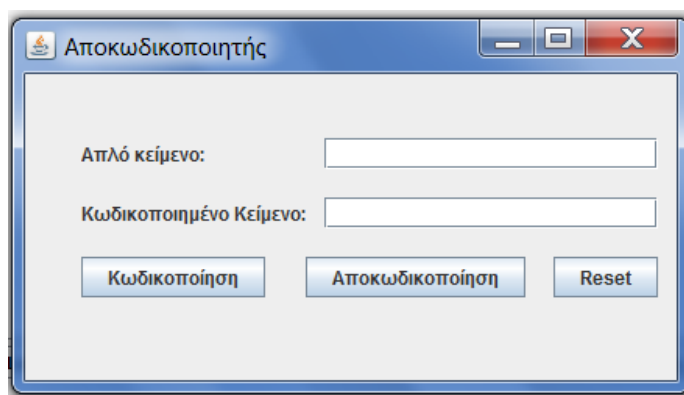


Εργαστήριο 1 - 3^η Άσκηση - Ανάλυση

Εκφώνηση: Δημιουργείστε εφαρμογή υλοποίησης ενός απλού κωδικοποιητή-αποκωδικοποιητή. Η εφαρμογή αποτελείται από τρία κουμπιά (Button) (Encode, Decode, Reset), δύο περιοχές κειμένου (Text Field) (μία για το code και μία για το decode) και δύο ετικέτες (Label) για το χαρακτηρισμό των περιοχών κειμένου. Αυτό που θα κάνει ο κωδικοποιητής είναι όλους τους χαρακτήρες που υπάρχουν στην πάνω περιοχή κειμένου θα τους αλλάζει στον επόμενο χαρακτήρα. Ο αποκωδικοποιητής θα κάνει το ανάποδο.

Υπόδειξη: Χρησιμοποιήστε την τάξη **StringBuilder** για να χρησιμοποιήσετε ένα αλφαριθμητικό ως πίνακα χαρακτήρων και την συνάρτηση **setCharAt()** για να «κυλίσετε» τους χαρακτήρες. Με τη συνάρτηση **getText()** διαβάζουμε κείμενο από ένα αντικείμενο.

Η εφαρμογή θα μοιάζει ως εξής:



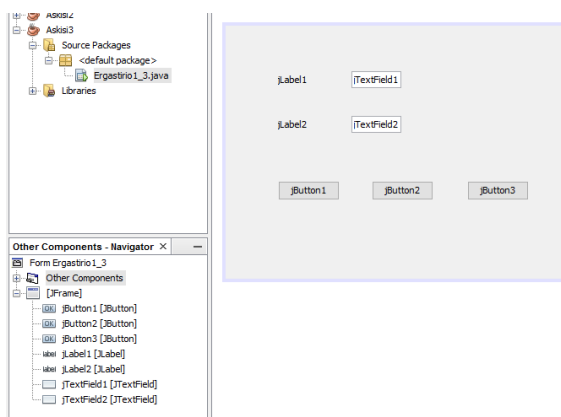
Ανάλυση και εξήγηση λύσης

Βήμα 1: Δημιουργούμε ένα νέο project στο NetBeans με όνομα Askisi3.

Βήμα 2: Δημιουργούμε ένα νέο Java αρχείο τύπου JFrame Form με όνομα Ergastirio1_3.

Βήμα 3: Ορισμός Ιδιοτήτων στο JFrame. Ορίζουμε ως τίτλο του JFrame το «Αποκωδικοποιητής».

Βήμα 4: Πρόσθεση αντικειμένων στον καμβά και μορφοποίησή τους. Προσθέτουμε τρία Button δύο Label και δύο TextField στον καμβά (JFrame) - Εικόνα 1.

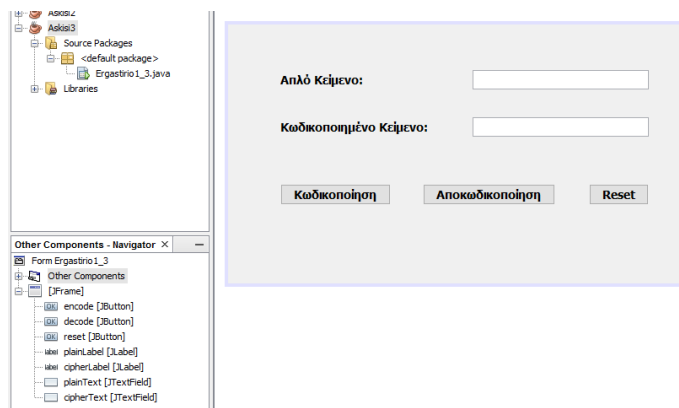


Εικόνα 1: Στο JFrame προστέθηκαν επτά αντικείμενα

Ορίζουμε τις παρακάτω ιδιότητες στα επτά αντικείμενα:

- **JButton1.**
 1. **Text:** «Κωδικοποίηση»
 2. **Variable name:** encode
 3. **Font:** Bold, 14 Size
- **JButton2.**
 1. **Text:** «Αποκωδικοποίηση»
 2. **Variable name:** decode
 3. **Font:** Bold, 14 Size
- **JButton3.**
 1. **Text:** «Reset»
 2. **Variable name:** reset
 3. **Font:** Bold, 14 Size
- **JLabel1.**
 1. **Text:** «Απλό Κείμενο»
 2. **Variable name:** plainLabel
 3. **Font:** Bold, 14 Size
- **JLabel2.**
 1. **Text:** «Κωδικοποιημένο Κείμενο»
 2. **Variable name:** cipherLabel
 3. **Font:** Bold, 14 Size
- **JTextField1.**
 1. **Text:** «»
 2. **Variable name:** plainText
 3. **Font:** 14 Size
- **JTextField2.**
 1. **Text:** «»
 2. **Variable name:** cipherText
 3. **Font:** 14 Size

Η νέα μορφή στον καμβά φαίνεται στην Εικόνα 2.



Εικόνα 2: Το JFrame μετά την υλοποίηση του 4ου Βήματος

Βήμα 5: Ορισμός γεγονότων.

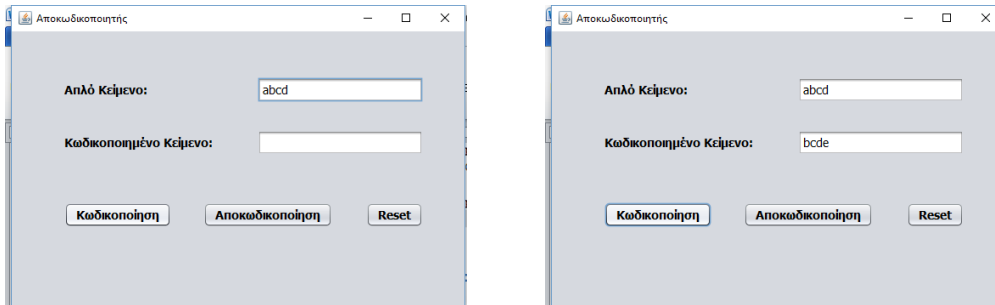
- encode.** Ορίζουμε στο encode το γεγονός Action. Όταν ο χρήστης θα πατάει το encode θα πρέπει να γίνουν τα παρακάτω (Εικόνα 3 / γραμμές 119-125) (Το αποτέλεσμα του γεγονότος φαίνεται στην Εικόνα 4):
 - Να διαβάζει το κείμενο από το plainText. Αυτό γίνεται στη γραμμή 120, στην παράμετρο της StringBuilder με την εντολή `plainText.getText()`.
 - Να μετατρέπει το κείμενο σε αντικείμενο τύπου `StringBuilder`, δηλαδή πίνακα χαρακτήρων (γραμμή 120).
 - Για κάθε κελί του πίνακα εκτελεί την εντολή της γραμμής 122 `pinakasstr.setCharAt(i, (char) (pinakasstr.charAt(i) + 1));` Με την εντολή αυτή τοποθετεί στον πίνακα `pinakasstr` έναν χαρακτήρα (`setCharAt`) στη θέση `i`, αφού πρώτα μετατρέψει τον χαρακτήρα που υπήρχε στη θέση `i` σε ASCII κωδικό (`pinakasstr.charAt(i)`) και αλλάξει τον ASCII κωδικό με τον επόμενο (`pinakasstr.charAt(i) + 1`) ASCII κωδικό. Τον νέο κωδικό τον μετατρέπει σε χαρακτήρα με την `(char)` από το τμήμα του κώδικα `(char) (pinakasstr.charAt(i) + 1)`. Αποτέλεσμα αυτής της γραμμής είναι αν ο χρήστης γράψει το «Α» στο plaintext τότε πατώντας στο encode θα γραφτεί το Β στο cipherText.
 - Εμφανίζει το αποτέλεσμα στο cipherText αφού μετατρέψει το αντικείμενο `pinakasstr` σε κείμενο.
- decode.** Ορίζουμε στο decode το γεγονός Action. Λειτουργεί παρόμοια με το encode. Ο κώδικας του αναπτύσσεται στην Εικόνα 3, στις γραμμές 132-138 και το αποτέλεσμα στην Εικόνα 5
- reset.** Ορίζουμε στο reset το γεγονός Action. Όταν ο χρήστης θα πατάει το reset θα αδειάζουν τα δύο TextField, `plainText` και `cipherText`. Εικόνα 3, γραμμές 127-130.

```

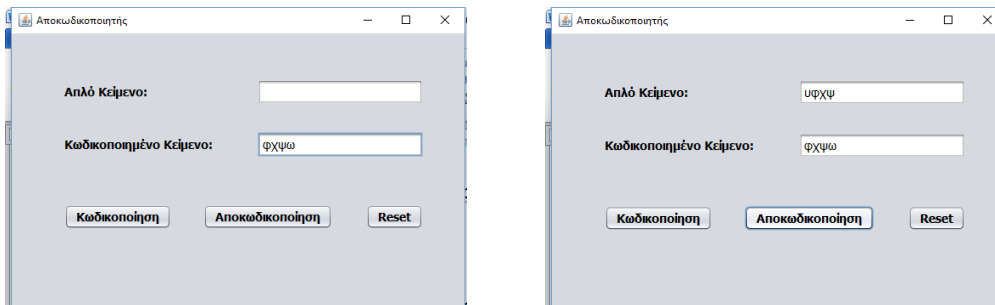
119 private void encodeActionPerformed(java.awt.event.ActionEvent evt) {
120     StringBuilder pinakasstr = new StringBuilder(plainText.getText());
121     for (int i=0; i < pinakasstr.length(); i++) {
122         pinakasstr.setCharAt(i, (char) (pinakasstr.charAt(i) + 1));
123     }
124     cipherText.setText(pinakasstr.toString());
125 }
126
127 private void resetActionPerformed(java.awt.event.ActionEvent evt) {
128     plainText.setText("");
129     cipherText.setText("");
130 }
131
132 private void decodeActionPerformed(java.awt.event.ActionEvent evt) {
133     StringBuilder pinakasstr = new StringBuilder(cipherText.getText());
134     for (int i=0; i < pinakasstr.length(); i++) {
135         pinakasstr.setCharAt(i, (char) (pinakasstr.charAt(i) - 1));
136     }
137     plainText.setText(pinakasstr.toString());
138 }
139

```

Εικόνα 3: Ο κώδικας της εφαρμογής



Εικόνα 4: Η εκτέλεση της κωδικοποίησης. Πριν και μετά.



Εικόνα 5: Η εκτέλεση της αποκωδικοποίησης. Πριν και μετά.