Κατανόηση και τεκμηρίωση του έργου

Περιγραφή της εφαρμογής

Το UPM-Swing είναι μία cross-platform εφαρμογή διαχείρισης στοιχείων σύνδεσης(Usernames και Passwords). Ο χρήστης δημιουργεί μία βάση δεδομένων στην οποία και αποθηκεύει τα στοιχεία σύνδεσης των διαφόρων λογαριασμών που έχει. Η πρόσβαση στην βάση γίνεται με τον κωδικό πρόσβασης που έχει επιλέξει ο χρήστης και το περιεχόμενό της είναι κρυπτογραφημένο με αλγόριθμο AES(BouncyCastle API →bcprov-jdk14-145.jar) που χρησιμοποιείτε από third-party library. Έτσι στο πακέτο crypto της εφαρμογής περιέχονται δύο κλάσεις η EncryptionService.class και η DESDecryptionService.class. Η πρώτη κλάση κρυπτογραφεί το περιεχόμενο της βάσης και η δεύτερη το αποκρυπτογραφεί(χρησιμοποιείται το master password της βάσης) ώστε το περιεχόμενό της να μπορεί να εμφανιστεί στον χρήστη. Επίσης ο χρήστης μπορεί να επιλέξει η βάση να κλειδώνεται αυτόματα και να ζητείται εκ νέου ο κωδικός πρόσβασης όταν η εφαρμογή παραμένει αδρανής για συγκεκριμένο χρονικό διάστημα(μετράται σε λεπτά και την ορίζει ο χρήστης από το μενού των επιλογών). Ακόμη ο χρήστης μπορεί να ορίσει την βάση που θέλει να ανοίγει αυτόματα κατά την εκκίνηση της εφαρμογής, να κάνει import και export την βάση από και σε CSV αρχεία αντίστοιχα(χωρίς κρυπτογράγηση).

Η εφαρμογή στηρίζεται στην βιβλιοθήκη Java.swing της java προκειμένου να παρέχει φιλική διεπαφή χρήστη. Ακόμη οι περισσότερες και βασικότερες λειτουργίες της εφαρμογής μπορούν να εκτελεστούν και μόνο από το πληκτρολόγιο με συνδυασμούς πλήκτρων(Streamline design) αυξάνοντας την ταχύτητα της εφαρμογής. Στο κεντρικό toolbar(MainWindow.class) υπάρχουν δύο Jbuttons που δίνουν την δυνατότητα στον χρήστη να κάνει Copy το username και το password του λογαριασμού που έχει επιλέξει χωρίς να χρειάζεται να ανοίξει την καρτέλα των στοιχείων του λογαριασμού και να κάνει selection και μετά Copy στα πεδία username και password. Οι παραπάνω λειτουργίες μπορούν να πραγματοποιηθούν και από το βασικό pop-up menu(Λογαριασμός → Αντιγραφή Ονόματος Χρήστη + Λογαριασμός → Αντιγραφή Κωδικού Πρόσβασης) της εφαρμογής όσο και με την χρήση μόνο του

πληκτρολογίου(ctr + U → Αντιγραφή Ονόματος Χρήστη και ctr + P →Αντιγραφή Κωδικού Πρόσβασης . Μια ακόμα υπηρεσία που παρέχεται στον χρήστη είναι η γεννήτρια παραγωγής τυχαίων κωδικών πρόσβασης με βάση το επιθυμητό μέγεθος που έχει ορίσει ο χρήστης από το μενού των επιλογών.

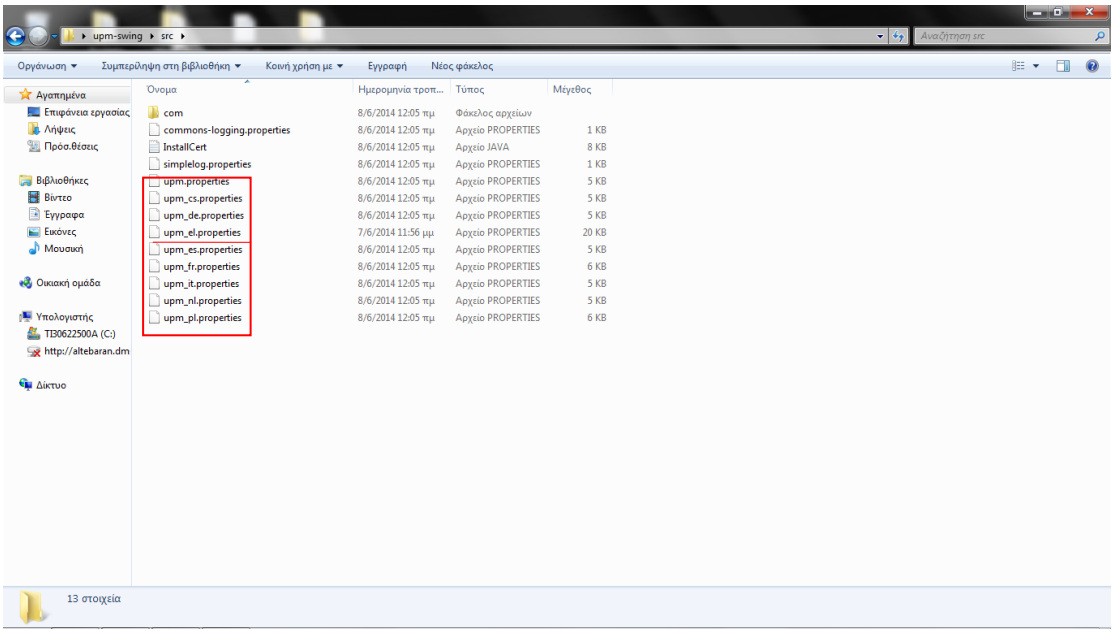
Δίδεται η δυνατότητα η βάση δεδομένων να αποθηκευτεί στο cloud(πχ Dropbox) και ο χρήστης να έχει πρόσβαση σε αυτήν από πολλές συσκευές(διαφορετικά pc με διαφορετικό λειτουργικό + android tablets και smartphones μιας και είναι διαθέσιμη δωρεάν στο Googleplay η εφαρμογή UPM για συσκευές android). Κάθε αλλαγή που κάνει ο χρήστης στη βάση μεταφέρεται σε όλες της συσκευές που χρησιμοποιεί μέσω συγχρονισμού.

Τεχνικά στοιχεία της εφαρμογής

- ✓ Άδεια: GNU General Public License (GPL)
- ✓ Γλώσσα υλοποίησης: java
- ✓ Σύστημα δόμησης Ant(build.xml). Basic Targets: ant compile(μεταγλώτιση πηγαίου κώδικα και κώδικα ελέγχου), ant package-jar(δημιουργία εκτελέσιμου αρχείου jar της εφαρμογής), ant package-tgz(Συμπερίληψη των external libraries –jar στο directory που έχει μεταγλωτιστεί η εφαρμογή ώστε να μπορούν να χρησιμοποιήθούν από την εφαρμογή), ant test(τρέχουν τα Junit tests και δημιουργούνται αρχεία κειμένου στα οποία καταγράφονται τα αποτελέσματα του ελέγχού), ant clean(καθαρισμός του working directory και διαγραφή όλων των αρχείων και των directories που δημιουργήθηκαν κατά την δόμηση της εφαρμογής)
- ✓ web site εφαρμογής: http://sourceforge.net/projects/upm/
- ✓ Git Repository: https://github.com/adrian/upm-swing
- ✓ Μέγεθος(σε γραμμές κώδικα exec:git ls-files | grep .java | xargs wc -l): 6831 γραμμές πηγαίου κώδικα + 1193 γραμμές από Junit tests = Σύνολο 8024 γραμμές
- ✓ 52 java αρχεία που μεταγλωτίζοντας δημιουργούν 104 κλάσεις (*.java και *.class αναζήτηση των windows στο src και test directory της εφαρμογής)

- ✓ Διαθέσιμο σε αρκετές γλώσσες → αγγλικά, τσέχικα, γερμανικά, ιταλικά, ισπανικά, γαλλικά, πολωνικά και ελληνικά(δική μου συνεισφορά)

Δομή Κώδικα



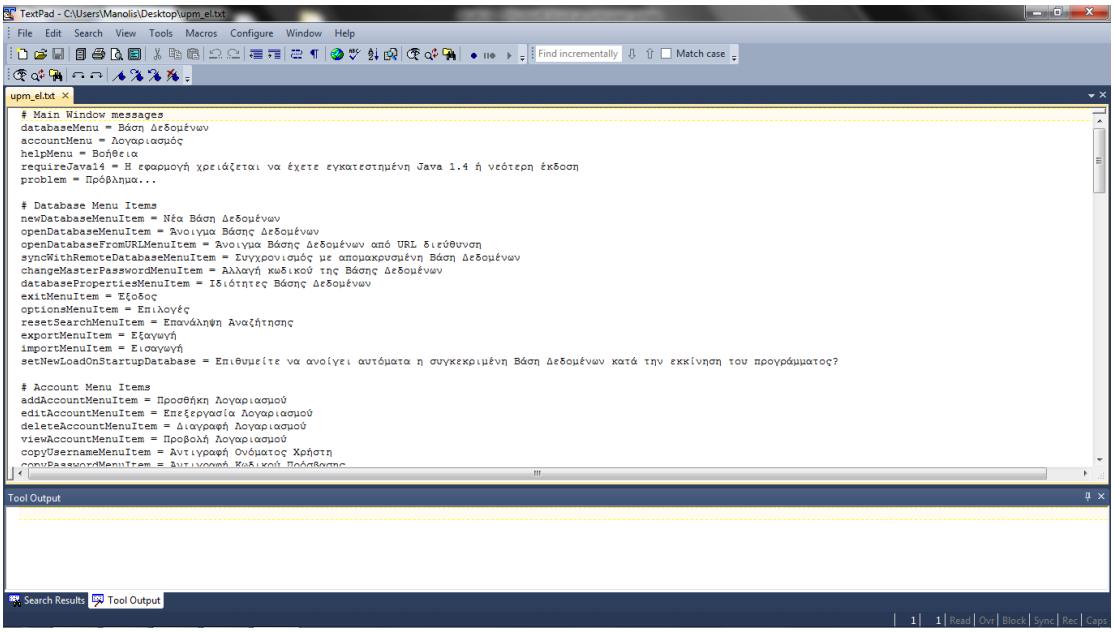
Στο παραπάνω Screenshot κάτω από το directory **src** φαίνονται τα αρχεία που περιέχουν τα μηνύματα της εφαρμογής στις διάφορες γλώσσες που είναι μεταφρασμένα πχ upm_es.properties τα μηνύματα της εφαρμογής στα ισπανικά. Έτσι η διεθνοποίηση της εφαρμογής γίνεται πάρα πολύ εύκολα. Απλά δημιουργείς ένα νέο properties αρχείο στο συγκεκριμένο directory(πχ upm_el.properties για τα μηνύματα της εφαρμογής στα ελληνικά) και μεταφράζεις τα μηνύματα στη συγκεκριμένη γλώσσα. Ακόμη προσθέτεις την καινούργια γλώσσα στην λίστα των υποστηριζόμενων γλωσσών της εφαρμογής(πχ new Locale("el")) στην κλάση Translator του πακέτου util. Ενδεικτικά παρατίθεται ένα μέρος του κώδικα:

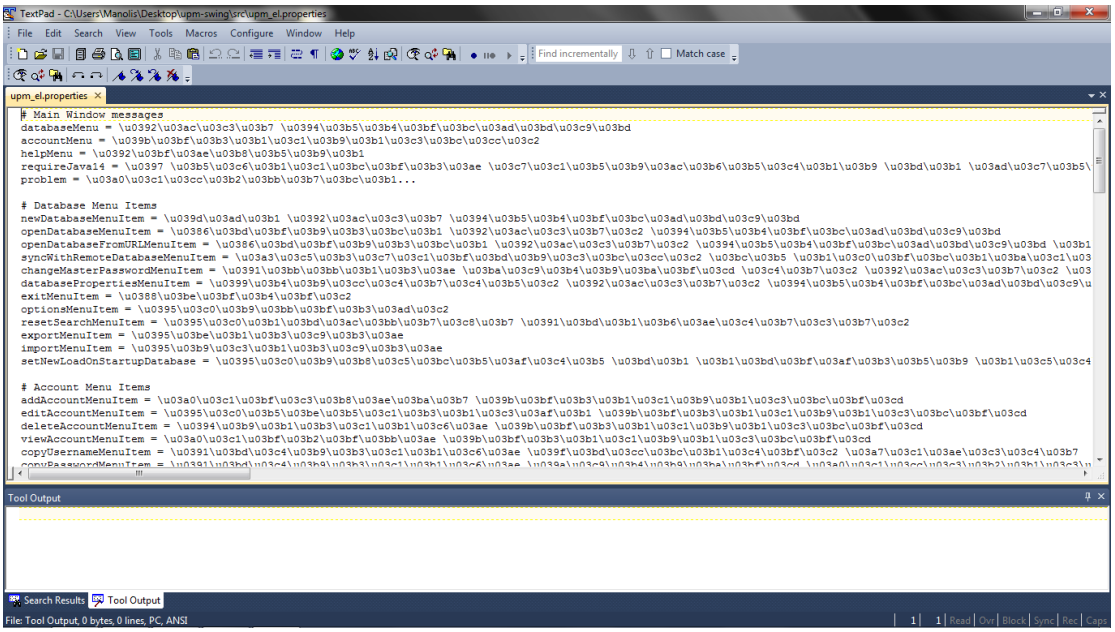
```
package com._17od.upm.util;

import java.text.MessageFormat;

import java.util.Locale;
```

```
import java.util.ResourceBundle;

public class Translator {

    public static Locale[] SUPPORTED_LOCALES = {

        new Locale("cs"), Locale.ENGLISH, Locale.FRENCH, Locale.GERMAN, Locale.ITALIAN, new Locale("es"), new Locale("pl"), new Locale("nl"), new Locale("el")};
```

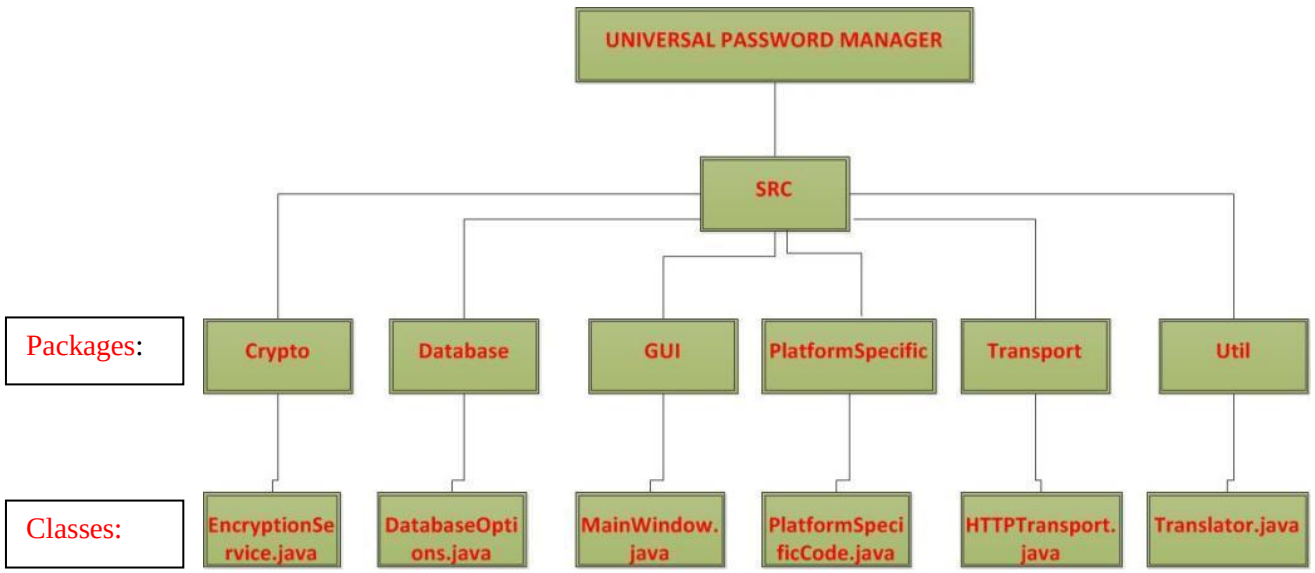
Έτσι για κάθε μήνυμα που εμφανίζει η εφαρμογή καλείται η μέθοδος translate της κλάσης Translator με όρισμα το όνομα του μηνύματος και επιστρέφει το μήνυμα στην γλώσσα που έχει επιλεγεί και αποθηκευτεί στην κλάση Preferences(πχ title = Translator.translate("addAccount");). Αν η προτίμηση γλώσσας είναι null(δηλαδή δεν έχει επιλεγεί ακόμα κάποια συγκεκριμένη γλώσσα από τον χρήστη τότε η εφαρμογή είναι by default στα αγγλικά. Όσον αφορά την προσθήκη της ελληνικής γλώσσας που αποτελεί μέρος της συνεισφοράς μου, εκτός από την μετάφραση των μηνυμάτων στα ελληνικά χρειάστηκε τα μηνύματα να μετατραπούν σε κωδικοποίηση ascii μιας και η κωδικοποίηση της ελληνικής γλώσσας δεν υποστηρίζεται από την java. Έτσι χρησιμοποίησα μια εντολή που υποστηρίζεται από το java SDK την native2ascii από το CMD των windows, για να δημιουργήσω από το αρχείο upm_el.txt(κωδικοποίηση utf-8) που περιέχει τα μηνύματα της εφαρμογής στα ελληνικά το αρχείο upm_el.properties με τα μηνύματα σε κωδικοποίηση ascii.

Ενδεικτικά παρατίθενται τα δύο επόμενα Screenshots:

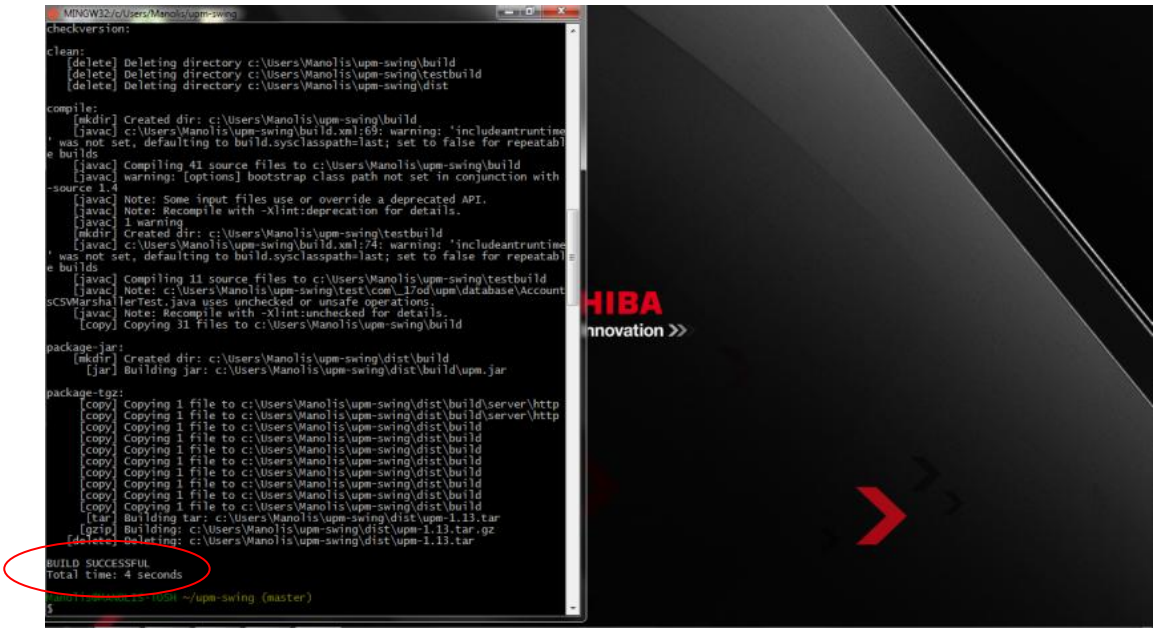




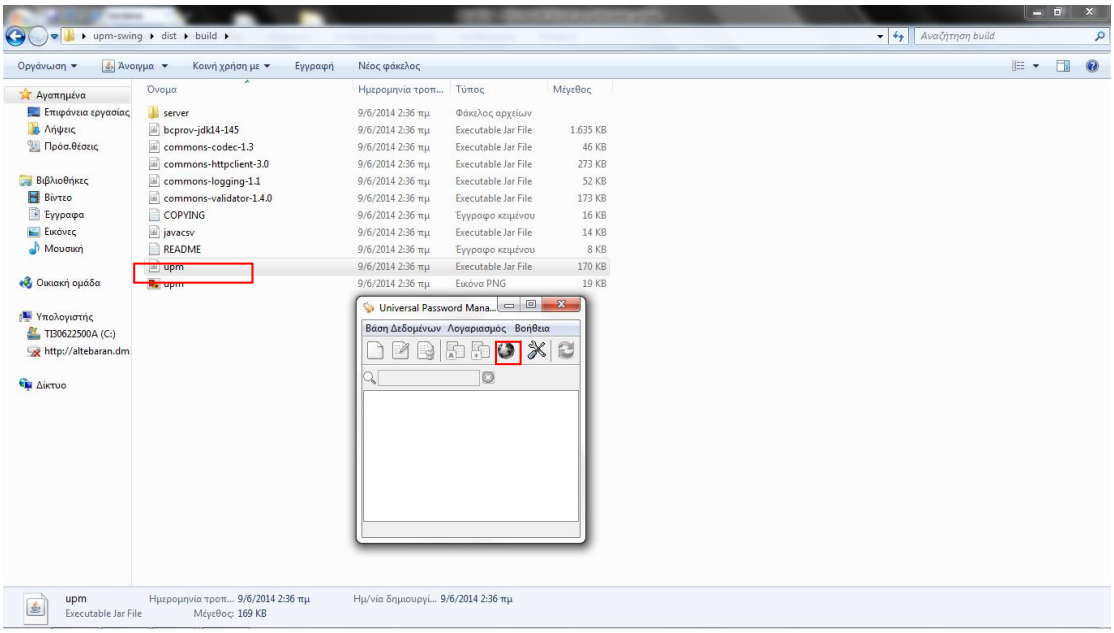
Στο παρακάτω διάγραμμα φαίνεται ο τρόπος που είναι δομημένη η εφαρμογή σε πακέτα(το όνομα του κάθε πακέτου είναι ενδεικτικό των λειτουργιών που υλοποιούνται σε αυτό). Κάτω από κάθε πακέτο έχω παραθέσει μια ενδεικτική κλάση του πακέτου.



Στο παρακάτω Screenshot φαίνεται η επιτυχής δόμηση της εφαρμογής με το εργαλείο ant στοχεύοντας τα targets του αρχείου build.xml(πχ ant compile, ant package-jar).



Στο επόμενο Screenshot φαίνεται το εκτελέσιμο αρχείο (upm.jar) που δημιουργήθηκε και εκκινεί την εφαρμογή. Το μαρκαρισμένο εικονίδιο είναι το disable_icon Launch URL(1^{ο} pull request). Το εικονίδιο που πατάει ο χρήστης για να ανοίξει το url του λογαριασμού που έχει επιλέξει και στην συγκεκριμένη περίπτωση είναι σκούρο γκρι(disable) μιας και δεν έχει επιλεγεί κάποιος λογαριασμός. Όπως θα φανεί στα επόμενα screenshots όταν επιλεγεί κάποιος λογαριασμός το εικονίδιο αλλάζει και γίνεται μπλέ(enable).

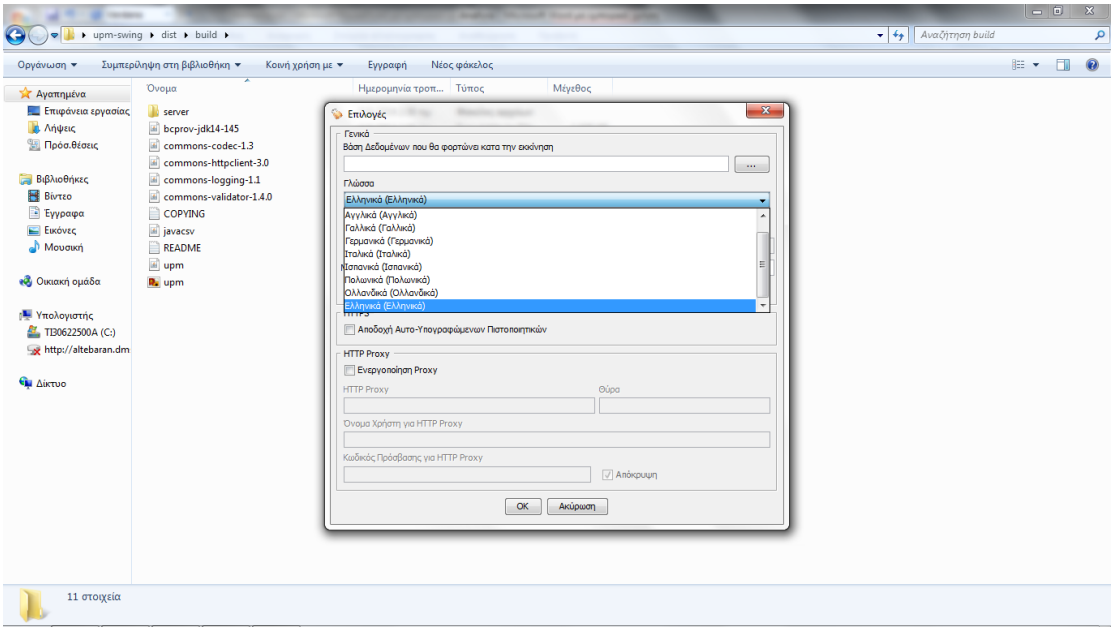


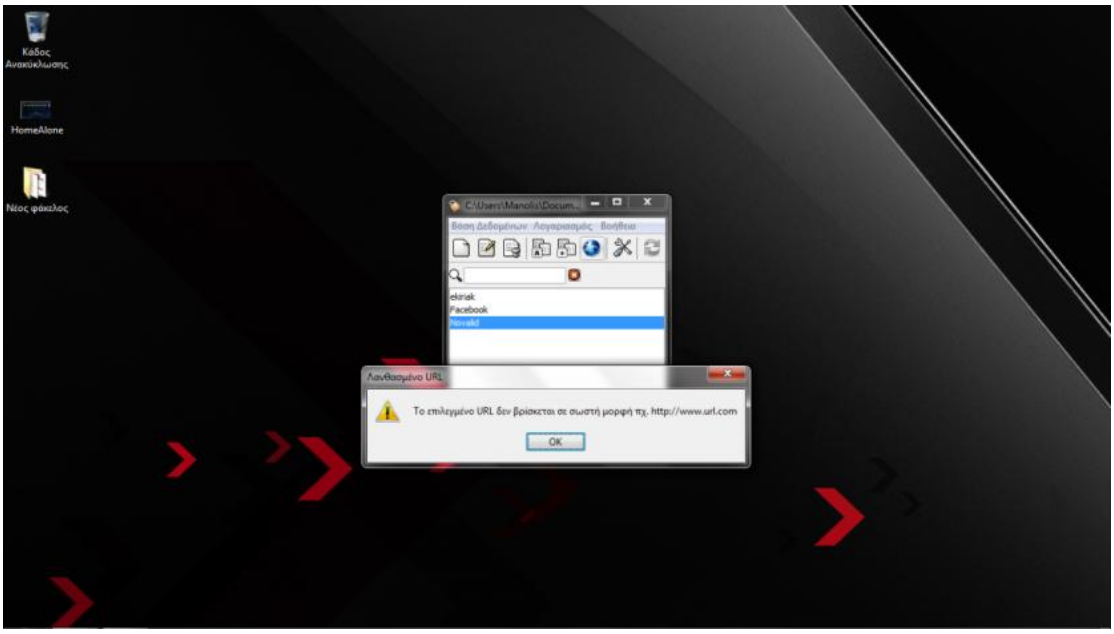
Πλάτος αλλαγών, Ποιότητα υλοποίησης και Ολοκλήρωση

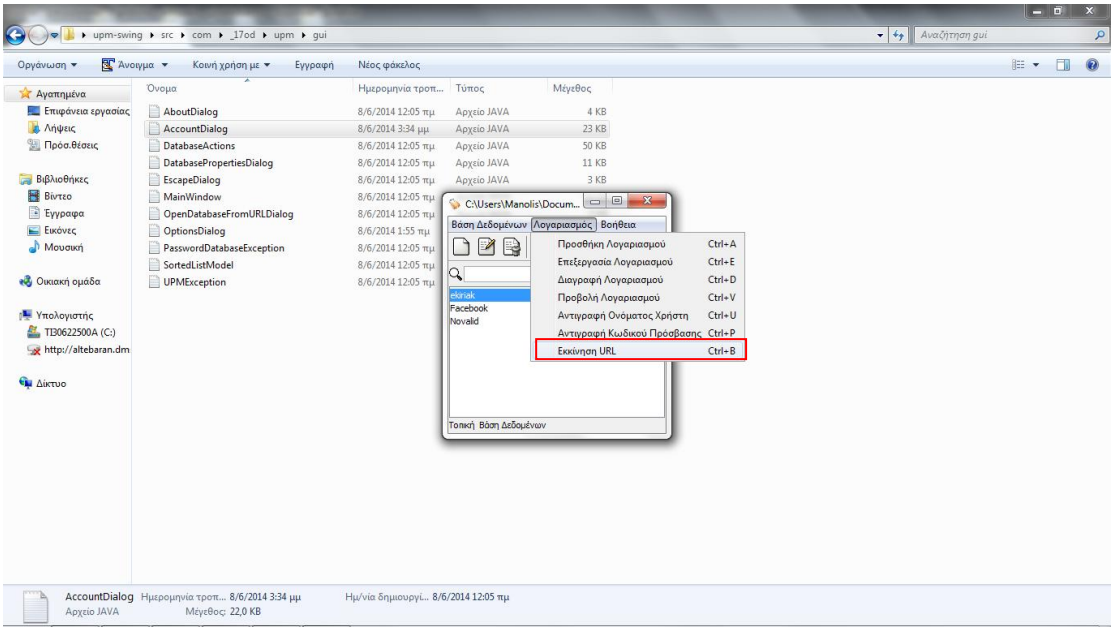
1ο Pull Request(Accepted/Merged)

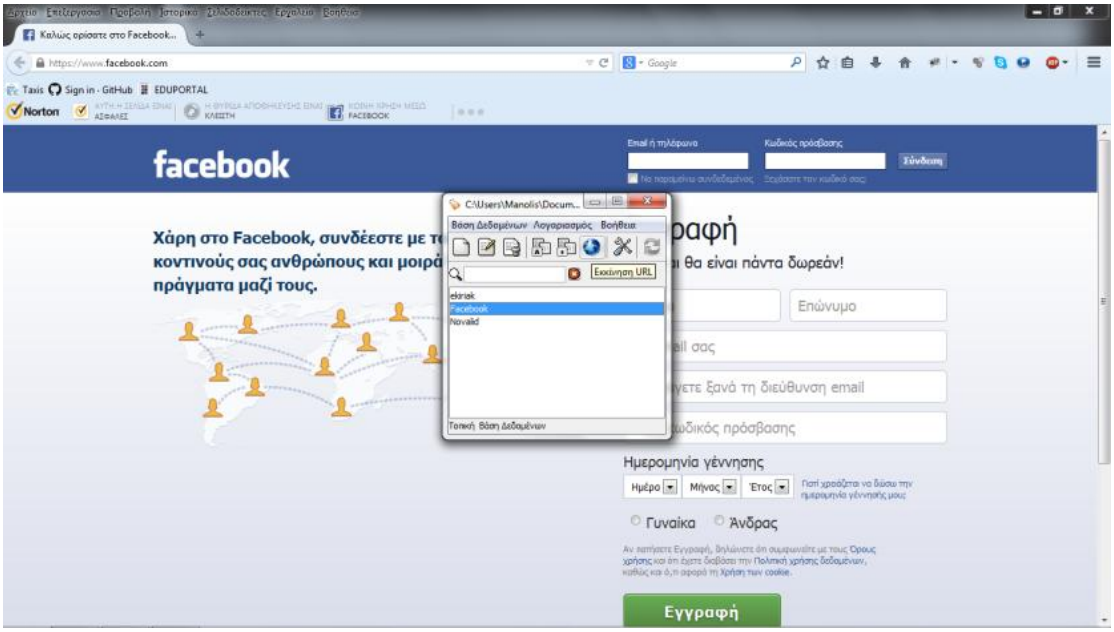
- Μετάφραση των μηνυμάτων της εφαρμογής στα ελληνικά και μετατροπή τους σε κωδικοποίηση ascii (με cmd command native2ascii) πχ databaseMenu=\u039e\u2019\u039e\u00ac\u039f\u0192\u039e\u00b7 \u039e\u201d\u039e\u00b5\u03.
- Προσθήκη εικονιδίου Launch URL στο βασικό toolbar της εφαρμογής. Πατώντας το ο χρήστης, ανοίγει το URL του λογαριασμού που έχει επιλέξει. Αν το συγκεκριμένο URL είναι κενό ή σε μη αποδεκτή μορφή πχ ww.url/com ο χρήστης ενημερώνεται με pop-up message(Joptionpane).

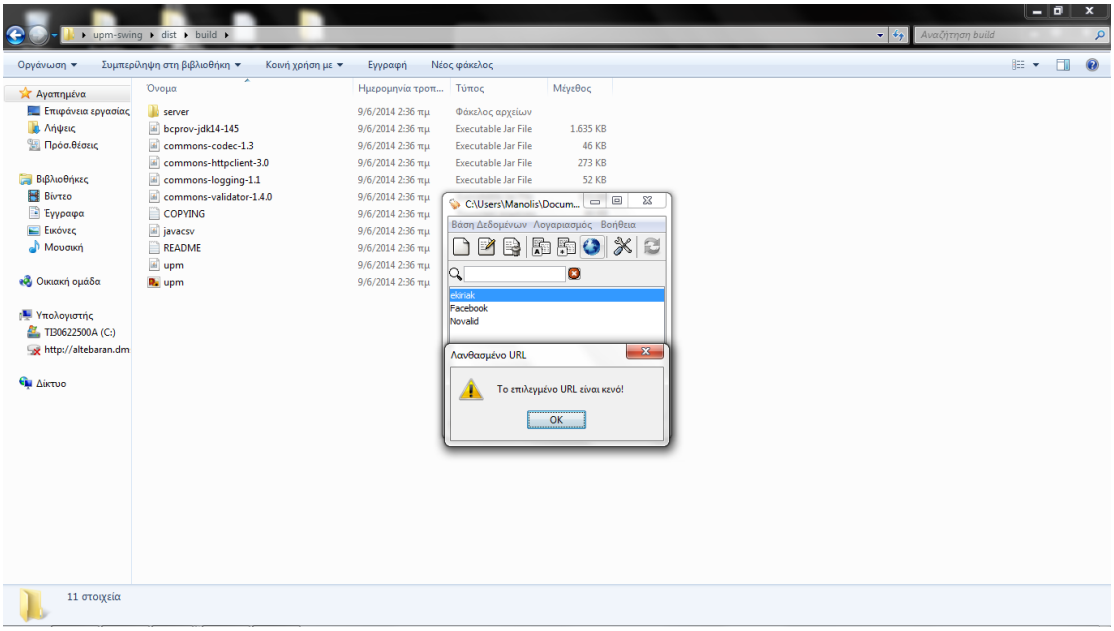
Screenshots της εφαρμογής μετά την υλοποίηση:


upm-swing ▸ dist ▸ build ▸
Αναζήτηση build
Οργάνωση
Συμπερίληψη στη βιβλιοθήκη
Κοινή χρήση με
Εγγραφή
Νέος φάκελος
Αγαπημένα
Επιφάνεια εργασίας
Λήψεις
Πρόσ.θέσεις
Βιβλιοθήκες
Βίντεο
Έγγραφα
Εικόνες
Μουσική
Οικιακή ομάδα
Υπολογιστής
TI30622500A (C:)
http://altebaran.dm
Δίκτυο
Όνομα
Ημερομηνία τροπ...
Τύπος
Μέγεθος
server
bcprov-jdk14-145
commons-codec-1.3
commons-httpclient-3.0
commons-logging-1.1
commons-validator-1.4.0
COPYING
javacsv
README
upm
upm
11 στοιχεία
Επιλογές
Γενικά
Βάση Δεδομένων που θα φορτώνει κατα την εκκίνηση
Γλώσσα
Ελληνικά (Ελληνικά)
Αγγλικά (Αγγλικά)
Γαλλικά (Γαλλικά)
Γερμανικά (Γερμανικά)
Ιταλικά (Ιταλικά)
Ισπανικά (Ισπανικά)
Πολωνικά (Πολωνικά)
Ολλανδικά (Ολλανδικά)
Ελληνικά (Ελληνικά)
Αποδοχή Αυτο-Υπογραφόμενων Πιστοποιητικών
HTTP Proxy
Ενεργοποίηση Proxy
HTTP Proxy
Θύρα
Όνομα Χρήστη για HTTP Proxy
Κωδικός Πρόσβασης για HTTP Proxy
Απόκρυψη
OK
Ακύρωση


Κάδος Ανακύκλωσης
HomeAlone
Νέος φάκελος
Βάση Δεδομένων
Λογαριασμός
Βοήθεια
ekiriak
Facebook
Λανθασμένο URL
Το επιλεγμένο URL δεν βρίσκεται σε σωστή μορφή πχ. http://www.url.com
OK


upm-swing ▸ src ▸ com ▸ _17od ▸ upm ▸ gui
Αναζήτηση gui
Οργάνωση
Άνοιγμα
Κοινή χρήση με
Εγγραφή
Νέος φάκελος
Όνομα
Ημερομηνία τροπ...
Τύπος
Μέγεθος
AboutDialog 8/6/2014 12:05 πμ Αρχείο JAVA 4 KB
AccountDialog 8/6/2014 3:34 μμ Αρχείο JAVA 23 KB
DatabaseActions 8/6/2014 12:05 πμ Αρχείο JAVA 50 KB
DatabasePropertiesDialog 8/6/2014 12:05 πμ Αρχείο JAVA 11 KB
EscapeDialog 8/6/2014 12:05 πμ Αρχείο JAVA 3 KB
MainWindow 8/6/2014 12:05 πμ
OpenDatabaseFromURLDialog 8/6/2014 12:05 πμ
OptionsDialog 8/6/2014 1:55 πμ
PasswordDatabaseException 8/6/2014 12:05 πμ
SortedListModel 8/6/2014 12:05 πμ
UPMException 8/6/2014 12:05 πμ
Αγαπημένα
Επιφάνεια εργασίας
Λήψεις
Πρόσ.θέσεις
Βιβλιοθήκες
Βίντεο
Έγγραφα
Εικόνες
Μουσική
Οικιακή ομάδα
Υπολογιστής
TI30622500A (C:)
http://altebaran.dm
Δίκτυο
C:\Users\Manolis\Docum...
Βάση Δεδομένων
Λογαριασμός
Βοήθεια
Προσθήκη Λογαριασμού Ctrl+A
Επεξεργασία Λογαριασμού Ctrl+E
Διαγραφή Λογαριασμού Ctrl+D
Προβολή Λογαριασμού Ctrl+V
Αντιγραφή Ονόματος Χρήστη Ctrl+U
Αντιγραφή Κωδικού Πρόσβασης Ctrl+P
Εκκίνηση URL Ctrl+B
ekiriak
Facebook
Novalid
Τοπική Βάση Δεδομένων
AccountDialog Ημερομηνία τροπ... 8/6/2014 3:34 μμ Ημ/νία δημιουργί... 8/6/2014 12:05 πμ
Αρχείο JAVA Μέγεθος: 22,0 KB

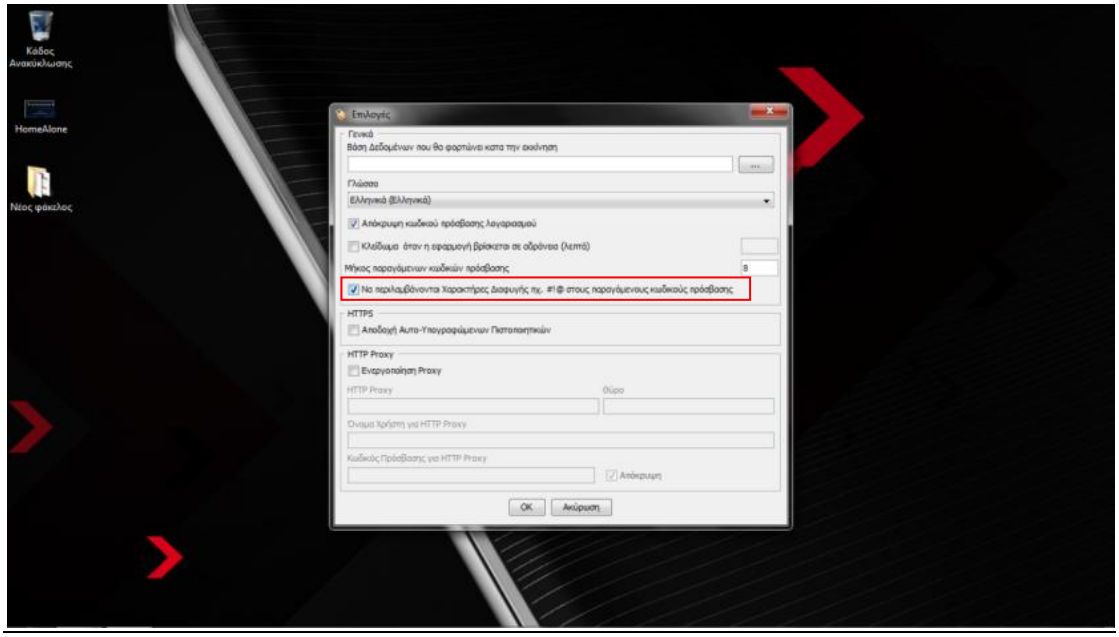

Καλώς ορίσατε στο Facebook...
https://www.facebook.com
Taxis
Sign in · GitHub
EDUPORTAL
Norton
FACEBOOK
facebook
Email ή τηλέφωνο
Κωδικός πρόσβασης
Σύνδεση
Χάρη στο Facebook, συνδέεστε με τ
κοντινούς σας ανθρώπους και μοιρά
πράγματα μαζί τους.
C:\Users\Manolis\Docum...
Βάση Δεδομένων
Λογαριασμός
Βοήθεια
Εκκίνηση URL
ekiriak
Facebook
Novalid
Τοπική Βάση Δεδομένων
Επώνυμο
Ημερομηνία γέννησης
Ημέρα
Μήνας
Έτος
Γυναίκα
Άνδρας
Εγγραφή

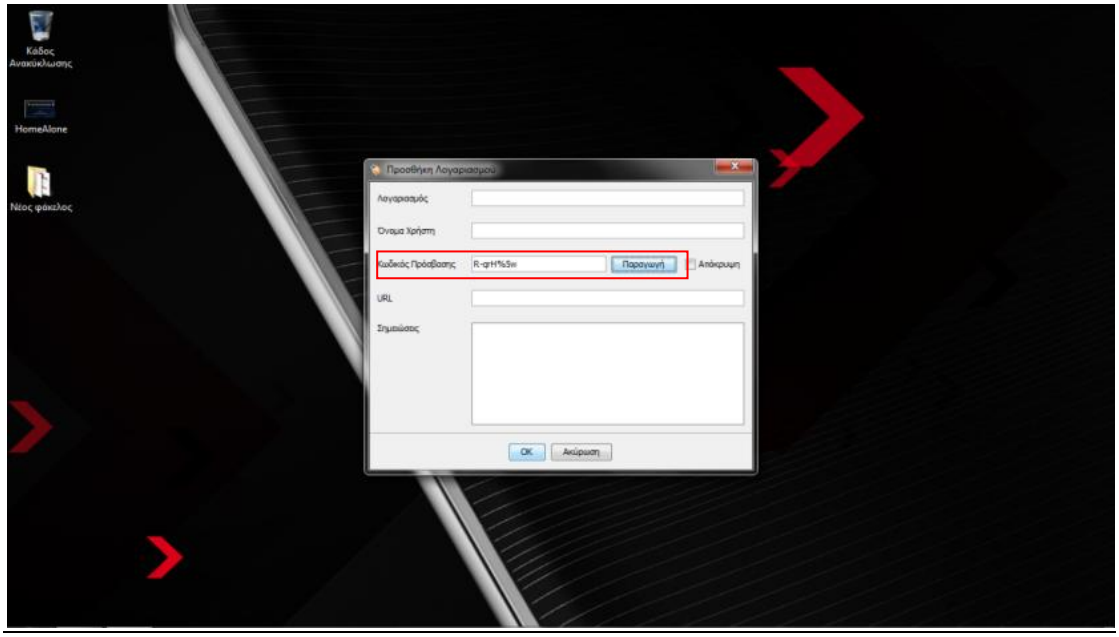


2ο Pull Request(Εν αναμονή αποδοχής-συγχώνευσης)

Προσθήκη checkbox(enable by default) που δίνει την δυνατότητα στον χρήστη να επιλέξει αν θα περιλαμβάνονται χαρακτήρες διαφυγής (!@#) στα password που παράγονται. Επίσης δημιουργία μεθόδων που εξασφαλίζουν ότι το παραγόμενο password σε περίπτωση που ο χρήστης έχει επιλέξει να περιλαμβάνονται χαρακτήρες διαφυγής περιέχει: τουλάχιστον 1 lower case + 1 upper case + 1 number + 1 escape character. Αλλιώς, αν ο χρήστης δεν έχει επιλέξει να περιλαμβάνονται χαρακτήρες διαφυγής περιέχει: τουλάχιστον 1 lower case + 1 upper case + 1 number.

Screenshots της εφαρμογής μετά την υλοποίηση:





Υλοποίηση LaunchURL Function

- Προσθήκη LaunchURL button(+icons) και LaunchURLMenuItem + Keyboard shortcut(ctr + B) B→Browse

- Προσθήκη private void LaunchSelectedURL() (cross platform function)
- Προσθήκη library commons-validator.jar
- Προσθήκη private boolean UrlisValid(SelectedUrl)

Ενδεικτικός κώδικας που έγραψα (κλάση Mainwindow package gui):

```
// The "Launch URL" button

launchURLButton = new JButton();
launchURLButton.setToolTipText(Translator.translate(LAUNCH_URL_TXT));
launchURLButton.setIcon(Util.loadImage("launch_URL.gif"));


launchURLButton.setDisabledIcon(Util.loadImage("launch_URL_d.gif"));;

        launchURLButton.addActionListener(new ActionListener() {

                public void actionPerformed(ActionEvent e) {

                        AccountInformation accInfo = dbActions.getSelectedAccount();

                        String uRl = accInfo.getUrl();

        //Check if the selected  url is null or emty and inform the user via JoptioPane message

        if ((uRl == null) || (uRl.length() == 0)) {
JOptionPane.showMessageDialog(null,Translator.translate("EmptyUrlJoptionpaneMsg"),Translator.translate("UrlErrorJoptionpaneTitle"),JOptionPane.WARNING_MESSAGE);

        //Check if the selected  url is a valid formated url(via urlIsValid() method)  and inform the user via JoptioPane message
```

```
                }     else if (!(urlIsValid(uRl))) {
        JOptionPane.showMessageDialog(null,Translator.translat
e("InvalidUrlJoptionpaneMsg"),Translator.translate("UrlErrorJo
ptionpaneTitle"),JOptionPane.WARNING_MESSAGE);

 //Call the method LaunchSelectedURL() using the selected url
as input

                } else {

                        LaunchSelectedURL(uRl);

            }   }

        });

        launchURLButton.setEnabled(false);

        toolbar.add(launchURLButton);

        toolbar.addSeparator();


// Create LaunchURL JoptionMenuItem

        launchURLMenuItem = new
JMenuItem(Translator.translate(LAUNCH_URL_TXT),
KeyEvent.VK_B);
launchURLMenuItem.setAccelerator(KeyStroke.getKeyStroke(
KeyEvent.VK_B,
Toolkit.getDefaultToolkit().getMenuShortcutKeyMask()));

//ctr+B shortcut

                accountMenu.add(launchURLMenuItem);

                launchURLMenuItem.addActionListener(new
ActionListener() { public void actionPerformed(ActionEvent e)
{
AccountInformation                accInfo                =
dbActions.getSelectedAccount();
                    String uRl = accInfo.getUrl();

                        .

                        .
```

.

```
launchURLMenuItem.setEnabled(false);

launchURLMenuItem.setActionCommand(LAUNCH_URL_TXT);
```

//Method that get(as input) the selected Account URL and open this URL via the default browser of our platform

```
    private void LaunchSelectedURL(String url) {
                if(Desktop.isDesktopSupported()){
                 Desktop desktop = Desktop.getDesktop();
                            try {
                        desktop.browse(new URI(url));
                            } catch (IOException  e) {
                 // TODO Auto-generated catch block
                            e.printStackTrace();
                }catch (URISyntaxException e) {
                        // TODO Auto-generated catch block
                          e.printStackTrace();
                         }
// Linux and Mac specific code in order to launch url
                }else{
              Runtime runtime = Runtime.getRuntime();
                                try {
                                    runtime.exec("xdg-open " + url);
                                    } catch (IOException e) {
```

```
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                      }
}
}
```

```
// Use com.apache.commons.validator library in order to check the validity(proper formating, e.x http://www.url.com) of the given url
    private boolean urlIsValid(String urL) {

                    UrlValidator urlValidator = new UrlValidator();
                    if (urlValidator.isValid(urL)) {
                      return true;
                    } else {
                      return false;
                    }

}
```

<u>Βελτίωση Generate Password function</u>

- Add public static final String INCLUDE_ESCAPE_CHARACTERS στην Preferences.class(αποθήκευση και ανάκτηση επιλογών χρήστη πχ μήκος παραγόμενων κωδικών πρόσβασης, συμπερίληψη η όχι χαρακτήρων διαφυγής κλπ)
- Προσθήκη private static final char[] Extended_Arraylist με επιπλέον 24 χαρακτήρες διαφυγής πχ #$^¨)(><

- Προσθήκη: private static final char[] UPPERCASE_CHARS, private static final char[] LOWERCASE_CHARS, private static final char[] NUMBER_CHARS, private static final char[] ESCAPE_CHARS για τον έλεγχο του παραγόμενου Password
- Προσθήκη private static String GeneratePassword(int PassLength, boolean InclEscChars)
- Προσθήκη private static boolean CheckPassStrong(String Pass, boolean InclEscChars)
- Προσθήκη private static boolean InclUpperCase(String GeneratedPass)
- Προσθήκη private static boolean InclLowerCase(String GeneratedPass)
- Προσθήκη private static boolean InclNumber(String GeneratedPass)
- Προσθήκη private static boolean InclEscape(String GeneratedPass)

Ενδεικτικός κώδικας που έγραψα (κλάση AccountDialog package gui):

```
// Extended CharArray list which include also 24 Escape characters for stronger password generation

    private static final char[] EXTRA_ALLOWED_CHARS = { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z',  '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '~', '!', '@', '#', '$', '%', '^', '&', '*', '(', ',', ')', '_','-', '+', '=', '|', '/', '<', '>', '.', '?', ';', ':' };

        /*We will use the (4)four above CharArray lists(UPPERCASE_CHARS, LOWERCASE_CHARS, NUMBER_CHARS, ESCAPE_CHARS) to ensure that the generated
```

```
        passwords will be more complex. If the user has
selected to include escape characters to generated passwords
and the length of the passwords is 4 or above, then we will
use some methods in order to  generate  passwords that will
have at least 1 lower case + 1 upper case + 1 number + 1
escape character. On the other hand, if the user has not
selected to include escape characters to generated passwords
and the length of the passwords is at least 3, then we will use
methods in order to  generate  passwords that will have at
least 1 lower case + 1 upper case + 1 number.

        */

   private static final char[] UPPERCASE_CHARS = { 'A',
'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q',
'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z' };

   private static final char[] LOWERCASE_CHARS = {'a',
'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r',
's', 't', 'u', 'v', 'w', 'x', 'y', 'z'};

   private static final char[] NUMBER_CHARS = { '0', '1',
'2', '3', '4', '5', '6', '7', '8', '9' };

   private static final char[] ESCAPE_CHARS = { '~', '!',
'@', '#', '$', '%', '^', '&', '*', '(', ',', ')', '_','-', '+', '=', '|', '/',
'<', '>', '.', '?', ';', ':'};


        generateRandomPasswordButton.addActionListener(new
        ActionListener() {public void actionPerformed(ActionEvent
        actionevent) {

 // Get the user's preference about including or not Escape
Characters to generated passwords

Boolean includeEscapeChars = new Boolean(

 Preferences


        .get(Preferences.ApplicationOptions.INCLUDE_ESCAPE_CHAR
        ACTERS, "true"));
```

```
Int pwLength =
Preferences.getInt(Preferences.ApplicationOptions.ACCOUNT_
PASSWORD_LENGTH, 8);

  String Password;


  if ((includeEscapeChars.booleanValue()) &&(pwLength>3)) {

      // Verify that the generated password satisfies the
criteria for strong passwords(including Escape Characters)
do{

Password =
GeneratePassword(pwLength,includeEscapeChars.booleanValu
e());

}while
(!(CheckPassStrong(Password,includeEscapeChars.booleanVal
ue())));

} else if (!(includeEscapeChars.booleanValue())
&&(pwLength>2)) {

        // Verify that the generated password satisfies the
criteria for strong passwords(excluding Escape Characters)

do{

Password =
GeneratePassword(pwLength,includeEscapeChars.booleanValu
e());

}while
(!(CheckPassStrong(Password,includeEscapeChars.booleanVal
ue())));

   } else {

        //     Else a weak password of 3 or less chars will be
produced

 Password =
GeneratePassword(pwLength,includeEscapeChars.booleanValu
e());}
```

```
 password.setText(Password);
        }
       });

    /* The above method takes as input  the user's preferences
    about password length and including or excluding Escape
    Characters and radomly generates
      a password. Then, the method returns the above password
    as a String.
       */
```

private static String GeneratePassword(int PassLength, boolean InclEscChars) {

```
            SecureRandom random = new SecureRandom();
                StringBuffer passwordBuffer = new StringBuffer();
                if (InclEscChars) {
                for(int i=0; i< PassLength; i++) {
        passwordBuffer.append(EXTRA_ALLOWED_CHARS[rand
    om.nextInt(EXTRA_ALLOWED_CHARS.length)]);
                }
        return passwordBuffer.toString();
        }else {
                for(int i=0; i< PassLength; i++) {
        passwordBuffer.append(ALLOWED_CHARS[random.next
    Int(ALLOWED_CHARS.length)]);
            }
       return passwordBuffer.toString();
    }
    }
```

```
/*The above method returns true if the generated password
satisfies the criteria of a strong password

including or excluding Escape Characters. If not, then returns
false.

*/

 private static boolean CheckPassStrong(String Pass,
boolean InclEscChars){

                    if (InclEscChars){

if ((InclUpperCase(Pass)) && (InclLowerCase(Pass)) &&
(InclNumber(Pass)) && (InclEscape(Pass))) {

                                    return true;

                    } else {

                                    return false;

                    }

                } else {

                        if ((InclUpperCase(Pass)) &&
      (InclLowerCase(Pass)) && (InclNumber(Pass)) ) {

                                    return true;

                    } else {

                                    return false;

                    }

                   }

            }



// The above method rerurns true if the generated password
contains at least one Upper Case character. If not, then the
method returns false.
```

```
    private static boolean  InclUpperCase(String
GeneratedPass) {
      char[] PassWordArray = GeneratedPass.toCharArray();
      boolean find = false;
      outerloop:
            for (int i=0; i < PassWordArray.length; i++) {
              for (int j=0; j < UPPERCASE_CHARS.length; j++)
{
                if (PassWordArray[i] == UPPERCASE_CHARS[j])
{
                  find = true;
                  break outerloop;
                }
              }
               }
              if (find) {
                        return true;
               }  else {
                      return false;
               }

   }
```

// The above method rerurns true if the generated password contains at least one Lower Case character. If not, then the method returns false.

private static boolean InclLowerCase(String GeneratedPass) { ……..}

// The above method rerurns true if the generated password contains at least one Number. If not, then the method returns false.

private static boolean InclNumber(String GeneratedPass) {……..}

// The above method rerurns true if the generated password contains at least one Escape character. If not, then the method returns false.

private static boolean InclEscape(String GeneratedPass) {……..}

*Σε όλον τον κώδικα που έγραψα στο πλαίσιο της συνεισφοράς μου έγραψα και τα αντίστοιχα σχόλια που τον τεκμηριώνουν όπως φαίνεται και από τα αποσπάσματα κώδικα που παρατέθηκαν πιο πάνω.

Έλεγχος

Δημιούργησα κάποια Junit tests προκειμένου να γίνει έλεγχος στις τροποποιήσεις που έκανα στην γεννήτρια παραγωγής τυχαίων κωδικών πρόσβασης. Έτσι, προσέθεσα την κλάση TestAccountDialog στο test directory της εφαρμογής(στον φάκελο gui που δημιούργησα) και έγραψα 9 test cases. Λόγω του ότι οι μέθοδοι που ήθελα να τεστάρω ήταν private χρησιμοποίησα μια τεχνική μου ονομάζεται **Reflection.**

Ενδεικτικός κώδικας που έγραψα:

public class TestAccountDialog extends TestCase {

```
//Reflection Strings for Private Method Name
 String InclUpperCaseMethod = "InclUpperCase";
 String InclLowerCaseMethod = "InclLowerCase";
 String InclNumberMethod = "InclNumber";
 String InclEscapeMethod = "InclEscape";
 String GeneratePasswordMethod = "GeneratePassword";

 public void testInclUpperCaseTrue() throws
SecurityException, NoSuchMethodException,
IllegalArgumentException, IllegalAccessException,
InvocationTargetException{
   //Using reflection to test private method
   Class myTarget = AccountDialog.class;
   Class params[] = new Class[1];
   params[0] = boolean.class;
   Method method =
myTarget.getDeclaredMethod(InclUpperCaseMethod,
String.class);
   method.setAccessible(true);
   String Password = "Klfjjru!234";
   Boolean result = (Boolean)method.invoke(method,
Password);
   assertTrue(result);
 }

public void testGeneratePassword() throws
SecurityException, NoSuchMethodException,
IllegalArgumentException, IllegalAccessException,
InvocationTargetException{
```

```
//Using reflection to test private method
Class myTarget = AccountDialog.class;
Class params[] = new Class[1];
params[0] = String.class;
Method method =
myTarget.getDeclaredMethod(GeneratePasswordMethod,
int.class, boolean.class);
method.setAccessible(true);
int Length= 10;
int Length2 = 7;
String result = (String)method.invoke(method, Length,
true);
String result2 = (String)method.invoke(method, Length,
false);
String result3 = (String)method.invoke(method, Length2,
true);
String result4 = (String)method.invoke(method, Length2,
false);


assertEquals(10,result.length(), 0.1);
assertEquals(10,result2.length(), 0.1);
assertNotSame(12,result.length());
assertNotSame(12,result2.length());
assertNotSame(9,result.length());
assertNotSame(9,result2.length());
assertEquals(7,result3.length(), 0.1);
assertEquals(7,result4.length(), 0.1);
```

```
        assertNotSame(12,result3.length());
        assertNotSame(12,result4.length());
        assertNotSame(9,result3.length());
        assertNotSame(9,result4.length());
    }
}
```

Αποτελέσματα που παρήχθησαν από τα Junit tests που έγραψα(ant test target):

Testsuite: com._17od.upm.gui.TestAccountDialog

Tests run: 9, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0,1 sec

1. Testcase: testInclUpperCaseFalse took 0,03 sec
2. Testcase: testInclUpperCaseTrue took 0 sec
3. Testcase: testInclLowerCaseTrue took 0 sec
4. Testcase: testInclLowerCaseFalse took 0 sec
5. Testcase: testInclNumberTrue took 0 sec
6. Testcase: testInclNumberFalse took 0 sec
7. Testcase: testInclEscapeTrue took 0 sec
8. Testcase: testInclEscapeFalse took 0 sec
9. Testcase: testGeneratePassword took 0,06 sec

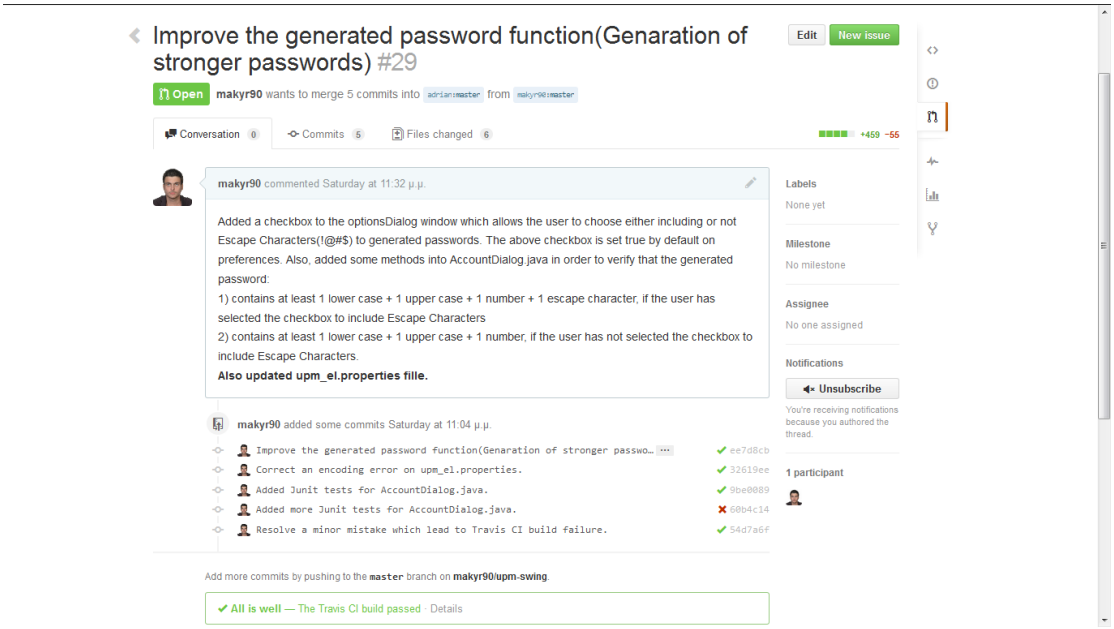
Μέγεθος Συνεισφοράς

- 317 γραμμές κώδικα (μαζί με σχόλια)
- 209 γραμμές κώδικα για Junit tests(9 test cases)
- Προσθήκη 5 νέων μηνυμάτων στο αρχείο μηνυμάτων της εφαρμογής

- Προσθήκη βιβλιοθήκης apache.commons.validator (lib + build.xml)
- Προσθήκη δύο εικονιδίων(enable-disable) για το function Launch URL

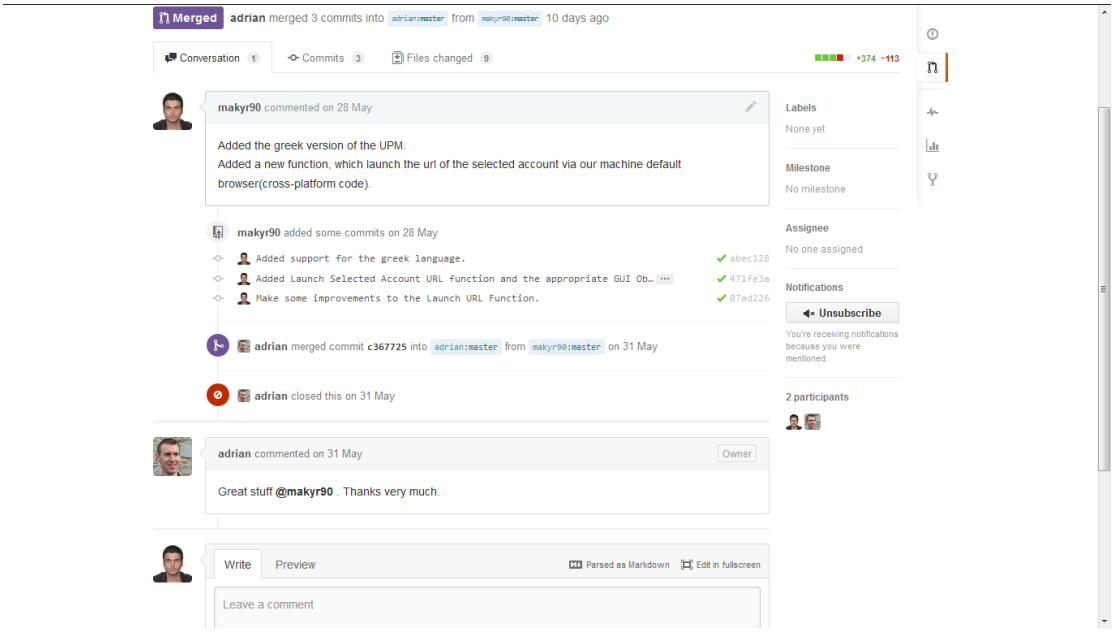
Οργάνωση στο github

Ο τρόπος που δούλεψα στο έργο ήταν ο εξής: Αρχικά έκανα **fork** το έργο στον λογαριασμό μου(https://github.com/makyr90/upm-swing) και από εκεί έκανα git clone και έσπρωχνα τα commits μου. Έπειτα όταν ολοκλήρωνα κάποια αλλαγή έκανα pull request για συγχώνευση(προσθέτοντας λεπτομερή περιγραφή των αλλαγών που έχω κάνει) με το master branch(https://github.com/adrian/upm-swing) της εφαρμογής. Το επόμενο Screenshot δείχνει το 2^{ο} pull request που έκανα και το οποίο βρίσκεται εν αναμονή έγκρισης από τον Author.



Συνεργασία με ομάδα ανάπτυξής

Η συνεργασία με την ομάδα ανάπτυξης περιορίστηκε στην έγκριση-συγχώνευση των αλλαγών που είχα κάνει στην εφαρμογή και ένα πολύ σύντομο ευχαριστήριο μήνυμα για την συνεισφορά μου στο έργο. Το επόμενο Screenshot αφορά την έγκριση-συγχώνευση του 1ου pull request που έκανα.



Κριτική στην δουλειά άλλης ομάδας

Είδα την αλλαγή που έκανε η ομάδα Komatos για το έργο JCloisterZone(https://github.com/farin/JCloisterZone/pull/42/commits) . Η συνεισφορά στο έργο ήταν σχετικά μικρή αλλά χρήσιμη κατά τη γνώμη μου. Αφορούσε ένα παιχνίδι που έπρεπε να ταιριάξεις κομμάτια για να σχηματίσεις ένα είδος παζλ. Τα κομμάτια αυτά περιστρέφονταν έτσι ώστε να μπορούν να ταιριάξουν. Όταν το πλακίδιο δεν ταίριαζε σε κανένα σημείο του παζλ μετά από κάποια περιστροφή τότε «θόλωνε» η εικόνα του για να δείξει στον παίκτη ότι στην συγκεκριμένη μοφή το πλακίδιο δεν μπορεί να ταιριάξει πουθενά. Η συνεισφορά του συνάδελφου Κωνσταντίνου Μάτου (ομάδα Komatos) ήταν να εμφανίζονται στον παίκτη μόνο οι περιστροφές των πλακιδίων που μπορούν να ταιριάξουν στο παζλ. Για παράδειγμα, έστω ένα πλακίδιο που οι 3 από τις 4 περιστροφές

του ταιριάζουν στο παζλ, όταν ό χρήστης το γυρίσει στην 3η περιστροφή και μετά κάνει κλικ να το γυρίσει στην 4η περιστροφή του(μη ταιριαστή μορφή) τότε γίνεται preview ξανά η 1η περιστροφή(ταιριαστή) του πλακιδίου και όχι η 4η περιστροφή θολωμένη.

Ένα ακόμη σημείο που κρίνω θετικά είναι ο έλεγχος που έγινε από τον συνάδελφο με την δημιουργία Junit tests για να εξακριβωθεί ο αντίκτυπος της αλλαγής που πραγματοποίησε στην εφαρμογή ακολουθώντας τον κανόνα "eating your own dog food", δηλαδή την υποχρέωση να ελέγχεις εσύ ο ίδιος τον κώδικα που προσθέτεις ή τροποποιείς.