



Διεθνές Πανεπιστήμιο της Ελλάδος

Έδρα στη Θεσσαλονίκη, τμήματα και παραρτήματα σε Θεσσαλονίκη, Σέρρες, Καβάλα, Δράμα,
Κατερίνη, Κιλκίς, Διδυμότειχο

Ομάδα Εργαστηρίου Μεθοδολογίας Προγραμματισμού Ε6

Όνομα project : Jave Game Engine

Μέλη:

- Γαρουφαλιάς Κωνσταντίνος-Μάριος 19121
- Γιαλαμάς Απόστολος 19099
- Τριανταφυλλίδου Κουλουριώτου Καλλιόπη 19111

ΠΕΡΙΕΧΟΜΕΝΑ

1.1 Αρχιτεκτονική του Project	3
1.2 Πακέτα.....	3
1.2.1 Dice Package.....	4
1.2.2 DefaultTimer ,Timer ,TimeScoreCal	4
1.2.3 Game Difficulty.....	5
1.2.4 Game Board.....	7
1.2.5 Json Parser.....	10
1.2.6 Other.....	11
1.2.7 Player.....	13
1.2.8 Main.....	14
2.1Εξήγηση τρόπου λειτουργίας SnakesAndLadders.....	15
3.1 Τελικό συμπέρασμα.....	18
3.2 Βιβλιογραφία	19

Τελική Παράδοση

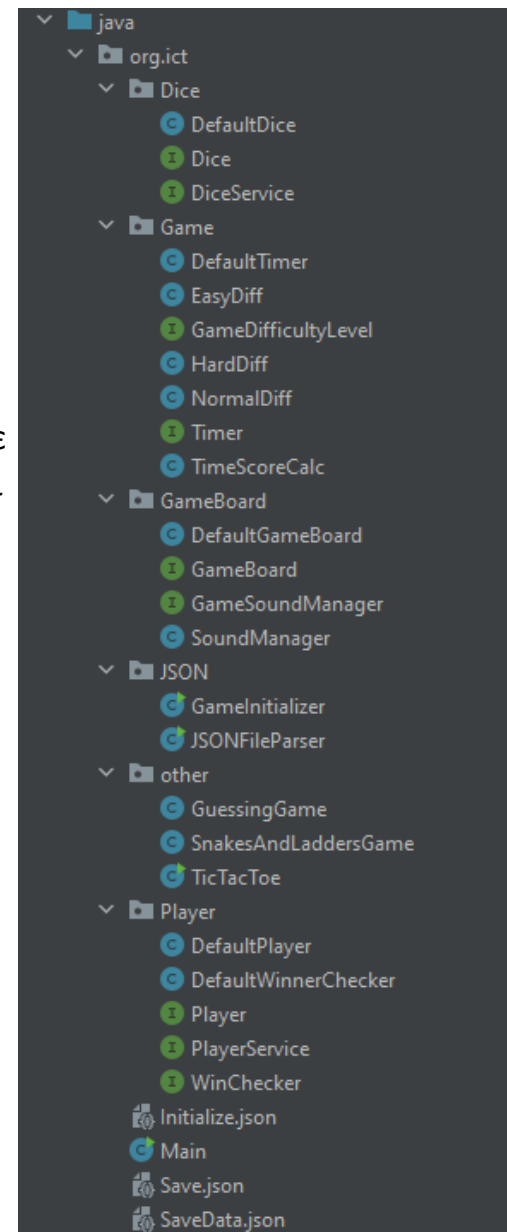
1.1 Αρχιτεκτονική του Project

Ο κώδικας αναπαριστά μια απλή μηχανή παιχνιδιού που υποστηρίζει παιχνίδια όπως το “Φιδάκι” ή “Snakes And Ladders” . Αυτό έχει σχεδιαστεί χρησιμοποιώντας αντικειμενοστραφείς αρχές όπως η Επεκτασιμότητα και η Συντηρησιμότητα του κώδικα .

Αρχικά , θα χρειαστεί να κατανοήσουμε την αρχιτεκτονική του κώδικα :

1.2 Packages

- I. org.ict.Dice: Περιέχει κλάσεις που σχετίζονται με τη λειτουργικότητα του ζαριού.
- II. org.ict.Game: Περιέχει κλάσεις που σχετίζονται με τη λογική του παιχνιδιού και τα επίπεδα δυσκολίας.
- III. org.ict.GameBoard: Περιέχει κλάσεις που σχετίζονται με την τεχνική πλευρά του πίνακα του παιχνιδιού (GUI) και τον ήχο.
- IV. org.ict.JSON: Περιέχει κλάσεις που σχετίζονται με την ανάγνωση των ρυθμίσεων του παιχνιδιού από ένα αρχείο Initialize.json.
- V. org.ict.other: Περιέχει κλάσεις για άλλες υλοποιήσεις παιχνιδιών ή για testing.
- VI. org.ict.Player: Περιέχει κλάσεις που σχετίζονται με τον παίκτη.



1.2.1 Dice

Το DefaultDice αναπαριστά την υλοποίηση του DiceService και του Dice Interface . Η μέθοδος roll() προσομοιώνει την κύλιση του ζαριού ανάλογα τον αριθμό ζαριών που ορίζουμε εμείς στο .json αρχείο :

```
DefaultDice.java

1 package org.ict.Dice;
2
3 public class DefaultDice implements DiceService {
4     private int diceNum;
5
6     public DefaultDice(int diceNum) {
7         this.diceNum = diceNum;
8     }
9
10    public int roll() {
11        return diceNum * ((int) (Math.random() * 6) + 1);
12    }
13
14    public void setDiceNum(int diceNum) {
15        this.diceNum = diceNum;
16    }
17
18    public int getDiceNum() {
19        return diceNum;
20    }
21 }
```

```
Dice.java

1 package org.ict.Dice;
2
3 public interface Dice {
4     int roll();
5 }
6
```

1.2.2 Game package : DefaultTimer , Timer , TimeScoreCalc

Η DefaultTimer είναι υλοποίηση του Interface Timer. Η μέθοδος getTime() επιστρέφει τον χρόνο του σύστηματος σε χιλιοστά του δευτερολέπτου , ενώ η TimeScoreCalc παρέχει μια στατική μέθοδο calculateScore() για τον υπολογισμό του σκορ βάσει των υπολειπόμενων μαντεψιών, της ώρας έναρξης και της ώρας λήξης:

```
Timer.java

1 package org.iot.Game;
2
3 public interface Timer {
4     long getTime();
5 }
6
7
```

```
TimerScoreCalc.java

1 package org.iot.Game;
2
3 public class TimeScoreCalc {
4     public static int calculateScore(long remainingGuesses,
5     long startTime, long endTime) {
6         double totalTimeMillis = endTime - startTime;
7         double totalTimeSeconds = totalTimeMillis / 1000.0;
8         int score = (int) (remainingGuesses * 100 - totalTimeSeconds);
9         if (score < 0) {
10             score = 0;
11         }
12         return score;
13     }
14 }
```

1.2.3 Game Difficulty Level

To GameDifficultyLevel Interface Ορίζει το difficulty παιχνιδιού με την μέθοδο getDifficultyMap() **[3]** η οποία παίρνει ένα JSONObject ως είσοδο και επιστρέφει ένα HashMap που αναπαριστά το επίπεδο δυσκολίας(Easy,Normal,Hard). Για παράδειγμα , στη περίπτωση του Snakes And Ladders το Easy δίνει περισσότερες ευκαιρίες στον παίχτη να ανεβεί μια σκάλα και λιγότερες ευκαιρίες πέσει σε παγίδια με φίδια και να πέσει :

```
EasyDiff.java

1 package org.iot.Game;
2
3 import org.json.simple.JSONObject;
4 import java.util.HashMap;
5 import java.util.Map;
6 public class EasyDiff implements GameDifficultyLevel {
7     @Override
8     public Map<Integer, Integer> getDifficultyMap(JSONObject jsonObject) {
9         Map<Integer, Integer> difficultyMap = new HashMap<>();
10         JSONObject easyDiffMap = (JSONObject) jsonObject.get("EasyDiffMap");
11         for (Object key : easyDiffMap.keySet()) {
12             Integer startingSquare = Integer.parseInt((String) key);
13             Integer endingSquare = ((Long) easyDiffMap.get(key)).intValue();
14             difficultyMap.put(startingSquare, endingSquare);
15         }
16         return difficultyMap;
17     }
18 }
```

```
GameDifficultyLevel.java

1 package org.ict.Game;
2
3 import org.json.simple.JSONObject;
4
5 import java.util.Map;
6
7 public interface GameDifficultyLevel {
8     Map<Integer, Integer> getDifficultyMap(JSONObject jsonObject);
9 }
10
```

Τα Maps των difficulty είναι αποθηκευμένα στο .json αρχείο και μπορεί ο παίχτης να τα αλλάξει όπως θέλει :

```
Initialize.json

1 {
2     "boardSize" : 50,
3     "diceNum" : 1,
4     "difficulty" : "Easy",
5     "playerNum" : 2,
6     "gameType" : "Snakes And Ladders",
7     "EasyDiffMap": {
8         "4" : 14,
9         "9" : 31,
10        "17" : 29,
11        "20" : 38,
12        "28" : 84,
13        "40" : 59,
14        "51" : 67,
15        "54" : 34,
16        "62" : 90,
17        "63" : 81,
18        "64" : 60,
19        "71" : 91,
20        "87" : 99,
21        "99" : 79
22    },
23     "remainingGuesses" : 10,
24     "randomNumber" : 8
25 }
26
```

1.2.4 Game Board

Το GameBoard Interface χτίζει το ταμπλό του παιχνιδιού με την μέθοδο initializeGameBoard() . Η μέθοδος paintPlayer() ενημερώνει τη θέση του παίκτη στον πίνακα του παιχνιδιού και ζωγραφίζει τα κουτιά στα οποία βρίσκονται οι παίκτες. Η DefaultGameBoard υλοποιεί το GameBoard interface για το Snakes And Ladders με τον παρακάτω κώδικα :

```
1 package org.ict.GameBoard;
2
3 public class DefaultGameBoard extends JPanel implements GameBoard {
4     private int boardSize;
5     private JPanel gameBoard;
6     private Color color;
7     private Player player1;
8     private Player player2;
9
10    public DefaultGameBoard(int boardSize, Color color, Player player1, Player player2) {
11        this.boardSize = boardSize;
12        this.color = color;
13        this.player1 = player1;
14        this.player2 = player2;
15        initializeGameBoard();
16    }
17
18    @Override
19    public void initializeGameBoard() {
20        setLayout(new GridLayout(1, 1));
21        gameBoard = new JPanel(new GridLayout(10, 10));
22        for (int i = 0; i < boardSize; i++) {
23            JPanel panel = new JPanel();
24            panel.setBorder(BorderFactory.createLineBorder(Color.black));
25            panel.setBackground(Color.white);
26            JLabel label = new JLabel("" + (boardSize-i), SwingConstants.CENTER);
27            panel.add(label);
28            gameBoard.add(panel);
29        }
30        add(gameBoard, BorderLayout.CENTER);
31    }
32
33    @Override
34    public void paintPlayer(int player, int position) {
35        Player currentPlayer;
36        if (player == 1) {
37            currentPlayer = player1;
38        } else {
39            currentPlayer = player2;
40        }
41        int oldPosition = currentPlayer.getPosition();
42        currentPlayer.setPosition(position);
43        Component newComponent = gameBoard.getComponent(boardSize - position);
44        if (player == 1){
45            newComponent.setBackground(Color.blue);
46        } else{
47            newComponent.setBackground(Color.red);
48        }
49        Component oldComponent = gameBoard.getComponent(boardSize - oldPosition);
50        oldComponent.setBackground(Color.WHITE);
51    }
52 }
53
```

Εδώ καλό είναι να σημειωθεί πως το μέγεθος του ταμπλό που δημιουργείται ορίζεται από τον παίχτη στο .json αρχείο (παράδειγμα με boardSize = 50 και boardSize = 15):

50	49	48	47	46	Player 1 rolled a: 5! Player 1 MOVED to BOX: 5 Player 2 rolled a: 5! Player 2 MOVED TO: 5 Player 1 rolled a: 3! Player 1 MOVED to BOX: 8 Player 2 rolled a: 5! Player 2 MOVED TO: 10 Player 1 rolled a: 2! Player 1 MOVED to BOX: 10 Player 2 rolled a: 2! Player 2 MOVED TO: 12 Player 1 rolled a: 5! Player 1 MOVED to BOX: 15 Player 2 rolled a: 1! Player 2 MOVED TO: 13
45	44	43	42	41	
40	39	38	37	36	
35	34	33	32	31	
30	29	28	27	26	
25	24	23	22	21	
20	19	18	17	16	
15	14	13	12	11	
10	9	8	7	6	
5	4	3	2	1	

15	14	Player 1 rolled a: 1! Player 1 MOVED to BOX: 1 Player 2 rolled a: 1! Player 2 MOVED TO: 1 Player 1 rolled a: 6! Player 1 MOVED to BOX: 7 Player 2 rolled a: 2! Player 2 MOVED TO: 3
13	12	
11	10	
9	8	
7	6	
5	4	
3	2	
1		

Επίσης , υπάρχει μουσική υπόκρουση και Sound Effects σε διάφορα σημεία του κώδικα(όπως το background Audio ,ο ήχος που κάνουν τα ζάρια όταν πατάει το κουμπί Roll Dice ο παίχτης και όταν κάποιος φτάνει στη γραμμή τερματισμού παίζει έναν ήχο νίκης). Συγκεκριμένα , το SoundManager υλοποιεί το GameSoundManager interface :

```
GameSoundManager.java

1 package org.ict.GameBoard;
2
3 public interface GameSoundManager {
4     void playDiceRollSound();
5     void playWinSound();
6
7     void playBackgroundAudio();
8 }
9
```

```
GameSoundManager.java

1 package org.ict.GameBoard;
2
3 public class SoundManager implements GameSoundManager {
4     private Clip backgroundAudio;
5     private Clip backgroundAudio2;
6     private Clip diceRollSound;
7     private Clip winSound;
8
9     public SoundManager() throws IOException, UnsupportedAudioFileException,
10 LineUnavailableException {
11         //Importing sound files....
12     }
13     @Override
14     public void playDiceRollSound() {
15         diceRollSound.setFramePosition(0);
16         diceRollSound.start();
17     }
18     @Override
19     public void playWinSound() {
20         winSound.setFramePosition(0);
21         winSound.start();
22     }
23
24     public void playBackgroundAudio() {
25         backgroundAudio.loop(Clip.LOOP_CONTINUOUSLY);
26         backgroundAudio.start();
27     }
28     public void playBackgroundAudio2() {
29         backgroundAudio2.loop(Clip.LOOP_CONTINUOUSLY);
30         backgroundAudio2.start();
31     }
32 }
```

1.2.5 GameInitializer / JsonParser

Σε αυτό το πακέτο γίνεται parsing του .json αρχείου με την κλάση GameInitializer η οποία επιστρέφει ρυθμίσεις που χρειαζόμαστε για το παιχνίδι . Οι τιμές αυτές φαίνονται παρακάτω [1],[5]:

A screenshot of a code editor window titled 'Initialize.json'. The window has a dark background with light-colored text. The code is a JSON object with the following structure: a root object containing 'boardSize' (80), 'diceNum' (1), 'difficulty' ('Normal'), 'playerNum' (2), 'gameType' ('Guessing Game'), 'EasyDiffMap' (an object with 10 key-value pairs), 'NormalDiffMap' (an object with two empty arrays), 'HardDiffMap' (an object with two empty arrays), 'remainingGuesses' (10), and 'randomNumber' (8). The code is numbered from 1 to 23 on the left side of the editor.

```
1 {  
2     "boardSize" : 80,  
3     "diceNum" : 1,  
4     "difficulty" : "Normal",  
5     "playerNum" : 2,  
6     "gameType" : "Guessing Game",  
7     "EasyDiffMap": {  
8 "4": 14, "9": 31, "17": 29, "20": 38, "28": 84,  
9 "40": 59, "51": 67, "54": 34, "62": 90, "63": 81,  
10 "64": 60, "71": 91, "87": 99, "99": 79  
11     },  
12     "NormalDiffMap": {  
13         //...//  
14         //...//  
15     },  
16     "HardDiffMap": {  
17         //...//  
18         //...//  
19     },  
20     "remainingGuesses" : 10,  
21     "randomNumber" : 8  
22  
23 }
```

Η μέθοδος initialize() διαβάζει το αρχείο JSON και ορίζει τις τιμές των ρυθμίσεων. Παρέχονται μέθοδοι parse για την πρόσβαση στις τιμές των ρυθμίσεων που χρειάζονται για το παιχνίδι :

```

1  package org.ict.JSON;
2
3  public class GameInitializer {
4
5      public void initialize() throws Exception {
6          FileReader reader = new FileReader("../Initialize.json");
7          JSONParser jsonParser = new JSONParser();
8          JSONObject jsonObject = (JSONObject) jsonParser.parse(reader);
9          boardSize = (long) jsonObject.get("boardSize");
10         diceNum = (long) jsonObject.get("diceNum");
11         playerNum = (long) jsonObject.get("playerNum");
12         difficulty = (String) jsonObject.get("difficulty");
13         gameType = (String) jsonObject.get("gameType");
14         remainingGuesses = (long) jsonObject.get("remainingGuesses");
15         randomNumber = (long) jsonObject.get("randomNumber");
16     }
17     public long getBoardSize() {return boardSize;}
18     public long getDiceNum() {return diceNum;}
19     public long getPlayerNum() {return playerNum;}
20     public String getDifficulty() {return difficulty;}
21     public String getGameType(){return gameType;}
22     public long getRemainingGuesses(){return remainingGuesses;}
23     public long getRandomNumber(){return randomNumber;}
24 }

```

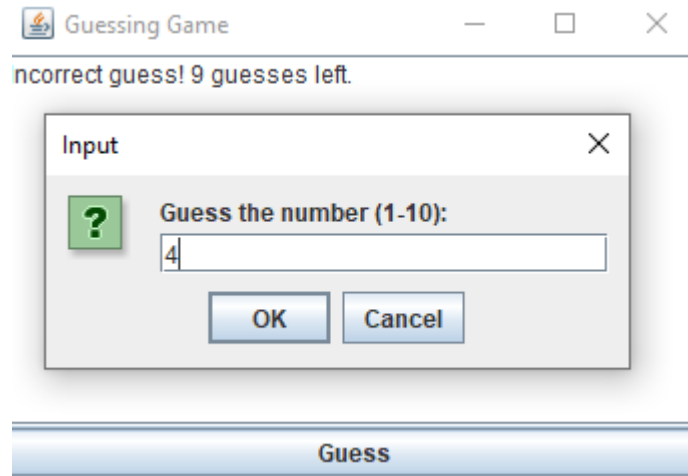
1.2.6 Other

Περιέχει κλάσεις για υλοποίηση άλλων παιχνιδιών που καλούνται στη main , όπως αυτή του GuessingGame. Στο παιχνίδι αυτό ο χρήστης δίνεται έναν άγνωστο αριθμό που πρέπει να μαντέψει. Παράλληλα, γίνεται χρήση του GameInitializer για τις τιμές αυτές (δηλαδή το remaining guesses και το random number). Επίσης , χρησιμοποιείται και το SoundManger [\[6\]](#) για να δώσει ήχο στο παιχνίδι ,παρομοίως με το Snakes And Ladders. Στο τέλος υπολογίζεται το σκόρ ενός παίχτη ανάλογα με τον χρόνο που έκανε να δώσει τη σωστή απάντηση με την χρήση της κλάσης TimeScoreCalc που αναφέραμε προηγουμένως. Με αυτόν τον τρόπο, ο κώδικας υπολογισμού βαθμολογίας είναι ενθουλακωμένος στην κλάση `TimeScoreCalc`, καθιστώντας τον πιο εύκολα επαναχρησιμοποιήσιμο και σε άλλα παιχνίδια :

```

1  package org.ict.other;
2
3  public class GuessingGame extends JFrame {
4      private final JTextArea gameInfo;
5      private final JButton guessButton;
6
7      private long startTime;
8      private long randomNumber;
9      private long remainingGuesses;
10     private int score;
11
12     SoundManager soundManager;
13
14     public GuessingGame() throws Exception{
15         super("Guessing Game");
16
17         GameInitializer initializer = new GameInitializer();
18         initializer.initialize();
19         remainingGuesses = initializer.getRemainingGuesses();
20         randomNumber = initializer.getRandomNumber();
21
22         soundManager = new SoundManager();
23         soundManager.playBackgroundAudio2();
24
25         DefaultTimer gameTimer = new DefaultTimer();
26
27         guessButton = new JButton("Guess");
28         guessButton.addActionListener(new ActionListener() {
29             public void actionPerformed(ActionEvent e) {
30                 int guess = Integer.parseInt(JOptionPane.showInputDialog(null,
31                     "Guess the number (1-10):"));
32                 if (remainingGuesses != 0) {
33                     gameTimer.getTime();
34                 }
35                 if (guess == randomNumber) {
36
37                     long endTime = gameTimer.getTime();
38                     score = TimeScoreCalc.calculateScore(remainingGuesses, startTime, endTime);
39
40                     gameInfo.append("You guessed correctly! It took you " + String.format("%.2f",
41                         (endTime - startTime) / 1000.0) + " seconds.\n");
42                     gameInfo.append("Your score is " + score + ".\n");
43                     soundManager.playWinSound();
44                     guessButton.setEnabled(false);
45                 } else {
46                     remainingGuesses--;
47                     gameInfo.append("Incorrect guess! " + remainingGuesses + " guesses left.\n");
48                     if (remainingGuesses == 0) {
49                         gameInfo.append("Game over! The correct number was " + randomNumber + ".\n");
50                         gameInfo.append("Your score is " + score + ".\n");
51                         guessButton.setEnabled(false);
52                     }
53                 }
54             }
55         });
56         //setup the game windows...
57     }
58 }
59
60

```



1.2.7 Player

Σε αυτό το Πακέτο γίνεται διεκπεραίωση δύο interface **[7],[8]** : του PlayerService και του WinChecker.

```
Player.java

1 package org.ict.Player;
2
3 public interface Player {
4     int getPosition();
5     void setPosition(int position);
6 }
```

```
WinChecker.java

1 package org.ict.Player;
2
3 public interface WinChecker {
4     boolean hasPlayerWon(int position,
5         int boardSize);
6 }
7
```

Η DefaultPlayer υλοποιεί την PlayerService. Οι μέθοδοι getPosition() και setPosition() υλοποιούνται για την ανάκτηση και τον ορισμό της θέσης του παίκτη στο board. Η defaultWinnerChecker παρέχει μεθόδους για τον έλεγχο νίκης ενός παίκτη. Η μέθοδος hasPlayerWon() ελέγχει αν ισχύει η συνθήκη νίκης με βάση τον παρεχόμενο πίνακα παιχνιδιού και τις θέσεις των παικτών και επιστρέφει true or false :

```
DefaultPlayer.java

1 package org.ict.Player;
2
3 public class DefaultPlayer implements PlayerService {
4     private int position;
5
6     public DefaultPlayer(int position) {
7         this.position = position;
8     }
9
10    @Override
11    public int getPosition() {
12        return position;
13    }
14
15    @Override
16    public void setPosition(int position) {
17        this.position=position;
18    }
19 }
```

```
DefaultWinnerChecker.java

1 package org.ict.Player;
2
3 public class DefaultWinnerChecker implements WinChecker {
4     @Override
5     public boolean hasPlayerWon(int position, int boardSize) {
6         return position >= boardSize;
7     }
8 }
```

1.2.8 Main

Το κεντρικό σημείο της εφαρμογής. Η main() μέθοδος διαβάζει το αρχείο .json και τρέχει το ανάλογο παιχνίδι που έχει τοποθετήσει ο παίχτης αρχικοποιώντας τις ρυθμίσεις που έχουμε θέσει με την GameInitializer. Ταυτόχρονα χτίζει το ταμπλό με GUI.

```
DefaultWinnerChecker.java

1 public Main() throws Exception {
2     super("Game Launcher");
3
4     GameInitializer initializer = new GameInitializer();
5     initializer.initialize();
6     gameType = initializer.getGameType();
7     System.out.println(gameType);
8
9     if (gameType.equalsIgnoreCase("Snakes and Ladders")) {
10         launchSnakesAndLadders();
11     } else if (gameType.equalsIgnoreCase("Tic Tac Toe")) {
12         launchTicTacToe();
13     } else if (gameType.equalsIgnoreCase("Guessing Game")) {
14         launchGuessingGame();
15     }
16 }
```


2.1 Εξήγηση τρόπου λειτουργίας SnakesAndLadders.

Το παιχνίδι τρέχει με τη βοήθεια της συνάρτησης `launchSnakesAndLadders` για 2 παίκτες [4]. Αρχικά, παίρνει τις κατάλληλες τιμές που χρειάζεται για την εκτέλεση του προγράμματος όπως το `diceNumber`, `difficulty` και το `boardSize`. Όταν γίνει κλικ στο κουμπί "Roll", η μέθοδος αναλαμβάνει τη λογική της κίνησης των παικτών με βάση την επιλογή του ζαριού και την τρέχουσα κατάσταση του παιχνιδιού. Επίσης, ελέγχει αν ο παίκτης έχει κερδίσει το παιχνίδι και ενημερώνει τις πληροφορίες του παιχνιδιού ανάλογα, όπως την `paintPlayer` που χρωματίζει το ανάλογο κουτί [2] που βρίσκεται ο παίκτης. Τέλος, υπάρχει και έλεγχος σε περίπτωση που ο παίκτης βγεί εκτός του `boardSize`:

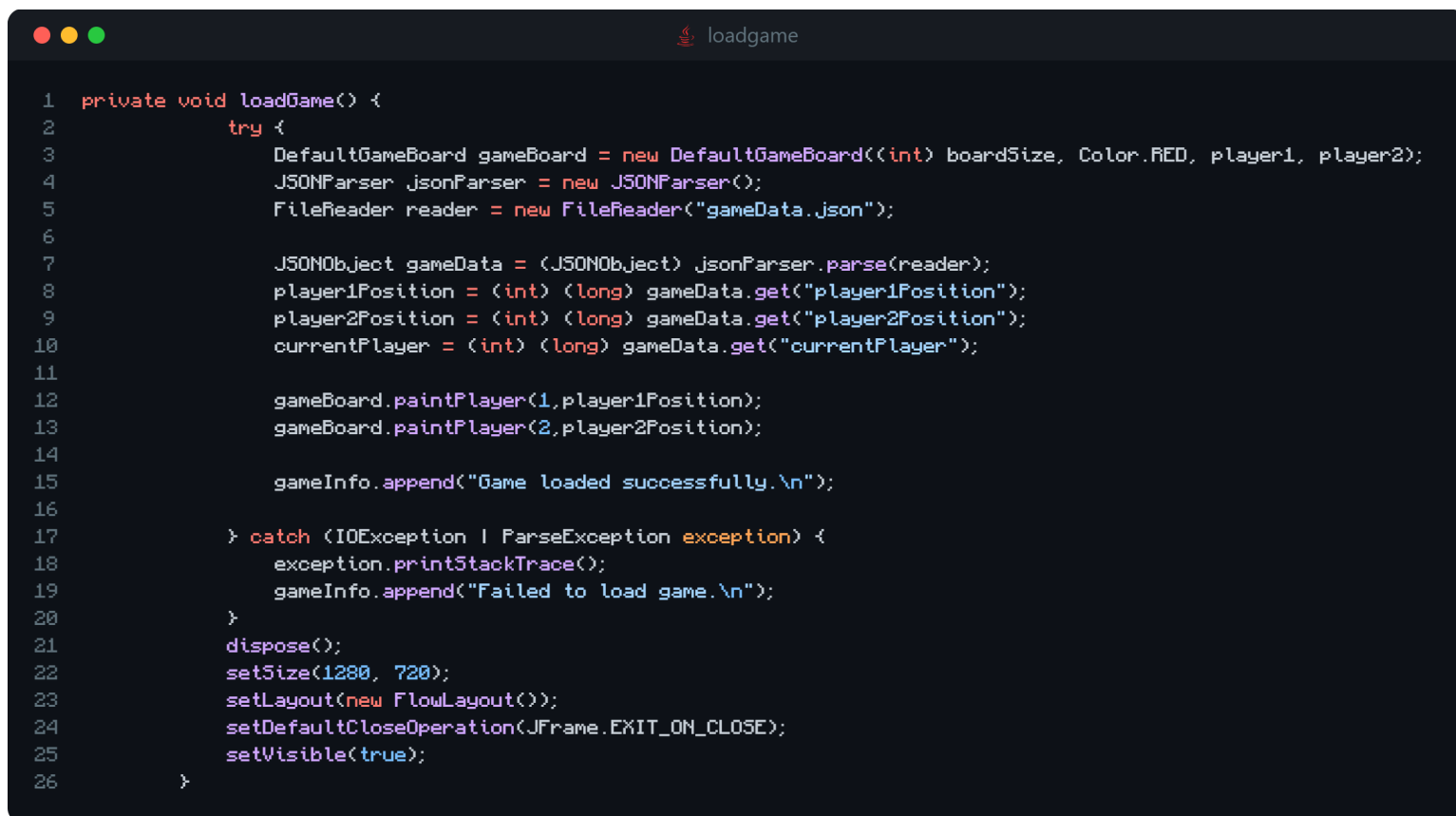
```
rollbutton function

1 rollButton.addActionListener(new ActionListener() {
2     public void actionPerformed(ActionEvent e) {
3         int roll = dice.roll();
4         gameInfo.append("Player " + (currentPlayer + 1) + " rolled a: " + roll + "!\n");
5         soundManager.playDiceRollSound();
6
7         Map<Integer, Integer> diffMap = gameDifficultyLevel.getDifficultyMap(jsonObject);
8         int currentPosition = (currentPlayer == 0) ? player1Position : player2Position;
9         currentPosition += roll;
10
11         Integer targetPosition = diffMap.get(currentPosition);
12         if (targetPosition != null) {
13             currentPosition = targetPosition;
14             gameInfo.append("Player " + (currentPlayer + 1) + " TELEPORTS to BOX " + targetPosition + "!!!\n");
15         }
16
17         gameInfo.append("Player " + (currentPlayer + 1) + " MOVED to BOX: " + currentPosition + "\n\n");
18         int playerPosition = (currentPlayer == 0) ? player1Position : player2Position;
19         playerPosition = Math.min(playerPosition, (int) boardSize);
20
21         if (defaultGameChecker.hasPlayerWon(playerPosition, (int) boardSize)) {
22             gameInfo.append("Player " + (currentPlayer + 1) + " wins!\n");
23             soundManager.playWinSound();
24             rollButton.setEnabled(false);
25             gameBoard.setBackground((currentPlayer == 0) ? Color.blue : Color.red);
26             gameBoard.paintPlayer(currentPlayer + 1, playerPosition);
27             return;
28         }
29
30         gameBoard.paintPlayer(currentPlayer + 1, playerPosition);
31         currentPlayer = (currentPlayer + 1) % 2;
32     }
33 });
```

Τέλος , ο παίχτης μπορεί να αποθηκεύει και να συνεχίζει το παιχνίδι του με τις μεθόδους `loadGame` και `saveGame` . Οι θέσεις των παιχτών στο ταμπλό αλλά και ποίος παίχτης έχει σειρά (`currentPlayer`) αποθηκεύονται στο αρχείο του προγράμματος `gameData.json` πατώντας το κουμπί “Save Game” και φορτώνονται αναλόγως πατώντας το κουμπί “Load Game” :



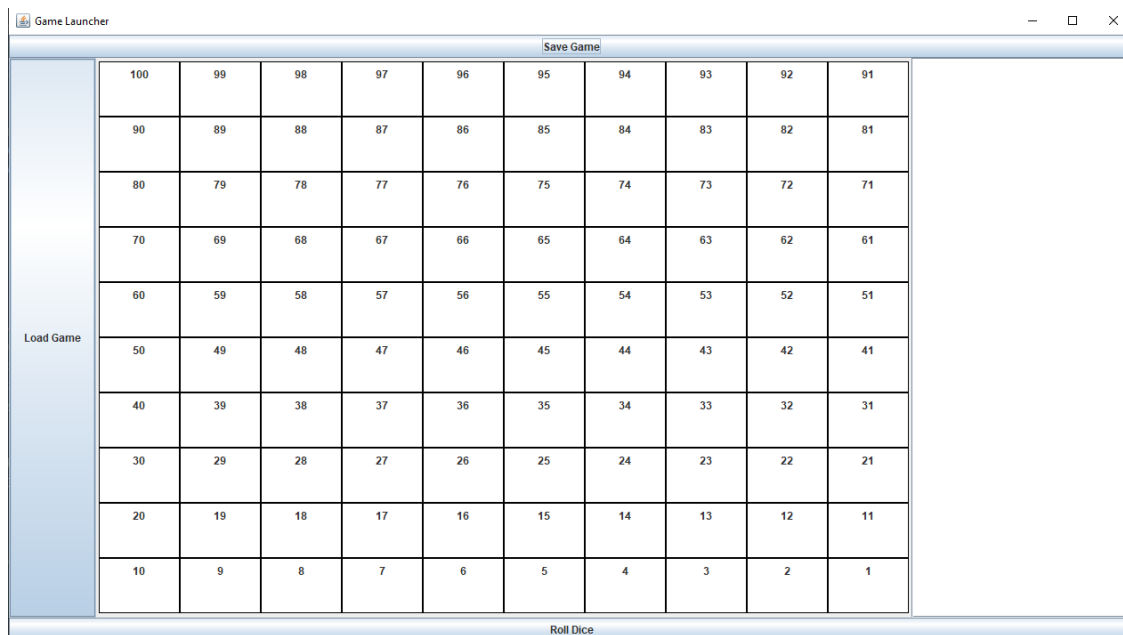
```
1 {
2   "currentPlayer":1,
3   "player1Position":44,
4   "player2Position":68
5
6 }
```



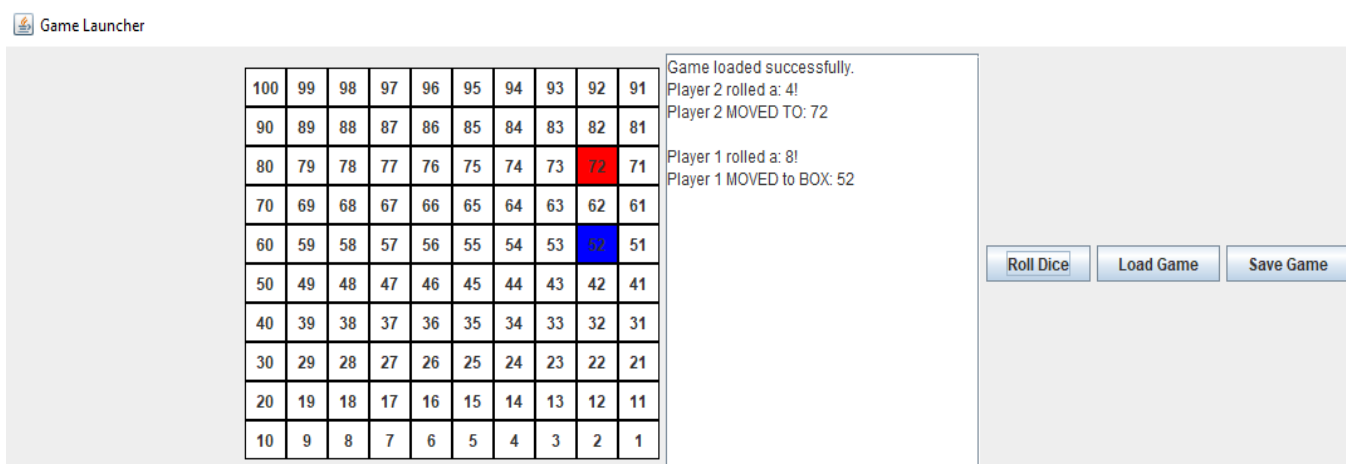
```
1 private void loadGame() {
2     try {
3         DefaultGameBoard gameBoard = new DefaultGameBoard((int) boardSize, Color.RED, player1, player2);
4         JSONParser jsonParser = new JSONParser();
5         FileReader reader = new FileReader("gameData.json");
6
7         JSONObject gameData = (JSONObject) jsonParser.parse(reader);
8         player1Position = (int) (long) gameData.get("player1Position");
9         player2Position = (int) (long) gameData.get("player2Position");
10        currentPlayer = (int) (long) gameData.get("currentPlayer");
11
12        gameBoard.paintPlayer(1,player1Position);
13        gameBoard.paintPlayer(2,player2Position);
14
15        gameInfo.append("Game loaded successfully.\n");
16
17    } catch (IOException | ParseException exception) {
18        exception.printStackTrace();
19        gameInfo.append("Failed to load game.\n");
20    }
21    dispose();
22    setSize(1280, 720);
23    setLayout(new FlowLayout());
24    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
25    setVisible(true);
26 }
```



```
1 private void saveGame() {
2     try {
3         JSONObject gameData = new JSONObject();
4         // Add the necessary game data to the JSON object
5         gameData.put("player1Position", player1Position);
6         gameData.put("player2Position", player2Position);
7         gameData.put("currentPlayer", currentPlayer);
8
9         // Write the JSON object to the file
10        FileWriter writer = new FileWriter("gameData.json");
11        writer.write(gameData.toJSONString());
12        writer.flush();
13        writer.close();
14
15        gameInfo.append("Game saved successfully.\n");
16
17    } catch (IOException e) {
18        e.printStackTrace();
19        gameInfo.append("Failed to save game.\n");
20    }
21 }
```



Μετά το Loading του παιχνιδιού :



3.1 Τελικό συμπέρασμα

Εν κατακλείδι, ο κώδικας παρέχει μια στηβαρή βάση για την κατασκευή διάφορων παιχνιδιών, εφαρμόζοντας μια gameEngine λογική, ενώ ταυτόχρονα χρησιμοποιεί κοινόχρηστα στοιχεία για την ρίξη των ζαριών, τη διαχείριση του ταμπλό παιχνιδιού, τη λειτουργικότητα των παικτών και τη διαχείριση του ήχου. Με αυτόν τον τρόπο, ο κώδικας επιτρέπει τη δημιουργία μιας ποικιλίας παιχνιδιών και την ευκολία συντήρησης και αναβάθμισής τους, παρέχοντας ένα ευέλικτο και αξιόπιστο πλαίσιο για δημιουργικότητα και φαντασία.

3.2 Βιβλιογραφία

Json Parser [1]:

<https://dzone.com/articles/how-can-we-read-a-json-file-in-java>

https://www.youtube.com/watch?v=yLf2r8w9lQ&ab_channel=BoostMyTool

https://www.youtube.com/watch?v=c6jlcyi4aMs&ab_channel=AwaisMirza

Java Swing[2] :

https://www.youtube.com/watch?v=Kmgo00avvEw&ab_channel=BroCode

<https://www.javatpoint.com/java-jscrollpane>

Java Map/Hash Maps[3] :

<https://www.edureka.co/blog/understanding-java-hashmaps/>

How to add 2 players in a game[4]:

<https://codereview.stackexchange.com/questions/126115/simple-two-players-dice-throwing-game>

How to parse Json In java[5]:

<https://stackoverflow.com/questions/2591098/how-to-parse-json-in-java>

How to add sound in java[6]:

<https://www.baeldung.com/java-play-sound>

<https://freesound.org/> (από εδώ έγινε χρήση ήχων σε .wav)

How to add interfaces and classes in a java project[7]:

https://www.w3schools.com/java/java_interface.asp

<https://www.geeksforgeeks.org/number-guessing-game-in-java/>

Java - load and save data with JSON-simple[8]:

https://www.youtube.com/watch?v=ywLKpHw1MjQ&ab_channel=MrCressey%27sClassVideos