παράδειγμα GR, GB)

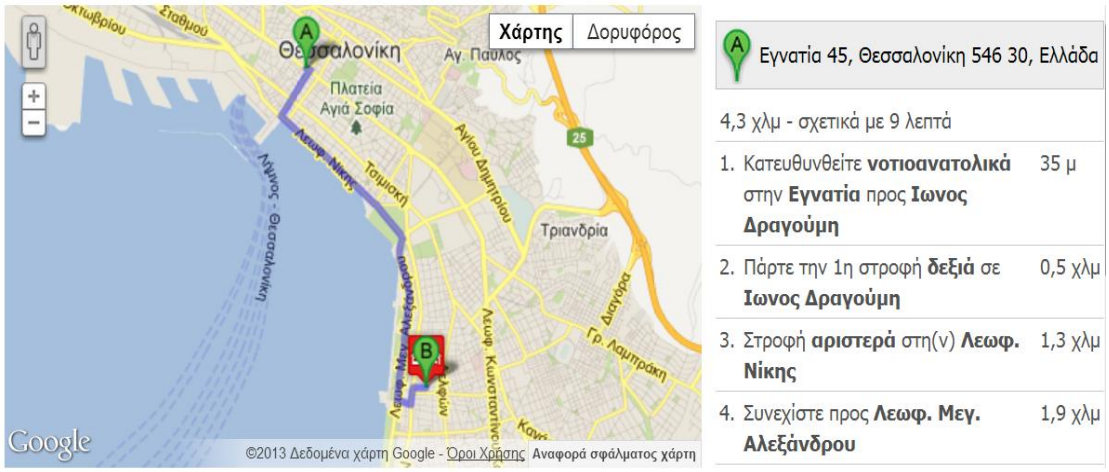
Συμπληρώνοντας αυτά τα πεδία ανάλογα με τις ανάγκες μας, έχουμε έτοιμη την αίτηση που θα σταλθεί στον διακομιστή. Το μόνο που απομένει είναι το πώς θα διαχειριστούμε τα αποτελέσματα.

2.10.3 Το αντικείμενο DirectionsRenderer

Τα αντικείμενα directionsService και directionRequest είναι αρκετά για να στείλουμε μια αίτηση για λήψη οδηγιών μεταξύ δυο σημείων. Ο στόχος μας όμως είναι φυσικά αυτές οι οδηγίες να εμφανίζονται στον χάρτη μας. Αυτή είναι η χρησιμότητα του directionsRenderer το οποίο δέχεται τις πληροφορίες που παίρνουμε από την μέθοδο route() και τις αναπαριστά στον χάρτη.

Για να σχεδιάσουμε την διαδρομή στον χάρτη αρχικά δημιουργούμε ένα στιγμιότυπο του αντικειμένου directionsRenderer. Για να ορίσουμε σε ποιον χάρτη θέλουμε να εμφανιστούν οι πληροφορίες χρησιμοποιούμε την μέθοδο setMap() η οποία παίρνει σαν μοναδικό όρισμα το όνομα του χάρτη. Τέλος, αφού η αίτηση για οδηγίες έχει σταλθεί και έχουμε τα αποτελέσματα (result), τα εισάγουμε σαν όρισμα στην μέθοδο setDirections(result).

Το αποτέλεσμα μετά από αυτή την διαδικασία φαίνεται στην παρακάτω εικόνα:



Εικόνα 2-7: Διαδρομή ανάμεσα σε δυο σημεία του χάρτη με αναλυτικές οδηγίες

2.10.4 Παράδειγμα υπηρεσίας Directions

Γνωρίζοντας πλέον ποια αντικείμενα και ποιες μεθόδους χρειαζόμαστε για την σχεδίαση οδηγιών στον χάρτη, μπορούμε πλέον να προχωρήσουμε σε ένα ολοκληρωμένο παράδειγμα κώδικα.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta content="text/html; charset=utf-8" http-equiv="Content-Type"
/>
<title>Directions Example</title>
<script
src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"
></script>
<script type="text/javascript">
      function initialize() {
        directionsDisplay = new google.maps.DirectionsRenderer();
        var chicago = new google.maps.LatLng(41.850033, -
87.6500523);
```

```
        var mapOptions = {
            zoom:7,
            mapTypeId: google.maps.MapTypeId.ROADMAP,
            center: chicago
        }
        map = new google.maps.Map(document.getElementById("map-
canvas"), mapOptions);
            directionsDisplay.setMap(map);
        }

        function calcRoute() {
            var start = Chicago;
            var end = Toronto;
            var request = {
                origin:start,
                destination:end,
                travelMode: google.maps.TravelMode.DRIVING
            };
            directionsService.route(request, function(result,
status) {
                if (status == google.maps.DirectionsStatus.OK) {
                directionsDisplay.setDirections(result);
                }
            });
        }

      google.maps.event.addDomListener(window, 'load',
initialize);
</script>
</head>
<body>
            <div id=map-canvas> </div>
            <input type="button" onclick="calcRoute()">
</body>
</html>
```

Αρχικά όπως είδαμε και σε προηγούμενο υποκεφάλαιο ορίζουμε τον χάρτη που θα περιέχει τις οδηγίες. Δημιουργούμε τρεις global μεταβλητές και αρχικοποιούμε την directionService με ένα νέο στιγμιότυπο της κλάσης.

Έπειτα, συμπληρώνουμε την μέθοδο initialize() για την δημιουργία του χάρτη. Η διαφορά με έναν απλό χάρτη είναι ότι εδώ αρχικοποιούμε την μεταβλητή directionsDisplay με ένα νέο αντικείμενο τύπου directionsRenderer και με την μέθοδο setMap() ορίζουμε που θα σχεδιαστούν οι οδηγίες.

Αφού έχουμε δημιουργήσει τον χάρτη το επόμενο βήμα είναι η μέθοδος που θα στέλνει τις αιτήσεις για οδηγίες και θα δέχεται τα αποτελέσματα. Ορίζουμε δυο μεταβλητές οι οποίες έχουν τον ρόλο της αφετηρίας και του προορισμού της διαδρομής μας. Παρατηρούμε ότι στο συγκεκριμένο παράδειγμα οι τιμές αυτές εισάγονται από αντικείμενα (όπως τα ορίζει το DOM σε ένα HTML έγγραφο) ενός HTML εγγράφου. Για καλύτερη κατανόηση ας υποθέσουμε ότι είναι δυο textboxes. Η Google μας δίνει την δυνατότητα εισάγοντας με απλό κείμενο τις πληροφορίες (πόλη, οδός, αριθμός – όχι αναγκαστικά με αυτή την σειρά) να μπορούμε να ορίσουμε το σημείο στον χάρτη το οποίο μας ενδιαφέρει (και μάλιστα με εξαιρετική ακρίβεια). Ο δεύτερος τρόπος είναι να εισάγουμε δυο αντικείμενα τύπου latlng τα οποία θα ορίσουν τις συντεταγμένες αφετηρίας και προορισμού.

Συμπληρώνουμε την αίτηση μας με τον τύπο οδηγιών που θέλουμε (TravelMode) ο οποίος στο συγκεκριμένο παράδειγμα είναι για αυτοκίνητα. Το τελευταίο μας βήμα είναι η αποστολή της αίτησης. Η μέθοδος route() του αντικειμένου directionService στέλνει τα δεδομένα μας και δέχεται τα αποτελέσματα και την κατάσταση της αίτησης. Αν το directionStatus περιέχει την τιμή OK τότε όλα πήγαν καλά με την αίτηση οπότε τα αποτελέσματα της (result) εισάγονται μέσω του αντικειμένου directionsDisplay (τύπου directionsRenderer) και της μεθόδου του, setDirections() στον χάρτη.

2.11 Υπηρεσία Geocoding

Ένα από τα πιο σημαντικά κομμάτια για έναν χάρτη είναι ο ακριβής και σωστός ορισμός διαφόρων τοποθεσιών. Οι τοποθεσίες που μας ενδιαφέρουν μπορούν να εμφανιστούν στον χάρτη με την μορφή ενός marker. Το κύριο πρόβλημα με τα markers είναι ότι λειτουργούν με συντεταγμένες (αντικείμενα τύπου LatLng) και όχι με διευθύνσεις η ονόματα πόλεων/περιοχών. Για να λυθεί αυτό το πρόβλημα και να κάνει την ζωή μας πιο εύκολη η Google προσφέρει την υπηρεσία Geocoding, ο ρόλος της οποίας είναι η μετατροπή συντεταγμένων σε διευθύνσεις και το αντίστροφο (reverse Geocoding). Η υπηρεσία προσφέρεται δωρεάν από την Google αλλά περιορίζεται στις 2.500 αιτήσεις για αναλύσεις διευθύνσεων ημερησίως. Όπως και με την υπηρεσία οδηγιών έτσι και εδώ κάνουμε χρήση αντικειμένων για την αποστολή αιτήσεων στον διακομιστή. Σε αντίθεση με την υπηρεσία directions εδώ χρησιμοποιούμε δυο αντικείμενα αφού αντίστοιχο του directionsRenderer δεν υπάρχει (και δεν χρειάζεται). Τα δυο αυτά αντικείμενα είναι τα εξής:

- Το αντικείμενο *Geocoder* το οποίο είναι υπεύθυνο για την αποστολή των αιτήσεων
- και το αντικείμενο *GeoCodeRequest* που περιέχει τις πληροφορίες για την αίτηση μας

2.11.1 Το αντικείμενο Geocoder

Το Geocoder είναι το αντικείμενο που δημιουργεί την σύνδεση με τον διακομιστή. Στέλνει αιτήσεις μέσω της μεθόδου geocode() για μετατροπή διευθύνσεων σε συντεταγμένες και δέχεται τα αποτελέσματα μαζί με μηνύματα ελέγχου για την πορεία της αίτησής μας.

Μέθοδος	Περιγραφή
geocode(request:**GeocoderRequest**, callback:function(Array.<**GeocoderResult**> , **GeocoderStatus**))	η μέθοδος η οποία στέλνει τις αιτήσεις μας στον διακομιστή. Επιστρέφονται μέσω συνάρτησης μηνύματα ελέγχου καθώς και ένας πίνακας με αποτελέσματα

Πίνακας 2-8: Η μέθοδος geocode()

Βλέπουμε ότι η geocode περιέχει ένα την αίτηση GeocoderRequest που θα αναλύσουμε αργότερα, και ότι δέχεται από τον διακομιστή έναν πίνακα τύπου GeocoderResult οπού βρίσκονται τα αποτελέσματα και ένα αντικείμενο GeocoderStatus δηλαδή το μήνυμα ελέγχου. Τα αποτελέσματα μας επιστρέφονται σε πινάκα γιατί υπάρχει περίπτωση να υπάρχουν περισσότερα από ένα αποτελέσματα.

Όσο για τα μηνύματα ελέγχου αυτά είναι τα εξής:

- **OK:** όλα πήγαν καλά με την διαδικασία και έχουν επιστραφεί αποτελέσματα
- **ZERO_RESULTS:** η αίτηση μας δεν επέστρεψε κανένα αποτέλεσμα. Συνήθως εμφανίζεται όταν έχουμε εισάγει διεύθυνση που δεν υπάρχει.
- **OVER_QUERY_LIMIT:** ένδειξη ότι έχουμε ξεπεράσει τον επιτρεπόμενο αριθμό αιτήσεων.
- **REQUEST_DENIED:** Η αίτηση μας απορριφθηκε
- **INVALID_REQUEST:** συνήθως πρόκειται για παράλειψη του πεδίου address η LatLng του αντικειμένου GeoCodeRequest

2.11.2 Το αντικείμενο GeoCodeRequest

Στο αντικείμενο GeoCodeRequest εισάγουμε τις πληροφορίες για την μετατροπή που χρειαζόμαστε. Αποτελεί την πρώτη παράμετρος της μεθόδου geocode().

Η γενική μορφή του αντικειμένου είναι η παρακάτω:

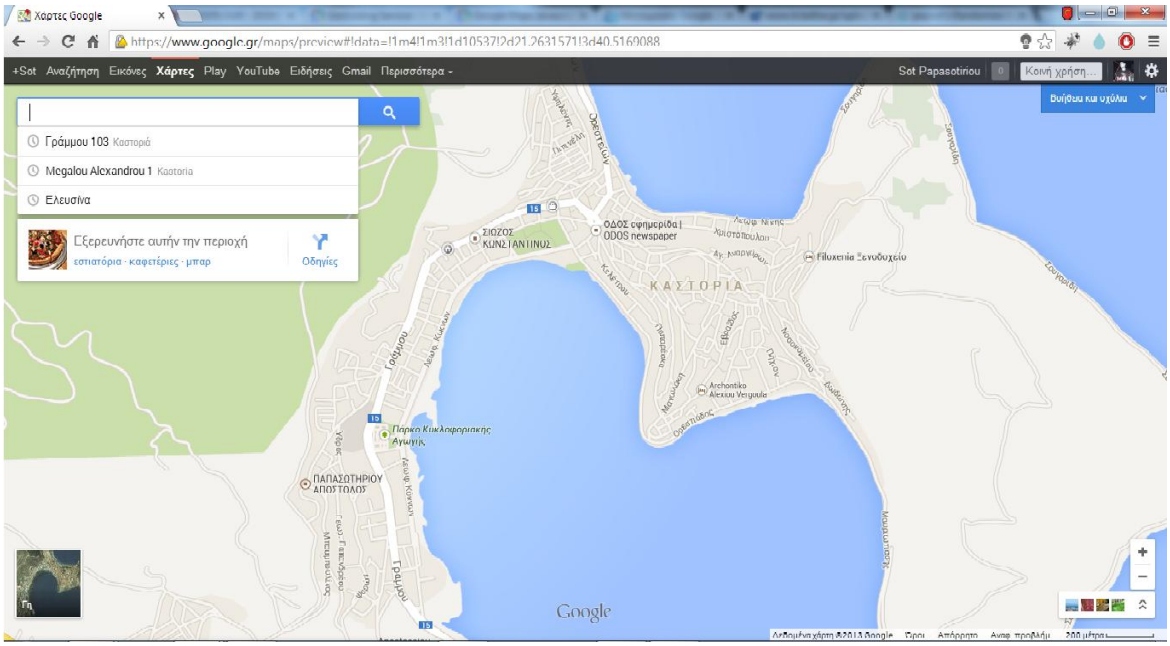
```
geoCodeRequest = {
 address: string,
 latLng: LatLng,
 bounds: LatLngBounds,
 region: string
}
```

Τα μόνα υποχρεωτικά πεδία είναι τα δυο πρώτα. Ωστόσο μπορούμε να εισάγουμε μόνο ένα από τα δυο ανάλογα με την μετατροπή που θέλουμε να κάνουμε. Για να μετατρέψουμε μια διεύθυνση σε συντεταγμένες εισάγουμε μια μεταβλητή τύπου string. Για μετατροπή συντεταγμένων σε διεύθυνση χρειαζόμαστε ένα αντικείμενο

LatLng. Η ιδιότητα bounds περιορίζει τα αποτελέσματα που ζητάμε σε μια συγκεκριμένη περιοχή ορισμένη από ένα σύνολο συντεταγμένων. Η ιδιότητα region κάνει το ίδιο αλλά ανάλογα με τον κωδικό περιοχής που έχουμε επιλέξει, για παράδειγμα GR για συντεταγμένες και διευθύνσεις που αφορούν την Ελλάδα.

Επίλογος - Το μέλλον του Google Maps

Το Google Maps είναι πλέον η πιο διαδεδομένη web based εφαρμογή στο διαδίκτυο. Η διαδραστικότητα και το πλήθος πληροφοριών που προσφέρει μέσω των εκατοντάδων mashups που υπάρχουν την έχουν κάνει με διαφορά την καλύτερη εφαρμογή χαρτογράφησης σε σχέση με άλλες παρόμοιες εφαρμογές όπως οι χάρτες της Microsoft (bing maps) ή της Yahoo. Εξελίσσεται συνεχώς και προσφέρει όλο και περισσότερες υπηρεσίες και πληροφορίες που χρειάζεται ο σημερινός χρήστης του διαδικτύου. Ήδη το επόμενο βήμα είναι κοντά: ένας χάρτης ο οποίος θα καλύπτει όλο το παράθυρο του browser μας και θα μας προσφέρει πληροφορίες για διαδρομές, μέσα μαζικής μεταφοράς, σημεία ενδιαφέροντος όπως εστιατόρια η καταστήματα, φωτογραφίες της περιοχής στην οποία βρισκόμαστε και πολλά άλλα. Όλα αυτά με ένα ακόμα πιο απλό τρόπο πλοήγησης και με νέα σχεδίαση για ολόκληρη την διεπαφή.



Εικόνα 2-8: Η νέα μορφή του Google Maps

Κεφάλαιο 3

Παρουσίαση της εφαρμογής

Εισαγωγή

Σε αυτό το κεφάλαιο θα γίνει μια αναλυτική περιγραφή της ιστοσελίδας του Μεσιτικού γραφείου. Θα αναφερθούν οι υπηρεσίες που προσφέρει καθώς και ο τρόπος με τον οποίο λειτουργούν.

3.1 Ιεραρχία του Μενού

Το κύριο μενού του site εμφανίζεται σε όλες τις σελίδες στο ίδιο σημείο πάντα (στο πάνω τμήμα του site δίπλα από το λογότυπο) για την πλοήγηση στις διάφορες υπηρεσίες. Οι επιλογές που ομαδοποιούνται κάτω από μια γενικότερη περιγραφή (για παράδειγμα η Αναζήτηση όπου εμφανίζονται δυο επιλογές αναζήτησης) εμφανίζονται, μόλις ο χρήστης τοποθετήσει τον κέρσορα πάνω από την επιλογή, σαν dropdown μενού. Η δομή του μενού είναι η εξής παρακάτω:

- Αρχική
    - Ακίνητα για Πώληση
    - Ακίνητα για Ενοικίαση
- Αναζήτηση Κατοικίας
    - Απλή Αναζήτηση
    - Αναζήτηση μέσω Google Maps
- Επιλογές Χρήστη
    - Σύνδεση – Αποσύνδεση
    - Δημιουργία Λογαριασμού
    - Το προφίλ μου
- Αποθηκευμένες Αναζητήσεις
- Επιλογές Διαχειριστή
    - Λίστα Ακινήτων
    - Προσθήκη Ακινήτου

Όλες αυτές οι επιλογές δεν εμφανίζονται σε κάθε χρήστη που επισκέπτεται την ιστοσελίδα μας. Γενικά οι χρήστες μας χωρίζονται σε τρεις κατηγορίες:

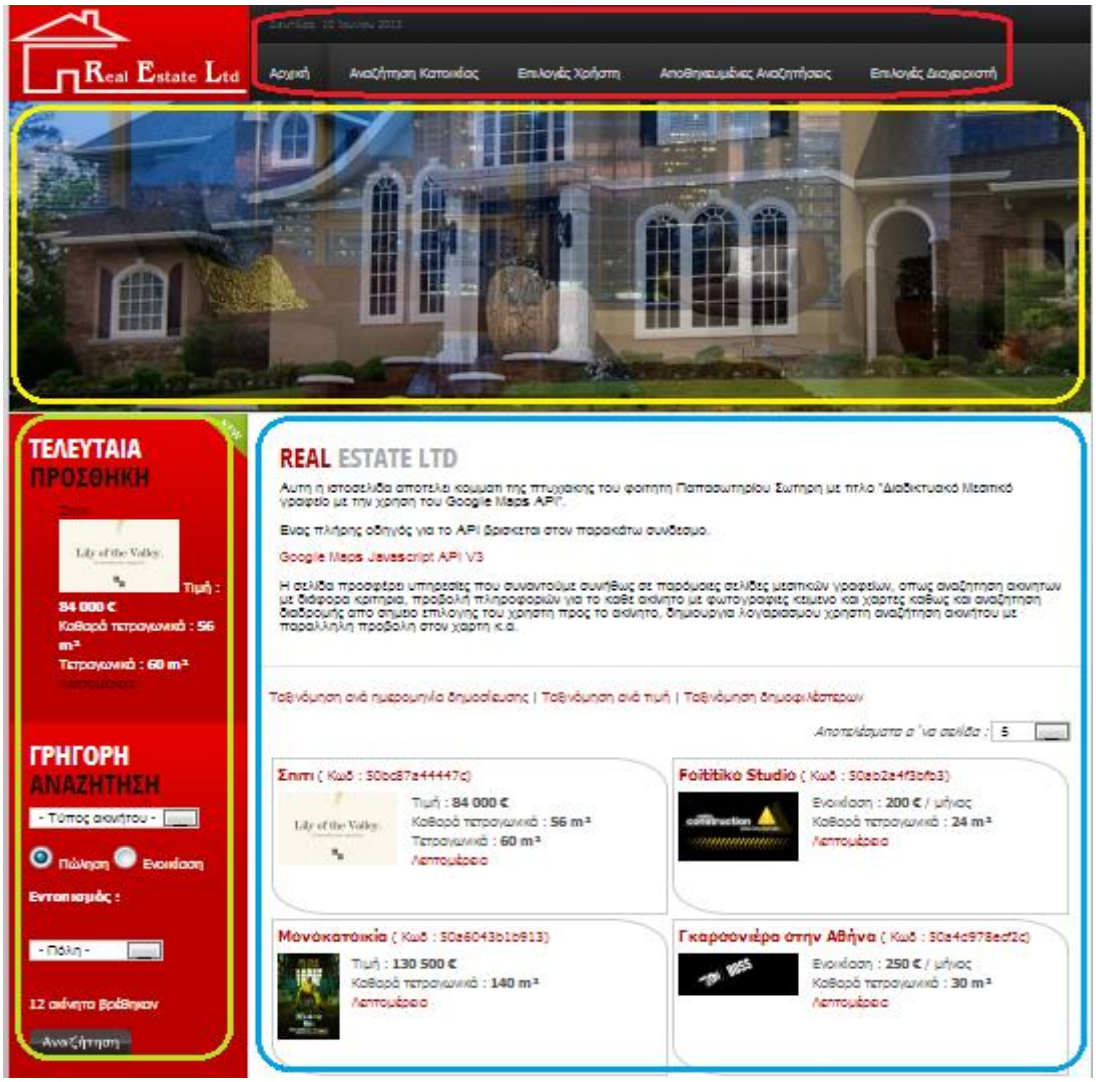
Guests: Επισκέπτες του site. Χρήστες οι οποίοι δεν έχουν δημιουργήσει λογαριασμό και για τους οποίους δεν έχουμε στοιχεία. Έχουν πρόσβαση στις επιλογές Αρχική (*Ακίνητα για πώληση*, *Ακίνητα για ενοικίαση*), Αναζήτηση Κατοικίας (*Απλή Αναζήτηση*, *Αναζήτηση μέσω Google Maps*) και τις Επιλογές Χρήστη (*Σύνδεση*, *Δημιουργία Λογαριασμού*).

Registered Users: Χρήστες του site που έχουν δημιουργήσει έναν προσωπικό λογαριασμό και μπορούν να συνδεθούν με τα στοιχεία τους. Έχουν πρόσβαση σε όλες τις επιλογές που έχουν και οι guests συν την επιλογή Αποθηκευμένες Αναζητήσεις και την σελίδα όπου εμφανίζονται οι προσωπικές τους πληροφορίες.

Special Users: Ειδική κατηγορία χρήστη. Ουσιαστικά ο διαχειριστής του site. Έχει πρόσβαση σε όλες τις επιλογές που αναφέρθηκαν παραπάνω συν δυο ακόμα επιλογές Διαχειριστή. Την εμφάνιση της ολοκληρωμένης λίστας ακινήτων του site και την προσθήκη ενός νέου ακινήτου στην λίστα.

3.2 Αρχική Σελίδα

Πληκτρολογώντας την διεύθυνση www.real-estate-thesis.gr στον browser μας βρισκόμαστε στην αρχική σελίδα του site. Η αρχική σελίδα μπορούμε να πούμε ότι χωρίζεται σε τέσσερα τμήματα. Το πρώτο τμήμα είναι το μενού του site το οποίο βρίσκεται στο πάνω μέρος της σελίδας. Ακριβώς από κάτω βλέπουμε έναν image slider ο οποίος αποτελεί το δεύτερο τμήμα. Πρόκειται για ένα module του Joomla το οποίο εναλλάσσει εικόνες σε τακτά χρονικά διαστήματα και το οποίο εμφανίζεται μόνο στην αρχική σελίδα. Συνεχίζοντας προς τα κάτω φτάνουμε στο τμήμα που φιλοξένει το κύριο περιεχόμενο του site μας. Συγκεκριμένα ένα κείμενο με πληροφορίες για την πτυχιακή και μια λίστα με όλες τα ακίνητα που υπάρχουν στην βάση δεδομένων και είναι δημοσιευμένα. Το τελευταίο τμήμα είναι η στήλη κόκκινου χρώματος στα αριστερά που χρησιμεύει για την εμφάνιση 2 modules. Το πρώτο εμφανίζει το πιο πρόσφατο ακίνητο του Μεσιτικού γραφείου και το δεύτερο είναι μια γρήγορη αναζήτηση ακινήτου χρησιμοποιώντας τα πιο σημαντικά κριτήρια.



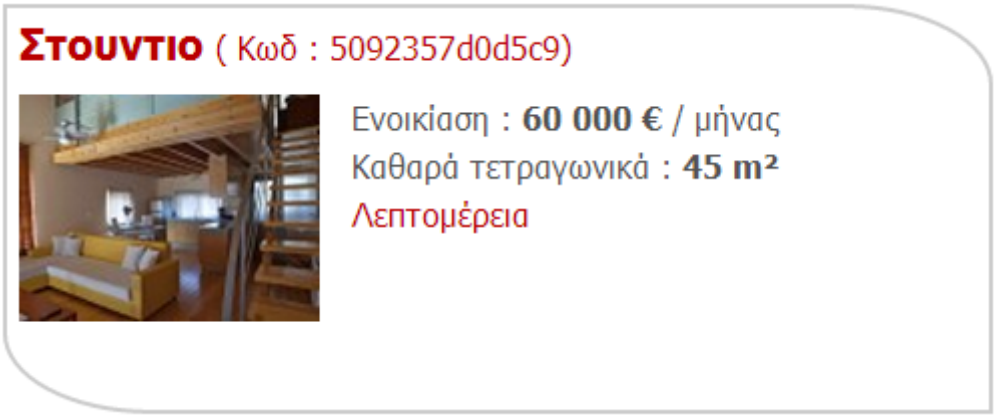
Εικόνα 3-1: Βασικά τμήματα της Αρχικής Σελίδας

Στην παραπάνω εικόνα φαίνονται τα τμήματα που αναφέραμε. Στο κόκκινο πλαίσιο είναι το μενού, στο κίτρινο ο Image Slider, στο πράσινο η στήλη με τα δυο Modules και στο γαλάζιο το κύριο περιεχόμενο.

3.2.1 Ακίνητα για Πώληση - Ενοικίαση

Η επιλογή αυτή μας μεταφέρει σε μια σελίδα η οποία περιέχει τις λίστες με τα αντίστοιχα ακίνητα δηλαδή μόνο για πώληση ή μόνο για Ενοικίαση. Η διαφορά με την αρχική σελίδα είναι ότι στο κύριο περιεχόμενο δεν έχουμε πλέον το κείμενο με τις πληροφορίες.

Τα ακίνητα εμφανίζονται με την εξής μορφή:



Εικόνα 3-2: Εμφάνιση του ακινήτου στην λίστα

Ο τίτλος του ακινήτου μαζί με το αναγνωριστικό του βρίσκονται στην κορυφή με έντονα κόκκινα γράμματα. Από κάτω μια από τις εικόνες που είναι διαθέσιμες για το ακίνητο και δίπλα της, σημαντικές πληροφορίες για την τιμή και τα τετραγωνικά. Επίσης πατώντας τον υπερσύνδεσμο “Λεπτομέρεια” μεταφερόμαστε στην σελίδα με αναλυτικές πληροφορίες του συγκεκριμένου ακινήτου.

3.3 Αναζήτηση Κατοικίας

Η λειτουργία αναζήτησης ακινήτων αποτελείται από δυο επιμέρους λειτουργίες. Μια κλασική λειτουργία αναζήτησης επιλέγοντας τις λεπτομέρειες των ακινήτων που ψάχνουμε και μια λειτουργία προβολής των ακινήτων (που ικανοποιούν τα κριτήρια μας) σε έναν χάρτη του Google Maps.

3.3.1 Απλή Αναζήτηση

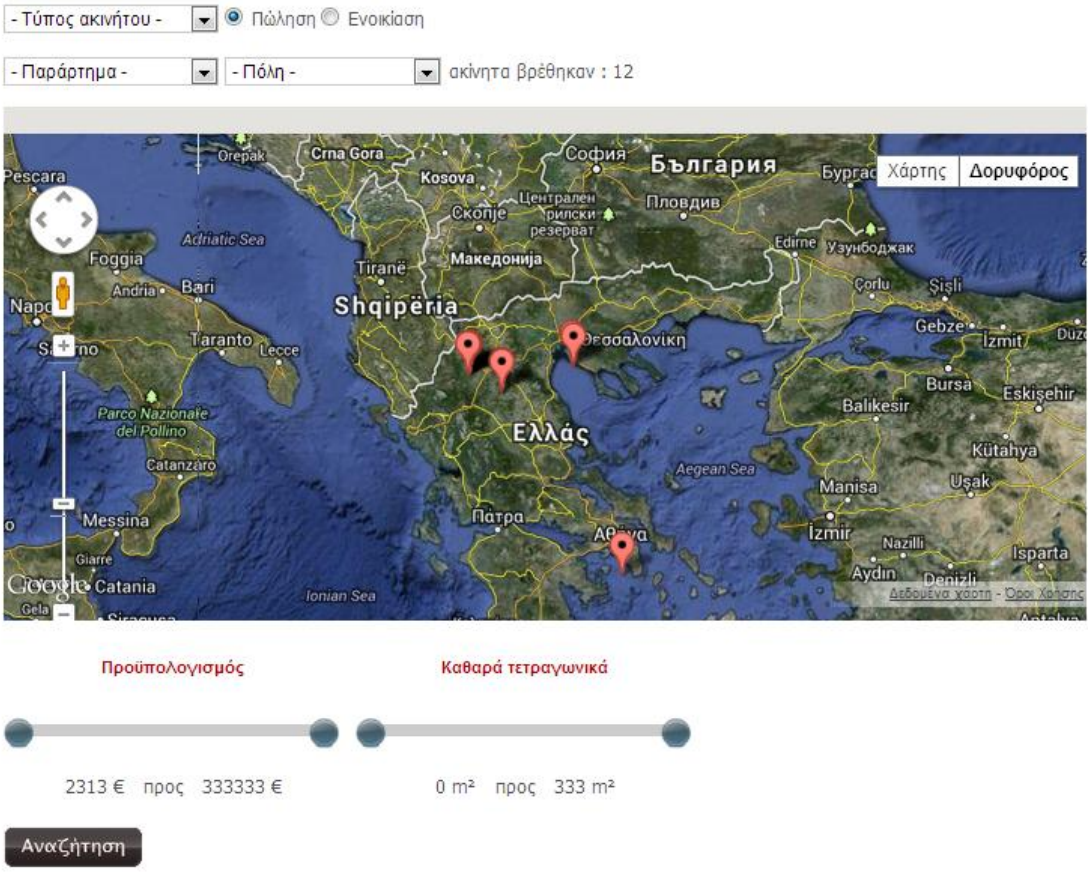
Η απλή αναζήτηση ακινήτων είναι ουσιαστικά μια φόρμα στην οποία συμπληρώνουμε τα κριτήρια σύμφωνα με τα οποία ψάχνουμε ένα ακίνητο. Περιέχει radio buttons, dropdown boxes και πεδία εισαγωγής στοιχείων για να μπορούμε να φιλτράρουμε καλύτερα την αναζήτηση μας. Τα κριτήρια με τα οποία μπορούμε να κάνουμε μια αναζήτηση είναι τα εξής:

- **Τύπος ακινήτου:** αν είναι διαμέρισμα, μονοκατοικία κ.α.
- **Πώληση ή Ενοικίαση**
- **Εντοπισμός:** η πόλη για την οποία ενδιαφερόμαστε
- **Προϋπολογισμός:** ορισμός τιμής ανάμεσα σε ελάχιστη και μέγιστη
- **Καθαρά τετραγωνικά:** η έκταση του ακινήτου
- **Ελάχιστος αριθμός δωματίων**
- **Τύπος θέρμανσης**

Αφού έχουμε εισάγει τα επιθυμητά κριτήρια μπορούμε είτε να προχωρήσουμε με την αναζήτηση (κουμπί αναζήτηση) είτε να σβήσουμε όλα τα στοιχεία που εισάγαμε και να ξεκινήσουμε από την αρχή (κουμπί καθαρισμός). Εκτός από τα δυο αυτά κουμπιά υπάρχει και ένα τρίτο κουμπί. Εμφανίζεται μόνο σε χρήστες που έχουν συνδεθεί στο site και χρησιμεύει στην αποθήκευση της αναζήτησής μας για μελλοντική χρήση. Η λειτουργία του θα αναλυθεί σε επόμενο υποκεφάλαιο.

3.3.2 Αναζήτηση μέσω Google Maps

Η λειτουργία αναζήτησης μέσω Google Maps μοιάζει με την απλή αναζήτηση γιατί και εδώ εισάγουμε πληροφορίες για τα κριτήρια της αναζήτησης. Οι κύριες διαφορές βρίσκονται στον τρόπο με τον οποίο εισάγουμε τα όρια για την επιθυμητή τιμή και τα τετραγωνικά του ακινήτου. Αυτό γίνεται με δυο slider bars στις οποίες μπορούμε να ορίσουμε το μέγιστο και ελάχιστο όριο της τιμής και των τετραγωνικών. Η πιο σημαντική διαφορά είναι ο χάρτης που βρίσκεται στο κέντρο της σελίδας και η εμφάνιση των αποτελεσμάτων αυτόματα πάνω του με την μορφή πολλαπλών markers. Αυτό δίνει την δυνατότητα στον χρήστη να ελέγχει την ακριβή τοποθεσία των ακινήτων που θα εμφανιστούν στα αποτελέσματα. Με κάθε αλλαγή στα κριτήρια της αναζήτησης γίνεται ανανέωση των δεδομένων στον χάρτη (χωρίς να ανανεωθεί ολόκληρη η σελίδα). Επιστρέφονται μόνο τα ακίνητα που πληρούν τις προϋποθέσεις και οι συντεταγμένες τους αποθηκεύονται σε έναν πίνακα με την χρήση JavaScript κώδικα. Έπειτα δημιουργούνται τα markers που θα εμφανιστούν στον χάρτη χρησιμοποιώντας τα αποθηκευμένα δεδομένα συντεταγμένων. Για την αναλυτική λίστα των ακινήτων ο χρήστης απλά πατάει το κουμπί αναζήτηση στο κάτω μέρος της σελίδας.



Εικόνα 3-3: Η αναζήτηση μέσω Google Maps

3.4 Επιλογές Χρήστη

Ακλουθώντας την πρώτη επιλογή (*Σύνδεση*) μεταφερόμαστε σε μια φόρμα εισαγωγής ονόματος χρήστη και κωδικού για να μπορέσουμε να συνδεθούμε με το site. Η επιλογή *Δημιουργία Λογαριασμοί* μας μεταφέρει σε μια φόρμα εισαγωγής στοιχείων (όνομα χρήστη και κωδικός, στοιχεία επικοινωνίας, e-mail κ.α.) για την δημιουργία ενός νέου λογαριασμού. Στην περίπτωση που έχουμε ήδη συνδεθεί η επιλογή σύνδεση μετατρέπεται σε “Αποσύνδεση” και προστίθεται η επιλογή “Το προφίλ μου” οπού μπορούμε να ελέγξουμε τον λογαριασμό μας, να εισάγουμε νέα στοιχεία κ.α.

3.5 Αποθηκευμένες Αναζητήσεις

Η ιστοσελίδα προσφέρει την επιλογή στους χρήστες της να αποθηκεύσουν μια αναζήτηση που έχουν κάνει ώστε να μπορούν να την επαναλάβουν στο μέλλον, χωρίς να χρειάζεται να ακολουθήσουν την πιο χρονοβόρα διαδικασία εισαγωγής όλων των κριτηρίων αναζήτησης. Η υπηρεσία αυτή προσφέρεται μόνο σε χρήστες που έχουν δημιουργήσει λογαριασμό και έχουν συνδεθεί στο site για να υπάρχει τρόπος να ομαδοποιηθούν οι αναζητήσεις στην βάση δεδομένων ανάλογα με τον χρήστη. Η αρχή γίνεται στην σελίδα της απλής αναζήτησης. Ο χρήστης μπορεί να επιλέξει την αποθήκευση της αναζήτησης του, πατώντας το αντίστοιχο κουμπί στο τέλος της σελίδας. Έπειτα όλα τα στοιχεία που έχει εισάγει ο χρήστης για την αναζήτησή του μεταφέρονται σε μια δεύτερη φόρμα HTML κρυφή από τον χρήστη. Το κουμπί αποθήκευση λειτουργεί όπως ένα απλό κουμπί υποβολής μιας HTML φόρμας. Εξαιτίας όμως της κατάστασης της φόρμας μας (δεν είναι ορατή στον χρήστη), είναι αναγκαίο το κουμπί να βρίσκεται έξω από αυτήν. Πλέον το να ορίσουμε την ιδιότητα type του κουμπιού στην τιμή submit δεν είναι αποτελεσματικό. Η υποβολή της φόρμας γίνεται τελικά μέσω της μεθόδου *post()* της jQuery.

Χρησιμοποιώντας την μέθοδο POST στέλνουμε τα δεδομένα της φόρμας στον διακομιστή οπού τα επεξεργάζεται το αρχείο *insert_data.php*. Σε αυτό το PHP αρχείο υλοποιείται η σύνδεση με την βάση δεδομένων και η εισαγωγή των δεδομένων στον πίνακα *user_search*. Ο πίνακας *user_search* περιέχει όλες τις αποθηκευμένες αναζητήσεις όλων των χρηστών του site. Για να μπορέσουμε να αντιστοιχήσουμε τις αναζητήσεις στους χρήστες χρησιμοποιούμε δυο κλειδιά. Το κύριο κλειδί είναι το πεδίο *search_id* και το ξένο κλειδί το πεδίο *user_id*. Με αυτή την διαδικασία ο κάθε χρήστης μπορεί να προσθέσει εγγραφές στον πίνακα *user_search*. Όπως αναφέρθηκε παραπάνω η επιλογή είναι ενεργή μόνο για χρήστες που έχουν συνδεθεί.

Αυτό σημαίνει ότι το κουμπί της λειτουργιάς εμφανίζεται μόνο σε αυτή την κατηγορία χρηστών. Για να υλοποιηθεί ο συγκεκριμένος περιορισμός πραγματοποιούμε έναν έλεγχο για το user_id του χρήστη. Αν είναι 0 αυτό σημαίνει ότι δεν έχει συνδεθεί κάποιος χρήστης και το κουμπί δεν εμφανίζεται. Αν είναι οτιδήποτε εκτός από 0 το κουμπί εμφανίζεται κανονικά.

Το μενού Αποθηκευμένες Αναζητήσεις εμφανίζει στον χρήστη μια λίστα με τις αναζητήσεις που έχει επιλέξει να αποθηκεύσει και τις λεπτομέρειες της κάθε μίας. Ακόμα, δίνεται η δυνατότητα για νέα αναζήτηση με τα ίδια κριτήρια πατώντας το κουμπί με τον μεγεθυντικό φακό. Φυσικά με την νέα αναζήτηση επιστρέφεται μια λίστα με ακίνητα που ικανοποιούν τα κριτήρια εκείνη την στιγμή και όχι τα αντίστοιχα όταν έγινε η αποθήκευση της αναζήτησης. Ένα παράδειγμα των αποθηκευμένων αναζητήσεων είναι το παρακάτω.

Τύπος Ακινήτου	Πωλ/ Ενοικ.	Πόλη	Προϋπ. ελάχιστος	Προϋπ. μέγιστος	τ.μ. ελάχιστα	τ.μ. μέγιστα	ελ. αριθ. δωματίων	τύπος θέρμανσης	Νέα Αναζήτηση
κένο	RENTING	Καστοριά	0	0	0	0	0	κένο	
κένο	SELLING	Καστοριά	0	0	0	0	0	κένο	
Διαμερίσματα	RENTING	Ηράκλειο	100	400	45	60	0	κένο	
Γκαρσονιέρες	SELLING	Κοζάνη	30000	50000	0	0	0	κένο	
Μονοκατοικίες	SELLING	Θεσσαλονίκη	100000	150000	0	100	5	κένο	

Εικόνα 3-4: Η λίστα με τις αποθηκευμένες αναζητήσεις ενός χρήστη

3.6 Επιλογές Διαχειριστή

Οι επιλογές διαχειριστή είναι ορατές μόνο σε περίπτωση που συνδεθεί κάποιος χρήστης με δικαιώματα διαχείρισης της σελίδας (μπορούμε να έχουμε παραπάνω από έναν διαχειριστές). Οι επιλογές που έχει ο διαχειριστής είναι η εμφάνιση μιας λίστας όλων των ακινήτων της σελίδας και η προσθήκη ενός νέου ακινήτου.

3.6.1 Λίστα Ακινήτων

Επιλέγοντας την λίστα ακινήτων από το κύριο μενού ο διαχειριστής μεταφέρεται σε μια σελίδα που περιέχει όλα τα ακίνητα, μαζί με κάποιες πληροφορίες για το καθένα, σε έναν πίνακα. Σε ένα πλήρως λειτουργικό site μεσιτικού γραφείου αυτή η λίστα μπορεί να είναι τεράστια οπότε έχουμε την επιλογή να μειώσουμε το

μέγεθος της, ορίζοντας το είδος συναλλαγής και τον τύπο ακινήτου που ψάχνουμε. Η λίστα μας προσφέρει τρεις λειτουργίες για την διαχείριση των ακινήτων. Στην πρώτη της στήλη, πατώντας τον κωδικό του ακινήτου μεταφερόμαστε σε μια σελίδα επεξεργασίας των πληροφοριών του. Στις δυο τελευταίες στήλες υπάρχουν οι υπόλοιπες λειτουργίες: η πρώτη μας δίνει την δυνατότητα να δημοσιεύσουμε (η να αποκρύψουμε) το ακίνητο στους υπολοίπους χρήστες και επισκέπτες του site. Η τελευταία επιλογή μας επιτρέπει να διαγράψουμε εντελώς το ακίνητο από το site και από την βάση δεδομένων.

- Είδος συναλλαγής - - Τύπος ακινήτου - - Επιλογή Γλώσσας -

Κωδ	Τύπος ακινήτου	Διεύθυνση	Πόλη	Καθαρά τετραγωνικά	Τιμή	Κατάσταση	Διαγραφή
50bc87a44447c	Μονοκατοικίες	Μιαούλη 14	Θεσσαλονίκη	56 m²	84 000 €		
50ab2a4f3bfb3	Γκαρσονιέρες	Ελευσίνος 9	Θεσσαλονίκη	24 m²	200 € / μήνας		
50a6043b1b913	Μονοκατοικίες	Πτολεμαίων 15	Κοζάνη	140 m²	130 500 €		
50a4d978ecf2c	Γκαρσονιέρες	Ιερά Οδός 68	Αθήνα	30 m²	250 € / μήνας		
50a0e1d4c1734	Βίλες	Μητροπολεως 40	Καστοριά	200 m²	70 000 €		
50939c3be96e2	Μονοκατοικίες	πιτσουλακη 45	Ηράκλειο	34 m²	30 000 € / μήνας		

Εικόνα 3-5: Η λίστα ακινήτων όπως την βλέπει ο διαχειριστής του site

3.6.2 Προσθήκη Ακινήτου

Το δεύτερο κομμάτι των επιλογών διαχειριστή είναι μια φόρμα προσθήκης νέου ακινήτου. Οι επιλογές μας για εισαγωγή στοιχείων είναι πάρα πολλές ώστε να μπορούν να δώσουν μια πολύ καλή εικόνα του ακίνητου στον χρήστη. Στο πρώτο κομμάτι της φόρμας μπορούμε να εισάγουμε τον τίτλο της κατοικίας, τον τύπο της καθώς και τον τύπο συναλλαγής (πώληση ή ενοικίαση). Μπορούμε να προσθέσουμε και μια περιγραφή χρησιμοποιώντας έναν WYSIWYG editor. Στο δεύτερο κομμάτι ορίζουμε την τοποθεσία του ακινήτου. Διεύθυνση Παράρτημα, Πόλη και Περιοχή είναι οι επιλογές που μπορούμε να επεξεργαστούμε. Υπάρχουν ακόμα δυο πεδία για τις συντεταγμένες του ακινήτου. Σε περίπτωση που έχουμε εισάγει μια σωστή διεύθυνση, πατώντας πάνω στον σύνδεσμο “Άνοιγμα Χάρτη” ενεργοποιείται η υπηρεσία Geocoder του Google Maps για να μεταφραστεί η διεύθυνση που έχουμε εισάγει σε συντεταγμένες. Αυτές θα χρησιμοποιηθούν στην

παρουσίαση των πληροφοριών του ακινήτου (στον χάρτη θα υπάρχει marker που θα αντιστοιχεί στην τοποθεσία του ακίνητου). Φυσικά έχουμε την επιλογή, μετά το άνοιγμα του χάρτη, να ορίσουμε εμείς τις συντεταγμένες μετακινώντας το marker στον χάρτη. Όταν ο χάρτης κλείσει οι συντεταγμένες θα έχουν συμπληρωθεί αυτόματα.

Μετά τις πληροφορίες για την τοποθεσία έχουμε την εισαγωγή οικονομικών πληροφοριών (τιμή αγοράς, μηνιαίο ενοίκιο κ.α.) και την εισαγωγή περισσότερων λεπτομερειών για το ακίνητο.

Αναλυτικότερα:

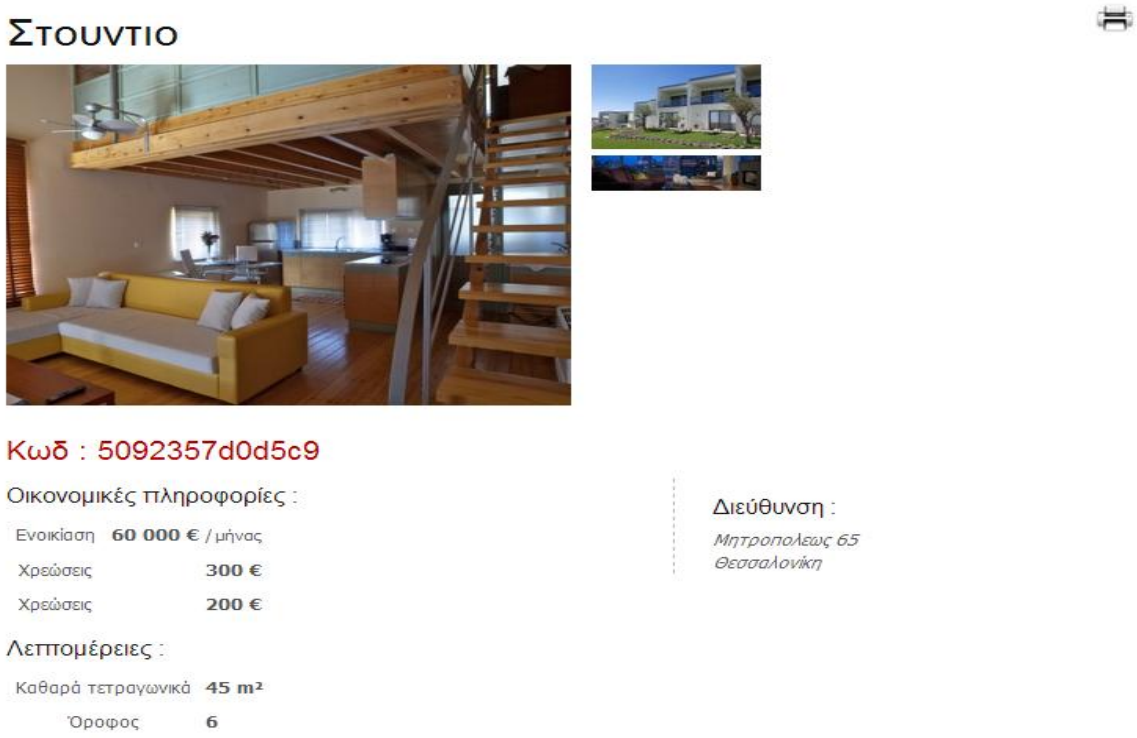
- Καθαρά τετραγωνικά
- Τετραγωνικά
- Ημερομηνία διάθεσης
- Κατάσταση ακινήτου
- Προσανατολισμός
- Όροφος
- Αριθμός ορόφων
- Αριθμός δωματίων
- Αριθμός υπνοδωματίων
- Αριθμός μπάνιων
- Αριθμός WC
- Τύπος θέρμανσης νερού
- Τύπος θέρμανσης

Οι πληροφορίες δημοσίευσης που ακολουθούν αφορούν περισσότερο το πότε και αν θέλουμε να δημοσιευτεί το ακίνητο στο site μας.
Τέλος μας δίνεται η δυνατότητα να προσθέσουμε φωτογραφίες του ακινήτου για να μπορεί ο χρήστης να έχει πραγματική εικόνα από τους εσωτερικούς η εξωτερικούς χώρους του.
Αφού έχουμε εισάγει αυτές τις πληροφορίες (η τμήμα τους) πατώντας το κουμπί “αποθήκευση” προσθέτουμε άλλο ένα ακίνητο στην λίστα με τα ήδη υπάρχοντα.

3.7 Εμφάνιση Πληροφοριών Ακινήτου

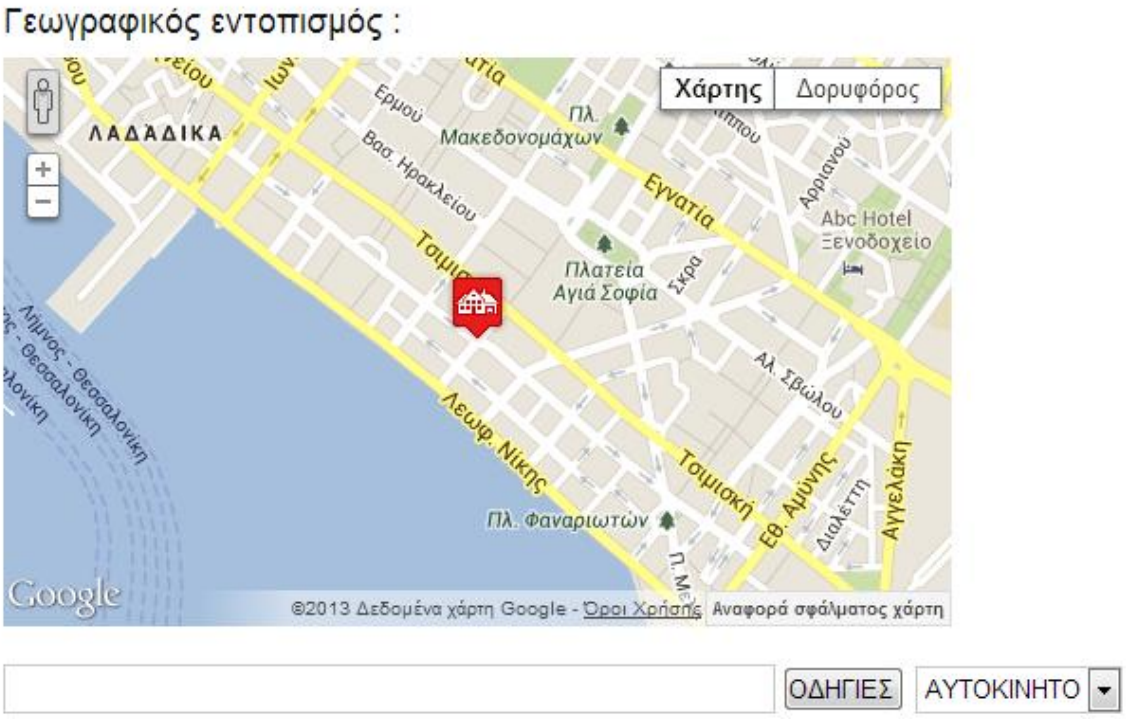
Η περιήγηση μας στο κύριο μενού έχει τελειώσει. Έχουμε δει και αναλύσει τις περισσότερες λειτουργίες του site. Το μόνο που μένει, και σίγουρα ένα από τα πιο σημαντικά κομμάτια όλης της ιστοσελίδας, είναι η εμφάνιση των πληροφοριών του ακινήτου. Προφανώς δεν υπάρχει κάποιο τμήμα του μενού που να εμφανίζει αυτές τις πληροφορίες. Θα πρέπει να επιλέξουμε τον τίτλο του ακινήτου η τον σύνδεσμο “λεπτομέρειες” που εμφανίζεται για κάθε ακίνητο σε κάθε λίστα (είτε στις αρχικές λίστες ακινήτων είτε σε αυτές που επιστρέφονται ως αποτελέσματα αναζήτησης). Αφού επιλέξουμε μια από τις παραπάνω ενέργειες μεταφερόμαστε στην σελίδα πληροφοριών του ακινήτου που έχουμε επιλέξει.

Στην αρχή της σελίδας εμφανίζονται ο τίτλος του ακινήτου και οι φωτογραφίες από εσωτερικούς η εξωτερικούς του χώρους. Ακριβώς από κάτω βλέπουμε περισσότερες πληροφορίες για την τιμή και την διεύθυνση του. Η περιγραφή του ακινήτου εμφανίζεται αμέσως μετά.



Εικόνα 3-6: Φωτογραφίες και βασικές πληροφορίες του ακινήτου

Μετά από τις βασικές πληροφορίες σειρά έχει ο χάρτης στον οποίο εμφανίζεται η τοποθεσία του ακινήτου. Ο χάρτης έχει δημιουργηθεί με την εφαρμογή Google Maps που αναλύθηκε στο προηγούμενο κεφάλαιο. Πάνω στον χάρτη παρατηρούμε ένα marker το οποίο υποδεικνύει την θέση του ακινήτου. Κάτω ακριβώς από τον χάρτη υπάρχουν ένα πεδίο εισαγωγής διεύθυνσης (η τοποθεσίας) ένα dropdown box με τις επιλογές “Αυτοκίνητο” και “Πεζός” και ένα κουμπί με το όνομα “Οδηγίες”. Με την χρήση των παραπάνω (εισαγωγή διεύθυνσης, επιλογή τρόπου ταξιδιού και πάτημα του κουμπιού) χρησιμοποιούμε την υπηρεσία Directions του Google Maps για την εμφάνιση οδηγιών στον χάρτη από το σημείο που επιλεγούμε στο ακίνητο. Στον κενό χώρο στα δεξιά του χάρτη εμφανίζονται αναλυτικές οδηγίες βήμα-βήμα για την διαδρομή μας σε μορφή κειμένου.



Εικόνα 3-7: Εμφάνιση του χάρτη και του τμήματος εύρεσης οδηγιών

Μετά τον χάρτη, το τελευταίο συστατικό της σελίδας είναι η φόρμα επικοινωνίας. Οποιοσδήποτε χρήστης μπορεί να συμπληρώσει την φόρμα και να στείλει ένα μήνυμα στον διαχειριστή της σελίδας για το συγκεκριμένο ακίνητο.

Επίλογος

Σε αυτό το κεφάλαιο παρουσιάστηκε η δομή και οι λειτουργιές της ιστοσελίδας Μεσιτικού γραφείου. Η ιστοσελίδα κατασκευάστηκε με κύριο μέλημα την ευκολία στην χρήση, τον ξεκάθαρο διαχωρισμό των υπηρεσιών που προσφέρει, για ευκολότερη πλοήγηση, την πληρότητα σε πληροφορίες για τα ακίνητα και την κατάσταση τους και τέλος, τον απλό και μινιμαλιστικό (όσο αυτό ήταν δυνατό) σχεδιασμό της εμφάνισης της. Οι υπηρεσίες που προστεθήκαν πιστεύω πως είναι σημαντικά κομμάτια για την επιτυχία μιας τέτοιας ιστοσελίδας και για την γρήγορη εξυπηρέτηση των χρηστών της.

Συμπεράσματα

Ξεκινώντας την διαδικασία ανάπτυξης της ιστοσελίδας για την πτυχιακή μου εργασία ήμουν αρχάριος στα περισσότερα τμήματα που την αποτελούσαν. Στην πορεία και μετά από μελέτη ανακάλυψα ότι τα εργαλεία που χρησιμοποίησα είναι εύκολα στην χρήση τους και μπορεί οποιοσδήποτε με βασικές γνώσεις προγραμματισμού να τα χειριστεί και να παράγει ιστοσελίδες υψηλού επιπέδου λειτουργικότητας και κομψού σχεδιασμού. Το σύστημα διαχείρισης περιεχομένου Joomla! αποτέλεσε το βασικό εργαλείο για την ολοκλήρωση του έργου και σε συνδυασμό με τα συστατικά κώδικα του extension Joomla Estate Agency μου έδωσαν την δυνατότητα να κατασκευάσω την ιστοσελίδα που παρουσιάστηκε σε αυτή την εργασία. Εξίσου σημαντικό εργαλείο για την ιστοσελίδα ήταν το API του Google Maps μιας από τις πιο εκλεπτυσμένες AJAX εφαρμογές αυτή την στιγμή στο διαδίκτυο. Οι επιλογές και οι δυνατότητες που προσφέρει για την επεξεργασία του χάρτη είναι τόσες πολλές που μπορεί να γραφτεί ολόκληρο βιβλίο πάνω στο API μόνο. Από την πλευρά μου προτίμησα να εισάγω λειτουργιές οι οποίες θα είχαν κάποια σχέση με ένα site μεσιτικού γραφείου, και προσπάθησα να καλύψω αυτές τις λειτουργιές όσο πιο αναλυτικά και απλά μπορούσα ώστε το αποτέλεσμα να είναι κατανοητό και να μην κουράζει. Με το τέλος της πτυχιακής μου εργασίας συνειδητοποίησα την πληθώρα εργαλείων που υπάρχουν στο διαδίκτυο για την δημιουργία ιστοσελίδων ή web based εφαρμογών, πολλά από τα οποία είναι ελευθέρα για χρήση. Με αυτά τα εργαλεία και τις εφαρμογές δημιουργούνται καθημερινά όλο και καλύτερα site ενώ εξελίσσονται για να προσφέρουν όλο και περισσότερες δυνατότητες και για να απλοποιήσουν ακόμα περισσότερο τις κλασσικές λειτουργίες ενός site αφήνοντας τον προγραμματιστή να ασχοληθεί με πιο δημιουργικά κομμάτια κώδικα η εμφάνισης.

Βιβλιογραφία-Πηγές

[1] Luke Welling, Laura Thompson (2009), PHP and MySQL Web development 4th edition

[2] http://www.php.net/usage.php PHP usage stats (30-4-2013)

[3] Σαλαμπάσης Μιχάλης, (2008), Εισαγωγή στον προγραμματισμό διαδικτυακών εφαρμογών

[4] Danny Goodman, Michael Morrison, Paul Novitski, Tia Gustaff Rayl, (2010), JavaScript Bible seventh edition

[5] http://www.w3.org/DOM/#what ορισμός Document Object Model (1-6-2013)

[6] Steve Suehring, Tim Converse, Joyce Park, (2009), PHP6 and MySQL bible

[7] Steven Holzner, Nancy Conner, (2009), Joomla for dummies

[8] http://forum.joomla.org/viewtopic.php?t=73 Mambo Open Source Development Team - Letter to the community - Discussion (4-6-2013)

[9] http://docs.joomla.org/What_are_components%2C_modules%2C_mambots_and_plugins%3F Extension Types – Ορισμοί (4-6-2013)

[10] Gabriel Svennerberg, (2010), Beginning Google Maps API 3

[11] http://news.cnet.com/Google-mapper-Take-browsers-to-the-limit/2100-1038_3-5808658.html History of Google Maps (17-5-2013)

[12] http://www.google.com/help/maps/helloworld/behind/history.html Landmarks of Google maps (17-5-2013)

[13] http://www.programmableweb.com/apis statistics for API's (15-5-2013)

[14] https://developers.google.com/maps/documentation/javascript/reference Τεκμηρίωση του Google Maps API (Ιούνιος 2013)

ΠΑΡΑΡΤΗΜΑ ΚΩΔΙΚΑ

Δημιουργία Πίνακα user_search

```
CREATE TABLE `realesta_db`.`user_search` (
`search_id` INT( 11 ) NOT NULL AUTO_INCREMENT ,
`user_id` INT( 11 ) NOT NULL ,
`type_id` INT( 11 ) NOT NULL ,
`transaction_type` VARCHAR( 15 ) COLLATE utf8_unicode_ci NOT NULL ,
`town_id` INT( 11 ) NOT NULL ,
`budget_min` INT( 11 ) NOT NULL ,
`budget_max` INT( 11 ) NOT NULL ,
`living-space-min` INT( 11 ) NOT NULL ,
`living-space-max` INT( 11 ) NOT NULL ,
`rooms_min` INT( 11 ) NOT NULL ,
`heatingtype` INT( 11 ) NOT NULL ,
PRIMARY KEY ( `search_id` )
) ENGINE = MYISAM DEFAULT CHARSET = utf8 COLLATE = utf8_unicode_ci;
```

Αρχείο user_search.php

```
<html>
    <head>
        <meta http-equiv="content-type" content="text/html; charset=UTF-
8" />
        <meta http-equiv="content-style-type" content="text/css" />
        <meta http-equiv="content-language" content="el" />
         <style>
body{
margin: 0px 0px 15px;
padding: 0px;
color: #525253;
font-family: Tahoma, Helvetica, sans-serif;
font-size: 11px;
line-height: 19px;

}

</style>
    </head>
    <body>
<?php
error_reporting(E_ERROR);
define( '_JEXEC', 1 );
define( 'JPATH_BASE', dirname(__FILE__) );
define( 'DS', '/' );

require_once ( JPATH_BASE .DS.'includes'.DS.'defines.php' );
require_once ( JPATH_BASE .DS.'includes'.DS.'framework.php' );

JFactory::getApplication('site')->initialise();
```

```
$user = JFactory::getUser();



$user_search= $user->id;

// create connection

$con=mysqli_connect("localhost","realesta_user","y8psZ7y}[H)n","realesta_
db");

// set the sql query
$result = mysqli_query($con,"SELECT distinct * FROM user_search where
user_id=$user_search") or die("Error: ".mysqli_error($con));
?>
</br>

<table border="1" width="100%" id="table1" style="border-collapse:
collapse">
    <tr style="border: 1px; border: solid; border-color: #ccc">
        <td><span lang="el" style="color:#B90000">Τύπος
Ακινήτου</span></td>
        <td><span lang="el">Πωλ/Ενοικ.</span></td>
        <td><span lang="el" style="color:#B90000">Πόλη</span></td>
        <td><span lang="el">Προϋπ. ελάχιστος </span></td>
        <td><span lang="el" style="color:#B90000">Προϋπ.
μέγιστος</span></td>
        <td><span lang="el">τ.μ. ελάχιστα</span></td>
        <td><span lang="el" style="color:#B90000">τ.μ.
μέγιστα</span></td>
        <td><span lang="el">ελ. αριθ. δωματίων</span></td>
        <td><span lang="el" style="color:#B90000">τύπος
θέρμανσης</span></td>
        <td><span lang="el">Νέα Αναζήτηση</span></td>
    </tr>


<?php

//loop the query and create a table row with data until empty
while($row = mysqli_fetch_array($result, MYSQLI_ASSOC))
  {
  ?>
    <tr style="border: 1px; border: solid; border-color: #ccc">
        <td><span lang="el">
<?php

switch ($row['type_id'])
{
case "2":
  echo "Apartments";
  break;
case "3":
  echo "Διαμερίσματα";
  break;
case "4":
  echo "Μονοκατοικίες";
  break;
case "5":
  echo "Βίλες";
```

```
  break;
case "6":
  echo "Γκαρσονιέρες";
  break;
default:
  echo "κένο";
}
?>
                                        </span></td>

        <td><span lang="el">
        <?php
        echo  $row['transaction_type'];
        if($row['transaction_type']=="SELLING")     {$transaction_type=1;}
else {$transaction_type=2;}


        ?></span></td>
        <td><span lang="el"><?php

switch ($row['town_id'])
{
case "6":
  echo "Αθήνα";
  break;
case "9":
  echo "Ηράκλειο";
  break;
case "8":
  echo "Θεσσαλονίκη";
  break;
case "5":
  echo "Καστοριά";
  break;
case "4":
  echo "Κοζάνη";
  break;
default:
  echo "κένο";
}
?>
                                    </span></td>

        <td><span     lang="el"><?php     echo          $row['budget_min'];
?></span></td>
        <td><span     lang="el"><?php     echo          $row['budget_max'];
?></span></td>
        <td><span    lang="el"><?php    echo       $row['living_space_min'];
?></span></td>
        <td><span    lang="el"><?php    echo       $row['living_space_max'];
?></span></td>
        <td><span     lang="el"><?php     echo          $row['rooms_min'];
?></span></td>
        <td><span lang="el"><?php

switch ($row['heatingtype'])
{
case "6":
  echo "Αυτόνομη";
  break;
case "9":
```

```
  echo "Δεν διαθέτει";
  break;
case "8":
  echo "Υγραέρειο";
  break;
default:
  echo "κένο";
}
?>
                                        </span></td>

        <td><span   lang="el"><a   href="./view_results.php?user_id=<?php
echo $user_search ?>&town_id=<?php echo $row['town_id'] ?>&type_id=<?php
echo  $row['type_id']  ?>&transaction_type=<?php  echo  $transaction_type
?>&budget_min=<?php  echo  $row['budget_min']  ?>&budget_max=<?php  echo
$row['budget_max']              ?>&living_space_min=<?php              echo
$row['living_space_min']          ?>&living_space_max=<?php          echo
$row['living_space_max']   ?>&rooms_min=<?php   echo   $row['rooms_min']
?>&heatingtype=<?php        echo        $row['heatingtype']        ?>"><img
src="./images/com_jea/thumb-medium/search.png"            title="Αναζήτση"
></a></span></td>
    </tr>
<?php
 }

// close database connection
mysqli_close($con);
?>

</table>
```

Αρχείο view_results.php

```
<html>
    <head>
        <meta http-equiv="content-type" content="text/html; charset=UTF-
8" />
        <meta http-equiv="content-style-type" content="text/css" />
        <meta http-equiv="content-language" content="el" />
         <style>
body{
margin: 0px 0px 15px;
padding: 0px;
color: #525253;
font-family: Tahoma, Helvetica, sans-serif;
font-size: 11px;
line-height: 19px;
}
</style>
    </head>
    <body>

<?php

// retrieve parameters
$user_id = $_GET['user_id'];
$type_id = $_GET['town_id'];
```

```
$type_id = $_GET['type_id'];
$transaction_type = $_GET['transaction_type'];
$budget_min = $_GET['budget_min'];
$budget_max = $_GET['budget_max'];
$living_space_min = $_GET['living_space_min'];
$living_space_max = $_GET['living_space_max'];
$rooms_min = $_GET['rooms_min'];
$heatingtype = $_GET['heatingtype'];

// create connection
$con=mysql_connect("localhost","realesta_user","y8psZ7y}[H)n");

mysql_query("SET NAMES 'utf8'", $con);

$db_select = mysql_select_db("realesta_db", $con);

// set up the sql statement
$sql_query= "select * from uwo60_jea_properties where ";

If ($transaction_type==1) { $sql_query= $sql_query . 'transaction_type like "SELLING"';} else { $sql_query= $sql_query . 'transaction_type like "RENTING"';}

If ($type_id>0) { $sql_query= $sql_query . " and type_id=$type_id";}

If ($town_id>0) { $sql_query= $sql_query . " and town_id=$town_id";}

If (($budget_max>0)&&($budget_min>0)) {$sql_query= $sql_query . " and price BETWEEN $budget_min AND $budget_max";}

If (($living_space_min>0)&&($living_space_max>0)) {$sql_query= $sql_query . " and living_space BETWEEN $living_space_min AND $living_space_max";}

If ($rooms_min>0) { $sql_query= $sql_query . " and rooms between $rooms_min and 1000";}

If ($heatingtype>0) { $sql_query= $sql_query . " and heatingtype=$heatingtype";}


// set the sql query
$result = mysql_query($sql_query);


$num = mysql_num_rows($result);

//loop the query and create a table row with data until empty
if ($num > 0 ) {
$i=0;
while ($i < $num) {
 //$row=mysql_result($result,$i);
 //$title=$row['title'];
$id=mysql_result($result,$i,"id");
  ?>


  <table border="1" width="100%" id="table1" style="border-collapse: collapse">
    <tr>
```

```
        <td rowspan="4"><img src="./images/com_jea/thumb-medium/<?php
        $str=mysql_result($result,$i,"images");
        $str_exp= explode(",", $str);
        $str_exp2= explode(":", $str_exp[0]);

        if ($str=='[]') {
        echo "no_photo.jpg";}
        else{
        if ($str_exp2[1]!=='""') {
        $str_rep=str_replace('"', '', $str_exp2[1]);
        echo $id.'-'.$str_rep;}
        else {
        echo "no_photo.jpg";}
        }


        ?>"></td>
        <td style="color:#B90000">Όνομα</td>
        <td><span lang="el"><?php  echo mysql_result($result,$i,"title");
?></span></td>
    </tr>
    <tr>
        <td>Τιμή</td>
        <td><?php  echo mysql_result($result,$i,"price");   ?></td>
    </tr>
    <tr>
        <td style="color:#B90000">Τετραγωνικά</td>
        <td><?php       echo    mysql_result($result,$i,"living_space");;
?></td>
    </tr>
    <tr>
        <td     colspan="2"><a     target="_top"     style="color:#B90000"
href="<?php   echo   './index.php?option=com_jea&view=property&id='.   $id
?>">Λεπτομέρειες</a></td>
    </tr>
</table>
  </br>
<?php

$i++;}
}
?>
</body>
<html>
```

Αρχείο insert_data.php

```
<?php
echo $_POST['user_id'];

// Get post val
$user_id = $_POST['user_id'];
$type_id = $_POST['type_id'];
$transaction_type = $_POST['transaction_type'];
$town_id= $_POST['town_id'];
$budget_min = $_POST['budget_min'];
$budget_max = $_POST['budget_max'];
```

```
$living_space_min= $_POST['living_space_min'];
$living_space_max = $_POST['living_space_max'];
$rooms_min = $_POST['rooms_min'];
$heatingtype = $_POST['heatingtype'];

echo 'gggg'. $user_id;



$db=mysqli_connect("localhost","realesta_user","y8psZ7y}[H)n","realesta_d
b");

if (mysqli_connect_errno($db))
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }

mysqli_query($db,"INSERT INTO user_search
(user_id,type_id,transaction_type,town_id,budget_min,budget_max,living_sp
ace_min,living_space_max,rooms_min,heatingtype) VALUES
('$_POST[user_id]','$_POST[type_id]','$_POST[transaction_type]','$_POST[t
own_id]','$_POST[budget_min]','$_POST[budget_max]','$_POST[living_space_m
in]','$_POST[living_space_max]','$_POST[rooms_min]','$_POST[heatingtype]'
)");

echo "1 record added";


mysqli_close($db);
?>
```

Αρχείο search.php (φόρμα “hidden” και JavaScript κώδικας για μεταφορά δεδομένων στην φόρμα)

```
<form method="post" style="visibility:hidden" id="HiddenForm"
action="./../../mysearch/insert_data.php">
<input name="user_id" type="text" id="user_id" value="<?php echo $user-
>id ?>"/>
<input name="type_id" type="text" id="type_id" />

<input name="transaction_type" type="text" id="transaction_type" />
<input name="town_id" type="text" id="town_id" />
<input name="budget_min" type="text" id="budget_min" />
<input name="budget_max" type="text" id="budget_max" />
<input name="living_space_min" type="text" id="living_space_min" />
<input name="living_space_max" type="text" id="living_space_max" />
<input name="rooms_min" type="text" id="rooms_min" />
<input name="heatingtype" type="text" id="heatingtype" />

</form>
<?php

if($user->id!=0){ ?>
<div style="width: 50%">
<input name="Button1" type="image" style="float: right"
src="http://www.real-estate-
thesis.gr/media/com_jea/images/save_search_inactive.png"
```

```
onmouseover="this.src='http://www.real-estate-
thesis.gr/media/com_jea/images/save_search_active.png'"
onmouseout="this.src='http://www.real-estate-
thesis.gr/media/com_jea/images/save_search_inactive.png'"  value="button"
id="sub" onclick="copyPasteFields()" />
</div>
<?php
}
?>

<script                                                type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js"></s
cript>

<script type="text/javascript">
function copyPasteFields(){

var type_id=document.getElementById("filter_type_id").value;


var town_id=document.getElementById("filter_town_id").value;

var budget_min=document.getElementById("jea-search-budget-min").value;
var budget_max=document.getElementById("jea-search-budget-max").value;
var     livingSpace_min=document.getElementById("jea-search-living-space-
min").value;
var     livingSpace_max=document.getElementById("jea-search-living-space-
max").value;
var search_rooms=document.getElementById("jea-search-rooms").value;
var heatingType=document.getElementById("filter_heatingtype").value;

if(document.getElementById("jea-search-selling").checked){
    var             transaction_type=document.getElementById("jea-search-
selling").value;
}else{
    var             transaction_type=document.getElementById("jea-search-
renting").value;
}

document.getElementById("type_id").value=type_id;
document.getElementById("transaction_type").value=transaction_type;
document.getElementById("town_id").value=town_id;
document.getElementById("budget_min").value=budget_min;
document.getElementById("budget_max").value=budget_max;
document.getElementById("living_space_min").value=livingSpace_min;
document.getElementById("living_space_max").value=livingSpace_max;
document.getElementById("rooms_min").value=search_rooms;
document.getElementById("heatingtype").value=heatingType;

}

$("#sub").click( function() {
$.post( $("#HiddenForm").attr("action"),
$("#HiddenForm :input").serializeArray()
);
});

$("#HiddenForm").submit( function() {
return false;
});
</script>
```

Αρχείο default_googlemap.php (JavaScript κώδικας για εμφάνιση χάρτη και υπηρεσίας οδηγιών)

```
var map = null;
var directionsDisplay;
var directionsService;
var markerArray = [];
var stepDisplay;


function initMap(mapOptions, MarkerLatlng) {
    map     =     new     google.maps.Map(document.id('jea_property_map'),
mapOptions);
    var marker = new google.maps.Marker({
        position: MarkerLatlng,
        map: map,
        title: '{$this->row->ref}',
        icon:                                    "http://www.real-estate-
thesis.gr/media/com_jea/images/house.png"
    });
    directionsService = new google.maps.DirectionsService();
    directionsDisplay = new google.maps.DirectionsRenderer();
    directionsDisplay.setMap(map);
    directionsDisplay.setPanel(document.getElementById('directions-
panel'));
}

window.addEvent("domready", function(){
    var longitude  = {$longitude};
    var latitude   = {$latitude};



    if (longitude && latitude) {
        var myLatlng = new google.maps.LatLng(latitude, longitude);
        var options = {
          zoom : 15,
          center : myLatlng,
          mapTypeId : google.maps.MapTypeId.ROADMAP
        };
        initMap(options, myLatlng);

    } else {
        var geocoder = new google.maps.Geocoder();
        var    opts    =    {'address':'$address',    'language':'$lang',
'region':'$region'};
        geocoder.geocode(opts, function(results, status) {
            if (status == google.maps.GeocoderStatus.OK) {
                var myLatlng = results[0].geometry.location;
                var options = {
                  center : myLatlng,
                  mapTypeId : google.maps.MapTypeId.ROADMAP
                };
                initMap(options, myLatlng);
                map.fitBounds(results[0].geometry.viewport);
             }
        });
    }
```

```
});

function getDirections() {
  var selectedMode = document.getElementById('mode').value;
  var start = document.getElementById("DirectionsStart").value;
  var end = new google.maps.LatLng($latitude, $longitude);
  var request = {
    origin:start,
    destination:end,
    travelMode: google.maps.TravelMode[selectedMode]
  };
  directionsService.route(request, function(result, status) {
    if (status == google.maps.DirectionsStatus.OK) {
      directionsDisplay.setDirections(result);
    }
  });
}



EOD;
$this->document->addScriptDeclaration($script);
?>
<style>
    #directions-panel {
        height: 300px;
        float: right;
        width: 290px;
        overflow: auto;
      }

      #map-canvas {
        float: left;
      }

</style>
<div id="directions-panel"></div>
<div id="jea_property_map"></div>
<input type="text" id="DirectionsStart" style="height:22px; width:50%;
text-align:center">
<input type="button" id="DirectionsButton" value="ΟΔΗΓΙΕΣ"
onclick="getDirections()">
<select id="mode" style="height: 25px;">
      <option value="DRIVING">ΑΥΤΟΚΙΝΗΤΟ</option>
      <option value="WALKING">ΠΕΖΟΣ</option>
</select>
```