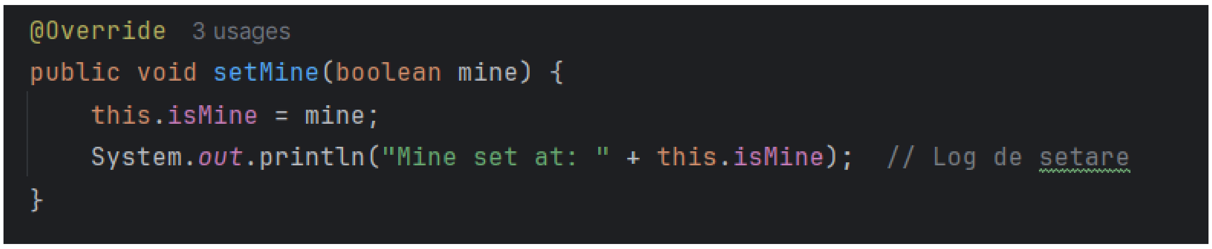
Proiectul Minesweeper este o implementare a unui joc clasic, dezvoltat in Java folosind JavaFX pentru interfata grafica. Scopul proiectului este de a oferi o aplicatie completa si modulara, respectand principiile de programare orientata pe obiecte. Proiectul se incadreaza in cerintele laboratorului de Java si acopera toate aspectele specificate in barem.

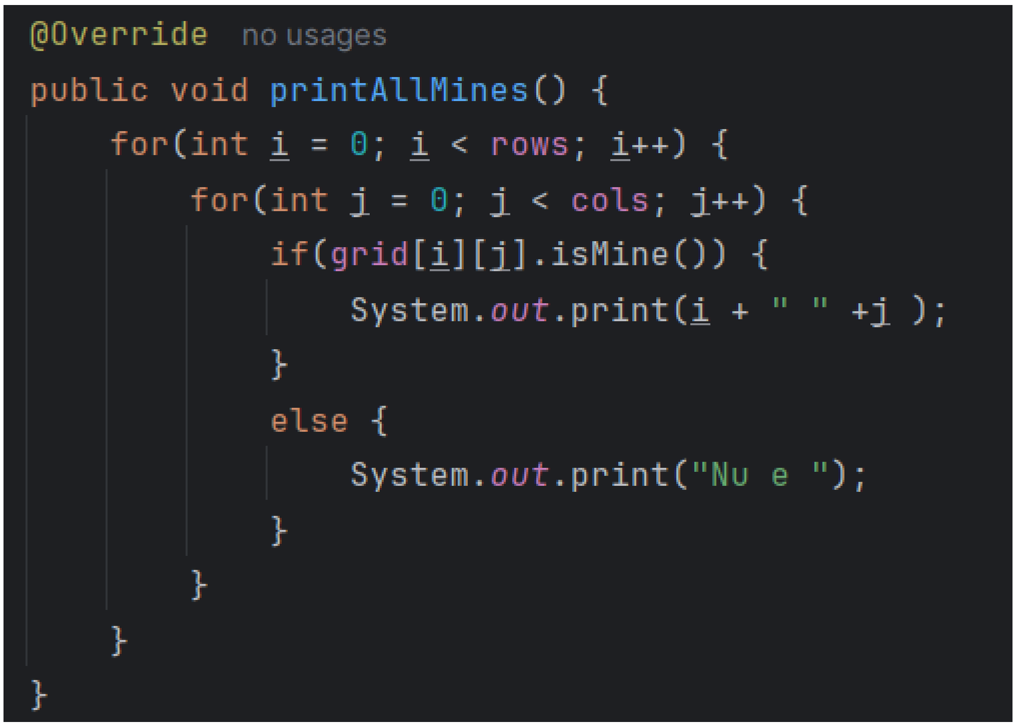
Corelarea cu Cerintele Laboratoarelor

Lab1: Introducere in Java (output, tipuri de valori, functii)



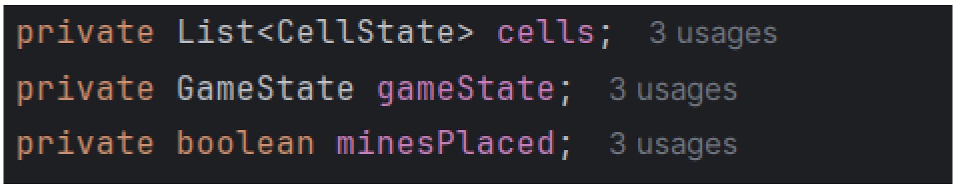
```java
@Override  3 usages
public void setMine(boolean mine) {
    this.isMine = mine;
    System.out.println("Mine set at: " + this.isMine);  // Log de setare
}
```

Lab2: Introducere in Java (input, for, while, switch, if)

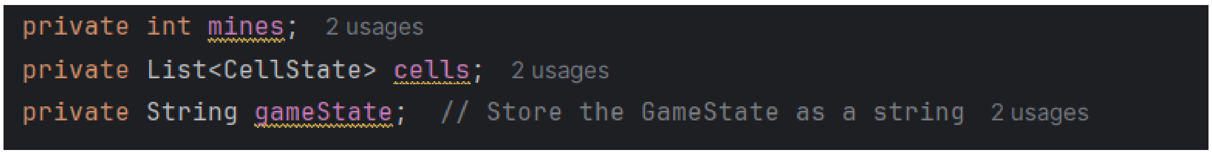


```java
@Override  no usages
public void printAllMines() {
    for(int i = 0; i < rows; i++) {
        for(int j = 0; j < cols; j++) {
            if(grid[i][j].isMine()) {
                System.out.print(i + " " +j );
            }
            else {
                System.out.print("Nu e ");
            }
        }
    }
}
```

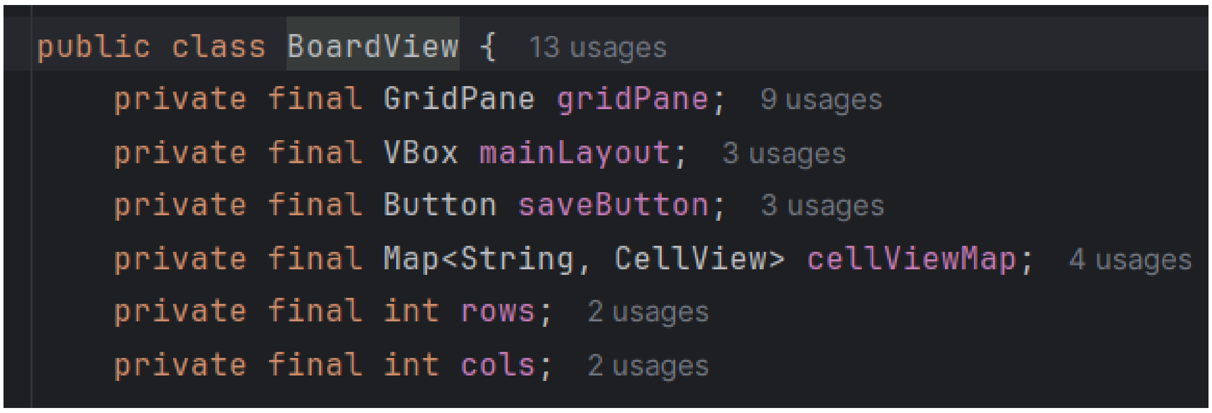
Lab3: Colectii Java (Array, List, Map)



```
private List<CellState> cells;  3 usages
private GameState gameState;  3 usages
private boolean minesPlaced;  3 usages
```

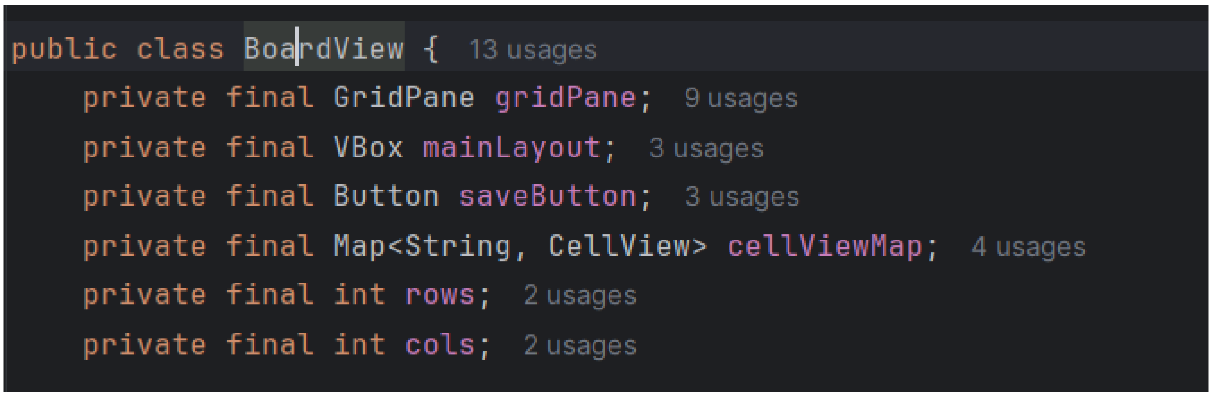


```
private int mines;  2 usages
private List<CellState> cells;  2 usages
private String gameState;  // Store the GameState as a string  2 usages
```

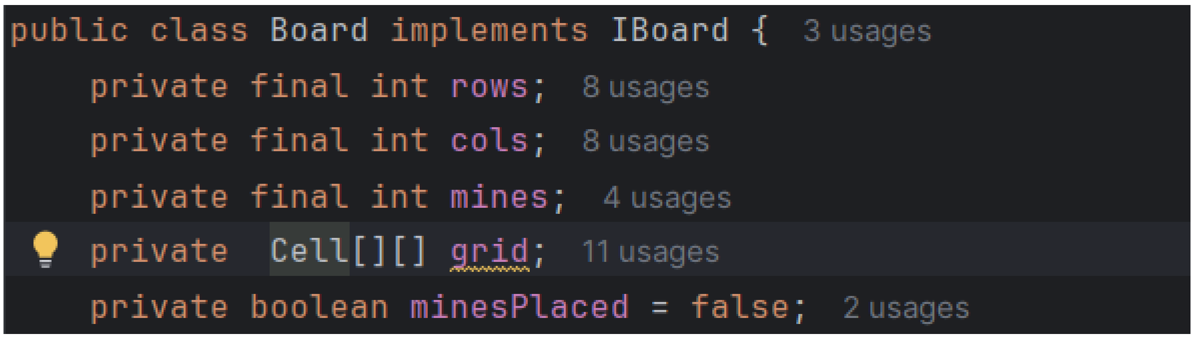


```
public class BoardView {  13 usages
    private final GridPane gridPane;  9 usages
    private final VBox mainLayout;  3 usages
    private final Button saveButton;  3 usages
    private final Map<String, CellView> cellViewMap;  4 usages
    private final int rows;  2 usages
    private final int cols;  2 usages
```

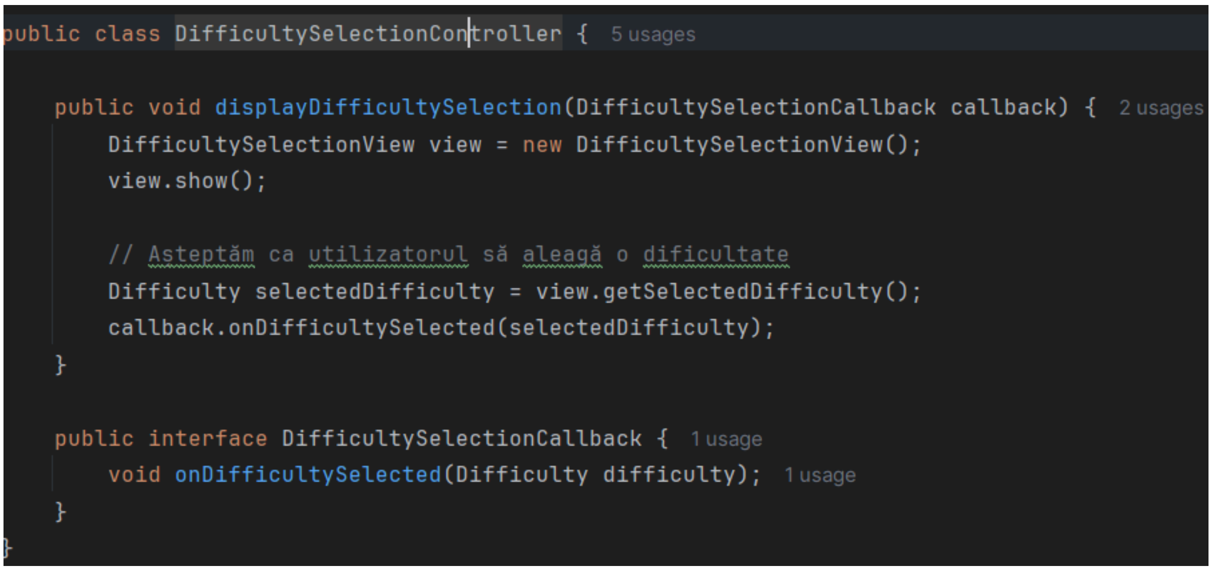
Lab4: Clase Java (clasa cu atribute si metode)



```java
public class BoardView {  13 usages
    private final GridPane gridPane;  9 usages
    private final VBox mainLayout;  3 usages
    private final Button saveButton;  3 usages
    private final Map<String, CellView> cellViewMap;  4 usages
    private final int rows;  2 usages
    private final int cols;  2 usages
```

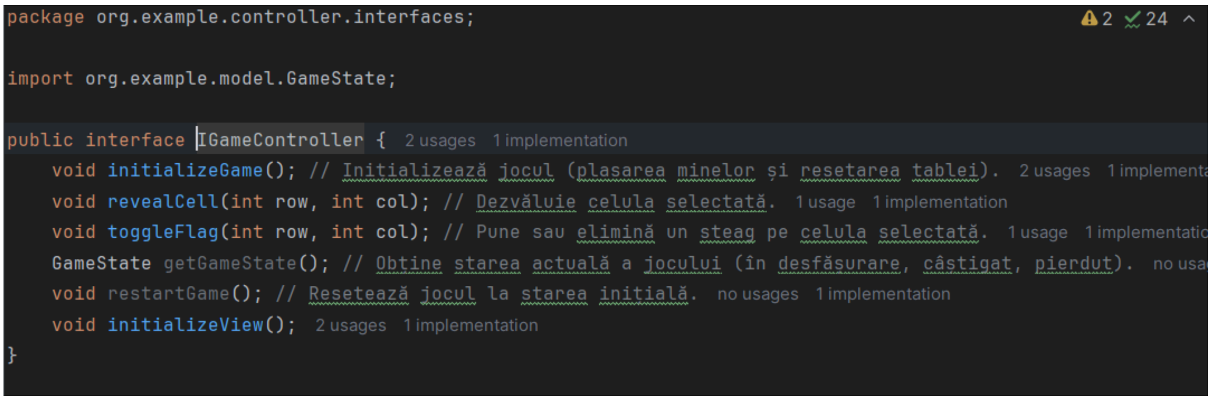


```java
public class Board implements IBoard {  3 usages
    private final int rows;  8 usages
    private final int cols;  8 usages
    private final int mines;  4 usages
    private  Cell[][] grid;  11 usages
    private boolean minesPlaced = false;  2 usages
```

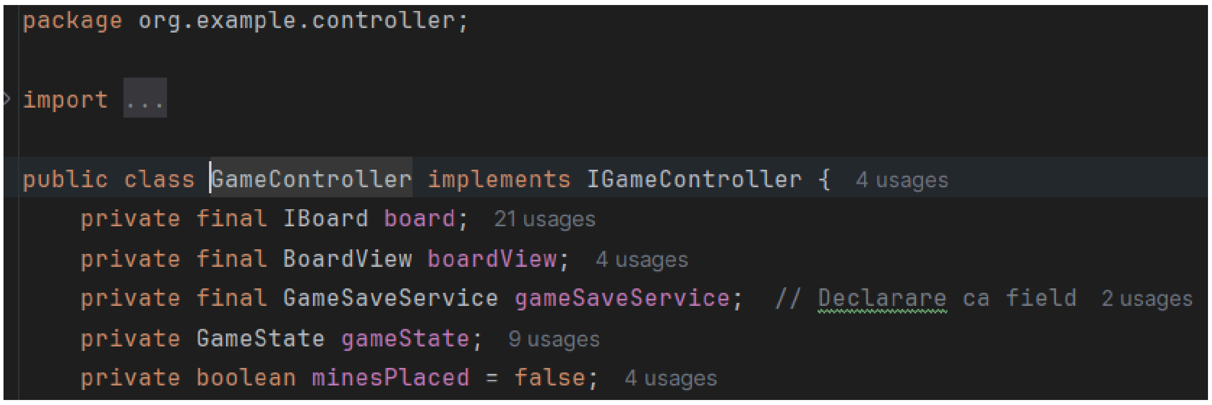


```java
public class DifficultySelectionController {  5 usages

    public void displayDifficultySelection(DifficultySelectionCallback callback) {  2 usages
        DifficultySelectionView view = new DifficultySelectionView();
        view.show();

        // Asteptăm ca utilizatorul să aleagă o dificultate
        Difficulty selectedDifficulty = view.getSelectedDifficulty();
        callback.onDifficultySelected(selectedDifficulty);
    }

    public interface DifficultySelectionCallback {  1 usage
        void onDifficultySelected(Difficulty difficulty);  1 usage
    }
}
```

Lab5: Mostenire in Java, clase abstracte

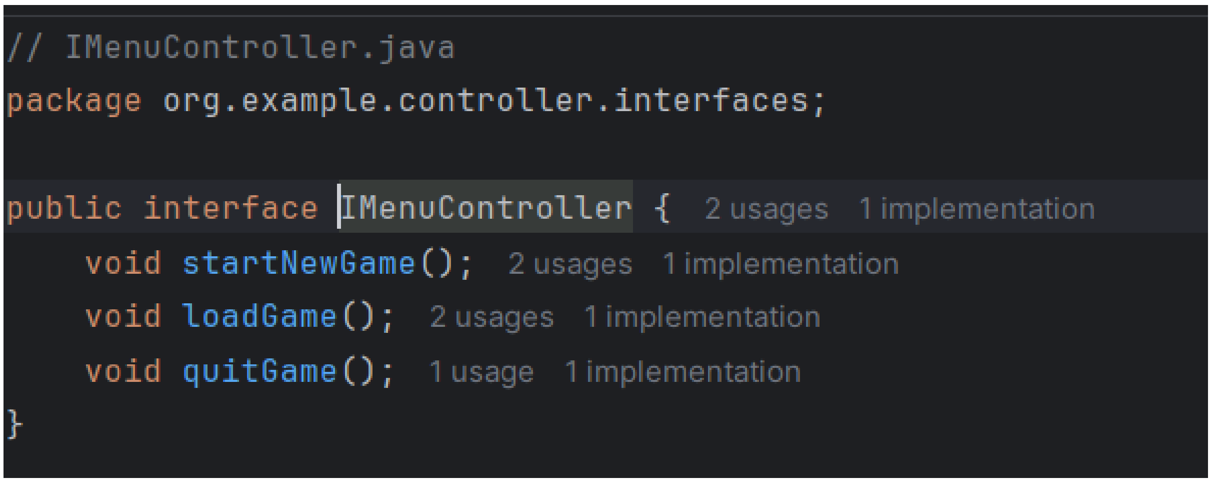


```
package org.example.controller.interfaces;                                              ⚠2 ✓24 ^

import org.example.model.GameState;

public interface IGameController {  2 usages  1 implementation
    void initializeGame(); // Initializează jocul (plasarea minelor și resetarea tablei).  2 usages  1 implementa
    void revealCell(int row, int col); // Dezvăluie celula selectată.  1 usage  1 implementation
    void toggleFlag(int row, int col); // Pune sau elimină un steag pe celula selectată.  1 usage  1 implementatio
    GameState getGameState(); // Obtine starea actuală a jocului (în desfășurare, câștigat, pierdut).  no usa
    void restartGame(); // Resetează jocul la starea initială.  no usages  1 implementation
    void initializeView();  2 usages  1 implementation
}
```



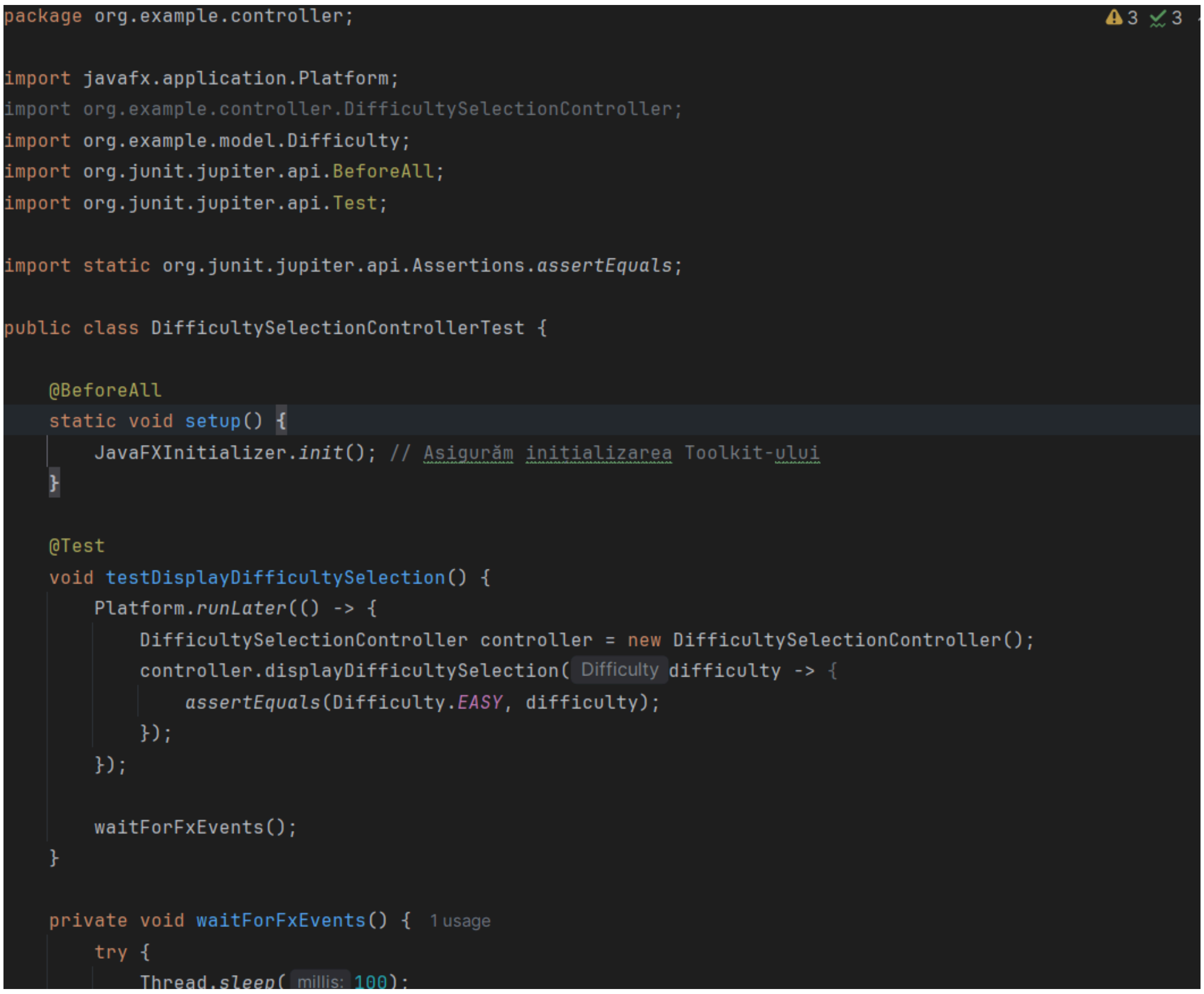
```
package org.example.controller;

import ...

public class GameController implements IGameController {  4 usages
    private final IBoard board;  21 usages
    private final BoardView boardView;  4 usages
    private final GameSaveService gameSaveService;  // Declarare ca field  2 usages
    private GameState gameState;  9 usages
    private boolean minesPlaced = false;  4 usages
```

Lab6: Interfete in Java

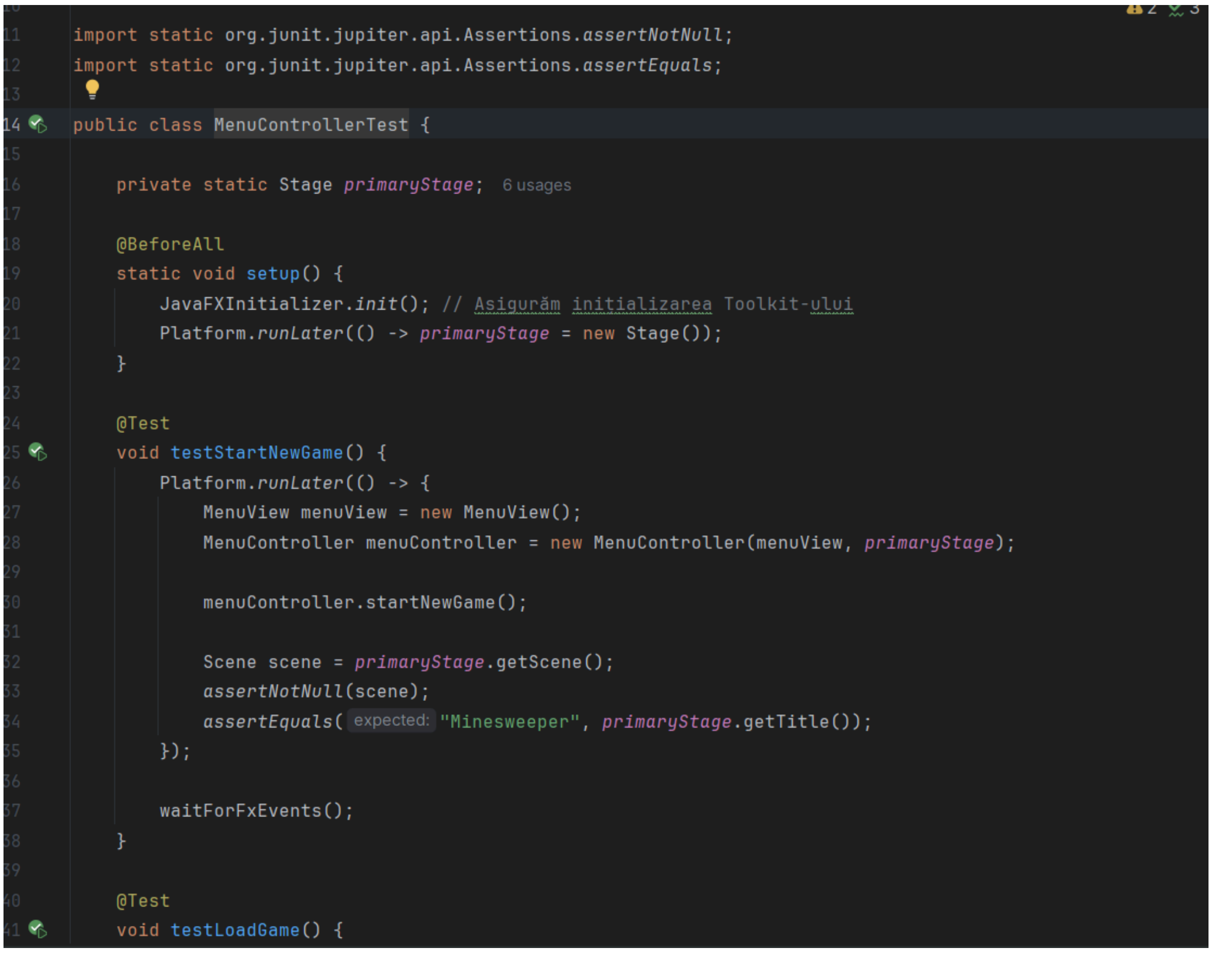


```
// IMenuController.java
package org.example.controller.interfaces;

public interface IMenuController {  2 usages  1 implementation
    void startNewGame();  2 usages  1 implementation
    void loadGame();  2 usages  1 implementation
    void quitGame();  1 usage  1 implementation
}
```

Lab7: Teste

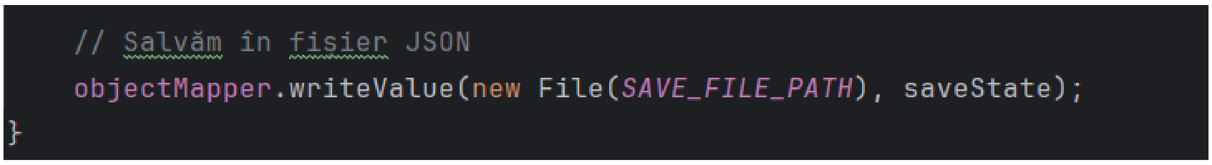


```
package org.example.controller;                                                          ⚠3 ✓3

import javafx.application.Platform;
import org.example.controller.DifficultySelectionController;
import org.example.model.Difficulty;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

public class DifficultySelectionControllerTest {

    @BeforeAll
    static void setup() {
        JavaFXInitializer.init(); // Asigurăm inițializarea Toolkit-ului
    }

    @Test
    void testDisplayDifficultySelection() {
        Platform.runLater(() -> {
            DifficultySelectionController controller = new DifficultySelectionController();
            controller.displayDifficultySelection( Difficulty difficulty -> {
                assertEquals(Difficulty.EASY, difficulty);
            });
        });

        waitForFxEvents();
    }

    private void waitForFxEvents() {  1 usage
        try {
            Thread.sleep( millis: 100);
```

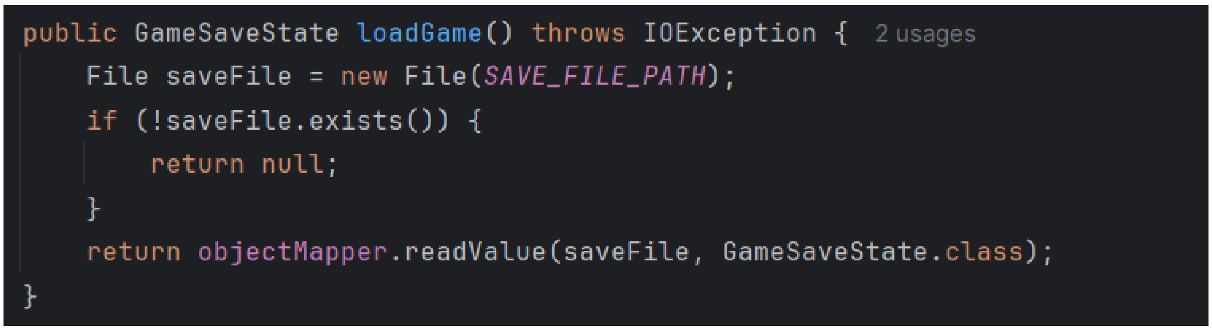


```java
import static org.junit.jupiter.api.Assertions.assertNotNull;
import static org.junit.jupiter.api.Assertions.assertEquals;

public class MenuControllerTest {

    private static Stage primaryStage;  6 usages

    @BeforeAll
    static void setup() {
        JavaFXInitializer.init(); // Asigurăm initializarea Toolkit-ului
        Platform.runLater(() -> primaryStage = new Stage());
    }

    @Test
    void testStartNewGame() {
        Platform.runLater(() -> {
            MenuView menuView = new MenuView();
            MenuController menuController = new MenuController(menuView, primaryStage);

            menuController.startNewGame();

            Scene scene = primaryStage.getScene();
            assertNotNull(scene);
            assertEquals( expected: "Minesweeper", primaryStage.getTitle());
        });

        waitForFxEvents();
    }

    @Test
    void testLoadGame() {
```

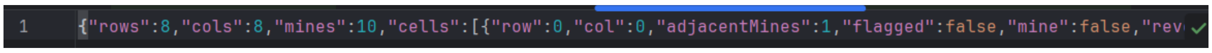
Lab8: Persistenta datelor



```java
    // Salvăm în fisier JSON
    objectMapper.writeValue(new File(SAVE_FILE_PATH), saveState);
}
```



```java
public GameSaveState loadGame() throws IOException {  2 usages
    File saveFile = new File(SAVE_FILE_PATH);
    if (!saveFile.exists()) {
        return null;
    }
    return objectMapper.readValue(saveFile, GameSaveState.class);
}
```



```
{"rows":8,"cols":8,"mines":10,"cells":[{"row":0,"col":0,"adjacentMines":1,"flagged":false,"mine":false,"rev
```

Diagrama sequence:

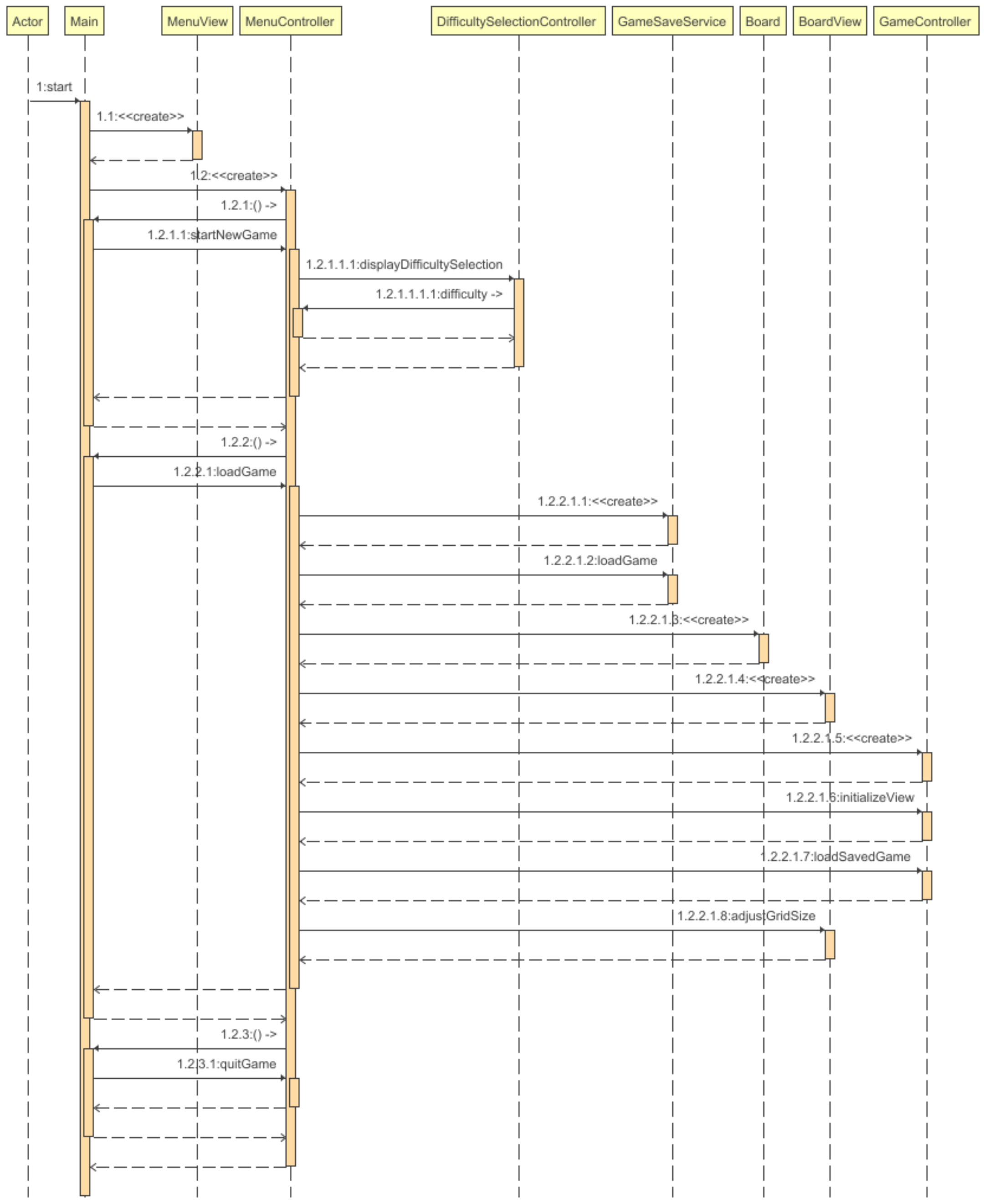


Diagrama class

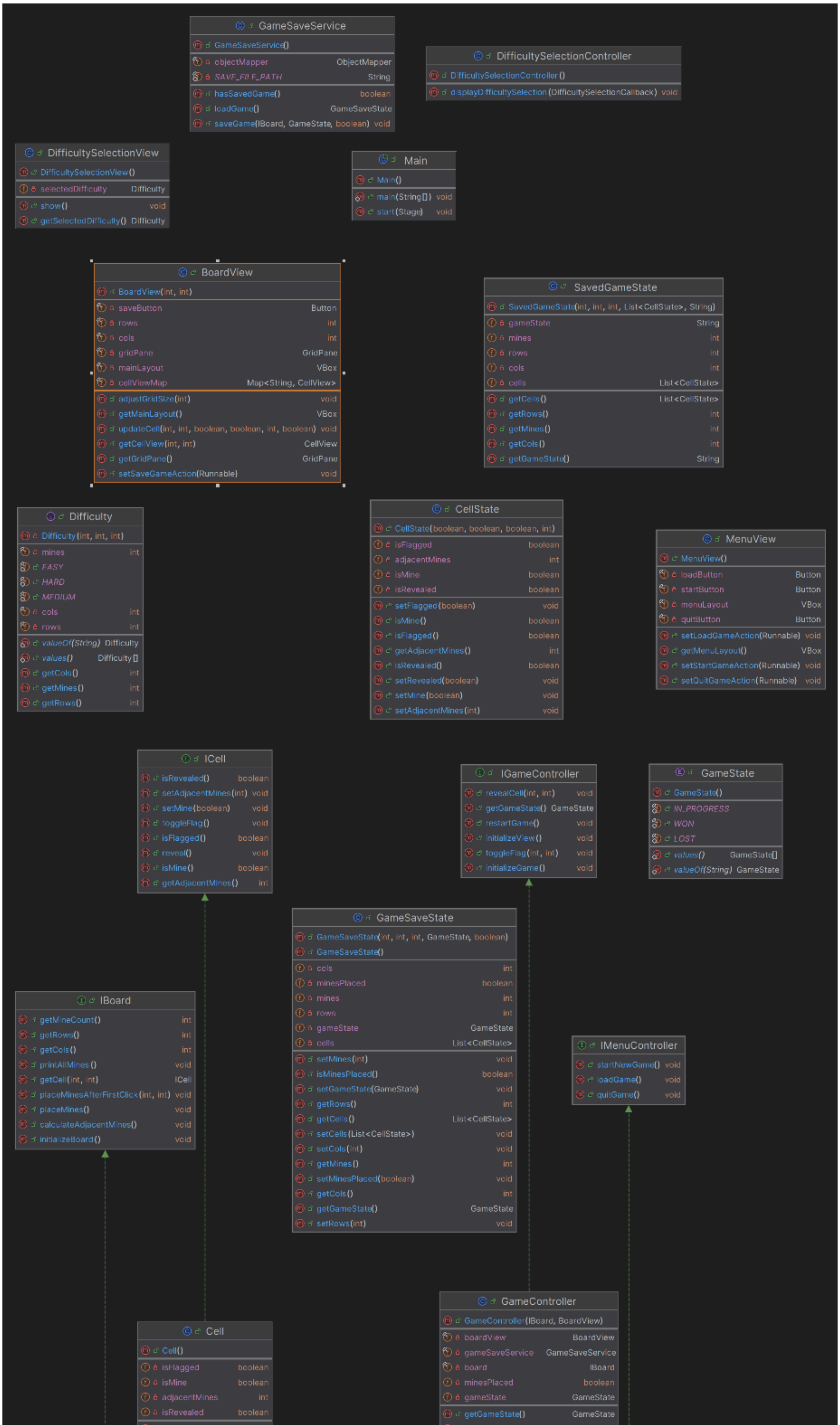
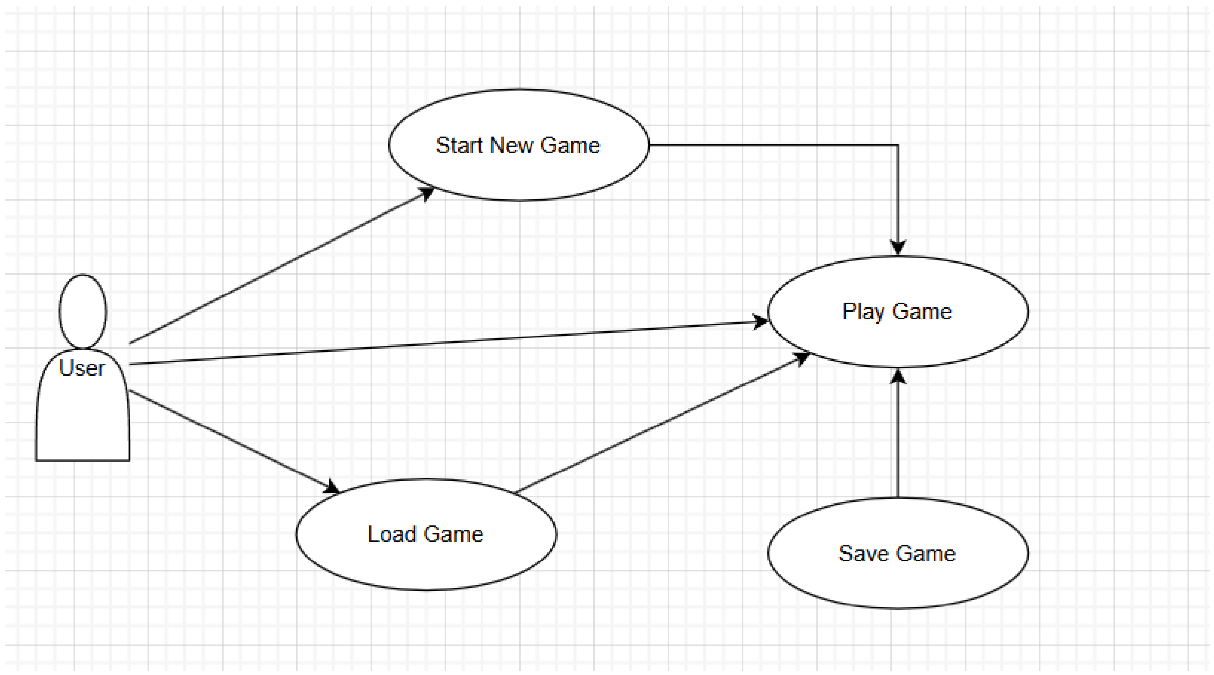

GameSaveService
GameSaveService()
objectMapper ObjectMapper
SAVE_FILE_PATH String
hasSavedGame() boolean
loadGame() GameSaveState
saveGame(IBoard, GameState, boolean) void
DifficultySelectionController
DifficultySelectionController()
displayDifficultySelection(DifficultySelectionCallback) void
DifficultySelectionView
DifficultySelectionView()
selectedDifficulty Difficulty
show() void
getSelectedDifficulty() Difficulty
Main
Main()
main(String[]) void
start(Stage) void
BoardView
BoardView(int, int)
saveButton Button
rows int
cols int
gridPane GridPane
mainLayout VBox
cellViewMap Map<String, CellView>
adjustGridSize(int) void
getMainLayout() VBox
updateCell(int, int, boolean, boolean, int, boolean) void
getCellView(int, int) CellView
getGridPane() GridPane
setSaveGameAction(Runnable) void
SavedGameState
SavedGameState(int, int, int, List<CellState>, String)
gameState String
mines int
rows int
cols int
cells List<CellState>
getCells() List<CellState>
getRows() int
getMines() int
getCols() int
getGameState() String
Difficulty
Difficulty(int, int, int)
mines int
EASY
HARD
MEDIUM
cols int
rows int
valueOf(String) Difficulty
values() Difficulty[]
getCols() int
getMines() int
getRows() int
CellState
CellState(boolean, boolean, boolean, int)
isFlagged boolean
adjacentMines int
isMine boolean
isRevealed boolean
setFlagged(boolean) void
isMine() boolean
isFlagged() boolean
getAdjacentMines() int
isRevealed() boolean
setRevealed(boolean) void
setMine(boolean) void
setAdjacentMines(int) void
MenuView
MenuView()
loadButton Button
startButton Button
menuLayout VBox
quitButton Button
setLoadGameAction(Runnable) void
getMenuLayout() VBox
setStartGameAction(Runnable) void
setQuitGameAction(Runnable) void
ICell
isRevealed() boolean
setAdjacentMines(int) void
setMine(boolean) void
toggleFlag() void
isFlagged() boolean
reveal() void
isMine() boolean
getAdjacentMines() int
IGameController
revealCell(int, int) void
getGameState() GameState
restartGame() void
initializeView() void
toggleFlag(int, int) void
initializeGame() void
GameState
GameState()
IN_PROGRESS
WON
LOST
values() GameState[]
valueOf(String) GameState
GameSaveState
GameSaveState(int, int, int, GameState, boolean)
GameSaveState()
cols int
minesPlaced boolean
mines int
rows int
gameState GameState
cells List<CellState>
setMines(int) void
isMinesPlaced() boolean
setGameState(GameState) void
getRows() int
getCells() List<CellState>
setCells(List<CellState>) void
setCols(int) void
getMines() int
setMinesPlaced(boolean) void
getCols() int
getGameState() GameState
setRows(int) void
IBoard
getMineCount() int
getRows() int
getCols() int
printAllMines() void
getCell(int, int) ICell
placeMinesAfterFirstClick(int, int) void
placeMines() void
calculateAdjacentMines() void
initializeBoard() void
IMenuController
startNewGame() void
loadGame() void
quitGame() void
Cell
Cell()
isFlagged boolean
isMine boolean
adjacentMines int
isRevealed boolean
GameController
GameController(IBoard, BoardView)
boardView BoardView
gameSaveService GameSaveService
board IBoard
minesPlaced boolean
gameState GameState
getGameState() GameState

Diagrama use case:


Start New Game
User
Play Game
Load Game
Save Game