Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Τεχνολογία Πολυμέσων-7ο Εξάμηνο
Αναφορά Εξαμηνιαίας Εργασίας

Όνομα: Χρήστος Παπαδημητρίου
Κωδικός: 03118017

Το IDE που χρησιμοποιήθηκε είναι το IntelliJ IDEA Community Edition 2021.3.1. Το project της εφαρμογής περιέχεται στον φάκελο multi_proj. Η εφαρμογή έχει σχεδιαστεί σε γενικές γραμμές σύμφωνα με την αρχιτεκτονική MVC (Model-View-Controller). Σε αυτό το πλαίσιο έχουμε τρία βασικά αρχεία κώδικα:

- Το hello-view.fxml
- Το Controller.java
- Το HelloApplication.java

Τα τρία αυτά αρχεία αναλαμβάνουν ξεχωριστό κομμάτι της υλοποίησης έκαστο. Παρακάτω γίνεται επιγραμματική αναφορά στον ρόλο κάθε αρχείου:

hello-view.fxml
Έχει το ρόλο της βασικής παρουσίασης της εφαρμογής. Δηλαδή με βάση αυτό φορτώνεται η κύρια οθόνη της εφαρμογής.

HelloApplication.java
Υλοποιεί (κατά βάση) το Model της MVC αρχιτεκτονικής. Περιέχει την κλάση HelloApplication (που κληρωνομεί την application). Εκεί γίνεται η έναρξη (launch) της εφαρμογής όταν τρέχουμε το project. Επίσης μέσα στην κλάση αποθηκεύουμε σε static μεταβλητές τις παραμέτρους που αφορούν την εξέλιξη του παιχνιδιού. Με βάση αυτό, αυτή η κλάση παίζει το ρόλο της βάσης δεδομένων του παιχνιδιού. Επισής οι μέθοδοι της κλάση καλούνται όταν πατάμε κουμπιά στην εφαρμογή για να αλλάξουν αναλόγως την κατάσταση των μεταβητών.

Controller.java
Περιέχει την κλαση Controller που λειτουργεί ως “διαμεσολαβητής” ανάμεσα στο View και το Model (HelloAppication). Συγκεκριμένα περιέχει διάφορες μεθόδους που έχουν συνδεθεί με τα αποκρίσιμα στοιχεία του view (κουμπιά, menu κλπ). Έτσι κάθε φορά που πατάμε ένα αποκρίσιμο στοιχείο μία τέτοια μέθοδος καλείται και υλοποιεί την απόκριση της εφαρμογής στο user input. Οι μέθοδοι της κλάσης Contoller καλούνε μεθόδους (getters) της κλάσης HelloApplication για να “ζητήσουν” δεδομένα που πρέπει να εμφανιστούν στην οθόνη. Επίσης καλούν μεθόδους για να αλλάξουν/ανανεώσουν το model ανάλογα με τις επιλογές του παίκτη.

Επιπλέον υπάρχει ένα αρχείο με όνομα **Dict.java**. Αυτό περιέχει την κλάση Dict, που υλοποιέι λειτουργικότητες που αφορούν τη διαχείρηση των λεξικών. Δηλαδή καλεί το API της OpenLibrary για την κατασκευή νέου λεξικού, ενώ αναλαμβάνει και την αποθήκευσή του. Επίσης, σε αυτό το αρχείο έχουν οριστεί τέσσερις τύποι Exceptions:

- **UndersizeException** (προκαλείται όταν το description του βιβλίου που μας ενδιαφέρει δίνει λιγότερες από 20 λέξεις – πχ: OL25121728M)
- **UnbalancedException** (προκαλείται όταν στο βιβλίο που μας ενδιαφέρει υπάρχουν λέξεις με 9 και πάνω γράμματα σε ποστοστό λιγότερο από 20% - πχ: OL32784436M)
- **InvalidDescriptionException** (προκαλείται όταν το βιβλίο που μας ενδιαφέρει δεν έχει description, ή το description δεν έχει πεδίο value – πχ: OL16991264M)
- **NotFoundException** (προκαλείται όταν δεν υπάρχει βιβλίο με το ζητούμενο OpenLibraryID)

Ως προς την παρουσίαση του παιχνιδιού, τα menu Application και Details έχουν προστεθεί σε ένα menu bar στο πάνω μέρος της οθόνης και διαθέτουν όλα τα ζητούμενα menu items. Η οθόνη έχει χωριστεί σε 4 κομμάτια.

- Όταν ένα παιχνίδι βρίσκεται σε εξέλιξη, στο πάνω μέρος της οθόνης βλέπουμε τα ζητούμενα στατιστικά για το παιχνίδι.
- Το μεσαία μέρος της οθόνης είναι χωρισμένο σε δύο μέρη. Στο αριστερό βλέπουμε μια εικόνα που αντιστοιχεί στο στάδιο που βρισκόμαστε μέχρι να ολοκληρωθεί το κρέμασμά μας. Υπάρχουν 7 τέτοιες εικόνες. Η πρώτη αντιστοιχεί στο να μην έχει γίνει κανένα λάθος. Οι επόμενες 6 αντιστοιχούν στα διαδοχικά στάδια του κρεμάσματος. Κάτω από την εικόνα παρουσιάζεται η κρυμμένη λέξη, όπου τα γράμματα που δεν έχουν ακόμα βρεθεί αντικαθιστούντε με παύλες. Στο δεξί μέρος της οθόνης έχουμε Drop Down Menus, ένα για κάθε γράμμα της λέξης. Κάθε Drop Down Menu περιλαμβάνει όλους του πιθανούς χαρακτήρες για το συγκεκριμένο γράμμα στη λέξη και την αντίστοιχη πιθανότητα σε παράνθεση. Τα Menu αυτά δεν είναι για να επιλέξουμε χαρακτήρα, αλλά μόνο για να δούμε τις επιλογές μας.
- Η επιλογή χαρακτήρα γίνεται στο κάτω μέρος της οθόνης. Εκεί υπάρχουν δύο Text Fields. Στο ένα βάζουμε την θέση στη λέξη, για την οποία θέλουμε να μαντέξουμε. Στο δεύτερο βάζουμε τον χαρακτήρα που πιστεύουμε ότι είναι ο σωστός.

Όλες οι ζητούμενες λειτουργικότητες έχουν υλοποιηθεί. Ακολουθούν κάποιες παραδοχές που έγιναν:

- Οποιαδήποτε στιγμή επιλεγεί νέο λεξικό (μέσω του menu Load) η κατάσταση της εφαρμογής γίνεται reset. Αυτό σημαίνει ότι αν υπάρχει κάποιο παιχνίδι σε εξέλιξη, αυτό διακόπτεται. Επίσης όλες οι λεπτομέριες για προηγούμενα ολοκληρωμένα παιχνίδια (με προηγούμενο λεξικό) διαγρλαφονται. Με λίγα λόγια επιστρέφουμε σε καταστασηη σαν αν είχαμε μόλις ανοίξει την εφαρμογή.
- Κατά την εξέλιξη του παιχνιδού, στη λίστα με τους ενεργούς χαρακτήρες για κάθε θέση συμπεριλαμβάνονται μόνο χαρακτήρες που έχουν θετική πιθανότητα να είναι σωστοί, σύμφωνα με το ενεργό λεξικό. Δηλαδή, δεν συμπεριλαμβάνονται χαρακτήρες με πιθανότητα 0.
- Δεδομένου ότι δεν συμπεριλαμβάνουμε χαρακτήρες με πιθανότητα 0, υπάρχει η περίπτωση μετά από κάποιες σωστές μαντεψιές να αποκλειστούν όλες οι άλλες πιθανές λέξεις από το λεξικό και να γίνει φανερό ποιά είναι η κρυμμένη λέξη πριν μαντέψουμε όλα τα γράμματα της λέξης.

Ακολουθεί η περιγραφή μίας απλής ακολουθίας κινήσεων για να φανεί η λειτουργικότητα της εφαρμογής:

- Επιλέγουμε Application → Create
- Στο Pop-Up παράθυρο εισάγουμε 1 ως Dictionary ID και OL31390631M ως Open Library ID.
- Επιλέγουμε Application → Load
- Στο Pop-Up παράθυρο εισάγουμε 1 ως Dictionary ID
- Επιλέγουμε Application → Start
- Στο κουτάκι position βάζουμε 1 (πρώτο γράμμα της λέξης)
- Στο κουτάκι position βάζουμε “a” (δεδομένου ότι όλες οι λέξεις είναι με κεφαλαία αυτή θα είναι πάντα λάθος επιλογή)
- Πατάμε 6 φορές το κουμπί Go και βλέπουμε τις φάσεις του παιχνιδιού μέχρι την ήττα μας

Σημειώνεται ότι η κλάση της οποίας οι public μέθοδοι έχουν τεκμηριωθεί σύμφωνα με τις προδιαγραφές του εργαλείου javadoc είναι η κλαση HelloApplication (που βρίσκεται στο αρχείο HelloApplication.java). Επίσης, υπάρχει η περίπτωση να πρέπει να προστεθεί ξανά στα dependencies το αρχείο json-simple-1.1.1.jar για να τρέξει η εφαρμογή σε άλλον υπολογιστή, καθώς το έχω προσθέσει στο project στον υπολογιστή μου, αλλά η αλλαγή περιβάλλοντος ίσως δημιουργήσει πρόβλημα. Πάντως το αρχείο συμπεριλαμβάνεται στο zip που παραδίδω, στον φάκελο json-simple. Η προσθήκη του στα dependencies γίνεται (στο IntelliJ) μέσω του μενού File>Project Structure>Modules>Dependencies και πατώντας στο Add(+).