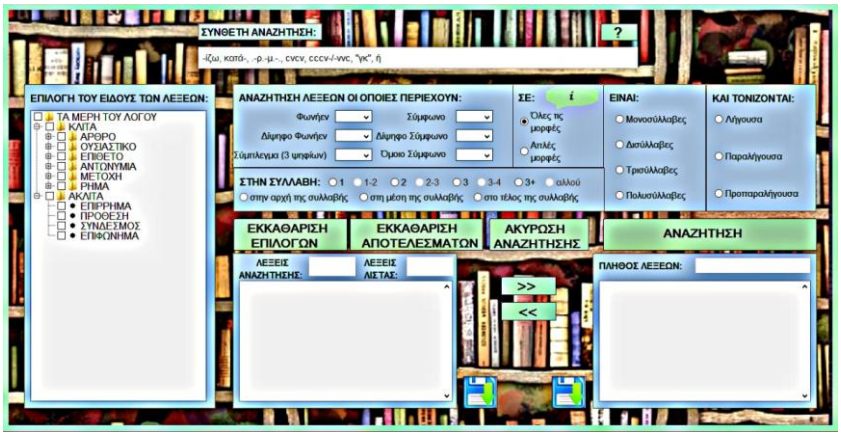
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

Σχολή Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών

Τομέας Μαθηματικών

“Σχεδίαση και Ανάπτυξη Εφαρμογής Αναζήτησης Λεξικού με Χρήση Κανονικών Εκφράσεων”



Διπλωματική Εργασία: Κούβελα Ελένη

Επιβλέπων: Συμβώνης Αντώνιος, Καθηγητής Ε.Μ.Π.

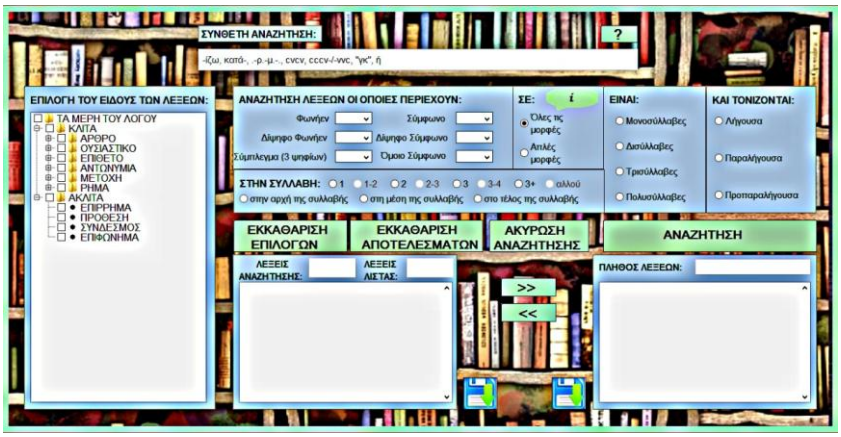
Αθήνα, Σεπτέμβριος 2015

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

Σχολή Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών

Τομέας Μαθηματικών

“Σχεδίαση και Ανάπτυξη Εφαρμογής Αναζήτησης Λεξικού με Χρήση Κανονικών Εκφράσεων”



Διπλωματική Εργασία: Κούβελα Ελένη

Επιβλέπων: Συμβώνης Αντώνιος, Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή:

……..	……..	……..
Συμβώνης Αντώνιος	Στεφανέας Πέτρος	Κολέτσος Ιωάννης
Καθηγητής Ε.Μ.Π.	Λέκτορας Ε.Μ.Π.	Επίκουρος Καθηγητής Ε.Μ.Π.

Αθήνα, Σεπτέμβριος 2015

.....

Κούβελα Ελένη
Διπλωματούχος Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών Ε.Μ.Π.

Ευχαριστίες

Για την εκπόνηση της παρούσας διπλωματικής εργασίας, την ανάπτυξη της εφαρμογής, αλλά και γενικότερα για την ολοκλήρωση του κύκλου σπουδών μου, νιώθω την ανάγκη να ευχαριστήσω όλους τους ανθρώπους που με βοήθησαν στο εγχείρημα αυτό.

Αρχικά, αποδίδω τις θερμές ευχαριστίες μου στον επιβλέποντα της διπλωματικής εργασίας μου κ. Συμβώνη Αντώνιο, Καθηγητή Ε.Μ.Π., για την ανάθεση της συγκεκριμένης εργασίας και την δυνατότητα ενασχόλησής μου με το συγκεκριμένο αντικείμενο.

Εν συνεχεία, θα ήθελα να ευχαριστήσω ξεχωριστά τα μέλη της επιτροπής, τον κ. Κολέτσο Ιωάννη, Επίκουρο Καθηγητή Ε.Μ.Π., καθώς και τον κ. Στεφανέα Πέτρο, Λέκτορα Ε.Μ.Π., για τη σημαντική συμβολή τους στην περαίωση της.

Ευχαριστίες θα ήθελα, επίσης, να απευθύνω στον υποψήφιο Διδάκτορα Λίτσα Χρήστο, για τη συνεχή και πολύτιμη καθοδήγηση και επίβλεψη που μου παρείχε κατά την πορεία εκπόνησης της εργασίας.

Τέλος, θα ήθελα να ευχαριστήσω ιδιαίτερα την οικογένειά μου, καθώς και τους συμφοιτητές και φίλους μου, οι οποίοι μου συμπαραστάθηκαν και με ενθάρρυναν με κάθε τρόπο, καθ' όλη την διάρκεια των σπουδών μου.

Περίληψη

Τα σύγχρονα τεχνολογικά επιτεύγματα αξιοποιούνται από πληθώρα επιστημονικών κλάδων. Η χρήση της τεχνολογίας, συγκεκριμένα, για εκπαιδευτικούς σκοπούς αποδεικνύεται εξαιρετικά ωφέλιμη στη διαδικασία μάθησης. Σκοπός της παρούσας διπλωματικής εργασίας είναι η σχεδίαση και ανάπτυξη εφαρμογής, η οποία να ενισχύει το έργο λογοθεραπευτών ή/και εκπαιδευτικών κατά την προσπάθειά τους να αντιμετωπίσουν, βελτιώσουν και τελικά αποκαταστήσουν διαταραχές και δυσκολίες που παρατηρούνται σε μερίδα παιδιών /μαθητών κατά την ομιλία τους. Η εφαρμογή αξιοποιεί το σύνολο των λέξεων της ελληνικής γλώσσας και παρέχει τις ομάδες εκείνων των λέξεων, οι οποίες εμφανίζουν δεδομένη ιδιαιτερότητα ανάλογα με την εκάστοτε διαταραχή. Η εφαρμογή αναζήτησης λεξικού επιτρέπει στους χρήστες της τον εμπλουτισμό του λεξιλογίου που απαιτείται για την εξάσκηση των παιδιών/μαθητών, προς ενδυνάμωση των ικανοτήτων τους στην άρθρωση του λόγου.

Abstract

Modern technological advances are used in many scientific fields. The use of technology, especially for educational purposes is proved to be extremely profitable in the learning process. The aim of this thesis is to design and develop a Java application that enhances the work of speech therapists or /and teachers in their attempt to encounter, improve and eventually restore difficulties observed in the speech of some children /students. The application leverages all the words of the Greek language in order to provide the groups of those words, which have a given feature depending on the disorder. The dictionary search application allows users to enrich the required vocabulary for the practice of children/students, with regard to strengthen their ability of speech articulation.

Περιεχόμενα

Περιεχόμενα Εικόνων

Κεφάλαιο 1

Εισαγωγή

1.1 Κίνητρο Διπλωματικής Εργασίας

Κατά τις τελευταίες δεκαετίες, η κοινωνική διείσδυση των ηλεκτρονικών υπολογιστών (Η/Υ) έχει αυξηθεί με γεωμετρική πρόοδο, με την εξέλιξη της να μοιάζει ατέρμονη. Σε όλους τους τομείς της σύγχρονης κοινωνίας, από την ιατρική και την μηχανική μέχρι την οικονομία και το εμπόριο, έχει εισχωρήσει η χρήση του Η/Υ. Προκύπτουν, διαρκώς, νέες ανάγκες, οι οποίες οδηγούν σε καινοτόμες ιδέες για την κάλυψή τους. Οι ιδέες αυτές υλοποιούνται με πληθώρα λογισμικών εφαρμογών (Application Software) να αναπτύσσονται. Το λογισμικό εφαρμογών, αποτελείται από προγράμματα που έχουν σχεδιαστεί, προκειμένου να βοηθήσουν τους χρήστες στην ολοκλήρωση των εργασιών τους, κατά τρόπο ταχύτερο, ευκολότερο και περισσότερο αποδοτικό. Είθισται να υπάρχει ένας βασικός διαχωρισμός μεταξύ δυο κατηγοριών, των διαδικτυακών εφαρμογών (web applications), όπως η εφαρμογή του ηλεκτρονικού ταχυδρομείου (email) και των πιο παραδοσιακών που διατίθενται για χρήση, χωρίς να απαιτείται σύνδεση σε δίκτυο, όπως οι διάφορες εφαρμογές επεξεργασίας κειμένου.

Η εκπόνηση της παρούσας διπλωματικής εργασίας έχει ως κίνητρο την ανάπτυξη λογισμικού εφαρμογής, εξειδικευμένης αναζήτησης λεξικού, ειδικού σκοπού, για χρήση εκτός σύνδεσης δικτύου. Η ιδέα της εφαρμογής αποσκοπεί στην αξιοποίηση ενός ειδικού, όχι εννοιολογικού χαρακτήρα, λεξικού, το οποίο αποτελείται από το σύνολο των λέξεων της ελληνικής γλώσσας με όλη την πληροφορία της εκάστοτε λέξης (σε ποιο μέρος του λόγου ανήκει, με ποιο τρόπο συλλαβίζεται, προφέρεται, κ.α.), ώστε να πραγματοποιείται η αναζήτηση και τελικά η εύρεση, εκείνων των λέξεων, οι οποίες έχουν, είτε κάποια ιδιαιτερότητα, είτε κάποιο συγκεκριμένο χαρακτηριστικό, ανάλογα με

τις ανάγκες του χρήστη της εφαρμογής. Πιο συγκεκριμένα, εάν ο χρήστης είναι ένας λογοθεραπευτής, η αξιοποίηση της εφαρμογής θα αφορά στην εύρεση λέξεων, ανάλογα με την μαθησιακή δυσκολία που πρέπει να αντιμετωπιστεί, ώστε να πραγματοποιηθεί εξάσκηση, διόρθωση και τελικά αποκατάσταση της γλωσσικής διαταραχής. Αντίστοιχα, εάν ο χρήστης της εφαρμογής είναι ένας φιλόγογος ή ένας γλωσσολόγος, οι αναζητήσεις που ενδιαφέρουν θα αφορούν, πιθανώς, σε λέξεις, οι οποίες θα παρουσιάζουν κάποιο συγκεκριμένο μοτίβο (pattern), λέξεις που αρχίζουν ή/και τελειώνουν με συγκεκριμένο τρόπο, λέξεις που αποτελούν, απλώς, κάποιο δεδομένο μέρος του λόγου.

1.2 Αντικείμενο Διπλωματικής Εργασίας

Στην παρούσα διπλωματική εργασία, αντικείμενο είναι η σχεδίαση των απαιτήσεων της εφαρμογής της αναζήτησης λεξικού και τελικώς η έξυπνη και αποδοτική ανάπτυξή της. Η γενική κατεύθυνση βασίζεται στη διάσπαση του αρχικού στόχου, σε επιμέρους μικρότερους, με εύρεση βέλτιστων λύσεων στα επιμέρους προβλήματα και τελική ολοκληρωμένη ανάπτυξη της εφαρμογής.

Αρχικά, η προσοχή εστιάζεται στην εύρεση κατάλληλου τρόπου διαχείρισης του αρχείου, στο οποίο θα είναι αποθηκευμένο το λεξικό, τόσο για την γρήγορη ανάκτησή του, όσο και την μικρή σε μνήμη αποθήκευσή του.

Εν συνεχεία, με γνώμονα την Νεοελληνική Γραμματική, εντοπίζεται το σύνολο των ιδιοτήτων των λέξεων της αλφαβήτου, με σκοπό να καθοριστούν με σαφήνεια οι δυνατότητες επιλογών αναζήτησης που θα δοθούν στον χρήστη της εφαρμογής.

Μελετάται αποδοτικός τρόπος διαπέρασης του συνόλου των λέξεων. Το αποτέλεσμα της αναζήτησης πρέπει να προκύπτει άμεσα, χωρίς καθυστερήσεις. Παράλληλα να εξασφαλίζεται ότι, πάρα την αυξημένη ταχύτητα, δίνεται ασφαλές - σωστό αποτέλεσμα, χωρίς κίνδυνο σφάλματος.

Τέλος, ερευνάται το καταλληλότερο πρότυπο παρουσίασης και αποθήκευσης της προκύπτουσας αναζήτησης, ώστε να παρέχεται η δυνατότητα στο χρήστη για περαιτέρω επεξεργασία ή/και ανάκτηση των αποτελεσμάτων, αντίστοιχα.

1.3 Διάρθρωση Διπλωματικής Εργασίας

Στην παρούσα διπλωματική εργασία μελετάται, η σχεδίαση και ανάπτυξη εφαρμογής, αναζήτησης λεξικού, με εκπαιδευτικό περιεχόμενο, ενώ εξετάζονται πιθανές μελλοντικές της επεκτάσεις. Η εργασία διαχωρίζεται από επτά θεματικές ενότητες.

Η πρώτη ενότητα είναι εισαγωγική με αναφορές στο κίνητρο και συγκεκριμένα στο αντικείμενο της παρούσας εργασίας.

Στο δεύτερο μέρος παρουσιάζεται αναλυτικά το τεχνολογικό υπόβαθρο που απαιτείται για την υλοποίηση της εφαρμογής και αναλύονται βασικές έννοιες που χρησιμοποιούνται στην σχεδίαση της εφαρμογής.

Ακολουθεί, εκτενής, παρουσίαση της εφαρμογής στην τρίτη θεματική ενότητα. Επεξηγείται, τόσο η λειτουργικότητά της, όσο και η χρησιμότητά της σε διάφορους κλάδους.

Το τέταρτο κεφάλαιο αναφέρεται στις μεθόδους σχεδίασης εφαρφογών, γενικότερα. Πραγματοποιείται, σχολαστική, παρουσίαση του τρόπου σχεδίασης της αναζήτησης λεξικού, συγκεκριμένα και το κεφάλαιο ολοκληρώνεται με αναφορά στο γραφικό περιβάλλον αλληλεπίδρασης χρήστη - εφαρμογής.

Το πέμπτο κεφάλαιο αναλύει την ανάπτυξη της εφαρμογής, με πλήρη εποπτεία των πακέτων και των κλάσεων που αυτή περιλαμβάνει.

Ακολουθεί η έκτη ενότητα, η οποία αφιερώνεται στην υλοποίηση της εφαρμογής, παρουσιάζοντας βασικές έννοιες στην ανάπτυξη κώδικα αντικειμενοστραφούς προγραμματισμού.Η ενότητα συμπληρώνεται από δείγμα του κώδικα που “κρύβεται” πίσω από την υλοποίησή της εφαρμογής και ολοκληρώνεται με την ανάλυση της διεπαφής προγραμματισμού εφαρμογών (API).

Το έβδομο και τελευταίο μέρος αφορά στην εξαγωγή συμπερασμάτων και τις επεκτάσεις που ενδεχομένως προκύψουν μελλοντικά.

Κεφάλαιο 2

Τεχνολογικό Υπόβαθρο

Σε όλα τα στάδια της εργασίας, γίνεται χρήση σύγχρονων τεχνικών και εργαλείων ανάπτυξης. Αναλύεται, λοιπόν, στο τρέχον κεφάλαιο, το τεχνολογικό πλαίσιο μέσα στο οποίο κινείται η παρούσα εργασία. Επιχειρείται μια συνοπτική παρουσίαση του κάθε εργαλείου χωριστά, ξεκινώντας από την γλώσσα προγραμματισμού πάνω στην οποία στήνεται η εφαρμογή και φτάνοντας μέχρι το λογισμικό εκείνο, με το οποίο πραγματοποιείται έλεγχος και αναθεώρηση του αποθηκευμένου, σε αρχείο κειμένου, κώδικα.

2.1 Java

Οι μεγαλύτερες προκλήσεις και πιο συναρπαστικές ευκαιρίες για τους προγραμματιστές λογισμικού σήμερα βρίσκονται στην εκμετάλλευση της δύναμης των δικτύων. Εφαρμογές που δημιουργούνται σήμερα, όποιο κι αν είναι το πεδίο εφαρμογής τους ή το κοινό στο οποίο απευθύνονται, είναι σχεδόν βέβαιο ότι θα τρέξουν σε μηχανές που συνδέονται με ένα παγκόσμιο δίκτυο υπολογιστικών πόρων. Η αυξανόμενη σημασία των δικτύων θέτει συνεχώς νέες απαιτήσεις για επιπλέον εργαλεία και τροφοδοτεί διαρκώς τη ζήτηση για ευφυέστερες εφαρμογές. Θέλουμε λογισμικό που λειτουργεί με συνέπεια, οπουδήποτε, σε οποιαδήποτε πλατφόρμα, και έχει την δυνατότητα συγχρονισμού με άλλες εφαρμογές. Θέλουμε δυναμικές εφαρμογές που εκμεταλλεύονται ένα συνδεδεμένο κόσμο, ώστε να έχουν πρόσβαση σε διαφορετικές πηγές πληροφόρησης. Θέλουμε πραγματικά λογισμικό που μπορεί να επεκταθεί και να αναβαθμιστεί εύκολα. Η Java είναι μια σύγχρονη γλώσσα προγραμματισμού που αντιμετωπίζει αυτά τα καθοριστικά μέτωπα: φορητότητα, ταχύτητα, ασφάλεια. Αυτός είναι ο λόγος που θεωρείται κυρίαρχη γλώσσα στον κόσμο του προγραμματισμού τις τελευταιές δεκαετίες.

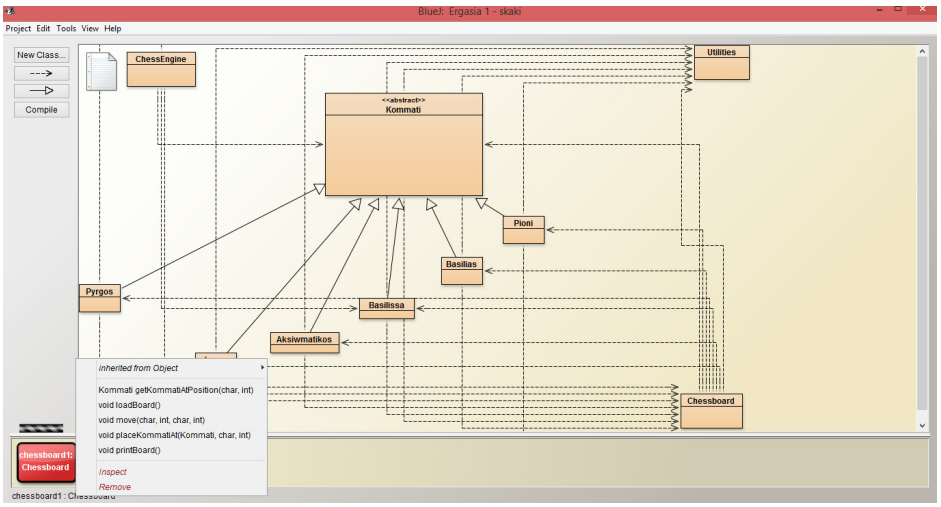
Είναι μια αντικειμενοστρεφής γλώσσα προγραμματισμού που σχεδιάστηκε από την εταιρεία πληροφορικής Sun Microsystems στις αρχές του 1991. Η εταιρεία υπό την καθοδήγηση του James Gosling, ο οποίος και θεωρείται ο "πατέρας" της Java, αναζητούσε το κατάλληλο εργαλείο για να αποτελέσει την πλατφόρμα ανάπτυξης λογισμικού σε μικρο-συσκευές (από έξυπνες οικιακές συσκευές έως πολύπλοκα συστήματα παραγωγής γραφικών). Ο πρώτος μεταγλωττιστής (compiler) της Java ήταν γραμμένος στη γλώσσα C από τον ίδιο τον James Gosling. Το 1994, ο A.Van Hoff ξαναγράφει τον μεταγλωττιστή της γλώσσας σε Java. Η επίσημη εμφάνισή της στη βιομηχανία της πληροφορικής ξεκινά από το Μάρτιο του 1995 όταν η Sun την ανακοίνωσε στο συνέδριο Sun World 1995 και οι πρώτες εταιρείες αρχίζουν να χρησιμοποιούν τη νέα αυτή προγραμματιστική γλώσσα για την ανάπτυξη λογισμικού, ώσπου το Νοεμβρίου του 2006 η Java έγινε πλέον μια γλώσσα ανοιχτού κώδικα όσον αφορά το μεταγλωττιστή (javac) και το πακέτο ανάπτυξης (JDK, Java Development Kit). Τελικά, τον Απριλίου 2010 πραγματοποιείται η εξαγορά της Sun Microsystems από την εταιρεία λογισμικού Oracle Corporation. Από το 2011, η Oracle είναι η δεύτερη μεγαλύτερη εταιρεία παραγωγής λογισμικού, με βάση τα έσοδα, μετά την Microsoft.

Το σημαντικότερο πλεονεκτήματα της Java έναντι των υπολοίπων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος που προσφέρει. Αυτό πρακτικά σημαίνει ότι τα προγράμματα που είναι γραμμένα σε Java τρέχουνε με τον ίδιο τρόπο σε οποιοδήποτε λειτουργικό σύστημα (Windows, Linux) χωρίς να χρειάζεται να ξαναγίνει μεταγλώττιση ή αλλαγή του πηγαίου κώδικα. Για να επιτευχθεί όμως αυτό χρειαζόταν κάποιος τρόπος, έτσι ώστε τα προγράμματα γραμμένα σε Java να μπορούν να είναι «κατανοητά» από κάθε υπολογιστή ανεξάρτητα του είδους επεξεργαστή (Intel x86, IBM) αλλά και λειτουργικού συστήματος (Windows, Unix, Linux, MacOS). Ο λόγος είναι ότι κάθε κεντρική μονάδα επεξεργασίας κατανοεί διαφορετικό κώδικα μηχανής. Ο συμβολικός κώδικας (assembly) που μεταφράζεται και εκτελείται σε Windows είναι διαφορετικός από αυτόν που μεταφράζεται και εκτελείται σε έναν υπολογιστή Linux. Η λύση δόθηκε με την ανάπτυξη της Εικονικής Μηχανής (Virtual Machine, VM). Η εικονική μηχανή της Java (JVM) είναι λογισμικό που εξαρτάται από το λειτουργικό σύστημα που θα χρησιμοποιηθεί, αναλαμβάνει να διαβάσει τον κώδικα Java, να τον μεταγλωττίσει (μέσω του javac) και εν συνεχεία να τον μεταφράσει σε γλώσσα μηχανής. Η εικονική μηχανή, ουσιαστικά είναι υπεύθυνη για την επικοινωνία μεταξύ χρήστη και υπολογιστή και παρέχει σημαντική ασφάλεια, διότι θα αναγνωρίσει και τελικά θα αποτρέψει σε κακόβουλο πρόγραμμμα να τρέξει στο σύστημα.

2.2 Ολοκληρωμένο Περιβάλλον Ανάπτυξης (IDE)

Για την ανάπτυξη κώδικα στην Java, είθισται να χρησιμοποιείται ένα ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment, IDE), το οποίο παρέχει κυρίως επεξεργαστή πηγαίου κώδικα, μεταγλωττιστή (compiler), αποσφαλματωτή (debugger). Υπάρχει μια μεγάλη ποικιλία επιλογής ολοκληρωμένων προγραμμάτων ανάπτυξης (IDEs) για ανάπτυξη Java κώδικα, με τα περισσότερα να είναι δωρεάν και ανοιχτού κώδικα (open source).

Ενδεικτικά παραδείγματα τέτοιων είναι το BlueJ, το οποίο περιλαμβάνει κάποια πρότυπα κώδικα, βιβλιοθήκες, πραγματοποιεί την μεταγλώττιση του πηγαίου κώδικα κ τρέχει την εφαρμογή, ενώ παρέχει ένα ενδιαφέρον γραφικό περιβάλλον για την ανάπτυξη των πακέτων και του κώδικα. Οι κλάσεις παρουσιάζονται ως πλαίσια, ώστε ανάλογα με τον κατασκευαστή, την κληρονομικότητα ή την διασύνδεση να επικοινωνούν μεταξύ τους με ανάλογα βέλη. Είναι ένα ιδιαίτερα βοηθητικό περιβάλλον ανάπτυξης για αρχάριους χρήστες, αφού δίνει την δυνατότητα πλήρους κατανόησης των βασικών αρχών του αντικειμενοστραφούς προγραμματισμού (οπτική αναπαράσταση των αντικειμένων, εύκολη επιλογή των μεθόδων του εκάστοτε αντικειμένου προς εκτέλεση).

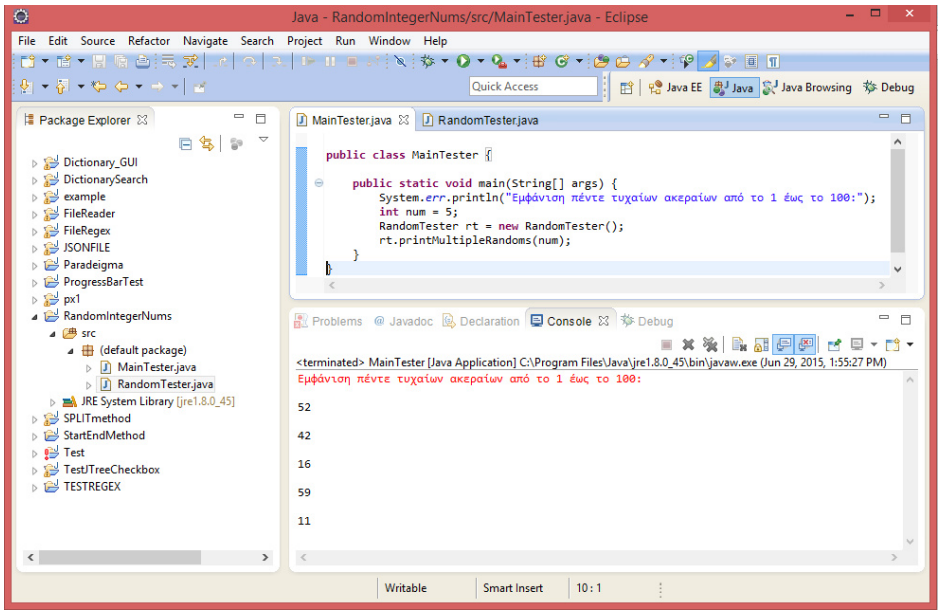


Σχήμα 2.1: Υλοποίηση εφαρμογής για σκάκι σε BlueJ

Το NetBeans, είναι ένα ακόμα ιδιαίτερα χρήσιμο προγραμματιστικό περιβάλλον ανάπτυξης σε Java με δυνατότητα ανάπτυξης εφαρμογών ακόμα και για χρήση σε φορητές συσκευές. Είναι ανοιχτού κώδικα (open source) και ελεύθερο προς εμπορική ή μη χρήση. Ο κώδικας πηγής είναι διαθέσιμος για επα-

ναχρησιμοποίηση κάτω από το Common Development and Distribution License (CDDL). Είναι γραμμένο σε Java, αλλά μπορεί να υποστηρίξει όλες τις γλώσσες προγραμματισμού. Έχει την δυνατότητα να ανιχνεύει λάθη κατά την πληκτρολόγηση και να προτείνει πιθανές λύσεις με ταχύτητα και απλότητα.

Στην παρούσα διπλωματική εργασία, για την ανάπτυξη της εφαρμογής της αναζήτησης λεξικού, έχει χρησιμοποιηθεί το Eclipse. Πρόκειται για ένα προγραμματιστικό περιβάλλον με πρότυπο γραφικό περιβάλλον ανάπτυξης και δυνατότητα επέκτασης του πηγαίου κώδικα, λόγω των πρόσθετων λειτουργιών (plug-ins) που μπορούν να εγκαταστηθούν σε αυτό. Για τον λόγο αυτό, είναι συμβατό με πολλές γλώσσες προγραμματισμού, ωστόσο είναι γραμμένο σε Java και προορίζεται κυρίως για αυτήν. Είναι αυτόματα συνδεδεμένο με την πλατφόρμα Maven, η οποία διαχειρίζεται με εξαιρετικά απλό τρόπο τις όποιες αλλαγές στον πηγαίο κώδικα και τις εξαρτήσεις του με άλλα πακέτα. Παρέχει εργαλεία για την εύκολη ανάπτυξη εφαρμογών γραφικού περιβάλλοντος χρήστη (Graphical User Interface, GUI), όπως το WindowBuilder, ενώ καθιστά εύκολη την επεξεργασία αρχείων τύπου CSV (Comma Seperated Values) και JSON (JavaScript Object Notation).



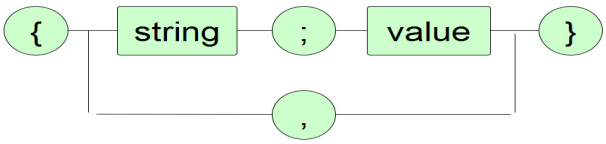
Σχήμα 2.2: Υλοποίηση εφαρμογής για την εμφάνιση πέντε τυχαίων ακέραιων αριθμών σε Eclipse

2.3 Json Αρχεία

Τα αρχεία τύπου JSON (JavaScript Object Notation) είναι μία ανοιχτή πρότυπη μορφή αρχείων που χρησιμοποιεί αναγνώσιμο από τον χρήστη κείμενο για την αποθήκευση και ανταλλαγή δεδομένων που αποτελούνται από ζεύγη χαρακτηριστικών-τιμών (key-value pairs). Χρησιμοποιείται, κυρίως, για τη μετάδοση δεδομένων μεταξύ του εξυπηρετητή (server) και κάποιας διαδικτυακής εφαρμογής (web application), ως εναλλακτική λύση της XML (Extensible Markup Language). Είναι απλή η κωδικοποίηση και αποκωδικοποίηση των δεδομένων και μπορούν να χρησιμοποιηθούν από διάφορες γλώσσες προγραμματισμού, παρά το γεγονός ότι στηρίζονται κυρίως στην JavaScript.

Ο πρώτος που καθόρισε και διέδωσε την μορφή των αρχείων JSON είναι ο Douglas Crockford στις αρχές του 2000. Όρισε ένα σύνολο κανόνων για την ηλεκτρονική κωδικοποίηση κειμένων. Σχεδίασε τη μορφή των αρχείων JSON δίνοντας έμφαση στην απλότητα, τη γενικότητα και τη χρησιμότητα στο Διαδίκτυο. Κατάφερε να ανταποκριθεί στις απαιτήσεις για φόρτωση δεδομένων με ταχύτητα, ασύγχρονα, στο παρασκήνιο κάποιας άλλης λειτουργίας, ώστε να μην παρατηρείται κάποια ουσιαστική καθυστέρηση.

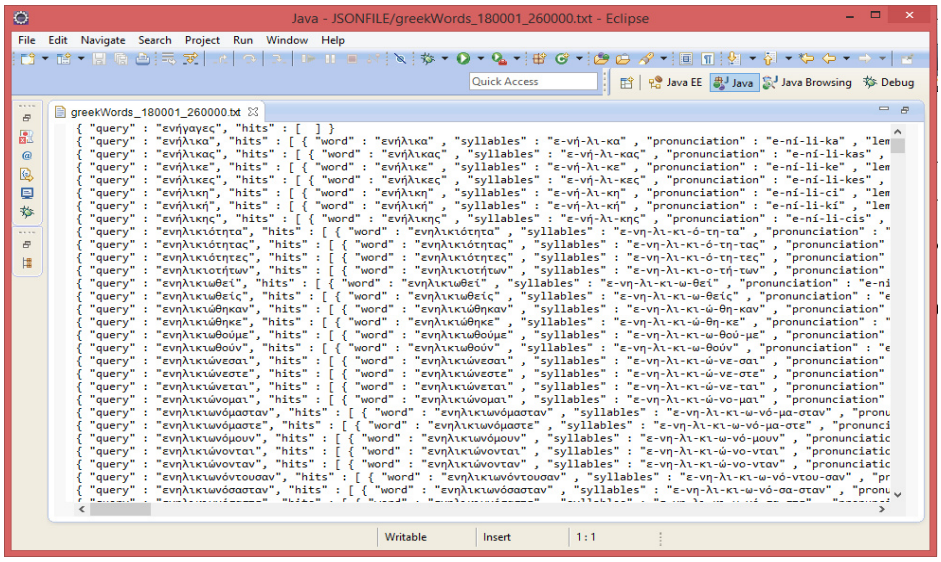
Η βασική κωδικοποίηση που χρησιμοποιείται είναι τα χαρακτηριστικά και οι τιμές των αντικειμένων να βρίσκονται μέσα σε εισαγωγικά και διαχωρισμένα μεταξύ τους με άνω-κάτω τελεία, ενώ κάθε επόμενο ζεύγος χαρακτηριστικού-τιμής διαδέχεται το προηγούμενο έπειτα από κόμμα. Το JSON αντικείμενο βρίσκεται εξ' ολοκλήρου μέσα σε αγκύλες ({“key1” : “value1”, “key2” : “value2”}). Επιπρόσθετες κωδικοποιήσεις είναι διαθέσιμες, τέτοιες ώστε τα αντικείμενα να παίρνουν ανάλογες μορφές, όπως λίστες ή διανύσματα.



Σχήμα 2.3: Βασική μορφή (format) JSON αντικειμένου

Στην εφαρμογή της αναζήτησης λεξικού, συγκεκριμένα, έχουν χρησιμοποιηθεί αρχεία τύπου JSON για την αποθήκευση (save) και άντληση (load) των δεδομένων του λεξικού. Αυτά κωδικοποιούνται (serialization) και αποκωδικοποιούνται (deserialization) κατά περίπτωση, ώστε να επιτευχθεί καλύτερη ταχύτητα εκτέλεσης, ελαχιστοποίηση αποθηκευτικού χώρου στο δίσκο και δέσμευση λιγότερης μνήμης. Τα αποτελέσματα της αναζήτησης που θα προκύψουν αποθηκεύονται σε αρχείο CSV, το οποίο παρουσιάζει το σύνολο των λέ-

ξεων που ταιριάζουν στην αναζήτηση που έχει ζητηθεί από τον χρήστη, με κάθε γραμμή του αρχείου να αντιπροσωπεύει μία λέξη με τις ιδιότητές της, διαχωρισμένες με κόμμα.



Σχήμα 2.4: Παράδειγμα αρχείου JSON

2.4 Κανονικές Εκφράσεις (Regular Expressions)

Μια κανονική έκφραση (regular expression) είναι μια ειδική συμβολοσειρά κειμένου, η οποία περιγράφει ένα μοτίβο αναζήτησης (pattern). Κάθε χαρακτήρας σε μία κανονική έκφραση (regular expression) είναι δυνατόν να αναπαριστά δύο ενδεχόμενα: είτε έναν χαρακτήρα που αντιπροσωπεύει κάποιο καθορισμένο μοτίβο αναζήτησης (metacharacter), είτε έναν χαρακτήρα με την κυριολεκτική του σημασία. Για παράδειγμα, ο χαρακτήρας της τελείας "." μπορεί να καθορίζει, είτε την αναζήτηση οποιουδήποτε συμβόλου σε ένα κείμενο (σε αυτή την περίπτωση λειτουργεί ως metacharacter), είτε την αναζήτηση του συγκεκριμένου συμβόλου με την κυριολεκτική του σημασία, δηλαδή της τελείας, στο κείμενο. Αναλυτικότερα, ορίζονται μια σειρά από σύμβολα με προκαθορισμένες λειτουργίες αναζήτησης και όταν χρειάζεται να γίνει κυριολεκτική χρήση των συμβόλων αυτών, τότε αυτό δηλώνεται με την εισαγωγή του συμβόλου "\". Ακολουθούν ενδεικτικά παραδείγματα χαρακτήρων, των οποίων η χρήση σε μια κανονική έκφραση αναλογεί σε κάποια αυστηρά καθορισμένη ενέργεια:

ΣΥΜΒΟΛΙΣΜΟΣ ΚΑΝΟΝΙΚΩΝ ΕΚΦΡΑΣΕΩΝ	ΣΗΜΑΣΙΑ ΤΟΥ ΣΥΜΒΟΛΙΣΜΟΥ ΣΤΗΝ ΑΝΑΖΗΤΗΣΗ ΚΕΙΜΕΝΟΥ
ΑΝΑΓΩΓΗ ΣΕ ΧΑΡΑΚΤΗΡΕΣ	
.	οποιοσδήποτε χαρακτήρας
[α|ε]	είτε ο χαρακτήρας "α", είτε ο χαρακτήρας "ε"
[^ω]	όχι ο χαρακτήρας "ω"
ΑΝΑΓΩΓΗ ΣΕ ΠΛΗΘΟΣ ΕΜΦΑΝΙΣΗΣ	
?	1 φορά εμφάνισης, προαιρετική
*	οσεσδήποτε φορές εμφάνισης, όλες προαιτερικές
+	τουλάχιστον 1 φορά εμφάνισης, οι επιπρόσθετες προαιρετικές
{min, max}	τουλάχιστον "min" φορές, αλλά όχι περισσότερες από "max" φορές
ΑΝΑΓΩΓΗ ΣΕ ΘΕΣΗ ΕΜΦΑΝΙΣΗΣ	
^	στην αρχή της γραμμής
$	στο τέλος της γραμμής
\<	στην αρχή της λέξης
/>	στο τέλος της λέξης

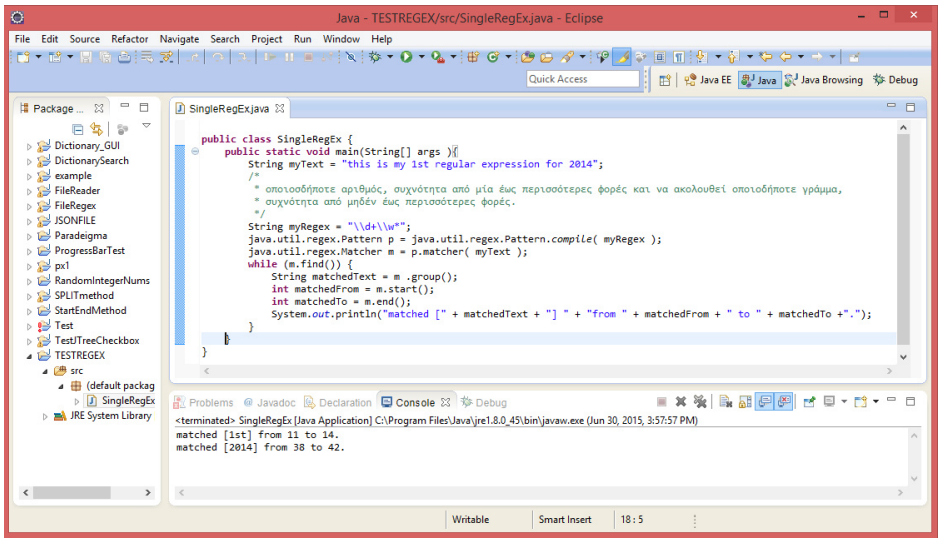
Σχήμα 2.5: Παράδειγμα χρήσης των regular expressions

Σε συστήματα που η συλλογή, εγγραφή, ανάκτηση, επεξεργασία, αποθήκευση και ανάλυση πληροφοριών είναι απαραίτητα, τότε οι κανονικές εκφράσεις (regular expressions) αποτελούν ένα πραγματικά πολύ χρήσιμο εργαλείο. Είναι τόσο απλές στην σύνταξη του μοτίβου, όσο και γρήγορες στην αναζήτηση του κειμένου. Ωστόσο, χρειάζεται προσοχή στην χρήση τους, ώστε πραγματικά το όφελος να είναι μεγάλο. Πληθώρα γλωσσών προγραμματισμού έχουν εξελιχθεί, με σκοπό να παρέχουν στους χρήστες τους τις δυνατότητες των κανονικών εκφράσεων (regular expressions). Κρύβονται πίσω από μηχανές αναζήτησης του διαδικτύου, από προγράμματα επεξεργασίας κειμένου, απλούς κειμενογράφους και αλλού.

Για κάθε αναζήτηση είναι πιθανό να υπάρχουν διαφορετικοί τρόποι να γραφεί μία κανονική έκφραση (regular expression). Επιπλέον, υπάρχουν τουλάχιστον τρεις διαφορετικοί αλγόριθμοι για να δώσουν απάντηση στο αν ένα μοτίβο εντοπίζεται και αν ναι με ποιο τρόπο μέσα σε ένα κείμενο. Ο παλιότερος και πιο γρήγορος είναι η μετατροπή μη ντετερμινιστικού αυτόματου (NFA) σε ντετερμινιστικό (DFA), το οποίο τρέχει το αλφαριθμητικό της εισόδου για κάθε ένα σύμβολο διαδοχικά. Η κατασκευή ντετερμινιστικού αυτόματου (DFA) για

μια κανονική έκφραση (regular expression) μεγέθους m δίνει πολυπλοκότητα $O(2^m)$, αλλά για να τρέξει σε κείμενο μεγέθους n χρειάζεται χρόνο $O(n)$. Μία εναλλακτική προσέγγιση είναι η κατασκευή ντετερμινιστικού αυτόματου (DFA) για κάθε ένα σύμβολο της εισόδου χωριστά, απορρίπτοντας σε κάθε επόμενο σύμβολο το προηγούμενο ντετερμινιστικό αυτόματο (DFA). Το αποτέλεσμα είναι από την μία να μειώνεται το εκθετικό κόστος της κατασκευής του DFA για την κανονική έκφραση, από την άλλη το κόστος της αναζήτησης στο κείμενο να αυξάνει σε $O(n * m)$. Ένας τρίτος αλγόριθμος πραγματοποιεί τον έλεγχο της ύπαρξης του μοτίβου στο κείμενο προς τα πίσω (backtracking) σε εκθετικό χρόνο χειρότερης περίπτωσης. Συχνά, γίνεται χρήση συνδυασμού των παραπάνω, προς εκμετάλλευση των πλεονεκτημάτων του κάθε αλγορίθμου ανάλογα την περίπτωση αναζήτησης [1].

Οι κανονικές εκφράσεις ξεκινάνε από το έργο του Αμερικανού μαθηματικού Stephen Kleene, ο οποίος υπήρξε ένα από τα πιο σημαίνοντα πρόσωπα στην ανάπτυξη της θεωρητικής επιστήμης των υπολογιστών. Ανέπτυξε τις κανονικές εκφράσεις ως συμβολισμό, για να περιγράψουν αυτό που αποκάλεσε "η άλγεβρα των κανονικών συνόλων". Το έργο του βρήκε το δρόμο του σε ορισμένες από τις πρώτες προσπάθειες ανάπτυξης υπολογιστικών αλγορίθμων αναζήτησης και από εκεί, σε μερικά από τα πρώτα εργαλεία διαχείρισης και αναζήτησης κειμένου στην πλατφόρμα Unix. Στην επιστήμη των υπολογιστών, το σύμβολο "*" έχει μείνει επίσημα γνωστό ως το "Αστέρι Κλέινι" (Kleene star).



Σχήμα 2.6: Παράδειγμα αναζήτησης μοτίβου σε κείμενο με χρήση των regular expressions

2.5 Σύστημα Ελέγχου και Αναθεώρησης, Bitbucket

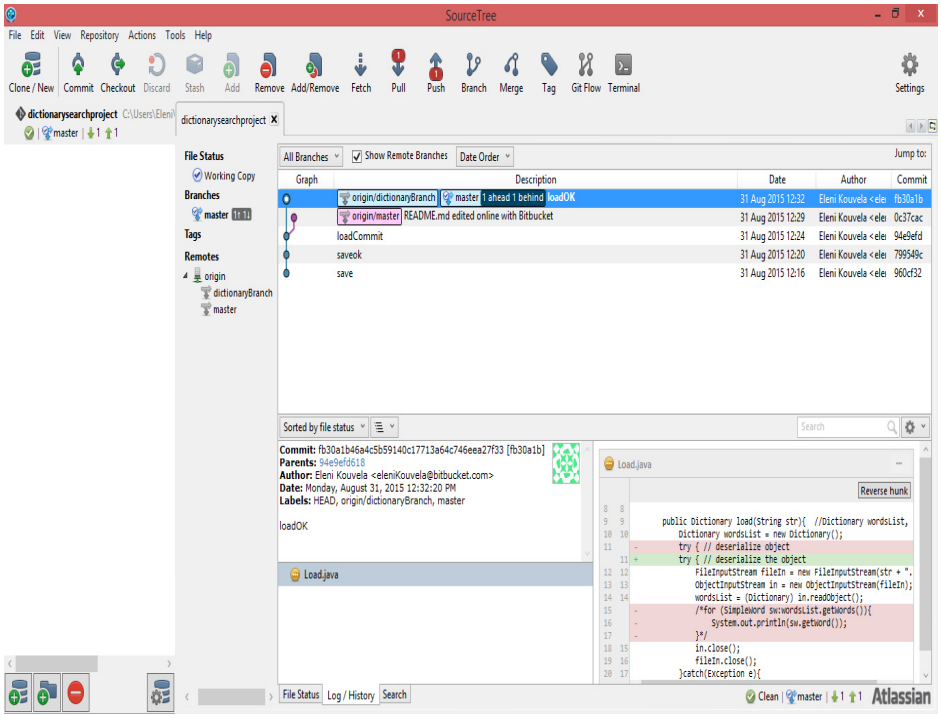
Κάθε είδους πρακτική που παρακολουθεί και παρέχει έλεγχο επί των αλλαγών σε έγγραφα, προγράμματα ηλεκτρονικών υπολογιστών, μεγάλες ιστοσελίδες ή άλλες συλλογές πληροφοριών, ονομάζεται Σύστημα Ελέγχου και Αναθεώρησης. Οι αλλαγές, συνήθως, αναγνωρίζονται με έναν κωδικό, που ονομάζεται "αριθμός αναθεώρησης", "επίπεδο αναθεώρησης", ή απλά "αναθεώρηση". Κάθε αναθεώρηση συνδέεται με μία χρονοσήμανση και το πρόσωπο που κάνει την αλλαγή. Οι αναθεωρήσεις μπορούν να συγκριθούν μεταξύ τους, να αποκατασταθούν, και σε περιπτώσεις με συγκεκριμένους τύπους αρχείων, να συγχωνευθούν. Τα συστήματα ελέγχου λειτουργούν, κυρίως, ως αυτόνομες εφαρμογές, αλλά ο έλεγχος αναθεώρησης είναι, επίσης, ενσωματωμένος σε διάφορους τύπους λογισμικού, όπως σε επεξεργαστές κειμένου. Τα πιο πολύπλοκα συστήματα ελέγχου και αναθεώρησης είναι εκείνα που χρησιμοποιούνται στην ανάπτυξη λογισμικού, όπου μια ομάδα ανθρώπων μπορεί να αλλάξει τα ίδια αρχεία.

Το Bitbucket, συγκεκριμένα, είναι σύστημα ελέγχου για την ανάπτυξη λογισμικού, γραμμένο σε Python, με σκοπό την ταχεία και ασφαλή διαχείριση και μετάδοση των αλλαγών σε αρχεία πηγαίου κώδικα. Κάθε χώρος εργασίας του είναι ένας αποθηκευτικός χώρος (repository) αρχείων καταγραφής κώδικα, με πλήρες ιστορικό των αλλαγών στον κώδικα, ανεξάρτητα από την πρόσβαση στο δίκτυο ή του κεντρικού διακομιστή. Η βασική έκδοση του Bitbucket προσφέρει δωρεάν λογαριασμούς με απεριόριστο αριθμό ιδιωτικών χώρων αποθήκευσης, ενώ επιτρέπει στους ιδιοκτήτες των λογαριασμών να προσθέσουν έως και πέντε επιπλέον χρήστες. Η χρήση του αποσκοπεί στην ασφαλή διαχείριση μεγάλων προγραμματιστικών έργων (projects), καθώς και στην έγκαιρη επικοινωνία μεταξύ ομάδων που εργάζονται πάνω στο ίδιο έργο. Πιο συγκεκριμένα, επιτρέπει να επανέλθει ένα αρχείο κώδικα σε μια προηγούμενη αναθεώρηση, η οποία είναι ζωτικής σημασίας για τους χρήστες, ενώ επιπρόσθετα επιτρέπει να παρακολουθεί ο ένας τις αλλαγές του άλλου, σε μια ομάδα, ώστε τελικά να διορθώνονται λάθη, και να κατοχυρώνεται η καταλληλότερη λύση σε ένα πρόβλημα.

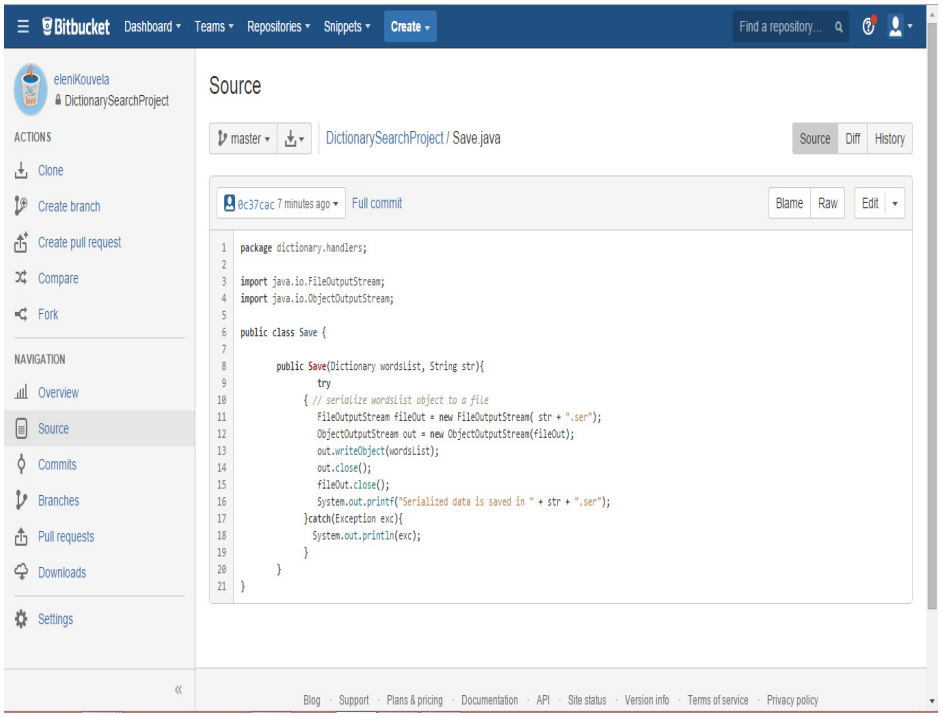
Δημιουργείται, αρχικά, ο απομακρυσμένος (remote) αποθηκευτικός χώρος των αρχείων, ο οποίος κλωνοποιείται σε ένα κοινόχρηστο σύστημα αρχείων, όπου επιτρέπεται η προσπέλασή τους από όλους τους χρήστες του λογαριασμού. Εν συνεχεία, κατασκευάζονται εικονικά κλαδιά (branches), των οποίων οι κόμβοι αντιπροσωπεύνουν τις νέες πληροφορίες (αρχεία πηγαίου κώδικα) που είναι πιθανόν να εισάγουν, με την εντολή "commit", οι χρήστες. Με την εισαγωγή νέου κόμβου ή/και με τις αλλαγές σε υπάρχον αρχείο κώδικα είναι αναγκαίο να ενημερωθούν όλοι οι χρήστες, με την εντολή "push", από εκείνον

που πραγματοποιεί τις εισαγωγές/αλλαγές, αλλά οφείλουν τα υπόλοιπα μέλη της ομάδας να πραγματοποιούν, ακρετά συχνά, αιτήματα ενημέρωσης, με την εντολή “pull”, ώστε να καταχωρούνται οι νέες πληροφορίες κώδικα στους ίδιους. Οι αλλαγές ή διαγραφές κώδικα επισημαίνονται με κόκκινο χρώμα στον κώδικα, ενώ η προσθήκη κώδικα ή σχολίου υποδεικνύεται με πράσινο χρώμα.

Ακολουθούν χαρακτηριστικά στιγμιότυπα, τόσο με ενδεικτικές προσθήκες και διαγραφές σε αρχεία του γραφήματος ενός project, με ταυτόχρονη προτροπή στις εντολές “push” και “pull”, ώστε τόσο να καταχωρηθούν, όσο και να γίνουν αντιληπτές από τους λοιπούς χρήστες, οι νέες πληροφορίες, όσο και με το γραφικό περιβάλλον διεπαφής του Bitbucket κατά την ανάκτηση πηγαίου κώδικα από το ιστορικό:



Σχήμα 2.7: Στιγμιότυπο SourceTree



Σχήμα 2.8: Στιγμιότυπο Bitbucket

Κεφάλαιο 3

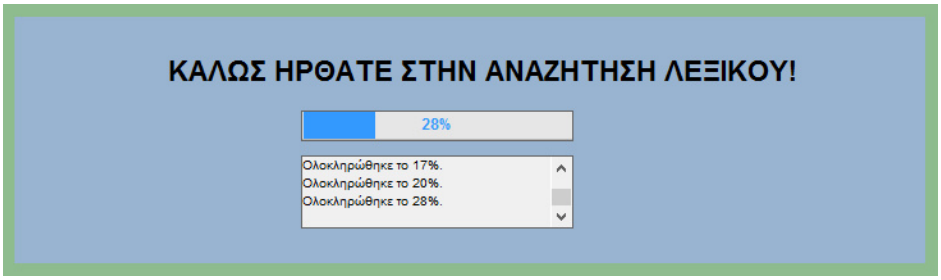
Παρουσίαση Εφαρμογής

Στο παρόν κεφάλαιο, προς διευκόλυνση του αναγνώστη, πραγματοποιείται μια πρώτη, αναλυτική, γνωριμία με την εφαρμογή της αναζήτησης λεξικού, ώστε καθ' όλη τη διάρκεια μελέτης της εργασίας, να υπάρχει πλήρης εποπτεία του συνόλου των λειτουργιών της εφαρμογής. Ταυτόχρονα, παρουσιάζονται λεπτομερώς οι περιπτώσεις χρήσης της εφαρμογής στους τομείς ενδιαφέροντος και δίνονται εκτενή παραδείγματα χρήσης ανάλογα με τον τομέα.

3.1 Πρώτη Γνωριμία με την Εφαρμογή

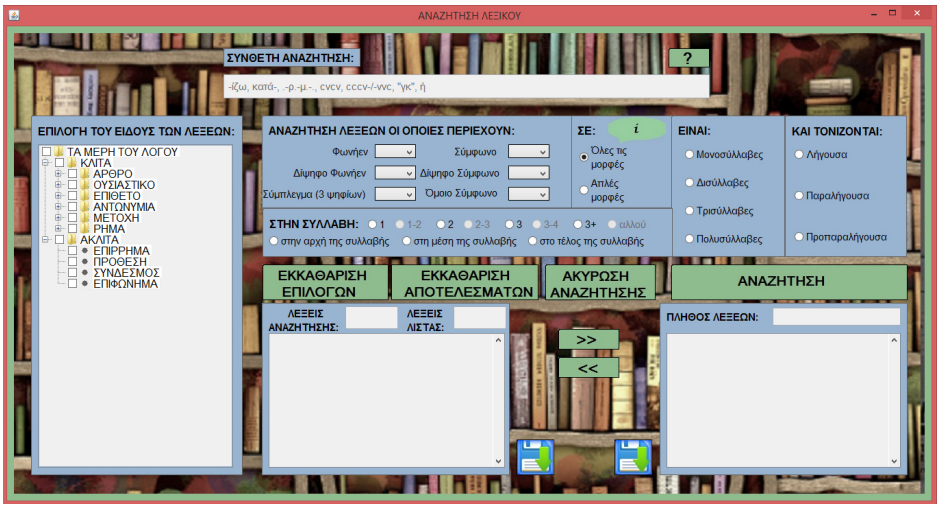
Ως ένα εγχειρίδιο της εφαρμογής, παρατίθενται, παρακάτω, όλα εκείνα τα παράθυρα, τα οποία θα εμφανιστούν στον χρήστη, από την εκκίνηση κιόλας της εφαρμογής.

1. **Αρχική**, μερικών δευτερολέπτων, **οθόνη** κατά την εκκίνηση της εφαρμογής.



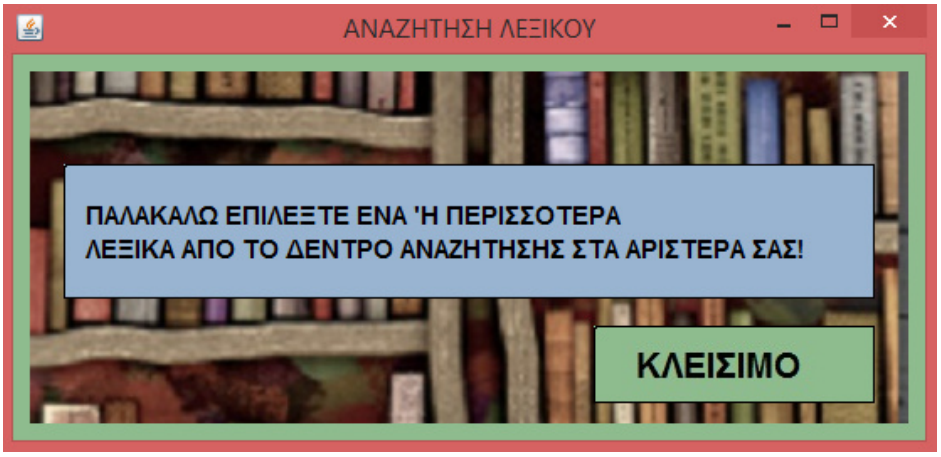
Σχήμα 3.1: Οθόνη Εκκίνησης Εφαρμογής

2. **Κύρια οθόνη επιλογής παραμέτρων αναζήτησης**, στην οποία παρουσιάζονται στον χρήστη τα κριτήρια όλων των δυνατών αναζητήσεων που θα μπορούσε να πραγματοποιήσει και με επιλογή του κάθε επιμέρους συστατικού της οθόνης πραγματοποιείται η ανάλογη ενέργεια.



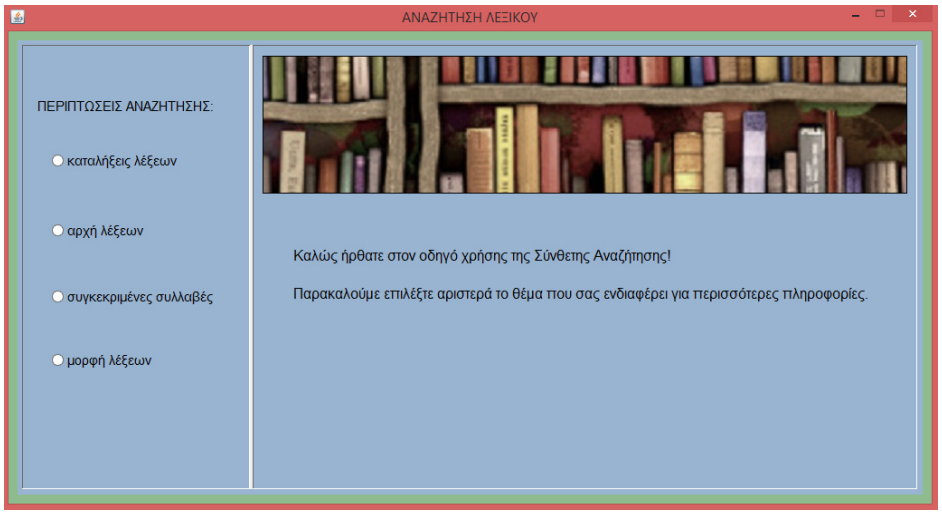
Σχήμα 3.2: Κύρια Οθόνη Εφαρμογής

3. **Βοηθητική οθόνη ενημέρωσης σφάλματος**, με την οποία ο χρήστης ενημερώνεται για ελλιπή επιλογή από το δέντρο αναζήτησης της εφαρμογής.



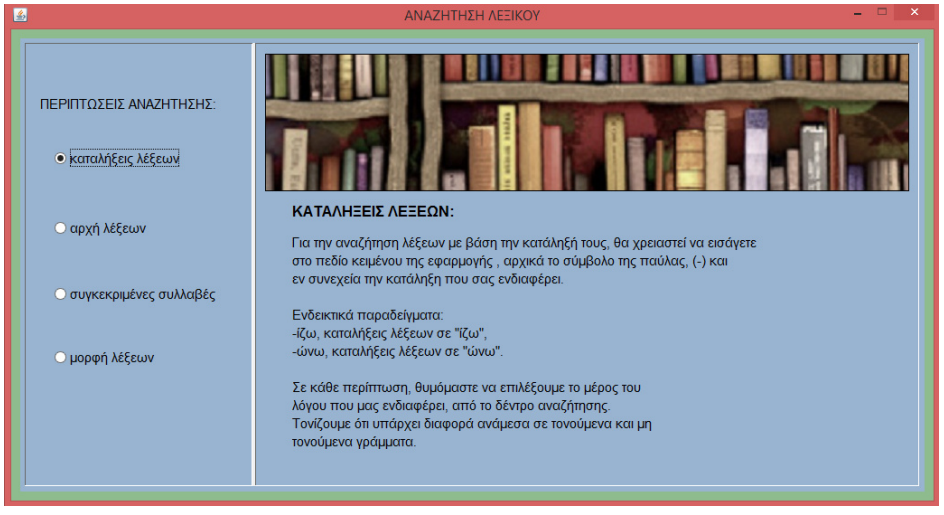
Σχήμα 3.3: Οθόνη Ενημέρωσης Σφάλματος Εφαρμογής

4. **Βοηθητική οθόνη σύνθετης αναζήτησης**, όπου ο χρήστης μπορεί να ενημερωθεί για τις επιλογές που παρέχει η σύνθετη αναζήτηση της εφαρμογής.



Σχήμα 3.4: Οθόνη Βοήθειας Σύνθετης Αναζήτησης Εφαρμογής

5. **Οθόνη ενημέρωσης 1ης περίπτωσης σύνθετης αναζήτησης**, η οποία αφορά σε αναζήτηση λέξεων με γνώμονα την κατάληξη τους.

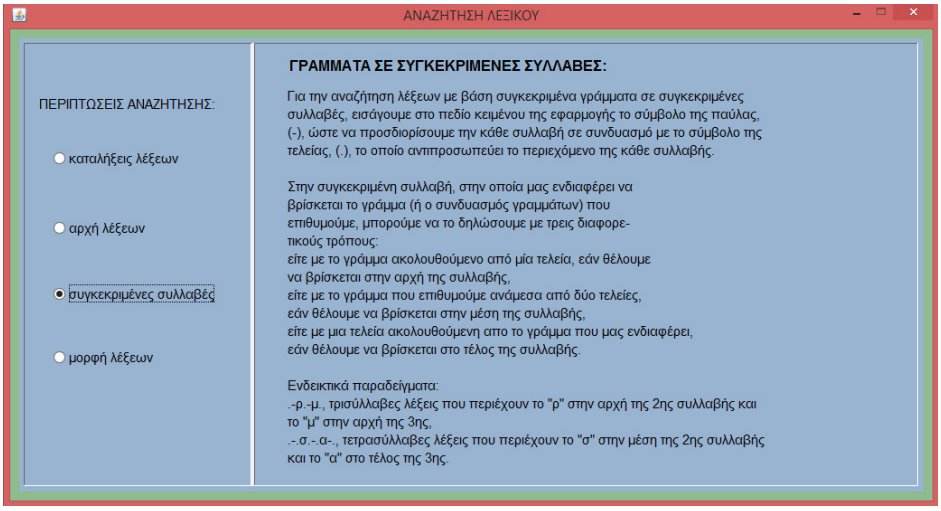


Σχήμα 3.5: Οθόνη Βοήθειας Σύνθετης Αναζήτησης Εφαρμογής (1η περίπτωση)

6. **Οθόνη ενημέρωσης 2ης περίπτωσης σύνθετης αναζήτησης**, με επεξήγηση σχετικά με τον τρόπο αναζήτησης βάσει της αρχής των λέξεων.

Σχήμα 3.6: Οθόνη Βοήθειας Σύνθετης Αναζήτησης Εφαρμογής (2η περίπτωση)

7. **Οθόνη ενημέρωσης 3ης περίπτωσης σύνθετης αναζήτησης**, με αναφορές σε αναζήτηση συγκεκριμένων φθόγγων σε συγκεκριμένες συλλαβές.



Σχήμα 3.7: Οθόνη Βοήθειας Σύνθετης Αναζήτησης Εφαρμογής (3η περίπτωση)

8. **Οθόνη ενημέρωσης 4ης περίπτωσης σύνθετης αναζήτησης**, για λέξεις με δεδομένο μοτίβο.

Σχήμα 3.8: Οθόνη Βοήθειας Σύνθετης Αναζήτησης Εφαρμογής (4η περίπτωση)

3.2 Περιπτώσεις Χρήσης Εφαρμογής

Τα τελευταία χρόνια εμφανίζεται ιδιαίτερο ενδιαφέρον για τα χαρακτηριστικά και τη διαχείριση των μαθησιακών δυσκολιών. Οι μαθησιακές δυσκολίες περιγράφουν ένα σύνολο διαταραχών που μειώνουν την ικανότητα ενός ατόμου να επικοινωνήσει ή να μάθει. Εκδηλώνονται με σημαντικές δυσκολίες στις ικανότητες ακρόασης, ομιλίας, ανάγνωσης, γραφής, συλλογισμού ή μαθηματικών ικανοτήτων. Οι διαταραχές αυτές αποδίδονται σε δυσλειτουργία του κεντρικού νευρικού συστήματος και μπορεί να υπάρχουν σε όλη τη διάρκεια της ζωής του ατόμου. Η Λογοθεραπεία είναι ο επιστημονικός κλάδος, ο οποίος έχει ως γνωστικό αντικείμενο τη μελέτη, έρευνα και εφαρμογή επιστημονικών γνώσεων, γύρω από την ανθρώπινη επικοινωνία, ομιλία, λόγο (προφορικό, γραπτό), μη λεκτική επικοινωνία και τις διαταραχές αυτών. Σκοπός της Λογοθεραπείας είναι η πρόληψη, η διάγνωση, και η αποκατάσταση των διαταραχών του λόγου, της ομιλίας, της φωνής και της επικοινωνίας. Στα πλαίσια της θεραπευτικής αγωγής παρέχεται, πλέον, εκπαιδευτικό λογισμικό για εξάσκηση, τόσο στον κλινικό χώρο, όσο και στο σπίτι, στοχεύοντας στη μεγιστοποίηση των ικανοτήτων-δεξιοτήτων του ατόμου με μαθησιακές δυσκολίες. Η εφαρμογή της αναζήτησης λεξικού σχεδιάστηκε, ώστε να αποτελέσει ένα ιδιαίτερα χρήσιμο βοήθημα των λογοθεραπευτών. Αξιοποιεί τις Τεχνολο-

γίες Πληροφοριών και Επικοινωνίας [1] με αλληλεπίδραση Η/Υ - χρήστη και δυνατότητα προσαρμογής στην εκάστοτε μαθησιακή δυσκολία, έτσι ώστε να επιτυγχάνεται το επιθυμητό αποτέλεσμα. Με απλό και δημιουργικό τρόπο παρουσιάζει άμεσο οπτικό αποτέλεσμα και επιτρέπει την αποθήκευση και μεταφορά του αποτελέσματος σε άλλα μέσα και υλικά.

Μελετώντας ένα απλό σενάριο χρήσης της εφαρμογής της αναζήτησης λεξικού στην περίπτωση που παιδί παρουσιάζει δυσκολία στην άρθρωση του γράμματος “ρ” της αλφαβήτου, θα πραγματοποιηθεί αναζήτηση δισύλλαβων λέξεων, οι οποίες περιέχουν το συγκεκριμένο γράμμα, ώστε να γίνει εξάσκηση με το σύνολο των λέξεων που κατέχουν την συγκεκριμένη ιδιότητα:

ΧΡΗΣΤΗΣ	ΕΦΑΡΜΟΓΗ
1. εκκίνηση εφαρμογής	
2. εμφάνιση αρχικής οθόνης	
3. επιλογή του είδους των λέξεων που ενδιαφέρει (ουσιαστικό, ρήμα, κλπ)	
4. επιλογή του γράμματος "ρ" να περιέχεται οπουδήποτε στην λέξη	
5. επιλογή δισύλλαβων λέξεων	
6. επιλογή όλων των περιπτώσεων τονισμού	
	7. δημιουργία της κανονικής έκφρασης (regular expression)
	8. αναζήτηση λέξεων με βάση τα επιλεγμένα κριτήρια
	9. εμφάνιση αποτελεσμάτων
10. επιλογή αποθήκευσης	
	11. υλοποίηση αποθήκευσης

Σχήμα 3.9: Σενάριο Χρήσης 1

Επιπρόσθετο σενάριο θα μπορούσε να αποτελεί η χρήση της εφαρμογής από επιστήμονες Γλωσσολογίας. Η Γλωσσολογία είναι σχετικά νέα επιστήμη, τουλάχιστον με τη μορφή που έχει λάβει τα τελευταία χρόνια. Επιδιώκει να απαντήσει σε θεμελιώδη ερωτήματα, όπως λόγου χάρη: τι γνωρίζουμε όταν γνωρίζουμε μια γλώσσα, πώς κατακτάται αυτή η γνώση, πώς χρησιμοποιείται. Εξετάζει τη γλώσσα, ως μέρος της ανθρώπινης συμπεριφοράς, τόσο από ψυχολογική, όσο και από κοινωνική και πολιτισμική άποψη και αναζητά να καθορίσει ποια χαρακτηριστικά είναι μοναδικά στην ανθρώπινη γλώσσα, ποια είναι καθολικά (σε όλες τις γλώσσες του κόσμου), πώς διαφοροποιούνται οι επιμέρους γλώσσες, πώς και γιατί αλλάζουν, “πεθαίνουν” ή “γεννιούνται”. Πιο συγκεκριμένα, σε θεωρητικό επίπεδο, η Γλωσσολογία μελετά τη γλώσσα

[1] Η Τεχνολογία Πληροφοριών και Επικοινωνίας (ΤΠΕ, αγγλ. IT ή ICT) είναι το σύνολο των επαγγελμάτων, τα οποία σχετίζονται με τη μελέτη, σχεδίαση, ανάπτυξη, υλοποίηση, συντήρηση και διαχείριση υπολογιστικών πληροφοριακών συστημάτων, κυρίως όσον αφορά εφαρμογές λογισμικού και υλικό υπολογιστών, με στόχο την παραγωγή, αποθήκευση, διαχείριση και μετάδοση πληροφοριών κάθε τύπου.

βάσει των επιπέδων ανάλυσής της, δηλαδή το φωνολογικό σύστημα, την άρθρωση και πρόσληψη των φθόγγων (φωνητική), το σχηματισμό λέξεων (μορφολογία), φράσεων και προτάσεων (σύνταξη), τη σημασία των γλωσσικών εκφράσεων (σημασιολογία), καθώς και τη χρήση της γλώσσας (πραγματολογία).

Αναλύοντας ένα τυπικό σενάριο χρήσης της εφαρμογής από γλωσσολόγους, το οποίο θα μπορούσε να αφορά γενικότερα φιλόλογους, θα μπορούσε να είναι η αναζήτηση λέξεων, οι οποίες παρουσιάζουν συγκεκριμένο μοτίβο (pattern), π.χ. λέξεις που έχουν την μορφή “σύμφωνο - φωνήεν - σύμφωνο - φωνήεν - σύμφωνο”:

ΧΡΗΣΤΗΣ	ΕΦΑΡΜΟΓΗ
1. εκκίνηση εφαρμογής	
2. εμφάνιση αρχικής οθόνης	
3. επιλογή του είδους των λέξεων που ενδιαφέρει (ουσιαστικό, ρήμα, κλπ)	
4. επιλογή σύνθετης αναζήτησης με μοτίβο "cvcvc"	
	5. δημιουργία της κανονικής έκφρασης (regular expression)
	6. αναζήτηση λέξεων με βάση τα επιλεγμένα κριτήρια
	7. εμφάνιση αποτελεσμάτων
8. επιλογή αποθήκευσης	
	9. υλοποίηση αποθήκευσης

Σχήμα 3.10: Σενάριο Χρήσης 2

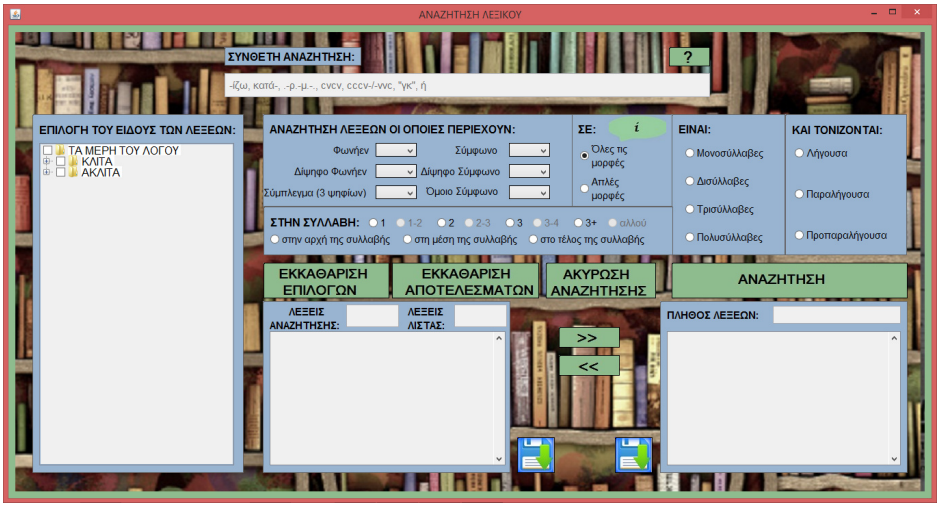
Τέλος, είναι δυνατή η χρήση της εφαρμογής από μαθητές, ως εκπαιδευτικό εργαλείο εκμάθησης παραγώγων λέξεων, καταλήξεων λέξεων, ορθογραφίας ή συλλαβισμού. Σε μια τέτοια περίπτωση, θα εξερευνούνταν, λόγου χάρη, ρήματα μέσω της σύνθετης αναζήτησης με μοτίβο “προ-” για ρήματα που ξεκινάνε με την πρόθεση “προ” ή με μοτίβο “-ίζω” για ρήματα με καταλήξη “ίζω”:

ΧΡΗΣΤΗΣ	ΕΦΑΡΜΟΓΗ
1. εκκίνηση εφαρμογής	
2. εμφάνιση αρχικής οθόνης	
3. επιλογή ρημάτων	
4. επιλογή σύνθετης αναζήτησης με μοτίβο "προ- ή -ίζω"	
	5. δημιουργία της κανονικής έκφρασης (regular expression)
	6. αναζήτηση λέξεων με βάση τα επιλεγμένα κριτήρια
	7. εμφάνιση αποτελεσμάτων
8. επιλογή αποθήκευσης	
	9. υλοποίηση αποθήκευσης

Σχήμα 3.11: Σενάριο Χρήσης 3

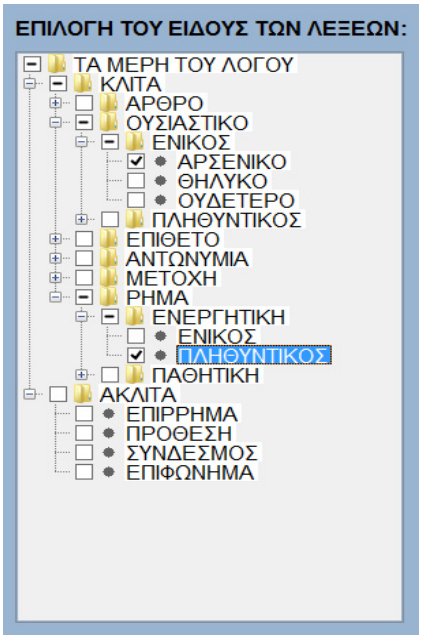
3.3 Παραδείγματα Χρήσης Εφαρμογής

Για την κατανόηση και την χρησιμότητα της εφαρμογής μελετώνται παραδείγματα των δυο βασικών λειτουργιών αναζήτησης: της απλής και της σύνθετης. Το αρχικό παράθυρο που θα εμφανιστεί στον χρήστη για αλληλεπίδραση με τις επιλογές της εφαρμογής είναι, όπως έχουμε παρουσιάσει, το ακόλουθο:



Σχήμα 3.12: Αρχική οθόνη αλληλεπίδρασης χρήστη-εφαρμογής

Στο σημείο αυτό, ο χρήστης έχει την δυνατότητα να περιηγηθεί στις λειτουργίες και να ορίσει τις επιθυμητές επιλογές του. Εστιάζοντας στο δέντρο αναζήτησης, είναι σημαντικό να γίνει επιλογή ενός ή περισσοτέρων κατηγοριών, οι οποίες αντιπροσωπεύουν τα λεξικά στα οποία θα πραγματοποιηθεί η αναζήτηση. Διαφορετικά, με την εντολή “ΑΝΑΖΗΤΗΣΗ”, εμφανίζεται σχετικό παράθυρο σφάλματος (σχήμα 3.3). Οι επιλογές δύναται να είναι εντελώς αόριστες, διαλέγοντας η αναζήτηση να αφορά σε όλα “ΤΑ ΜΕΡΗ ΤΟΥ ΛΟΓΟΥ”, έως πολύ συγκεκριμένες, δίνοντας περιορισμένες προτιμήσεις.

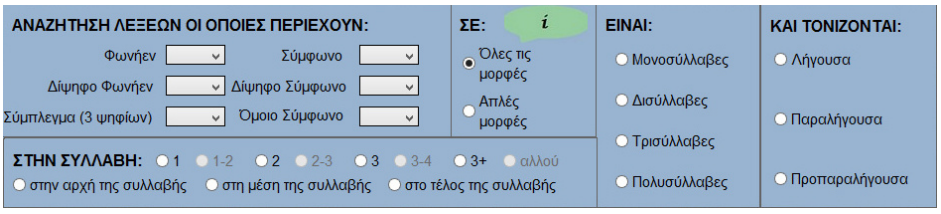


Σχήμα 3.13: Επιλογές από το δέντρο αναζήτησης

Με δεδομένη την επιλογή των επιθυμητών λεξικών, πραγματοποιείται, πλέον, μετάβαση σε περαιτέρω ενέργειες.

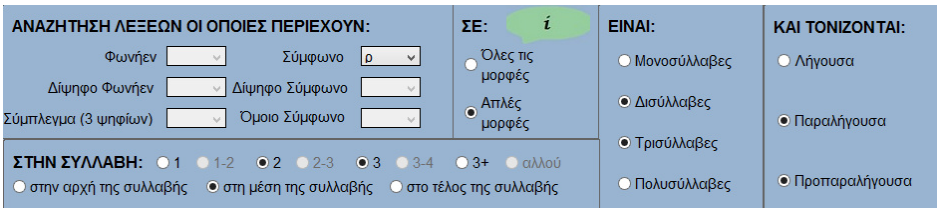
3.3.1 Παραδείγματα Απλής Αναζήτησης

Για χρήση της απλής αναζήτησης, ο χρήστης επικεντρώνεται στις ακόλουθες επιλογές:



Σχήμα 3.14: Απλή αναζήτηση

Σύμφωνα με τις ανάγκες του, γίνεται επιλογή των παραμέτρων αναζήτησης:



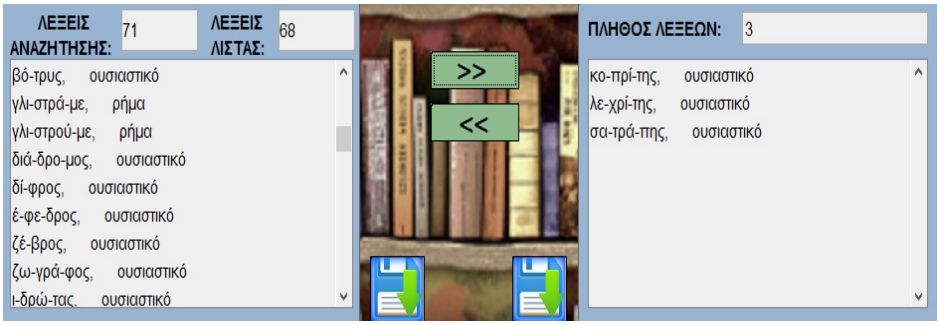
Σχήμα 3.15: Πιθανές επιλογές απλής αναζήτησης

Τα αποτελέσματα της αναζήτησης παρουσιάζονται αλφαβητικά στην αριστερή λίστα, συνοδευόμενα από το πλήθος των λέξεων που βρέθηκαν να πληρούν τα δεδομένα κριτήρια. Πρόκειται για απλές μορφές[2], δισύλλαβων και τρισύλλαβων ουσιαστικών, ενικού αριθμού, γένους αρσενικού και ρημάτων, ενεργητικής φωνής, πληθυντικού αριθμού, που περιέχουν τον φθόγγο “ρ”, στη μέση της 2ης και 3ης συλλαβής, ενώ τονίζονται στην παραλήγουσα και προπαραλήγουσα.

Σχήμα 3.16: Παρουσίαση αποτελεσμάτων απλής αναζήτησης

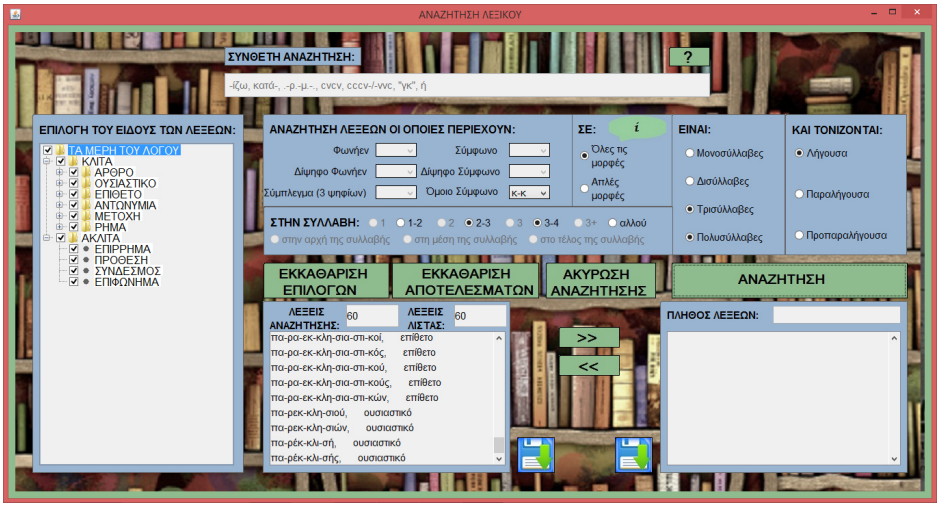
[2]Η απλή μορφή αφορά σε λέξεις α’ προσώπου, ονομαστικής πτώσης, ενώ όλες οι μορφές περιλαμβάνουν λέξεις σε όλα τα πρόσωπα και σε όλες τις πτώσεις.

Μετακίνηση ανεπιθύμητων λέξεων στην δεξιά λίστα, ή αντιστρόφως, είναι εφικτή.



Σχήμα 3.17: Προσθαφαίρεση λέξεων στις λίστες

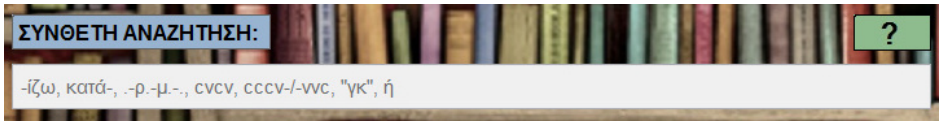
Αν επιλεγεί αναζήτηση για το σύνολο των τρισύλλαβων και πλέον λέξεων που περιέχουν τον όμοιο φθόγγο “κκ” μεταξύ 2ης - 3ης ή 3ης - 4ης συλλαβής και τονίζονται στη λήγουσα, τότε η μορφή της οθόνης καταλήγει στην παρακάτω:



Σχήμα 3.18: Εφαρμογή απλής αναζήτησης με βασικό κριτήριο (μεταξύ άλλων) λέξεις που περιέχουν τον όμοιο φθόγγο “κκ”

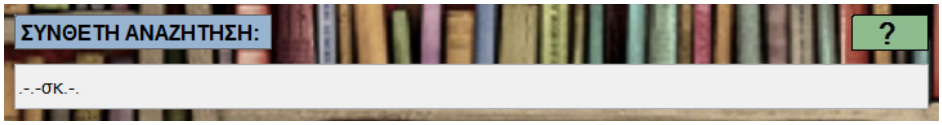
3.3.2 Παραδείγματα Σύνθετης Αναζήτησης

Η σύνθετη αναζήτηση περιλαμβάνει ένα πεδίο κειμένου, εμπλουτισμένο με λεπτομέρειες καθοδήγησης για τις λειτουργίες που καλύπτει και πλήκτρο παροχής εκτενέστερων οδηγιών (με αναδυόμενα παράθυρα όπως απεικονίζονται στα σχήματα 3.4, 3.5, 3.6, 3.7 και 3.8).



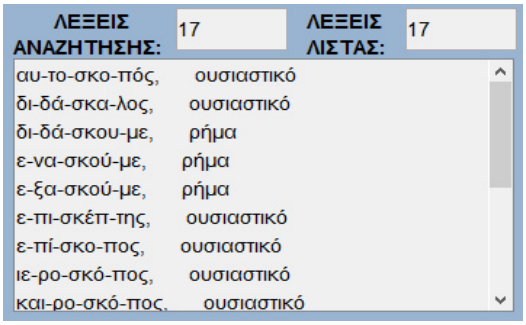
Σχήμα 3.19: Σύνθετη αναζήτηση

Διατηρώντας τα ήδη επιλεγμένα λεξικά, πληκτρολογείται αναζήτηση τετρασύλλαβων λέξεων, στις οποίες η 3η συλλαβή ξεκινάει με τον δίφθογγο “σκ”:



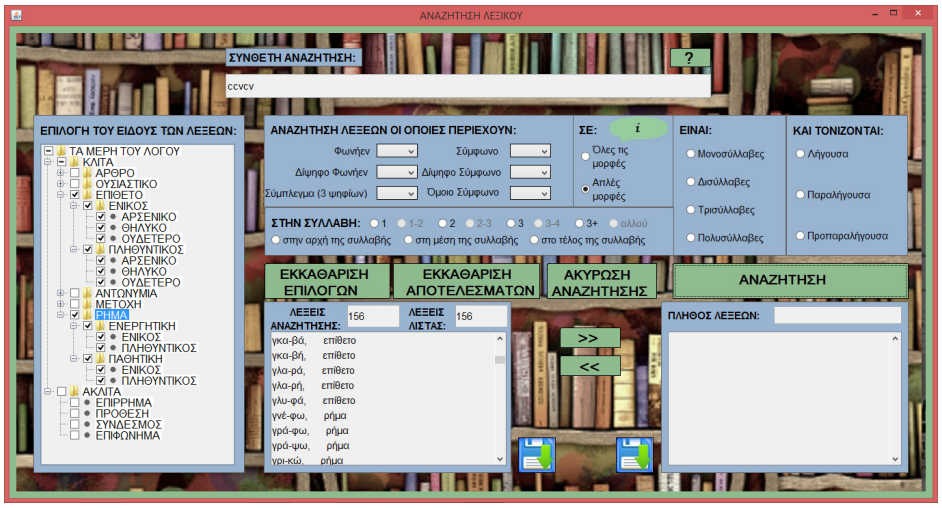
Σχήμα 3.20: Πιθανή επιλογή σύνθετης αναζήτησης

Ακολουθούν τα αποτελέσματα:



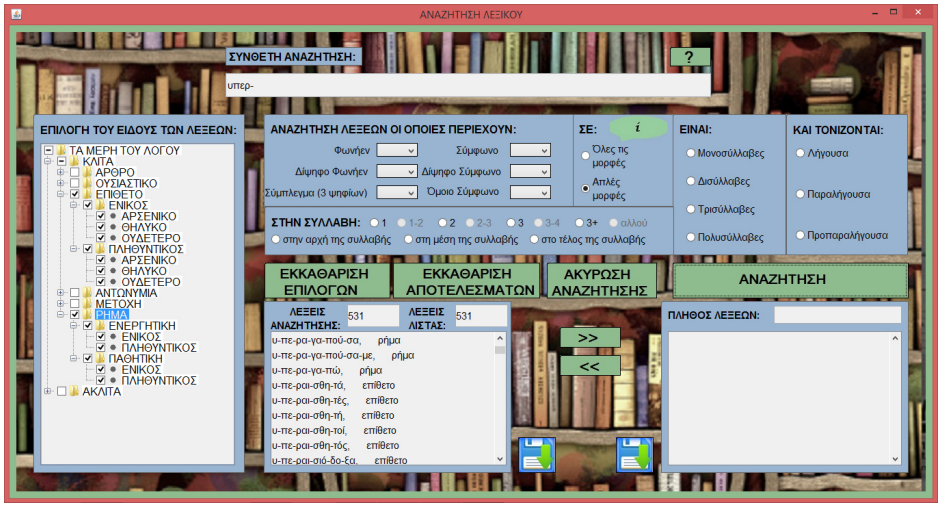
Σχήμα 3.21: Παρουσίαση αποτελεσμάτων αναζήτησης

Επιπρόσθετα, παρουσιάζεται εφαρμογή της σύνθετης αναζήτησης, στο σύνολο των επιθέτων και των ρημάτων, προς εύρεση λέξεων με μοτίβο “σύμφωνο - σύμφωνο - φωνήεν - σύμφωνο - φωνήεν”:



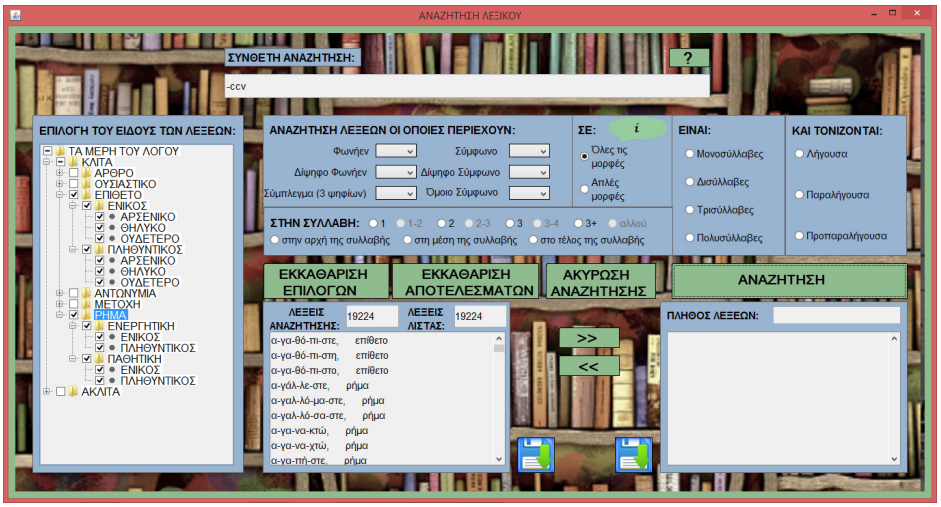
Σχήμα 3.22: Εφαρμογή σύνθετης αναζήτησης για συγκεκριμένο μοτίβο λέξεων

Όμοια, αναζήτηση συναρτήσει όμοιων φθόγγων στην αρχή των λέξεων:



Σχήμα 3.23: Εφαρμογή σύνθετης αναζήτησης για λέξεις που ξεκινάνε με τον ίδιο συνδυασμό φθόγγων

Ή βάσει κοινού μοτίβου στην κατάληξη των λέξεων:

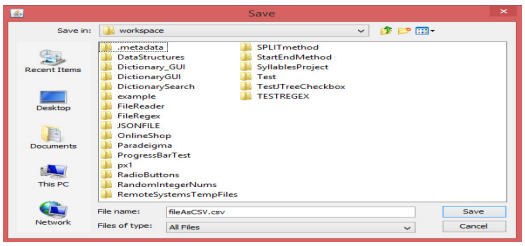


Σχήμα 3.24: Εφαρμογή σύνθετης αναζήτησης για λέξεις με κατάληξη κοινού μοτίβου (κατάληξη σε σύμφωνο-σύμφωνο-φωνήεν)

3.3.3 Αποθήκευση και Ανάκτηση Αποτελεσμάτων

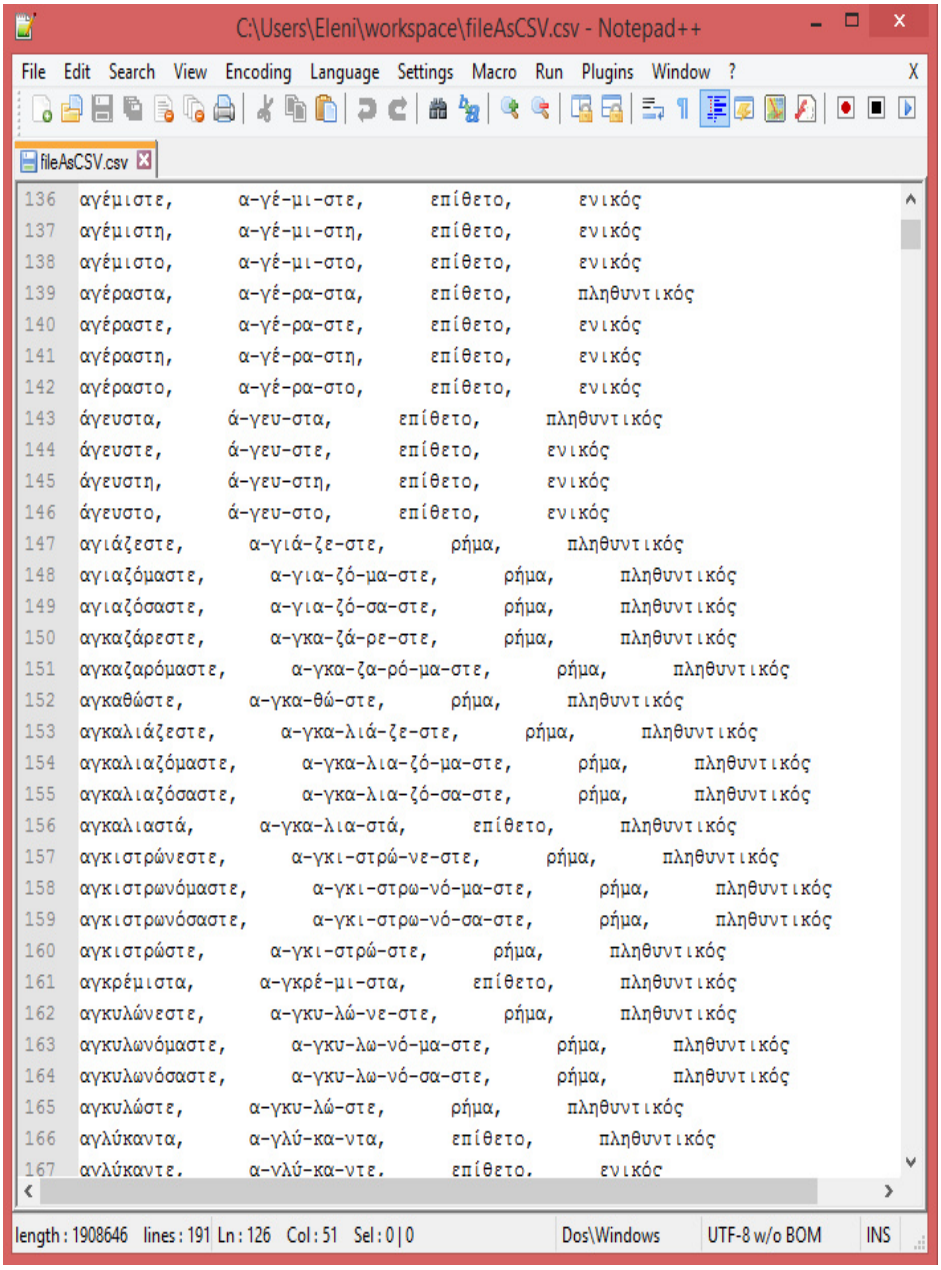
Τελικός, βέβαια, στόχος αποτελεί η δυνατότητα αποθήκευσης και ανάκτησης του συνόλου των λέξεων που προκύπτουν, ύστερα από τις διαδοχικές αναζητήσεις. Η δυνατότητα αυτή δίνεται από το πλήκτρο  αποθήκευσης της εφαρμογής, το οποίο είναι ξεχωριστό για καθε μια από τις δυο λίστες. Με το πάτημα του κουμπιού, θα εμφανιστεί παράθυρο επιλογής αρχείου προτίμησης, προς αποθήκευση της εκάστοτε λίστας, είτε σε χώρο της εσωτερικής μνήμης του Η/Υ, είτε σε εξωτερικό χώρο αποθήκευσης.

Επιλέγοντας αποθήκευση της λίστας των αποτελεσμάτων που δίνονται στο σχήμα 3.24, εμφανίζεται το παράθυρο επιλογών:



Σχήμα 3.25: Παράθυρο επιλογής αρχείου αποθήκευσης

Ύστερα από επαλήθευση της επιλογής μας, με το πάτημα του κουμπιού SAVE του παραπάνω παραθύρου - διαλόγου, υπάρχει διαθέσιμη, σε αρχείο του υπολογιστή με το όνομα που επιλέγεται (από προεπιλογή δίνεται το όνομα "fileAsCSV.csv"), η λίστα με τα αποτελέσματα της αναζήτησης.



Σχήμα 3.26: Ανάκτηση αρχείου αποθήκευσης

Κεφάλαιο 4

Σχεδίαση Εφαρμογής

Στην παρούσα ενότητα, επιδιώκεται μια σύντομη περιγραφή, τόσο βασικών εννοιών, όσο και μεθοδολογίας, σχετικά με την σχεδίαση και την ανάλυση αλγορίθμων. Στόχος είναι η ομαλή μετάβαση στον τρόπο σχεδίασης της ίδιας της εφαρμογής της αναζήτησης λεξικού.

4.1 Βασικές Έννοιες

Εργαλείο για την επίλυση ενός καλά καθορισμένου υπολογιστικού προβλήματος αποτελεί ο αλγόριθμος. Η διατύπωση του προβλήματος καθορίζει σε γενικές γραμμές την επιθυμητή σχέση εισόδου - εξόδου. Ο αλγόριθμος περιγράφει μια συγκεκριμένη υπολογιστική διαδικασία για την επίτευξη αυτής της σχέσης εισόδου - εξόδου. Συγκεκριμένα, ο όρος ”αλγόριθμος” αναφέρεται σε οποιαδήποτε καλά ορισμένη υπολογιστική διαδικασία που δέχεται κάποια τιμή ή κάποιο σύνολο τιμών ως είσοδο και δίνει κάποια τιμή ή κάποιο σύνολο τιμών ως έξοδο. Συνεπώς, ένας αλγόριθμος είναι μια ακολουθία υπολογιστικών βημάτων που μετασχηματίζει την είσοδο στην έξοδο.

Η ανάπτυξη ενός αλγορίθμου, συνοδεύεται από την ανάλυση του, η οποία είναι, ουσιαστικά, η απόδειξη ότι ο αλγόριθμος δίνει το σωστό αποτέλεσμα και η αξιολόγηση της επίδοσής του. Ένας αλγόριθμος, λοιπόν, χαρακτηρίζεται ”ορθός”, εάν για κάθε ακολουθία εισόδου, τερματίζει δίνοντας την ορθή έξοδο. Ένας ορθός αλγόριθμος, δηλαδή, επιλύει το δεδομένο υπολογιστικό πρόβλημα. Ένας μη ορθός αλγόριθμος μπορεί να μην τερματίζει καν για ορισμένες εισόδους, ή μπορει να τερματίζει δίνοντας διαφορετικό από το ζητούμενο, αποτέλεσμα.

Ένας αλγόριθμος οφείλει να καθορίζεται από συγκεκριμένα κριτήρια και πρότυπα. Ειδικότερα, τα κριτήρια που ικανοποιεί συνοψίζονται στα εξής:

- ✓ καθοριστικότητα (definiteness), κάθε κανόνας του ορίζεται επακριβώς και η αντίστοιχη διεργασία είναι συγκεκριμένη.

- ✓ περατότητα (finiteness), κάθε εκτέλεση είναι πεπερασμένη, δηλαδή τελειώνει ύστερα από έναν πεπερασμένο αριθμό διεργασιών ή βημάτων.

- ✓ αποτελεσματικότητα (effectiveness), κάθε μεμονωμένη εντολή του αλγορίθμου να είναι απλή (και όχι σύνθετη). Δηλαδή μία εντολή δεν αρκεί να έχει ορισθεί, αλλά πρέπει να είναι και εκτελέσιμη.

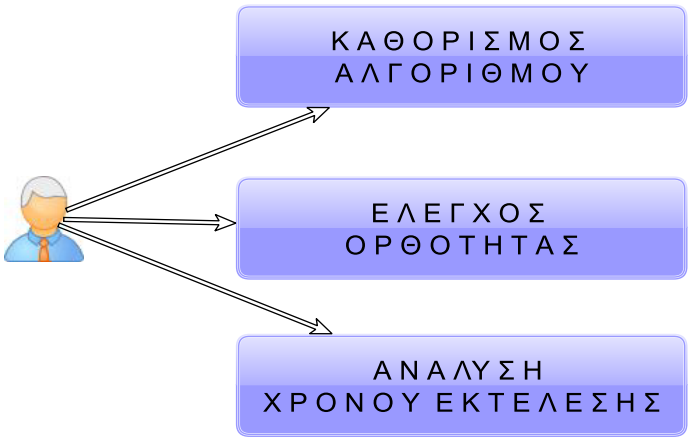
- ✓ επεκτασιμότητα (scalability), η δυνατότητα του λογισμικού να διαχειριστεί μεγαλύτερα προβλήματα και άρα μεγαλύτερο όγκο δεδομένων με σχετικά μικρή αύξηση στο κόστος του αλγόριθμου.

- ✓ να έχει είσοδο δεδομένων, επεξεργασία και έξοδο αποτελεσμάτων (input - output), κατά την εκκίνηση εκτέλεσης του αλγορίθμου μία ή περισσότερες τιμές δεδομένων πρέπει να δίνονται ως είσοδοι στον αλγόριθμο και εν συνεχεία ο αλγόριθμος πρέπει να δημιουργεί, τουλάχιστον, μία τιμή (δεδομένων), ως αποτέλεσμα προς το χρήστη ή προς ένα άλλο αλγόριθμο.

Οι τρόποι αναπαράστασης ενός αλγορίθμου ποικίλλουν. Είναι πιθανόν να γίνει με τον πιο αδόμητο τρόπο, ως ελεύθερο κείμενο, μέχρι τον πιο συνοπτικό και συμπαγή που εκπληρεί τις προϋποθέσεις του Δομημένου Προγραμματισμού [1] και είναι η κωδικοποίησή του σε ψευδογλώσσα. Μια εναλλακτική αναπαράσταση με πιο γραφικό χαρακτήρα είναι εκείνη του διαγράμματος ροής (flowchart).

Συνοψίζοντας, καθορίζεται επακριβώς το πρόβλημα που είναι προς επίλυση, ανάγεται σε αλγόριθμο, προκειμένου να προσδιοριστούν κομβικά σημεία (είσοδος - έξοδος) και διαδικασίες/λειτουργίες. Το γεγονός αυτό θα οδηγήσει σε περαιτέρω διερεύνηση, βέλτιστων και αποδοτικών λύσεων για την κάθε μία διαδικασία/λειτουργία ξεχωριστά. Τελικά, από το σύνολο της διαδικασίας της σχεδίασης θα επαληθευτεί η ορθότητα του αποτελέσματος και θα αναλυθεί ο χρόνος εκτέσης των διαδικασιών.

[1] Δομημένος Προγραμματισμός (structured programming) είναι μία προσέγγιση στον προγραμματισμό, η οποία βασίζεται στην έννοια της διαδικασίας. Η διαδικασία, γνωστή επίσης και ως ρουτίνα, υπορουτίνα, ή μέθοδος, είναι ένα αυτοτελές σύνολο εντολών προς εκτέλεση.



Σχήμα 4.1: Σχεδίαση Αλγορίθμου

4.2 Μέθοδοι Σχεδίασης Εφαρμογών

Σύμφωνα με τα προηγούμενα, πριν την ανάπτυξη μιας προγραμματιστικής εφαρμογής είναι χρήσιμο να γίνει σχεδίαση των απαιτήσεων της εφαρμογής. Αυτό συμβαίνει, διότι ένα πρόβλημα είναι πιθανόν να μην επιλύεται με μία μοναδική λύση, αλλά με περισσότερες. Η λύση σε ένα πρόβλημα, δηλαδή, μπορεί να προέλθει από ποικίλες προσεγγίσεις, τεχνικές και μεθόδους. Συνεπώς, είναι επιτακτική η ανάγκη να πραγματοποιείται μια καλή σχεδίαση του κάθε προβλήματος, ώστε τελικά να προτείνεται συγκεκριμένη μεθοδολογία και ακολουθία βημάτων. Βασικός στόχος είναι η πρόταση τόσο έξυπνων, όσο και αποδοτικών λύσεων.

Η σχεδίαση ενός προβλήματος περιλαμβάνει καταγραφή της υπάρχουσας πληροφορίας, αναγνώριση των ιδιαιτεροτήτων του προβλήματος, αποτύπωση των συνθηκών και προϋποθέσεων υλοποίησης του και τελικά πρόταση επίλυσης και υλοποίηση. Η τεχνική που χρησιμοποιείται ονομάζεται ιεραρχική σχεδίαση προγράμματος, με στόχο την διάσπαση του αρχικού προβλήματος σε μία σειρά από απλούστερα υποπροβλήματα, τα οποία να είναι εύκολο να επιλυθούν οδηγώντας στην επίλυση του αρχικού προβλήματος. Με αυτό τον τρόπο, το πρόγραμμα χωρίζεται σε τμήματα με αποτέλεσμα την:

- ✓ διευκόλυνση της δημιουργίας του προγράμματος
- ✓ σημαντική μείωση της πιθανότητας λάθους

- ✓ ευκολότερη παρακολούθηση, κατανόηση και συντήρηση του προγράμματος.

Την διάσπαση του αρχικού προβλήματος σε υποπροβλήματα θα ακολουθήσει η ιδέα της αντικειμενοστραφούς σχεδίασης (object-oriented design). Για την επίλυση του κάθε υποπρογράμματος θα γίνει αντικειμενοστραφής προσέγγιση, έτσι ώστε να αναπτυχθούν προγράμματα ευέλικτα και επαναχρησιμοποιήσιμα, δεδομένου ότι με την αντικειμενοστραφή σχεδίαση εκλαμβάνονται ως πρωτεύοντα δομικά στοιχεία ενός προγράμματος τα δεδομένα, από τα οποία δημιουργούνται με κατάλληλη μορφοποίηση τα αντικείμενα (objects), που αντιπροσωπεύουν αφηρημένες οντότητες και περιγράφονται μέσω των κλάσεων (classes). Μια κλάση, δηλαδή καθορίζει τις κοινές ιδιότητες και συμπεριφορές (μέθοδοι-methods, μεταβλητές-variables) σε έναν τύπο αντικειμένου. Ως πλεονεκτήματα της αντικειμενοστραφούς σχεδίασης θεωρούνται:

- ✓ η ενθυλάκωση (encapsulation), η ιδιότητα που προσφέρουν οι κλάσεις να ”κρύβουν” τα ιδιωτικά δεδομένα τους από το υπόλοιπο πρόγραμμα και να εξασφαλίζουν πως μόνο μέσω των δημόσιων μεθόδων τους θα μπορούν αυτά να προσπελαστούν.

- ✓ ο πολυμορφισμός (polymorphism), επιτρέπει το χειρισμό τιμών διαφορετικών τύπων δεδομένων με χρήση μιας ομοιόμορφης διεπαφής. Η έννοια αυτή εφαρμόζεται τόσο στους τύπους δεδομένων, όσο και στις συναρτήσεις.

- ✓ η κληρονομικότητα (inheritance), η ιδιότητα των κλάσεων να επεκτείνονται σε νέες κλάσεις, ρητά δηλωμένες ως κληρονόμους (υποκλάσεις), οι οποίες μπορούν να επαναχρησιμοποιήσουν τις μεταβιβάσιμες μεθόδους και ιδιότητες της κλάσης-πατέρα, αλλά και να προσθέσουν δικές τους.

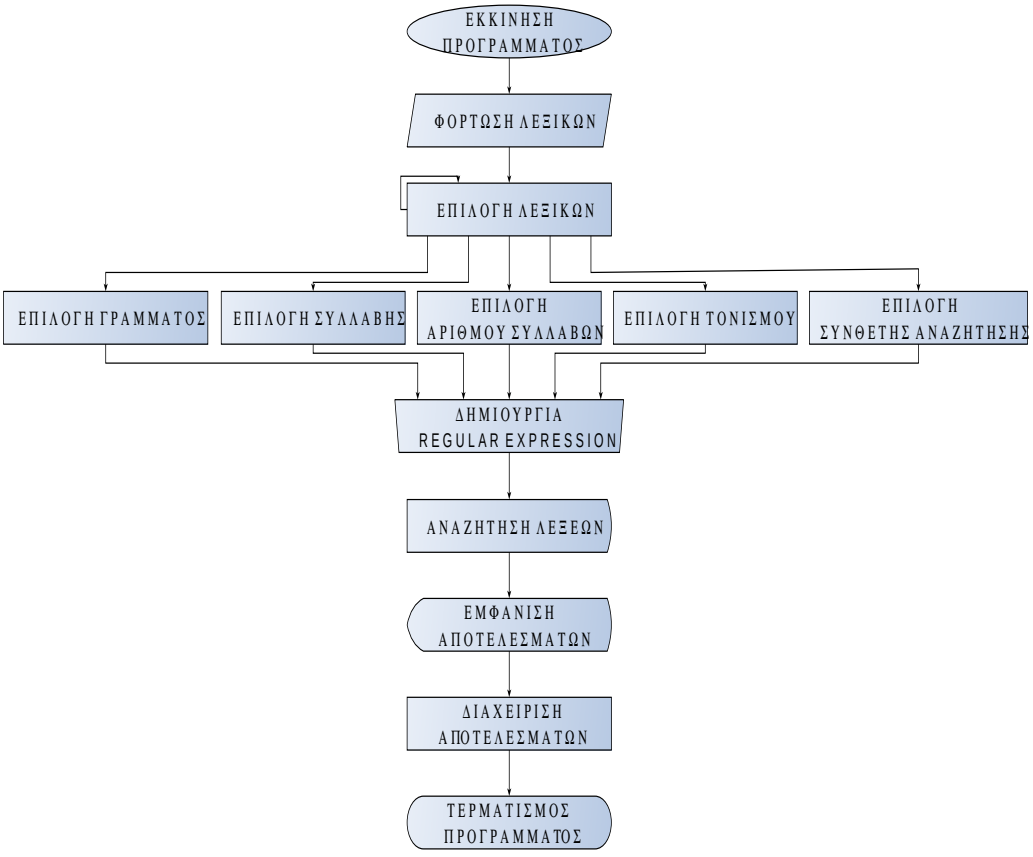
- ✓ η πολυδιεργασία νήματος (thread), η ιδιότητα που επιτρέπει σε ένα πρόγραμμα να εκτελεί δύο ή περισσότερες λειτουργίες ταυτόχρονα.

4.3 Σχεδίαση και Ανάλυση Εφαρμογής Αναζήτησης Λεξικού

Πρόκειται, όπως έχει αναφερθεί, για μία εφαρμογή αναζήτησης λεξικού, η οποία έχει ως στόχο την εύρεση λέξεων, οι οποίες πληρούν συγκεκριμένες προϋποθέσεις, τις οποίες καθορίζει ο χρήστης ανάλογα με τις ανάγκες του. Η διαδικασία είναι, αρχικά, ο προσδιορισμός της εισόδου του προγράμματος. Εν συνεχεία, καθορίζονται τα κριτήρια αναζήτησης που θα παρέχονται, ώστε να

καλύπτονται όλες οι πιθανές απαιτήσεις των χρηστών. Μελετάται ο τρόπος και η τεχνική με την οποία θα γίνει η διεργασία της αναζήτησης με τις συγκεκριμένες δοσμένες παραμέτρους, οι οποίες δίνονται από την επιλογή των κριτηρίων αναζήτησης. Εν τέλει, προκύπτει το αποτέλεσμα της διεργασίας της εισόδου, δηλαδή η έξοδος του προγράμματος, ενώ με τον έλεγχο αυτού, εξασφαλίζουμε την ορθότητά του. Η διαδικασία έχει ολοκληρωθεί με την εύρεση αποδοτικής και ορθής λύσης, δεδομένου ότι σε κάθε βήμα, έχει προταθεί και κατασκευαστεί η καταλληλότερη επίλυση των υποπροβλημάτων.

Παρουσιάζεται, παρακάτω, το διάγραμμα ροής της εφαρμογής, ώστε να υπάρχει μια εποπτεία και ακολουθεί εκτενής ανάλυση των βημάτων/υποπροβλημάτων:



Σχήμα 4.2: Διάγραμμα Ροής (flowchart)

ΒΗΜΑ $1^{ο}$: Η εφαρμογή είναι εφοδιασμένη με ένα σύνολο από 574.962 λέξεις της ελληνικής γλώσσας, κάθε μία εκ των οποίων παρέχει επιπλέον πληρο-

φορίες, αναφορικά με τις ιδιότητές της. Η διαδικασία της αναζήτησης αφορά το δεδομένο αλφαριθμητικό σύνολο, το οποίο, κατά συνέπεια, οφείλει να παρέχει δυνατότητες ευέλικτης αποθήκευσης, επεξεργασίας καθώς και ανάκτησης των δεδομένων του. Επιλέγεται, ως καταλληλότερος, ο καταμερισμός του συνόλου των λέξεων σε αρχεία μορφής JSON. Πρόκειται για 11 αρχεία, ολικού μεγέθους $248.68MB$:

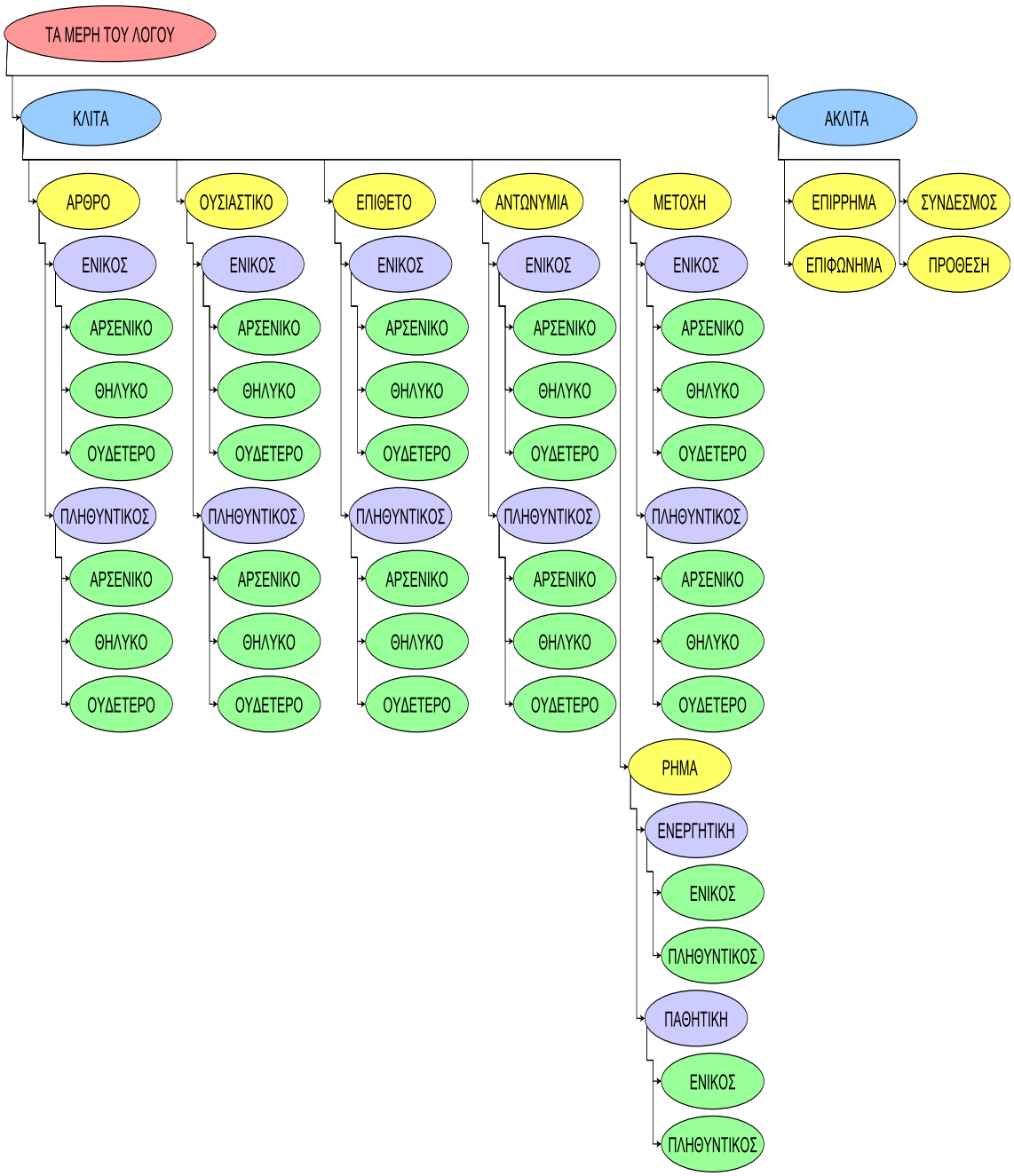
	ΟΝΟΜΑ ΑΡΧΕΙΟΥ	ΜΕΓΕΘΟΣ ΣΤΟ ΔΙΣΚΟ (ΣΕ MB)
1	greekWords_1_10000	4.48
2	greekWords_10001_40000	13.5
3	greekWords_40001_80000	17.1
4	greekWords_80001_120000	18.2
5	greekWords_120001_180000	26.4
6	greekWords_180001_260000	34.9
7	greekWords_260001_320000	26
8	greekWords_320001_380000	24
9	greekWords_380001_440000	25.4
10	greekWords_440001_500000	26.2
11	greekWords_500001_574962	32.5
	ΣΥΝΟΛΙΚΟ ΜΕΓΕΘΟΣ:	248.68

Σχήμα 4.3: Αρχεία Λεξικού

Είναι στο σημείο αυτό, όπου θα τεθεί υπό αμφισβήτηση, εάν κατά την εκκίνηση της εφαρμογής, είναι αποδοτικό να δοθούν ως είσοδο, όλα τα παραπάνω αρχεία, με όλες τις λέξεις της αλφαβήτου. Αφού τα αρχεία παρέχουν πληροφορίες για το μέρος του λόγου που αποτελεί η κάθε λέξη, καθώς και για πολλές ακόμα λεπτομέρειες, κρίνεται απαραίτητο να γίνει ένας τέτοιου είδους διαχωρισμός στα αρχεία. Με μια πρωτογενή διάκριση ανάμεσα στα κλιτά και άκλιτα μέρη του λόγου [2], ο διαχωρισμός παίρνει την μορφή της ακόλουθης

[2] *Κλιτά μέρη του λόγου* είναι εκείνα τα οποία κλίνονται, δηλαδή το καθένα από αυτά παίρνει στο λόγο διάφορες μορφές, σε αντίθεση με τα *άκλιτα μέρη του λόγου*, τα οποία δεν κλίνονται, παρουσιάζονται πάντοτε στο λόγο με την ίδια μορφή.[2]

αναπαράστασης:



Σχήμα 4.4: Διάσπαση Αρχικού Λεξικού σε Επιμέρους Κατηγορίες

Με αυτόν τον τρόπο, οργανώνονται τα δεδομένα σε μια δομή δέντρου, όπου διευκολύνεται, τόσο η προσπέλαση, όσο και η τροποποίηση τους. Με μια καθοδική διερεύνηση του δέντρου [3], παρατηρείται ότι ο διαχωρισμός των αρχείων, με βάση τις διάφορες κατηγορίες των λέξεων, δεν είναι, προφανώς, τυχαίος. Έχοντας τονίσει με διαφορετική χρωματική απόχρωση τους κόμβους που ανήκουν στο ίδιο επίπεδο του δέντρου, διαφαίνεται ότι τα φύλλα του (οι τελευταίοι κόμβοι του - κόμβοι που δεν έχουν "παιδιά", ανεξαρτήτως επιπέδου) θα καθορίσουν, τελικώς, την είσοδο του προγράμματος. Παρατηρώντας το δέντρο συνολικά, με επιμονή στα φύλλα του, διαπιστώνεται ότι στον μη αυθαίρετο αυτό διαχωρισμό, ικανοποιούνται θεμελιώδεις προϋποθέσεις:

- ✓ η ένωση του συνόλου των δεδομένων που αναπαριστούν τα φύλλα ενσωματώνει το σύνολο των λέξεων του λεξικού,
- ✓ η τομή του συνόλου των δεδομένων που αντιπροσωπεύουν τα φύλλα κρύβει το κενό σύνολο.

Ως εκ τούτου, επιβεβαιώνεται τόσο το γεγονός ότι ελέγχονται όλες οι λέξεις του αρχικού λεξικού, αλλά, επιπλέον, βεβαιώνεται ότι η κάθε λέξη θα ελεγχθεί μια και μοναδική φορά, διοτί δεν θα είναι πολλαπλά περασμένη στα επιμέρους λεξικά.

Επιπλέον, η κάθε λέξη του αρχείου διαθέτει λεπτομέρειες για ιδιότητες της λέξης, οι οποίες αφορούν στην προφορά της, τον χρόνο και την έγκλισή της εάν πρόκειται για ρήμα, οι οποίες παραλείπονται, δεδομένου ότι δεν συνεισφέρουν πληροφορία χρήσιμη για την αναζήτηση στο παρόν λεξικό, ενώ ταυτόχρονα δεν επηρεάζεται μελλοντική επέκταση της εφαρμογής.

Δίνεται, τελικά, η απάντηση για την είσοδο της εφαρμογής. Επιβεβαιώνοντας την ορθότητα του παραπάνω συλλογισμού και αναλύοντας την αποδοτικότητά του, προκύπτει η διαπίστωση ότι, πλέον, διατίθενται στην εφαρμογή αρχεία, σαφώς, κατά πολύ μικρότερου μεγέθους στο δίσκο από τα αρχικά, και με δεδομένα προσανατολισμένου περιεχομένου, για περιορισμένου ενδιαφέροντος αναζητήσεις, αν χρειαστεί.

Ακολουθεί πίνακας με τα νέα αρχεία του λεξικού και τα αντίστοιχα μεγέθη τους, με την ουσία να εντοπίζεται στο συνολικό μέγεθος των αρχείων, το οποίο δίνει το άθροισμα των $56027kB$, δηλαδή των $54,71MB$. Μέγεθος, περίπου, 4.5 φορές μικρότερο από το αρχικό.

[3] *Καθοδική διερεύνηση* θεωρείται η διερεύνηση που επεκτείνεται προς μεγαλύτερα "βάθη" στο γράφημα, οποτεδήποτε αυτό είναι δυνατόν.[3]

	ΟΝΟΜΑ ΑΡΧΕΙΟΥ	ΜΕΓΕΘΟΣ ΣΤΟ ΔΙΣΚΟ (ΣΕ ΚΒ)		ΟΝΟΜΑ ΑΡΧΕΙΟΥ	ΜΕΓΕΘΟΣ ΣΤΟ ΔΙΣΚΟ (ΣΕ ΚΒ)
1	ΑΚΛΙΤΑ, ΕΠΙΡΡΗΜΑ	57	11	ΚΛΙΤΑ, ΑΡΘΡΟ, ΕΝΙΚΟΣ,	2
2	ΑΚΛΙΤΑ, ΕΠΙΦΩΝΗΜΑ	4	12	ΚΛΙΤΑ, ΑΡΘΡΟ, ΕΝΙΚΟΣ,	2
3	ΑΚΛΙΤΑ, ΠΡΟΘΕΣΗ	3	13	ΚΛΙΤΑ, ΑΡΘΡΟ, ΕΝΙΚΟΣ,	1
4	ΑΚΛΙΤΑ, ΣΥΝΔΕΣΜΟΣ	5	14	ΚΛΙΤΑ, ΑΡΘΡΟ, ΠΛΗΘΥΝΤΙΚΟΣ,	1
5	ΚΛΙΤΑ, ΑΝΤΩΝΥΜΙΑ, ΕΝΙΚΟΣ, ΑΡΣΕΝΙΚΟ	15	15	ΚΛΙΤΑ, ΑΡΘΡΟ, ΠΛΗΘΥΝΤΙΚΟΣ, ΘΗΛΥΚΟ	1
6	ΚΛΙΤΑ, ΑΝΤΩΝΥΜΙΑ, ΕΝΙΚΟΣ, ΘΗΛΥΚΟ	8	16	ΚΛΙΤΑ, ΑΡΘΡΟ, ΠΛΗΘΥΝΤΙΚΟΣ, ΟΥΔΕΤΕΡΟ	1
7	ΚΛΙΤΑ, ΑΝΤΩΝΥΜΙΑ, ΕΝΙΚΟΣ, ΟΥΔΕΤΕΡΟ	2	17	ΚΛΙΤΑ, ΕΠΙΘΕΤΟ, ΕΝΙΚΟΣ, ΑΡΣΕΝΙΚΟ	5487
8	ΚΛΙΤΑ, ΑΝΤΩΝΥΜΙΑ, ΠΛΗΘΥΝΤΙΚΟΣ, ΑΡΣΕΝΙΚΟ	11	18	ΚΛΙΤΑ, ΕΠΙΘΕΤΟ, ΕΝΙΚΟΣ, ΘΗΛΥΚΟ	2601
9	ΚΛΙΤΑ, ΑΝΤΩΝΥΜΙΑ, ΠΛΗΘΥΝΤΙΚΟΣ, ΘΗΛΥΚΟ	4	19	ΚΛΙΤΑ, ΕΠΙΘΕΤΟ, ΕΝΙΚΟΣ, ΟΥΔΕΤΕΡΟ	119
10	ΚΛΙΤΑ, ΑΝΤΩΝΥΜΙΑ, ΠΛΗΘΥΝΤΙΚΟΣ, ΟΥΔΕΤΕΡΟ	3	20	ΚΛΙΤΑ, ΕΠΙΘΕΤΟ, ΠΛΗΘΥΝΤΙΚΟΣ, ΑΡΣΕΝΙΚΟ	4573

	ΟΝΟΜΑ ΑΡΧΕΙΟΥ	ΜΕΓΕΘΟΣ ΣΤΟ ΔΙΣΚΟ (ΣΕ ΚΒ)		ΟΝΟΜΑ ΑΡΧΕΙΟΥ	ΜΕΓΕΘΟΣ ΣΤΟ ΔΙΣΚΟ (ΣΕ ΚΒ)
21	ΚΛΙΤΑ, ΕΠΙΘΕΤΟ, ΠΛΗΘΥΝΤΙΚΟΣ, ΘΗΛΥΚΟ	1449	31	ΚΛΙΤΑ, ΟΥΣΙΑΣΤΙΚΟ, ΕΝΙΚΟΣ, ΟΥΔΕΤΕΡΟ	1761
22	ΚΛΙΤΑ, ΕΠΙΘΕΤΟ, ΠΛΗΘΥΝΤΙΚΟΣ, ΟΥΔΕΤΕΡΟ	1347	32	ΚΛΙΤΑ, ΟΥΣΙΑΣΤΙΚΟ, ΠΛΗΘΥΝΤΙΚΟΣ, ΑΡΣΕΝΙΚΟ	1905
23	ΚΛΙΤΑ, ΜΕΤΟΧΗ, ΕΝΙΚΟΣ, ΑΡΣΕΝΙΚΟ	997	33	ΚΛΙΤΑ, ΟΥΣΙΑΣΤΙΚΟ, ΠΛΗΘΥΝΤΙΚΟΣ, ΘΗΛΥΚΟ	2501
24	ΚΛΙΤΑ, ΜΕΤΟΧΗ, ΕΝΙΚΟΣ, ΘΗΛΥΚΟ	656	34	ΚΛΙΤΑ, ΟΥΣΙΑΣΤΙΚΟ, ΠΛΗΘΥΝΤΙΚΟΣ, ΟΥΔΕΤΕΡΟ	1864
25	ΚΛΙΤΑ, ΜΕΤΟΧΗ, ΕΝΙΚΟΣ, ΟΥΔΕΤΕΡΟ	1	35	ΚΛΙΤΑ, ΡΗΜΑ, ΕΝΕΡΓΗΤΙΚΗ, ΕΝΙΚΟΣ	5762
26	ΚΛΙΤΑ, ΜΕΤΟΧΗ, ΠΛΗΘΥΝΤΙΚΟΣ, ΑΡΣΕΝΙΚΟ	1105	36	ΡΗΜΑ, ΕΝΕΡΓΗΤΙΚΗ, ΠΛΗΘΥΝΤΙΚΟΣ	6968
27	ΚΛΙΤΑ, ΜΕΤΟΧΗ, ΠΛΗΘΥΝΤΙΚΟΣ, ΘΗΛΥΚΟ	366	37	ΚΛΙΤΑ, ΡΗΜΑ, ΠΑΘΗΤΙΚΗ, ΕΝΙΚΟΣ	4808
28	ΚΛΙΤΑ, ΜΕΤΟΧΗ, ΠΛΗΘΥΝΤΙΚΟΣ, ΟΥΔΕΤΕΡΟ	369	38	ΚΛΙΤΑ, ΡΗΜΑ, ΠΑΘΗΤΙΚΗ, ΠΛΗΘΥΝΤΙΚΟΣ	5703
29	ΚΛΙΤΑ, ΟΥΣΙΑΣΤΙΚΟ, ΕΝΙΚΟΣ, ΑΡΣΕΝΙΚΟ	2287		ΣΥΝΟΛΙΚΟ ΜΕΓΕΘΟΣ:	56027
30	ΚΛΙΤΑ, ΟΥΣΙΑΣΤΙΚΟ, ΕΝΙΚΟΣ, ΘΗΛΥΚΟ	3278			

Σχήμα 4.5: Νέα Αρχεία λεξικού

Τα 38 αυτά νέα λεξικά, θα εισαχθούν ταυτόχρονα, κατά την έναρξη της εφαρμογής, για εξοικονόμηση χρόνου εκκίνησης με την βοήθεια της τεχνικής της πολυνημάτωσης (multithreading). Με τον όρο αυτό εννοείται η εκτέλεση μιας διεργασίας με την ύπαρξη πολλαπλών νημάτων (threads). Τα νήματα αυτά μοιράζονται τους πόρους της διεργασίας και μπορούν να εκτελούνται ανεξάρτητα, χωρίς αναμονή εισόδου από άλλα. Η υλοποίηση των νημάτων και των διεργασιών διαφέρει από το ένα λειτουργικό σύστημα στο άλλο. Σε έναν απλό επεξεργαστή, η πολυνημάτωση (multithreading) πραγματοποιείται με τη μέθοδο της πολυπλεξίας με διαίρεση χρόνου. Ο επεξεργαστής μεταπηδάει μεταξύ των διάφορων νημάτων με αποτέλεσμα η εναλλαγή αυτή μεταξύ των διεργασιών να συμβαίνει σε πολύ τακτά χρονικά διαστήματα, τέτοια ώστε ο χρήστης να έχει την εντύπωση ότι τα νήματα εκτελούνται την ίδια στιγμή.

Μόνο σε έναν επεξεργαστή με πολλούς επεξεργαστικούς πυρήνες, τα νήματα εκτελούνται πραγματικά ταυτόχρονα και κάθε πυρήνας εκτελεί ένα συγκεκριμένο νήμα.

ΒΗΜΑ 2ο: Θα χρειαστεί, στο σημείο αυτό, να προσδιοριστούν με σαφήνεια, όλα τα πιθανά κριτήρια μιας αναζήτησης. Με ομαδοποίηση των ιδιοτήτων και των χαρακτηριστικών των λέξεων, προκύπτει το ζητούμενο. Στις λέξεις της ελληνικής αλβαβήτου, επομένως, συναντούνται περιφραστικά οι εξής ιδιότητες:

1. αντιπροσωπεύουν κάποιο μέρος του λόγου,
2. περιέχουν κάποια γράμματα της αλφαβήτου,
3. χωρίζονται σε συλλαβές τηρώντας κάποιους κανόνες,
4. τονίζονται σε κάποιο γράμμα, ώστε να δίνουν την απαραίτητη φωνητική πληροφορία για την προφορά της λέξης.

Κάθε ιδιότητα αντιπροσωπεύει έναν τομέα αναζήτησης, που είναι δυνατόν να χρειαστεί να γίνει συνδυαστική αναζήτηση αυτών, ή ακόμα να χρειαστεί εύρεση λέξεων με κάποια ιδιαιτερότητα, η οποία δεν καλύπτεται από τις παραπάνω ιδιοτήτες.

ΒΗΜΑ 3ο: Με την παραμετροποίηση των κριτηρίων αναζήτησης, έρχεται το στάδιο της προγραμματιστικής δημιουργίας της κανονικής έκφρασης (regular expression), βάσει της οποίας θα γίνει αργότερα η εξερεύνηση των λέξεων. Με χρήση κανονικών εκφράσεων, προκειμένου να πραγματοποιηθεί η αναζήτηση, προσδοκάται γρήγορη και ορθή απόδοση αποτελέσματος.

ΒΗΜΑ 4ο: Πραγματοποιείται η διεργασία της αναζήτησης, με χρήση μεθόδων, τις οποίες παρέχει το πακέτο της βιβλιοθήκης της Java, για τις κανονικές εκφράσεις.

ΒΗΜΑ 5ο: Επιλογή και καθορισμός του τρόπου εμφάνισης, οργάνωσης, αλλά και αποθήκευσης των αποτελεσμάτων, ώστε να παρέχεται δυνατότητα περαιτέρω επεξεργασίας τους ή μεταγενέστερης ανάκτησης παρελθοντικών αποτελεσμάτων αναζήτησης.

Έχοντας διασπάσει την αρχική ιδέα της εφαρμογής σε επιμέρους, αυτοτελείς διαδικασίες που λειτουργούν αυτόνομα, παρέχοντας άμεσα και σωστά αποτελέσματα η κάθεμια, σημαίνει πως έχει ολοκληρωθεί η σχεδίαση και ανάλυση της εφαρμογής. Θα ακολουθήσει η ανάπτυξη του κώδικα που θα δόσει νόημα στις παραπάνω τεχνικές και ιδέες. Ωστόσο, η λειτουργικότητα της εφαρμογής καθορίζεται, όχι μόνο από την σχεδίασή του αλγορίθμου της, αλλά και από το περιβάλλον, το οποίο θα παρέχει στον χρήστη, προς αλληλεπίδραση με τις διαδικασίες της.

4.4 Γραφικό Περιβάλλον Επικοινωνίας/Γραφική Διεπαφή Χρήστη (GUI)

Είναι προφανές ότι, πρέπει να δίνεται η δυνατότητα στον χρήστη να επιλέγει τα κριτήρια αναζήτησης των λέξεων που επιθυμεί. Αυτό επιτυγχάνεται με ένα σύνολο γραφικών στοιχείων που εμφανίζονται στην οθόνη του υπολογιστή με το άνοιγμα της εφαρμογής και εξυπηρετούν στην αλληλεπίδραση του χρήστη με την εφαρμογή.

Για την σχεδίαση του γραφικού περιβάλλοντος επικοινωνίας του χρήστη με την εφαρμογή (Graphical User Interface, GUI), η JAVA παρέχει ένα σύνολο βιβλιοθηκών κλάσεων, το JFC (Java Foundation Classes), το οποίο περιλαμβάνει την βιβλιοθήκη γραφικών συστατικών Abstract Window Toolkit (AWT), την βιβλιοθήκη πακέτων Swing, και τις λειτουργίες γραφικών δυο διαστάσεων JAVA 2D. Μαζί, παρέχουν την δυνατότητα ανάπτυξης μιας ολοκληρωμένης διεπαφής χρήστη για προγράμματα σε Java, ανεξάρτητα από το λειτουργικό περιβάλλον στο οποίο εργάζεται ο χρήστης (Windows, Mac OS ή Linux).

Συγκεκριμένα, η βιβλιοθήκη Abstract Window Toolkit (AWT) περιέχει κλάσεις οι οποίες υλοποιούν τα συστατικά (components) του γραφικού περιβάλλοντος. Με τον όρο συστατικά εννοούμε όλους εκείνους τους μηχανισμούς που επιτρέπουν την επικοινωνία της εφαρμογής με τον χρήστη. Η επικοινωνία υλοποιείται με την ενεργοποίηση κάποιου γεγονότος (event). Όταν ένα γεγονός ενεργοποιηθεί, ειδοποιεί το συστατικό με το οποίο είναι συνδεδεμένο για κάποιο εξωτερικό συμβάν που έγινε και το αφορά, ώστε να ανταποκριθεί ανάλογα. Βασικά συστατικά του πακέτου είναι: Button, Label, CheckBox, List κ.α., με γεγονότα που αφορούν στην χρήση του ποντικιού ή του πληκτρολογίου.

Η βιβλιοθήκη πακέτων Swing είναι, ουσιαστικά, μία επέκταση του πακέτου Abstract Window Toolkit (AWT), δεδομένου ότι περιλαμβάνει μία πληθώρα επιπλέον συστατικών για την δημιουργία γραφικού περιβάλλοντος διεπαφής (GUI), κάνοντας χρήση του ίδιου μοντέλου χειρισμού γεγονότων. Βασικά συστατικά του πακέτου είναι: JTree, JToolTip, JPopupMenu, JList, JLabel, JComboBox, JProgressBar και πολλά ακόμα.

Όσον αφορά το πακέτο JAVA 2D, πρόκειται για την δημιουργία σύνθετων γραφικών με ποικιλία γραμμών και σχημάτων, περιστροφή και αλλαγή μεγέθους γραφικών, επεξεργασία εικόνας, εξελιγμένο χειρισμό κειμένου και διαχείρισης χρωμάτων. Χρησιμοποιεί σύστημα συντεταγμένων και δημιουργεί γραφικά με την κλάση Graphics. Επιπλέον, υποστηρίζει δυνατότητες αναπαραγωγής ήχων στο παρασκήνιο κάποιας εφαρμογής, την εμφάνιση σελίδας απεικόνισης εκτύπωσης και διαμόρφωσης σελίδας, και προσδιορίζει τα χαρακτηριστικά εκτύπωσης.

AWT		SWING		JAVA 2D	
Accessibility	Drag & Drop	Input Methods	Image I/O	Print Service	Sound

Σχήμα 4.6: Πακέτο JFC

Στην εφαρμογή του λεξικού, για την σχεδίαση του γραφικού περιβάλλοντος έχει γίνει χρήση συνδυασμού των παραπάνω βιβλιοθηκών με μεγαλύτερη έμφαση στο πακέτο swing. Συγκεκριμένα, το κύριο παράθυρο της εφαρμογής αποτελείται από 5 βασικά μέρη:

1. το δέντρο αναζήτησης. Συστατικό του πακέτου swing, το οποίο εμφανίζει ένα σύνολο ιεραρχικών δεδομένων, στην περίπτωσή μας τα μέρη του λόγου της ελληνικής γλώσσας. Κάθε κόμβος του δέντρου είναι ένα πλαίσιο ελέγχου, επίσης συστατικό του swing,

2. τον πίνακα επιλογών του χρήστη με ένα ευρύ φάσμα κριτηρίων αναζήτησης. Έχουν χρησιμοποιηθεί συστατικά του swing, όπως τα ραδιοπλήκτρα που είναι κουμπιά επιλογών,

3. τις λίστες εμφάνισης και διαχείρισης των αποτελεσμάτων αναζήτησης με περιοχές κειμένου, οι οποίες δίνουν πληροφορία που αφορά είτε στο πλήθος των λέξεων που εντοπίστηκαν για τα συγκεκριμένα κριτήρια στα οποία αφορά η αναζήτηση που πραγματοποιείται, είτε στο γενικό σύνολο των λέξεων της λίστας που πιθανόν έχουν συσσωρευτεί από προηγούμενες αναζητήσης. Τόσο οι λίστες, όσο και τα πεδία κειμένου, είναι συστατικά που παρέχονται από το πακέτο swing,

4. την περιοχή προτροπής του χρήστη να πραγματοποιήσει σύνθετη αναζήτηση για ανάγκες που δεν καλύπτονται από τις υπάρχουσες στον πίνακα επιλογών.

5. τα κουμπιά επικοινωνίας του χρήστη με το πρόγραμμα, ώστε να πραγματοποιηθεί συγκεκριμένη λειτουργία, όπως αυτή ορίζεται στο κείμενο κάθε κουμπιού.

Για τα χρώματα της εφαρμογής, την εικόνα φόντου, την γραμματοσειρά καθώς και για τους διαχειριστές διάταξης του πλαισίου έχει γίνει χρήση της βιβλιοθήκης AWT (Abstract Window Toolkit).

Κεφάλαιο 5

Ανάπτυξη Εφαρμογής

Κάθε μια από τις δυνατότητες και τις λειτουργίες της εφαρμογής της αναζήτησης λεξικού, μεταφράζεται σε μια σειρά από πακέτα, κλάσεις, διεπαφές. Δεδομένου ότι μια επιτυχημένη σχεδίαση, ακολουθείται από την ανάπτυξη κώδικα για τα επιμέρους υποπροβλήματα, παρουσιάζεται, στην συγκεκριμένη ενότητα, η οργάνωση των πακέτων και η δημιουργία των κλάσεων ή/και διεπαφών της εφαρμογής.

5.1 Περιγραφή Πακέτων

Σε ένα μακροσκελές πρόγραμμα, για την ασφαλή υλοποίηση των διάφορων τύπων δεδομένων από κλάσεις και διαπροσωπίες, για την αποφυγή συγκρούσεων στην ονοματοδοσία, καθώς και για τον έλεγχο της πρόσβασης, προτείνεται να διαχωρίζονται συναφή είδη σε πακέτα (packages). Με τον τρόπο αυτό επιτυγχάνονται τα ακόλουθα:

- ✓ εύκολος καθορισμός των συσχετίσεων μεταξύ των διαφορετικών τύπων που χρησιμοποιούνται,
- ✓ τα ονόματα που χρησιμοποιούνται δεν έρχονται σε σύγκρουση με ονόματα άλλων πακέτων, διότι το κάθε πακέτο δημιουργεί ένα νέο χώρο ονομάτων,
- ✓ επιτρέπεται σε τύπους εντός του πακέτου να έχουν απεριόριστη πρόσβαση μεταξύ τους, ενώ ταυτόχρονα εξακολουθεί να περιορίζεται η πρόσβαση για είδη εκτός του πακέτου.

Τα πακέτα είναι αντίστοιχα με τους φακέλους στο λειτουργικό σύστημα ενός υπολογιστή. Η ίδια η Java παρέχει μία τεράστια βιβλιοθήκη από έτοιμα πακέτα που μπορεί κανείς να χρησιμοποιήσει, η οποία περιλαμβάνει από διαχείριση αρχείων μέχρι γραφικό περιβάλλον και δικτυακές επικοινωνίες. Στο εκάστοτε πρόγραμμα μπορούν να χρησιμοποιηθούν λειτουργίες από άλλα πακέτα, απλά υποδηλώνοντας την χρήση τους με την λέξη κλειδί "import" και το όνομα του πακέτου.

Για την δημιουργία πακέτου ορίζεται τύχαια ένα όνομα και είναι αναγκαίο να δηλώνεται αυστηρά στην αρχή κάθε τύπου που προορίζεται να είναι μέρος του δεδομένου πακέτου. Στην περίπτωση που δεν δηλωθεί κανένα πακέτο, οι τύποι του προγράμματος που πρόκειται να δημιουργηθούν (κλάσεις, διαπροσωπείες, κλπ) ορίζονται αυτόματα ως μέρη προεπιλεγμένου ανώνυμου πακέτου. Ένα ανώνυμο πακέτο αποτελεί λύση αποκλειστικά για μικρές ή προσωρινές εφαρμογές ή όταν έχει μόλις αρχίσει η διαδικασία ανάπτυξης του κώδικα. Διαφορετικά, κλάσεις και διεπαφές ανήκουν σε πακέτα με συγκεκριμένο όνομα.

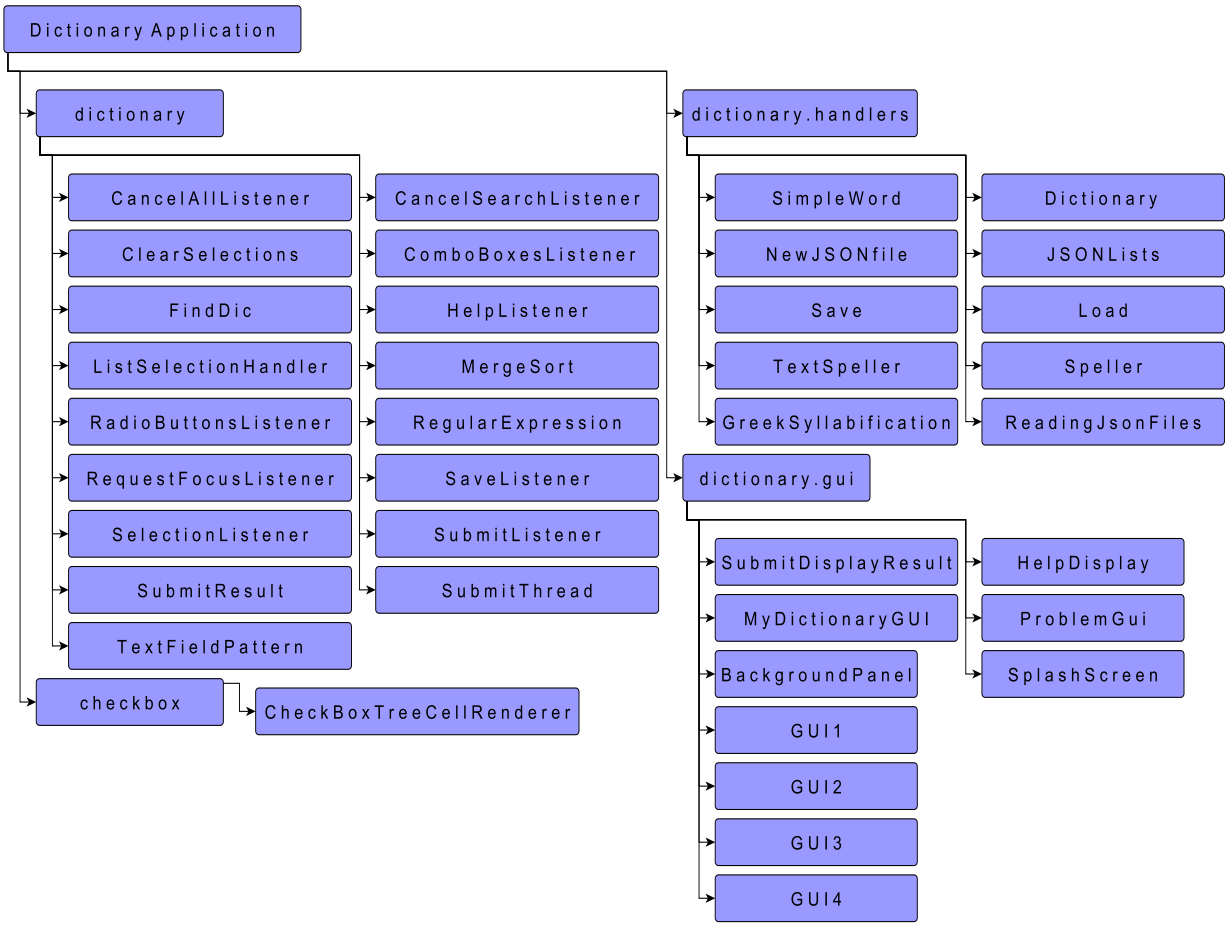
Αναλυτικά, στην ανάπτυξη της παρούσας εφαρμογής έχει γίνει δημιουργία τριών βασικών πακέτων και ενός βοηθητικού, καθένα από τα οποία περιλαμβάνει τις κλάσεις/διεπαφές εκείνες με σχετικό μεταξύ τους περιεχόμενο, δηλαδή σχετικές μεταξύ τους λειτουργίες. Ο διαχωρισμός έχει ως εξής:

(i) *dictionary.handlers*, πακέτο με τύπους που αφορούν στην διαχείριση των αρχικών αρχείων JSON, στα οποία είναι αποθηκευμένο το σύνολο των ελληνικών λέξεων με όλη την πληροφορία τους, με τρόπο τέτοιο ώστε τα αρχεία JSON να αντιπροσωπεύουν, τελικά, τα επιμέρους λεξικά που μας αφορούν, όπως έχουν παρουσιαστεί στο σχήμα **??**.

(ii) *dictionary*, πακέτο με τύπους που αφορούν καθαρά στις λειτουργίες της εφαρμογής.

(iii) *dictionary.gui*, πακέτο με κλάσεις που αφορούν αποκλειστικά στο γραφικό περιβάλλον επικοινωνίας του χρήστη με την εφαρμογή.

(iv) *checkbox*, βοηθητικό πακέτο, το οποίο περιλαμβάνει την κλάση εκείνη, με την οποία κατασκευάζεται, προγραμματιστικά, αντικείμενο δέντρου με πλαίσια ελέγχου σε κάθε κόμβο του, αφού δεν παρέχεται ως έτοιμο αντικείμενο από κάποια βιβλιοθήκη της JAVA.



Σχήμα 5.1: Οργανωτική Δομή Κλάσεων/Διεπαφών σε Πακέτα

5.2 Περιγραφή Κλάσεων

5.2.1 Κλάσεις Πακέτου *dictionary.handlers*

Στο συγκεκριμένο πακέτο πραγματοποιείται όλη η απαραίτητη επεξεργασία των αρχείων του λεξικού. Κατασκευάζονται κλάσεις για την δημιουργία των αντικειμένων της ”λεξης” και του ”λεξικού”. Συλλαβίζεται η κάθε μία λέξη χωριστά και η πληροφορία αυτή καταχωρείται εκ νέου στα αρχεία JSON, το οποία αποθηκεύονται κωδικοποιημένα. Περιγραφικά η διαδικασία αποθήκευσης του συλλαβισμού των λέξεων στα αρχεία JSON έχει ως εξής:

1. Διαπερνάται κάθε λέξη από κάθε ένα από τα αρχεία JSON και κρατεί-

ται, από τα αντικείμενα των αρχείων, το 1ο στοιχείο (ζεύγος key : value), το οποίο αφορά στην γραφή της λέξης.

2. Για κάθε στοιχείο κάθε αρχείου υλοποιείται μέθοδος για τον συλλαβισμό της λέξης, ο οποίος αποθηκεύεται σε αρχείο κειμένου (.txt).

3. Επαναλαμβάνεται η διαπέραση των αρχείων JSON, κρατώντας αυτή τη φορά το 2ο στοιχείο κάθε λέξης των αρχείων, το οποίο αφορά στον συλλαβισμό της λέξης και στο οποίο καταχωρείται, τελικά, ο συλλαβισμός των λέξεων, όπως διαβάζεται από το αρχείο κειμένου που έχει αποθηκευτεί.

Συνοπτική παρουσίαση των κλάσεων του πακέτου:

Όνομα κλάσης: *GreekSyllabification*	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	String	stringToSpell
	ArrayList<String>	vowels
	ArrayList<String>	consonants
	ArrayList<String>	result
Μέθοδοι	void	setStringToSpell(String stringToSpell)
	void	performSpelling()
	ArrayList<String>	getTokens()

Σχήμα 5.2: Μεταβλητές και Μέθοδοι Κλάσης *"GreekSyllabification"*

Όνομα κλάσης: *ReadingJsonFiles*	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	ArrayList<String>	ws
Μέθοδοι	ArrayList<String>	readFileLines(String fileName)
	String	readFile(String fileName)

Σχήμα 5.3: Μεταβλητές και Μέθοδοι Κλάσης *"ReadingJsonFiles"*

Όνομα κλάσης: *Dictionary*	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	ArrayList<SimpleWord>	words
Μέθοδοι	void	setWords(ArrayList<SimpleWord> words)
	ArrayList<SimpleWord>	getWords()

Σχήμα 5.4: Μεταβλητές και Μέθοδοι Κλάσης *"Dictionary"*

Όνομα κλάσης: *SimpleWord*	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	String	word, syllables, position, number, gender, voice, cases, person, tense
Μέθοδοι	void	setWord(String word), setSyllables(String syllables), setPosition(String position), setNumber(String number), setGender(String gender), setVoice(String voice), setCases(String cases), setPerson(String person), setTense(String tense)
	String	getWord(), getSyllables(), getPosition(), getNumber(), getGender(), getVoice(), getCases(), getPerson(), getTense()

Σχήμα 5.5: Μεταβλητές και Μέθοδοι Κλάσης *"SimpleWord"*

Η κλάση *"NewJSONfile"* περιλαμβάνει μία *main* μέθοδο, η οποία δημιουργεί καινούριο αρχείο JSON, μετατρέποντας τις λέξεις της αλφαβήτου σε αντικείμενα τύπου "SimpleWord", με το σύνολο τους να αποτελεί αντικείμενο τύπου "Dictionary".

Όμοια, η κλάση *"JSONLists"* περιλαβάνει την μέθοδο *main*, από την οποία θα προκύψουν τα 38 επιμέρους λεξικά που θα κωδικοποιηθούν (serialize) [1] με την κλάση *"Save"* και αργότερα θα αποκωδικοποιηθούν (deserialize) [2] με την κλάση *"Load"*.

[1] Κωδικοποίηση (serialization) είναι η μετατροπή ενός αντικειμένου σε μια σειρά από bytes, έτσι ώστε το αντικείμενο να μπορεί εύκολα να αποθηκευτεί ή να μεταδοθεί σε ένα δίαυλο επικοινωνίας.

[2] Το κωδικοποιημένο ρεύμα των bytes μετατρέπεται σε ένα πιστό αντίγραφο του αρχικού αντικειμένου με την διαδικασία της αποκωδικοποίησης (deserialization).

Τέλος, με την κλάση *"TextSpeller"* και κάνοντας χρήση των μεθόδων της διαπροσωπείας *"Speller"* υλοποιείται ο κατάλληλος συλλαβισμός για κάθε αντικείμενο "SimpleWord" του "Dictionary".

5.2.2 Κλάσεις Πακέτου *dictionary*

Πρόκειται για το πακέτο που περιέχει την κλάση, από την οποία θα σχηματιστεί η κατάλληλη κανονική έκφραση (regular expression). Αυτό πραγματοποιείται βάσει των επιλεγμένων από τον χρήστη παραμέτρων, κάθε μια από τις οποίες αποτελεί ένα ξεχωριστό γεγονός (event) και συνδέεται με τον αντίστοιχο ακροατή (listener), ο οποίος θα αποκριθεί κατάλληλα στο εκάστοτε γεγονός. Παρουσιάζοντας, αναλυτικά, τις κλάσεις που αποτελούν τους ακροατές για τα γεγονότα της εφαρμογής, έχουμε:

Όνομα κλάσης: SelectionListener	Διασύνδεση που υλοποιεί: TreeSelectionListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	String	selectedNodeName
	TreePath[]	treeCheck
	ArrayList<String>	collect
Μέθοδοι	void	valueChanged(TreeSelectionEvent se)
	void	setCollect(TreePath[] treeCheck)
	ArrayList<String>	getCollect()

Σχήμα 5.6: Ακροατής *"SelectionListener"* για την διαχείριση των επιλογών από το δέντρο αναζήτησης.

Όνομα κλάσης: ComboBoxesListener	Διασύνδεση που υλοποιεί: ItemListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	String	myItem
Μέθοδοι	void	itemStateChanged(ItemEvent e)
	void	setMyItem(String myItem)
	String	getMyItem()

Σχήμα 5.7: Ακροατής *"ComboBoxesListener"* για την διαχείριση των επιλογών από τα μενού επιλογών[3].

[3]Το μενού επιλογής (comboBox) είναι μία πτυσσόμενη λίστα, η οποία παρέχει στον χρήστη μια ομάδα από επιλογές. Κάθε γραμμή της λίστας αντιστοιχεί σε μια επιλογή.

Όνομα κλάσης: RadioButtonsListener	Διασύνδεση που υλοποιεί: ActionListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	String	whatform
	ArrayList<String>	sylCheck, intonationCheck, inSylCheck, whereInSylCheck, betweenSylCheck
Μέθοδοι	void	actionPerformed(ActionEvent e)
	ArrayList<String>	getsylCheck(), getIntonation(), getInSyl(), getWhereInSyl(), getBetweenSyl()
	String	getWhatForm()

Σχήμα 5.8: Ακροατής *"RadioButtonsListener"* για την διαχείριση των επιλογών από τα ραδιοπλήκτρα (radioButtons).

Όνομα κλάσης: ClearSelections	Διασύνδεση που υλοποιεί: ActionListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	-	-
Μέθοδοι	void	actionPerformed(ActionEvent arg0)

Σχήμα 5.9: Ακροατής *"ClearSelections"* για την διαχείριση του κουμπιού "ΕΚΚΑΘΑΡΙΣΗ ΕΠΙΛΟΓΩΝ".

Όνομα κλάσης: CancelAllListener	Διασύνδεση που υλοποιεί: ActionListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	-	-
Μέθοδοι	void	actionPerformed(ActionEvent arg0)

Σχήμα 5.10: Ακροατής *"CancelAllListener"* για την διαχείριση του κουμπιού "ΕΚΚΑΘΑΡΙΣΗ ΑΠΟΤΕΛΕΣΜΑΤΩΝ".

Όνομα κλάσης: CancelSearchListener	Διασύνδεση που υλοποιεί: ActionListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	SubmitListener	sub
Μέθοδοι	void	actionPerformed(ActionEvent e)

Σχήμα 5.11: Ακροατής *"CancelSearchListener"* για την διαχείριση του κουμπιού "ΑΚΥΡΩΣΗ ΑΝΑΖΗΤΗΣΗΣ".

Όνομα κλάσης: HelpListener	Διασύνδεση που υλοποιεί: MouseListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	-	-
Μέθοδοι	void	mousePressed(MouseEvent e), mouseClicked(MouseEvent arg0), mouseEntered(MouseEvent arg0), mouseExited(MouseEvent arg0), mouseReleased(MouseEvent arg0)

Σχήμα 5.12: Ακροατής *"HelpListener"* για την διαχείριση του κουμπιού βοήθειας "?".

Όνομα κλάσης: ListSelectionHandler/ List_1SelectionHandler	Διασύνδεση που υλοποιούν: ActionListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	-	-
Μέθοδοι	void	actionPerformed(ActionEvent arg0)

Σχήμα 5.13: Ακροατές *"ListSelectionHandler"* και *"List*$_1$*SelectionHandler"* για την διαχείριση των κουμπιών ">>" και "<<", αντίστοιχα.

Όνομα κλάσης: RequestFocusListener	Διασύνδεση που υλοποιούν: AncestorListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	-	-
Μέθοδοι	void	ancestorAdded(final AncestorEvent e)
	void	ancestorMoved(AncestorEvent e)
	void	ancestorRemoved(AncestorEvent e)

Σχήμα 5.14: Ακροατής *"RequestFocusListener"* για την διαχείριση της εστίασης του κέρσορα με την εκκίνηση της εφαρμογής.

Όνομα κλάσης: SaveListener	Διασύνδεση που υλοποιούν: ActionListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	-	-
Μέθοδοι	void	actionPerformed(ActionEvent ae)

Σχήμα 5.15: Ακροατής *"SaveListener"* για την διαχείριση των πλήκτρων αποθήκευσης.

Όνομα κλάσης: SubmitListener	Διασύνδεση που υλοποιούν: ActionListener	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	SubmitThread	st
Μέθοδοι	void	actionPerformed(ActionEvent e)
	void	displayMyPattern()
	void	requestStop()

Σχήμα 5.16: Ακροατής *"SaveListener"* για την διαχείριση του πλήκτρου "ΑΝΑΖΗΤΗΣΗ".

Πέρα από τους ακροατές για τον χειρισμό των γεγονότων, το συγκεκριμένο πακέτο, περιλαμβάνει, όπως έχει ήδη αναφερθεί, κλάσεις για την δημιουργία της κανονικής έκφρασης (όπως η κλάση SubmitResult), η οποία δημιουργείται σε δύο φάσεις. Σε πρώτο επίπεδο, ελέγχεται και δημιουργείται η ανάλογη κανονική έκφραση από το πεδίου κειμένου (textField) της σύνθετης αναζήτησης, ενώ σε δεύτερο επίπεδο, ελέγχονται όλες οι υπόλοιπες παράμετροι, με άμεση ανανέωση της κανονικής έκφρασης.

Επιπλέον, το πακέτο *dictionary* απαρτίζεται από κλάσεις που πραγματοποιούν την τελική ταξινόμηση των λέξεων, όπως παρουσιάζονται στην λίστα των αποτελεσμάτων και στο αρχείο τύπου CSV, ύστερα από το συμβάν (event) της αποθήκευσης. Επιπρόσθετα, περιλαμβάνει κλάση που συσχετίζει τους επιλεγμένους κόμβους του δέντρου με τα αρχεία τύπου "Dictionary" που έχουν αποκωδικοποιηθεί. Τέλος, περιέχει κλάση που χειρίζεται το νήμα (thread) που δημιουργείται στο παρασκήνιο, κατά την διεργασία της ακύρωσης της αναζήτησης. Συνοπτικά:

Όνομα κλάσης: TextFieldPattern	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	Dictionary	dictionarySearch
	String	consonant, vowel, name
	ArrayList<Dictionary>	dicts
Μέθοδοι	Dictionary	myPattern(String name, ArrayList<Dictionary> dicts)
	Dictionary	getDictionarySearch()
	void	setDictionarySearch(Dictionary dictionarySearch)

Σχήμα 5.17: Κλάση αναζήτησης *"TextFieldPattern"* με παραμετροποίηση της μεθόδου της βάσει της κανονικής έκφρασης που έχει κατασκευαστεί και της λίστας των λεξικών που έχουν επιλεγεί.

Όνομα κλάσης: RegularExpression	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	ComboBoxesListener	cbl
	String	myItem, myRegEx
	RadioButtonsListener	rdl
	ArrayList<String>	posInWord, posInSyl, wordsForm, syllablesList, intonationList
Μέθοδοι	String	myRegEx(String myItem, ArrayList<String> posInWord, ArrayList<String> posInSyl, ArrayList<String> wordsForm, ArrayList<String> syllablesList, ArrayList<String> intonationList)
	String	getMyRegEx()
	void	setMyRegEx(String myRegEx)

Σχήμα 5.18: Κλάση *"RegularExpression"* δημιουργίας κανονικής έκφρασης.

Όνομα κλάσης: MergeSort	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	List<String>	Words
Μέθοδοι	List<String>	sort(List<String> Words)
	List<String>	getWords()
	void	setWords(List<String> Words)

Σχήμα 5.19: Κλάση *"MergeSort"* ταξινόμησης ελληνικών λέξεων.

Όνομα κλάσης: FindDic	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	ArrayList<String>	path
	ArrayList<Dictionary>	dicts
Μέθοδοι	ArrayList<Dictionary>	ArrayList<Dictionary> findDic(ArrayList<String> path)
	ArrayList<String>	getPath()
	void	setPath(ArrayList<String> path)
	void	clearDic()

Σχήμα 5.20: Κλάση *"FindDic"* συσχέτισης επιλογών δέντρου αναζήτησης με τα αρχεία λεξικών JSON.

Όνομα κλάσης: SubmitThread	Διασύνδεση που υλοποιεί: Runnable	
	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	boolean	stopFlag
	FindDic	find
	SubmitResult	res
	SubmitDisplayResult	sdr
Μέθοδοι	void	setStopFlag()
	void	run()
	getters και setters για τις μεταβλητές find, res και sdr, οι οποίες προέρχονται από διαφορετικές κλάσεις	

Σχήμα 5.21: Κλάση *"SubmitThread"* υπεύθυνη για την διεργασία του νήματος κατά την ακύρωση της αναζήτησης.

5.2.3 Κλάσεις Πακέτου *dictionary.gui*

Στο πακέτο αυτό ορίζονται όλες εκείνες οι κλάσεις, οι οποίες θα αποδώσουν γραφικά την εφαρμογή της αναζήτησης λεξικού, με στόχο την επικοινωνία του χρήστη με την εφαρμογή. Οι κλάσεις για τις οθόνες της εφαρμογής είναι οι ακόλουθες:

Όνομα κλάσης: SplashScreen	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	-	-
Μέθοδοι	Void	doInBackground()
	void	done()
	void	actionPerformed(ActionEvent evt)
	void	propertyChange(PropertyChangeEvent evt)

Σχήμα 5.22: Μεταβλητές και Μέθοδοι Κλάσης *"SplashScreen"*

Η υλοποίηση γίνεται με την κλάση *"MyDictionaryGUI"*, η οποία περιέχει έναν κατασκευαστή και μεθόδους *get* και *set* για κάθε ένα συστατικό της οθόνης ξεχωριστά. Συνδυάζοντας, επιπλέον, την κλάση *"BackgroundPanel"* ορίζεται η εικόνα φόντου της οθόνης.

Όνομα κλάσης: BackgroundPanel	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	Paint	painter
	Image	image
	int	style
	float	alignmentX
	float	alignmentY
	boolean	isTransparentAdd
Μέθοδοι	void	setImage(Image image)
	void	setStyle(int style)
	void	setPaint(Paint painter)
	void	setImageAlignmentX(float alignmentX)
	void	setImageAlignmentY(float alignmentY)
	void	setTransparentAdd(boolean isTransparentAdd)
	void	paintComponent(Graphics g)

Σχήμα 5.23: Μεταβλητές και Μέθοδοι Κλάσης *"BackgroundPanel"*

Όνομα κλάσης: ProblemGui	
Κατασκευαστής	ProblemGui()

Σχήμα 5.24: Κλάση *"ProblemGui"*

Όνομα κλάσης: HelpDisplay	
Κατασκευαστής	HelpDisplay()

Σχήμα 5.25: Κλάση *"HelpDisplay"*

Όνομα κλάσης: GUI1	
Κατασκευαστής	GUI1()

Σχήμα 5.26: Κλάση *"GUI1"*

Τέλος, με την κλάση *"SubmitDisplayResult"* γίνεται η εμφάνιση των αποτελεσμάτων στην λίστα της κύριας οθόνης της εφαρμογής.

Όνομα κλάσης: SubmitDisplayResult	
Κατασκευαστής	SubmitDisplayResult(final Dictionary name)

Σχήμα 5.27: Κλάση *"SubmitDisplayResult"*

5.2.4 Κλάσεις Πακέτου *checkbox*

Το συγκεκριμένο πακέτο περιλαμβάνει μία κλάση με όνομα *"CheckBoxTreeCellRenderer"*, η οποία υλοποιεί την διαπροσωπεία *"TreeCellRenderer"*.

Όνομα κλάσης: *CheckBoxTreeCellRenderer*	Τύπος Μεταβλητής/Μεθόδου	Όνομα Μεταβλητής/Μεθόδου
Μεταβλητές	TreeCellRenderer	renderer
	JCheckBox	checkBox
	boolean	mouseInCheck
	MouseHandler	handler
	JTree	tree
Μέθοδοι	Component	getTreeCellRendererComponent(JTree tree, Object value, boolean selected, boolean expanded, boolean leaf, int row, boolean hasFocus)
	boolean	isInCheckBox(Point where)
	TreePath[]	getCheckedPaths()

Σχήμα 5.28: Μεταβλητές και Μέθοδοι Κλάσης *"CheckBoxTreeCellRenderer"*

Με τον τρόπο αυτό, υλοποιείται η δομή του δέντρου αναζήτησης με πλαίσια ελέγχου επιλογής, έτσι ώστε να είναι δυνατή η πολλαπλή επιλογή στοιχείων του δέντρου (κάτι το οποίο δεν συμβαίνει σε ένα απλό δέντρο αναζήτησης).

Κεφάλαιο 6

Υλοποίηση Εφαρμογής

Στην παρούσα ενότητα, παρουσιάζεται μέρος του κώδικα της εφαρμογής, ο οποίος μπορεί να εκληφθεί ως σημείο αναφοράς για κάποιες από τις βασικές έννοιες, όπως εκείνη του αντικειμένου και των μεθόδων του, στον αντικειμενοστρεφή προγραμματισμό. Επιπροσθέτως, παρουσιάζεται ο κώδικας υλοποίησης του API της εφαρμογής και διευκρινίζεται η σημασία του.

6.1 Βασικές Έννοιες Κώδικα Java

Αντικείμενο (Object)
Αντικείμενο είναι μία δομή κώδικα με συσχετιζόμενη κατάσταση και συμπεριφορά. Τα αντικείμενα στον προγραμματισμό, συχνά, χρησιμοποιούνται για να παραστήσουν πραγματικά αντικείμενα που βρίσκονται στην καθημερινότητα. Η κατάσταση ενός αντικειμένου περιγράφεται με πεδία(fields), ενώ η συμπεριφορά με κομμάτια κώδικα που ονομάζονται μέθοδοι. Αυτή η δομή προγραμματισμού αναπτύχθηκε για να παρέχεται ενθυλάκωση της πληροφορίας της κατάστασης ενός αντικειμένου (δηλαδή των πεδίων) και της συμπεριφοράς του (data encapsulation). Το αντικείμενο ενθυλακώνει την πληροφορία του και ο μόνος τρόπος αλληλεπίδρασης με αυτό είναι μέσω των μεθόδων του (συμπεριφοράς).

Κλάση (Class)
Η κλάση είναι το πρωτότυπο από το οποίο δημιουργούνται τα αντικείμενα. Στον κώδικα περιγράφονται κλάσεις, οι οποίες μετουσιώνονται σε αντικείμενα κατά την εκτέλεση του προγράμματος. Από ένα πρωτότυπο (κλάση) είναι δυνατόν να δημιουργηθούν πολλαπλά στιγμιότυπα (instances) αντικειμένων με διαφορετικές καταστάσεις, αλλά ίδια πάντα συμπεριφορά που ορίζεται από την κλάση. Με την δημιουργία μιας κλάσης, δημιουργείται ένα στιγμιότυπο

(object) από αυτή. Ο μόνος τρόπος για να δημιουργηθεί ένα τέτοιο στιγμιότυπο είναι μέσα από τον κώδικα κάποιας άλλης κλάσης. Η αρχική κλάση μιας εφαρμογής περιέχει πάντα τον ορισμό μιας μεθόδου ”main” η οποία και εκτελείται πρώτη. Εκεί δημιουργούνται τα πρώτα στιγμιότυπα της εφαρμογής. Η λέξη κλειδί ”new” χρησιμοποιείται για την δημιουργία στιγμιοτύπων.

Συνοψίζοντας, τα πλεονεκτήματα αυτού του σχεδιασμού είναι:

- ✓ Ατομικότητα, ο κώδικας ενός αντικειμένου μπορεί να γραφτεί και να συντηρηθεί ανεξάρτητα από τον κώδικα άλλων αντικειμένων. Μόλις δημιουργηθεί ένα αντικείμενο μπορεί να προωθείται εύκολα μέσα στο σύστημα.

- ✓ Απόκρυψη Πληροφοριών, με την αλληλεπίδραση μόνο με τις μεθόδους του αντικειμένου, οι λεπτομέρειες της εσωτερικής υλοποίησης παραμένουν κρυφές από τον έξω κόσμο.

- ✓ Επαναχρησιμοποίηση κώδικα, εάν ένα αντικείμενο υπάρχει ήδη (ενδεχομένως να έχει γραφτεί από άλλο προγραμματιστή), μπορεί να χρησιμοποιηθεί σε οποιοδήποτε άλλο πρόγραμμα. Αυτό επιτρέπει σε εξειδικευμένους προγραμματιστές να υλοποιούν/δοκιμάζουν/εκσφαλματώνουν πολύπλοκα και συγκεκριμένης λειτουργίας αντικείμενα, τα οποία μπορούν να χρησιμοποιηθούν μεταγενέστερα σε καινούριο κώδικα.

- ✓ Ευκολία σύνθεσης και εκσφαλμάτωσης, αν ένα συγκεκριμένο αντικείμενο αποδειχθεί προβληματικό, μπορεί κανείς να το αφαιρέσει και να προσθέσει στη θέση του ένα άλλο. Όπως στον πραγματικό κόσμο όταν ένα εξάρτημα μιας μηχανής είναι ακατάλληλο ή προβληματικό, τότε αυτό μπορεί να αντικατασταθεί.

6.2 Δείγμα Κώδικα Εφαρμογής

Ολόκληρη η εφαρμογή στηρίζεται στην δημιουργία και περαιτέρω χρήση δυο πολύ βασικών αντικειμένων (objects): της έννοιας της *λέξης* (SimpleWord) και της έννοιας του *λεξικού* (Dictionary).

Το αντικείμενο *SimpleWord* έχει δυο διαφορετικούς κατασκευαστές που δίνουν, αναλόγως, τις αρχικές τιμές στις παραμέτρους του κάθε στιγμιότυπου και μεθόδους που ορίζουν όλες τις ιδιότητές του. Για αντικείμενο τύπου *SimpleWord* έχουμε τις ιδιότητες που προσδιορίζουν πλήρως την έννοια μιας λέξης να είναι: η ίδια η λέξη (word), ο συλλαβισμός της (syllables), το μέρος του λόγου (position), ο αριθμός (number), το γένος (gender), η πτώση (cases), το

πρόσωπο (person), η φωνή (voice) και ο χρόνος (tense).

```
1  package dictionary.handlers;
2  public class SimpleWord implements java.io.Serializable {
3    private static final long serialVersionUID =
       -7552705379953435 22L;
4    private String word, syllables, position, number, gender,
       voice, cases, person, tense;
5    public SimpleWord() {
6      super();
7      this.word = "";
8      this.syllables = "";
9      this.position = "";
10     this.number = "";
11     this.gender = "";
12     this.cases = "";
13     this.person = "";
14     this.voice = "";
15     this.tense = "";
16   }
17   public SimpleWord(String word, String syllables, String
       position,
18       String number, String gender, String voice, String cases,
       String person, String tense) {
19     super();
20     this.word = word;
21     this.syllables = syllables;
22     this.position = position;
23     this.number = number;
24     this.gender = gender;
25     this.cases = cases;
26     this.person = person;
27     this.voice = voice;
28     this.tense = tense;
29   }
30   public String getWord() {
31     return word;
32   }
33   public void setWord(String word) {
34     this.word = word;
35   }
36   public String getSyllables() {
37     return syllables;
38   }
39   public void setSyllables(String syllables) {
40     this.syllables = syllables;
41   }
42   public String getPosition() {
```

```
43      return position;
44    }
45    public void setPosition(String position) {
46      this.position = position;
47    }
48    public String getNumber() {
49      return number;
50    }
51    public void setNumber(String number) {
52      this.number = number;
53    }
54    public String getGender() {
55      return gender;
56    }
57    public void setGender(String gender) {
58      this.gender = gender;
59    }
60    public String getCases() {
61      return cases;
62    }
63    public void setCases(String cases) {
64      this.cases = cases;
65    }
66    public String getPerson() {
67      return person;
68    }
69    public void setPerson(String person) {
70      this.person = person;
71    }
72    public String getVoice() {
73      return voice;
74    }
75    public void setVoice(String voice) {
76      this.voice = voice;
77    }
78    public String getTense() {
79      return tense;
80    }
81    public void setTense(String tense) {
82      this.tense = tense;
83    }
84  }
```

Σε συνέχεια της δημιουργίας της κλάσης που δημιουργεί αντικείμενα τύπου *SimpleWord*, έρχεται η κλάση εκείνη από την οποία δημιουργούνται στιγμιότυπα τύπου *Dictionary*. Πρόκειται για αντικείμενα, τα οποία αποτελούν λί-

στα (ArrayList) αντικειμένων τύπου *SimpleWord*. Όμοια, με παραπάνω, υπάρχει διπλός κατασκευαστής. Ένας για την δημιουργία κενού αντικειμένου και εκείνος στον οποίο αποδίδονται οι αρχικές τιμές της παραμέτρου. Μέθοδοι για την μεταγενέστερη κλήση και ανάκτηση του στιγμιότυπου συμπληρώνουν την κλάση.

```
1  package dictionary.handlers;
2  import java.util.ArrayList;
3  public class Dictionary implements java.io.Serializable {
4    private static final long serialVersionUID =
       3243435789519887458L;
5    private ArrayList<SimpleWord> words;
6    public Dictionary() {
7      super();
8      this.words = new ArrayList<>();
9    }
10   public Dictionary(ArrayList<SimpleWord> words) {
11     super();
12     this.words = words;
13   }
14   public ArrayList<SimpleWord> getWords() {
15     return words;
16   }
17   public void setWords(ArrayList<SimpleWord> words) {
18     this.words = words;
19   }
20 }
```

Στο ίδιο πακέτο, έχουν δημιουργηθεί οι κλάσεις *Save* και *Load* για την κωδικοποίηση και αποκωδικοποίηση, αντίστοιχα, των αρχείων του λεξικού.

```
1  package dictionary.handlers;
2  import java.io.FileOutputStream;
3  import java.io.ObjectOutputStream;
4  public class Save {
5    public Save(Dictionary wordsList, String str){
6            try { // to serialize "wordsList" object to file named
       "str"
7                  FileOutputStream fileOut = new FileOutputStream(
       str + ".ser");
8                  ObjectOutputStream out = new ObjectOutputStream(
       fileOut);
9                  out.writeObject(wordsList);
```

```
                out.close();
                fileOut.close();
                System.out.printf("Serialized data is saved in "
    + str + ".ser");
            }catch(Exception e){
                System.out.println(e);
            }
        }
}
```

```
package dictionary.handlers;
import java.io.FileInputStream;
import java.io.InputStream;
import java.io.ObjectInputStream;
public class Load {
  public Dictionary load(String str){ // load from the file
   named "str"
    Dictionary wordsList = new Dictionary();
    try { //  to deserialize object
      FileInputStream fileIn = new FileInputStream(str + ".ser")
    ;
        ObjectInputStream in = new ObjectInputStream(fileIn);
        wordsList = (Dictionary) in.readObject();
        in.close();
        fileIn.close();
          }catch(Exception e){
        System.out.println(e.toString());
        e.printStackTrace();
          }
          return wordsList;
        }
}
```

6.3 Δημιουργία Διεπαφής Προγραμματισμού Εφαρμογών (API)

Με τον όρο διεπαφή προγραμματισμού εφαρμογών (Application Programming Interface - API) εννοούμε την διεπαφή των προγραμματιστικών διαδικασιών, που παρέχει ένα λειτουργικό σύστημα, βιβλιοθήκη ή εφαρμογή, προκειμένου να επιτρέπει να γίνονται προς αυτά αιτήσεις από άλλα προγράμματα ή/και

ανταλλαγή δεδομένων. Πρόκειται, δηλαδή, για μια διεπαφή λογισμικού προς λογισμικό και όχι για μια διεπαφή χρήστη. Ένας από τους βασικούς σκοπούς μίας διεπαφής είναι να ορίζει και να διατυπώνει το σύνολο των λειτουργιών-υπηρεσιών που μπορεί να παρέχει μια βιβλιοθήκη ή ένα λειτουργικό σύστημα σε άλλα προγράμματα, χωρίς να επιτρέπει πρόσβαση στον κώδικα που υλοποιεί αυτές τις υπηρεσίες. Η διεπαφή διαχωρίζει την προγραμματιστική υλοποίηση κάποιων υπηρεσιών από τη χρήση τους.

Κατασκευάζονται, λοιπόν, οι μέθοδοι της εφαρμογής, με κριτήριο την δημιουργία διεπαφής (API), σύμφωνα με την οποία η αναζήτηση λέξεων πραγματοποιείται ανεξάρτητα από τις υπηρεσίες που προσφέρονται στον χρήστη. Τα πλεονεκτήματα εντοπίζονται στο γεγονός ότι η διάθεση των δεδομένων μέσω API είναι ικανή να βελτιώσει την ποιότητα των δεδομένων και να προσφέρει μεγαλύτερη ευελιξία στην παροχή των υπηρεσιών. Τα δεδομένα μπορούν, με αυτό τον τρόπο, να ενσωματωθούν πιο εύκολα σε άλλες εφαρμογές ή ιστοσελίδες και να διατηρούνται ενημερωμένα με την πάροδο του χρόνου (up-to-date), διασφαλίζοντας, έτσι, ομαλή και ολοκληρωμένη εμπειρία για τον χρήστη.

```
import java.util.ArrayList;
import dictionary.SubmitResult;
import dictionary.handlers.Dictionary;
import dictionary.handlers.Load;
public class TestAPI {
  public static void main(String[] args) {
    ArrayList<Dictionary> dicts = new ArrayList<Dictionary>();
    Dictionary dic = new Dictionary();
    Load loader = new Load();
    dic = loader.load("rhmaEnergEn");
    dicts.clear();
    dicts.add(dic);
    SubmitResult t = new SubmitResult();
    Dictionary res = t.submit("cvcvcv", dicts);
    for (int i = 0; i < res.getWords().size(); i++) {
      System.out.println(res.getWords().get(i).getSyllables());
    }
  }
}
```

Με τον παραπάνω κώδικα, δίνονται τα αποτελέσματα της αναζήτησης λέξεων, από ρήματα ενεργητικής φωνής, ενικού αριθμού, που ικανοποιούν το μοτίβο (pattern) ”σύμφωνο - φωνήεν - σύμφωνο - φωνήεν - σύμφωνο - φωνήεν”.

Σχήμα 6.1: Εμφάνιση μέρους αποτελεσμάτων στην κονσόλα του Eclipse

Κεφάλαιο 7

Επίλογος

7.1 Συμπεράσματα

Αντικείμενο της παρούσας διπλωματικής εργασίας, αποτέλεσε η ανάπτυξη εφαρμογής αναζήτησης λεξικού, με πλήρη εκμάθηση της σύγχρονης δυναμικής γλώσσας προγραμματισμού JAVA, καθώς και εξοικείωση με θέματα αντικειμενοστρεφούς προγραμματισμού, σε ένα από τα πλέον διαδεδομένα ολοκληρωμένα περιβάλλοντα ανάπτυξης εφαρμογών (IDE), το Eclipse.

Η μελέτη των παραπάνω τεχνολογιών πραγματοποιήθηκε με ορίζοντα την εκβάθυνση σε θέματα διαχείρισης και επεξεργασίας αρχείων, κατανάλωσης μεγάλου αποθηκευτικού χώρου στο δίσκο του Η/Υ και μνήμης τυχαίας προσπέλασης (RAM), κατά την ανάκτησή τους από την εκτέλεση κάποιας εφαρμογής.

Καταλληλότερη λύση αποτέλεσε η αποθήκευση των δεδομένων σε αρχεία τύπου JSON. Κυριότερο πλεονέκτημα της μορφής JSON, έναντι άλλων μορφών, αποτέλεσε το γεγονός, τόσο ότι είναι απλή στην σύνταξή και ανάγνωσή της, όσο και το ότι δημιουργεί μικρότερης έκτασης αρχεία.

Για την ανάπτυξη της εφαρμογής της αναζήτησης λεξικού, θεωρήθηκε ωφέλιμη η αξιοποίηση των κανονικών εκφράσεων (regular expressions). Η χρήση τους ενδείκνυται, άλλωστε, σε περιπτώσεις όπου τα δεδομένα βρίσκονται σε τυποποιημένη, αυστηρά ορισμένη, μορφή και προκύπτει ανάγκη αναζήτησης συγκεκριμένου μοτίβου (pattern) μέσα σε ένα σύνολο αλφαριθμητικών τύπων.

7.2 Μελλοντικές Επεκτάσεις

Πιθανές μελλοντικές επεκτάσεις της παρούσας διπλωματικής εργασίας θα μπορούσαν να αφορούν τρία επίπεδα:

✓ *Μετατροπές στην υλοποίηση της εφαρμογής.*

Οι αλλαγές αυτές αναφέρονται στον τρόπο αποθήκευσης και ανάκτησης των δεδομένων. Προτείνεται η δημιουργία μιας συλλογής των τυποποιημένων δεδομένων, δηλαδή η δημιουργία μιας βάσης δεδομένων (database), ώστε να αποτρέπονται περιπλοκές, που πιθανόν προκύπτουν, από την διαχείριση πολλών αρχείων και λιστών. Στόχος, η μελέτη του χρόνου απόκρισης της εφαρμογής στις δυο διαφορετικές περιπτώσεις και τελικά η εξαγωγή συμπερασμάτων.

Μια ακόμα πρόταση αποτελεί ο εμπλουτισμός της εφαρμογής με αρχεία εικόνων και ήχου, για την οπτική αναπαράσταση και ηχητική πληροφορία της προφοράς, της κάθε λέξης του λεξικού. Η πρόταση αυτή αποσκοπεί σε πιο πλούσιες σε πληροφορίες αναζητήσεις, με ευρύτερο οπτικοακουστικό αποτέλεσμα.

✓ *Ανάπτυξη επιπρόσθετων εφαρμογών, ψυχαγωγικού χαρακτήρα, με εκπαιδευτικό περιεχόμενο.*

Η παρούσα εφαρμογή δίνει το έναυσμα ανάπτυξης καινοτόμων ιδεών στον τομέα των προγραμματιστικών εφαρμογών. Με αξιοποίηση των δεδομένων του λεξικού, αξίζει το ενδιαφέρον η ανάπτυξη εφαρμογής, η οποία θα υλοποιεί το παιχνίδι του σταυρόλεξου. Πρόκειται για ένα παιχνίδι μυαλού και γνώσης, στο οποίο ο λύτης επιδιώκει να βρει κρυμμένες λέξεις, οι οποίες διασταυρώνονται μεταξύ τους. Απαραίτητη προϋπόθεση είναι η προσθήκη στο λεξικό των ορισμών των λέξεων, ώστε κατά την εκκίνηση του σταυρόλεξου να περιγράφονται περιφραστικά οι κρυμμένες λέξεις. Η υλοποίηση αποσκοπεί στην τυχαία διαλογή λέξεων σε κάθε νέο παιχνίδι, με προγραμματιστικό τρόπο, ο οποίος θα ικανοποιεί τους κανόνες, σύμφωνα με τους οποίους πρέπει να επιλεγούν και να διασταυρωθούν οι λέξεις. Πρόκειται, δηλαδή, για την υλοποίηση μιας γεννήτριας παραγωγής σταυρόλεξων.

✓ *Επέκταση χρήσης της εφαρμογής.*

Πιθανή επέκταση χρήσης της εφαρμογής θα μπορούσε να αφορά στην μετατροπή της σε mobile εφαρμογή. Αυτό σημαίνει ότι με κατάλληλες τροποποιήσεις, η εφαρμογή θα μπορεί να εγκατασταθεί και να λειτουργήσει σε ”έξυπνα” κινητά τηλέφωνα (smartphones). Επίσης, θα μπορούσε να μετατραπεί

σε εφαρμογή υποστήριξης πολλαπλών γλωσσών (multi-language application) ή ακόμα και σε διαδυκτιακή πλατφόρμα με δημιουργία προσωπικού λογαριασμού και σύνδεση (login) σε αυτόν, με δυνατότητες δημιουργίας προσωπικού προφίλ χρήστη.

Βιβλιογραφία

[1] Jeffrey E. F. Friedl. Mastering regular expressions (3rd edition), 2006.

[2] Μανόλης Τριανταφυλλίδης. Νεοελληνική Γραμματική, 1996.

[3] Ronald L. Rivest Clifford Stein Thomas H. Cormen, Charles E. Leiserson. Εισαγωγή στους Αλγορίθμους (Τόμος i), 2007.

[4] Patrick Niemeyer και Daniel Leuck. Learning java (4th edition), 2013.

[5] David J. Barnes και Michael Kölling. Objects first with java: A practical introduction using bluej (5th edition), 2012.

[6] Παναγιώτης Δ. Μποζάνης. Δομές Δεδομένων, 2006.

[7] Java. https://docs.oracle.com/en/.

[8] Bitbucket. https://bitbucket.org/.

[9] Eclipse ide. http://www.eclipse.org/home/index.php.

[10] Regular expressions. http://www.regular-expressions.info.

[11] Json. http://www.json.org/.