Τελική εργασία στο μάθημα Τεχνολογία Πολυμέσων

Περιγραφή υλοποίησης

Νικόλαος Φρυγανίωτης - 03116444 (nikolasfryganiotis@yahoo.com)

Περιγραφή Υλοποίησης

Η εφαρμογή χρησιμοποιεί την κλάση Main.java η οποία φορτώνει το βασικό παράθυρο της εφαρμογής(Main.FXML). Η κλάση που υλοποιεί ασύγχρονα μεθόδους που καλούνται από την αλληλεπίδραση του χρήστη με την κύρια οθόνη(πατώντας διάφορα κουμπιά ή χρησιμοποιώντας το MenuBar) είναι η MainController.Java. Από την εφαρμοφή δημιουργούνται και άλλα παράθυρα pop up, όπως το Load.FXML και άλλα που θα αναλύσουμε στη συνέχεια. Η ανάπτυξη της εφαρμογής έγινε στο eclipse IDE με έκδοση της Java JDK-15.0.1 και χρήση του JavaFX-SDK-15.0.1.
Για την υλοποίηση της γραφικής απεικόνισης της εφαρμογής χρησιμοποιήθηκαν:

- AnchorPane
- MenuBar
- Menu
- MenuItem
- Label
- Pane
- Button
- TextField
- VBox
- TableView
- TableColumn

Για την προσομοίωση έχουν δημιουργηθεί οι παρακάτω κλάσεις με βάση τις προδιαγραφές που δόθηκαν στην εκφώνηση.

- Για την υλοποίηση των χαρακτηριστικών του παίκτη, είτε είναι ο player(που θεωρούμε ότι είναι ο παίχτης του οποίου βλέπουμε το ταμπλό και ουσιαστικά εκτελούμε για εκείνον τις βολές), είτε ο enemy(του οποίου το ταμπλό δεν βλέπουμε) χρησιμοποιείται η κλάση Player. Τα χαρακτηρηστικά αυτά αφορούν το πλήθος ενεργών πλοίων κάθε παίκτη, το πλήθος ενεργειών που έχει κάνει, τους συνολικούς του πόντους, το ποσοστό επιτυχιών του στις βολές, τις θέσεις των ενεργών του πλοίων, τα εναπομείναντα μεγέθη κάθε πλοίου, την κατάσταση κάθε πλοίου του, το αν έχει παίξει σε αυτόν τον γύρο, τον αριθμό των επιτυχημένων βολών του και το αποτέλεσμα της βολής του αντιπάλου του σε αυτόν τον γύρο(αν χτυπήθηκε πλοίο του σε αυτό τον γύρο, και αν ναι ποιο).
- Για την υλοποίηση του Load του ID που επιλέξαμε χρησιμοποιήθηκε η κλάση ReadFile.java.
- Για την τοποθέτηση των πλοίων, όσον αφορά τη γραφική διεπαφή, στο 10×10 ταμπλό χρησιμοποιήθηκε η κλάση Piece.java.
- Για την τοποθέτηση των πλοίων στο εσωτερικό του, και για τον έλεγχο περί ορθότητας των θέσεων που δόθηκαν στα πλοία(διαφορετικά αν δημιουργείται πρόβλημα κατά την τοποθέτηση των πλοίων δημιουργεί μια εξαίρεση, την οποία χειρίζεται δίνοντας το κατάλληλο μήνυμα στην κονσόλα),χρησιμοποιείται η κλάσση Table.Java.
- Στην περίπτωση που λείπει πληροφορία για κάποιο πλόιο είτε του player είτε του enemy εμφανίζεται το μήνυμα "There is some information missing for the input". Π.χ. αν για κάποιο πλοίο δεν δίνεται ο τύπος του. Ή αν λείπει όλη η πληροφορία για κάποιο πλοίο. Την εξαίρεση αυτή υλοποιεί η κλάση MissingInputException.java.
- Στην περίπτωση που ένα πλοίο βγαίνει εκτός των ορίων του ταμπλό εμφανίζεται το μήνυμα "The ship is out of table's bounds".Επίσης τυπώνει τις συντεταγμένες που δημιουργούν πρόβλημα. Την εξαίρεση αυτή υλοποιεί η κλάση OversizeException.java.

- Στην περίπτωση που ένα πλοίο τοποθετείται σε κελί που ήδη έχει άλλο πλοίο εμφανίζεται το μήνυμα "Ship in the same position". Την εξαίρεση αυτή υλοποιεί η κλάση OverlapTilesException.java.

- Στην περίπτωση που ένα πλοίο εφάπτεται κάθετα ή οριζόντια με άλλο πλοίο ,έστω και σε ένα κελί, εμφανίζεται το μήνυμα "Tangent ships". Την εξαίρεση αυτή υλοποιεί η κλάση AdjacentTilesException.java.

- Στην περίπτωση που υπάρχουν περισσότερα από ένα πλοία για κάθε τύπο εμφανίζεται το μήνυμα "There is more than one ships for a type of ship". Την εξαίρεση αυτή υλοποιεί η κλάση InvalidCountException.java.

- Για την υλοποίηση του πίνακα που αφορά την κατάσταση των πλοίων του αντιπάλου χρησιμοποιείται η κλάση EnemyShips.java. Η κλάση αυτή έχει ως attributes τον τύπο του πλοίου(Carrier,Battleship κ.λ.π.) και την κατάσταση του. Αν το πλοίο είναι ανέπαφο, η κατάσταση του είναι Intact(αυτό εμφανίζεται στον πίνακα). Αν η κατάσταση του είναι χτυπημένο, τότε η κατάσταση του θα είναι Chipped. Αλλιώς αν είναι βυθισμένο η κατάσταση του θα είναι Sunk.

- Για την υλοποίηση των πινάκων των τελευταίων 5 βολών του player και του enemy χρησιμοποιείται η κλάση PlayerShots.java. Η κλάση αυτή έχει ως attributes τις συντεταγμένες x,y του σημείου κάθε βολής, το αποτέλεσμα της βολής(αν είναι Successful Shot ή Unsuccessful Shot), και αν έχουμε Successful Shot δίνεται και ο τύπος του πλοίου που χτυπήθηκε.

Απομένουν άλλες 3 κλάσεις να αναφέρουμε, τις οποίες θα εξετάσουμε στην συνέχεια αναφερόμενοι στις παραδοχές που έχουν γίνει αλλά και στο πως υλοποιήθηκε η συγκεκριμένη εφαρμογή και το αποτέλεσμα της υλοποίησης στην γραφικό περιβάλλον. Οι κλάσεις αυτές είναι η Main.java, MainController.java, LoadController.java.

Απαιτήσεις

- Αρχικά μέσω της Main Class γίνεται η δημιουργία του simulation instance μέσω του public constructor του simulation. Αυτό που γίνεται είναι να δημιουργείται το κύριο παράθυρο της εφαρμογής και να φορτώνεται το αρχείο Main.FXML που υλοποιεί την γραφική απεικόνιση του βασικού παραθύρου. Στο πάνω μέρος της γραφικής απεικόνισης βλέπουμε να υπάρχει ένα MenuBar, ένας μετρητής για τους γύρους(ROUND), και επίσης κάποια χαρακτηρηστικά για τον player και για τον enemy. Στο μεσαίο κομμάτι της γραφικής διεπαφής φαίνονται 2 10×10 grids που στην αρχή είναι άδεια, καθώς δεν έχουμε φορτώσει τα αρχεία player-SCENARIO-ID.txt και enemy-SCENARIO-ID.txt ώστε να τοποθετηθούν τα πλοία. Επίσης φαίνεται η αρίθμηση τους και η κατεύθυνση του Coordinate X, Coordinate Y. Στο κάτω μέρος της οθόνης βλέπουμε δύο TextFields που ο παίκτης επιλέγει τις συντεταγμένες του σημείου που θέλει να ρίξει. Για να ρίξει πρέπει αφού επιλέξει τις συντεταγμένες, να πατήσει το κουμπί Shoot.

- Μπορεί η γραφική διεπαφή να φορτώνεται από το Main.Java, όμως όλη η διαδραστικότητα της εφαρμογής γίνεται μέσω του MainController.java. Δηλαδή με το που φορτωθεί το Main.FXML καλείται το public void initialize() του MainController και στα 2 Pane που έχει φορτώσει το Main.FXML δημιουργεί τα 2 grids. Επίσης γίνεται η αρχικοποίηση των αντικειμένων που θα χρειαστούν για την διαδραστικότητα. Αρχικά δεν έχει γίνει φόρτωση των player-SCENARIO-ID.txt και enemy-SCENARIO-ID.txt και δεν έχουν τοποθετηθεί πλοία. Έχουμε θεωρήσει ότι ο παίκτης δεν πρόκειται να πατήσει το κουμπί Shoot, όσο δεν έχουν φορτωθεί τα αρχεία που αναφέραμε παραπάνω, δηλαδή όσο δεν έχει αρχίσει το παιχνίδι. Επίσης θεωρούμε ότι δεν πρόκειται να επιλέξει κάτι από το Menu Details καθώς κάτι τέτοιο δεν έχει νόημα όσο το παιχνίδι δεν έχει αρχίσει(αν και δεν θα δημιουργούσε κάποιο πρόβλημα στην εκτέλεση του προγράμματος καθώς οι constructors έχουν αρχικοποιήσει τα αντικείμενα που παράγουν τους πίνακες). Τώρα έστω ότι πάει να επιλέξει το κουμπί Start από το Menu Application. Αν πατηθεί το MenuItem Start καλείται η συνάρτηση public void startFunction(ActionEvent Event) που χρησιμοποιεί ένα try catch που προσπαθεί να ανοίξει το αρχείο "medialab/game.txt" που θα εξηγήσουμε στη συνέχεια τι κάνει. Αν αυτό το αρχείο δεν υπάρχει, που σημαίνει ότι δεν έχει επιλεχθεί κάποιο SCENARIO ακόμα, εμφανίζεται το μήνυμα "There isn't a Loaded game file.". Τα 2 grid παραμένουν αρχικοποιημένα χωρίς πλοία. Για να αρχίσει το παιχνίδι πρέπει να επιλέξουμε το MenuItem Load και να επιλέξουμε ένα κατάλληλο.

- Για να αρχίσει το παιχνίδι πρέπει να επιλέξουμε το MenuItem Load και να επιλέξουμε ένα κατάλληλο ID που το SCENARIO να υπάρχει και τα 2 αρχεία, player-SCENARIO-ID.txt και enemy-SCENARIO-ID.txt να μην δημιουργούν εξαιρέσεις κανένα από τα δύο. Πατώντας το Load λοιπόν καλείται η public void loadFunction του MainController η οποία φορτώνει ένα popup παράθυρο, με ένα TextField που επιλέγεις ID, και ένα κουμπί(Submit) που το κάνεις submit, με την χρήση του Load.FXML. Πατώντας το submit καλείται η public void chooseId του αρχείου LoadController.java που ελέγχει αν υπάρχουν τα 2 παραπάνω αρχεία με το ID που επιλέξαμε. Αν δεν υπάρχουν εμφανίζει το μήνυμα "This ID doesn't exist."(το εμφανίζει ακόμα κι αν ο παίκτης πατήσει αμέσως submit, χωρίς να επιλέξει κάποιο ID). Αν το ID τώρα υπάρχει άλλα κάποιο από τα 2 αρχεία παράγει κάποια εξαίρεση, σύμφωνα με τον τρόπο που είπαμε παραπάνω, τυπώνει το αντίστοιχο μήνυμα. Τα παρακάτω SCENARIOS δημιουργούν μια εξαίρεση ανάλογα με το ID τους(βρίσκονται στον φάκελο medialab).

  - ID = 0: MissingInputException
  - ID = 2: OverlapTilesException
  - ID = 3: OversizeException
  - ID = 4: AdjacentTilesException
  - ID = 5: InvalidCountExeception

  Στις παραπάνω περιπτώσεις που αναφέραμε, αν κλείσουμε το pop up παράθυρο και πατήσουμε Start θα εμφανιστεί και πάλι το μήνυμα There isn't a Loaded game file. Τώρα πέραν των εξαιρέσεων που αναφέραμε, θεωρούμε ότι κατα τα άλλα θα είναι κατάλληλη η μορφή των αρχείων. Για ID = 1 υπάρχει SCENARIO και μάλιστα είναι κατάλληλο. Στην περίπτωση επιλογής ID με κατάλληλο SCENARIO δημιουργείται το αρχείο game.txt. Αυτό το αρχείο χρειάζεται το start για να ξεκινήσει καινούρια παρτίδα. Το αρχείο game.txt περιέχει το διάστημα που λαμβάνει κάθε πλοίο στο αντίστοιχο grid. Πρώτα για τον player με την εξής διάταξη:

  1. 1. Carrier
  2. 2. Battleship
  3. 3. Cruiser
  4. 4. Submarine
  5. 5. Destroyer

  Και στην συνέχεια για τον enemy με την ίδια διάταξη. Αφού φορτωθεί κατάλληλο SCENARIO θεωρούμε ότι ο παίχτης θα κλείσει το pop-up window. Αξίζει να πούμε ότι κάθε φορά που γίνεται κλήση της συνάρτησης loadFunction του MainController.java το αρχείο game.txt σβήνεται. Αυτό σημαίνει ότι αν πατήσω το load πρέπει να ξαναεπιλέξω κατάλληλο SCENARIO ώστε αν πατήσω το Start να ξεκινήσει παρτίδα.

- Έστω ότι λοιπόν φορτώνεται ένα κατάλληλο SCENARIO και ο παίκτης πατάει Start.Καλείται η public void startFunction η οποία αρχικοποιεί τα 2 αντικείμενα τύπου Player(player, enemy). Αρχικά τυπώνεται μήνυμα "New game starts!!!". Στην συνέχεια η startFunction αρχικοποιεί το grid και κάνει την τοποθέτηση των πλοίων. Ως πλοία έχουν χρησιμοποιηθεί Circles του JavaFX που μπαίνουν στο grid με διαφορετικό χρώμα ανάλογα με τον τύπο του πλοίου(έχουν επιλεγεί ίδια χρώματα με αυτά στο setup_files.pdf). Η αρχικοποίηση των πλοίων του player και του enemy γίνεται με βάση το game.txt. Πράγματα που αφορούν πόντους, ευστοχία κ.λ.π. αρχικοποιούνται στο 0, τα πλοία αρχικοποιούνται στο κατάλληλο μέγεθος ανάλογα με τον τύπο, και κατάσταση("Intact"). Τα διαθέσιμα πλοία κάθε παίκτη στην αρχή είναι 17. Για κάθε παίχτη έχουμε μια boolean μεταβλητή που δείχνει αν έχει παίξει στον γύρο, η οποία αρχικοποιείται σε false. Επίσης υπάρχει και μια String μεταβλητή που δείχνει τον τύπο του τελευταίου πλοίου κάθε παίκτη που χτυπήθηκε, η οποία αρχικοποιείται σε "". Επίσης αρχικοποιούνται οι int μεταβλητές enemyPreviousX, enemyPreviousY, που έχουν τις συντεταγμένες της προηγούμενης επιτυχούς βολής του enemy, στο -1. Θα εξηγήσουμε στην συνέχεια πως χρησιμοποιούνται. Αυτό που γίνεται είναι να ρίχνουμε ένα νόμισμα, ώστε να αποφασίζουμε με πιθανότητα 1/2 ποιος παίκτης παίζει πρώτος. Για την προσομοίωση του flipping coin χρησιμοποιούμε την συνάρτηση random της κλάσης Math, που αναθέτει μια τιμή στην double μεταβλητή coin. Αν coin>0.5 ο player παίζει πρώτος. Αλλιώς

παίζει ο enemy πρώτος. Εμφανίζεται ένα μήνυμα στην οθόνη που ενημερώνει ποιος παίζει πρώτος. Σε περίπτωση που παίζει ο enemy πρώτος η πρώτη επίθεση από αυτόν γίνεται από τον κώδικα του startFunction. Θα εξηγήσουμε πως γίνεται η επίθεση στην συνέχεια. Γίνεται με τον ίδιο τρόπο στην shoot. Τέλος πρέπει να αναφέρουμε για την start πως όσο δεν έχουμε πατήσει Load ώστε να σβηστεί το game.txt μπορούμε να πατήσουμε όσες φορές θέλουμε Start, ακόμα και αν έχουν περάσει κάποιοι γύροι του παιχνιδιού , ή ακόμα να έχει τελείωσει και γίνεται reset του παιχνιδιού(όλα επιστρέφουν στην αρχική τους κατάσταση).

- Η μέθοδος public void shoot του MainController.java είναι αυτή που ουσιαστικά παίζει το παιχνίδι. Δηλαδή το παιχνίδι μπορεί να έχει αρχίσει στην start αν παίζει ο enemy πρώτος, αλλά για να συνεχίσει, πρέπει ο παίχτης να επιλέξει ένα στόχο και να πατήσει Shoot. Έχουμε θεωρήσει ότι ο παίχτης αρχικά θα βάλει τιμές και για τις 2 συντεταγμένες, οι οποίες θα είναι κατάλληλες, δηλαδή και οι 2 τιμές θα είναι ακέραιες και θα ανήκουν στο διάστημα [0-9], και μετά θα πατήσει Shoot. Επίσης για επιτυχημένη βολή χρησιμοποιούνται Circles με κόκκινο χρώμα. Επιτρέπεται να χτυπήσεις το ίδιο σημείο παραπάνω από μία φορά, οπότε εύστοχη είναι η πρώτη βολή που θα ευστοχήσεις σε σημείο που υπήρχε πλοίο. Οι υπόλοιπες βολές είναι άστοχες και τις αναπαριστούμε πάλι με Circles, αλλά αυτή τη φορά με μαύρο χρώμα. Αν χτυπήσω 2η φορά σε σημείο που είχα ευστοχήσει θα μετατραπεί από Circle χρώματος κόκκινου, σε μαύρο. Στη shoot αυτό που γίνεται είναι να ελέγχεται αν μια βολή του player είναι εύστοχη και ανάλογα με το αποτέλεσμα να ενημερώνεται το grid του enemy, τα Enemy Ships, τα στοιχεία που αφορούν την απόδοση του παίχτη κ.λ.π., ποιο πλοίο του enemy χτυπήθηκε, αν χτυπήθηκε. Αντίστοιχα επιτίθεται και ο enemy. Μόνο που επιλέγει το σημείο που θα επιτεθεί με διαφορετικό τρόπο. Αν η προηγούμενη βολή που έριξε ήταν άστοχη, επιλέγει ένα σημείο τυχαία. Αν όμως ήταν εύστοχη επιλέγει με πιθανότητα 1/4 μια γειτονική θέση από αυτή που πέτυχε τον στόχο, δηλαδή πάνω, κάτω, αριστερά ή δεξιά. Για να κρατάμε τις συντεταγμένες της εύστοχης βολής χρησιμοποιούμε τις μεταβλητές enemyPreviousX, enemyPreviousY που ανέφερα νωρίτερα. Αν αστοχήσω ξανά αρχικοποιούνται στο -1. Αφού επιτεθούν οι 2 παίκτες ενημερώνουν και τους πίνακες playerShots,enemyShots. Αφού παίξουν και οι 2 πάμε στον επόμενο γύρο με τον παίχτη που παίζει πρώτος να είναι αυτός που έπαιξε πρώτος και στον πρώτο γύρο. Κάθε φορά που βυθίζεται ένα πλοίο το η εφαρμογή μας ενημερώνει. Το παιχνίδι τελειώνει όταν είτε ένας παίχτης ξεμείνει από πλοία είτε αν γίνουν 40 γύροι κερδίζει αυτός με τους περισσότερους πόντους. Και στις 2 περιπτώσεις το παιχνίδι μας ενημερώνει για τον νικητή. Η μεταβλητή winner γίνεται true, η οποία είχε αρχικοποιηθεί με false στην start. Επειδή όλος ο κώδικας βρίσκεται σε ένα if αυτή η μεταβλητή να μην είναι true, σταματάει να εκτελείται και δεν μπορεί να υπάρξει άλλος γύρος. Για να γίνει reset του παιχνιδιού ο παίχτης πρέπει να πατήσει Start, αν θέλει να παίξει με την ίδια διάταξη πλοίων, αλλιώς πρέπει να κάνει Load άλλο κατάλληλο SCENARIO.

- Οι συναρτήσεις enemyShips(MenuItem Enemy Ships) ,
playerShots(MenuItem Player Shots), enemyShots(MenuItem Enemy Shots) του MainController.java δημιουργούν pop-up παράθυρα με πίνακες που επιδεικνύουν τα στοιχεία που ζητήθηκαν από την εκφώνηση.

- Τέλος θεωρούμε ότι ο παίχτης για να σταματήσει πατάει το κουμπί Exit του Menu Application, και δεν πατάει το X, για να κλήσει το βασικό παράθυρο της εφαρμογής. Κι αυτό γιατί η exitFunction του MainController.java που καλείται σβήνει το αρχείο game.txt και τυπώνει το μήνυμα "Exit", πριν κλείσει την εφαρμογή.

Δημιουργία γραφικής διεπάφης

Πραγματοποιήθηκε κάθε τμήμα που αναφερόταν στην εκφώνηση. Το μόνο που προστέθηκε είναι το Total Actions στο κύριο παράθυρο και ο μετρητής των γύρων(ROUND).

Λοιπές απαιτήσεις

Οι αρχές σχεδίασης του αντικειμενοστραφούς προγραμματισμού, έχουν τηρηθεί.
Έχει γίνει τεκμηρίωση των μεθόδων της κλάσης Player με τις προδιαγραφές Javadoc