```java
package maze;

import java.awt.Point;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.Timer;

import maze.data.MazeContainer;
import maze.data.MazeElem;
import maze.data.MazeUtils.Direction;
import maze.data.MazeUtils.Player;
import maze.display.GraphicDisplay;
import maze.display.KeyboardListener;
import maze.solvers.AStar;

/**
 * Game logic for moving the players and selectively showing the solution
 *
 * @author Pierre-Andre Mudry, Romain Cherix
 * @date February 2012
 * @version 1.4
 */
public class MazeGame {

    public MazeElem[][] maze;
    int width, height;

    GraphicDisplay gd;
    MazeContainer mc;
    KeyboardListener kl;

    /**
     * By default, you are the first player but this can change..
     */
    Player player = Player.PLAYER1;

    MazeGame MazeContainer mc) {
        gd = new GraphicDisplay(mc, 12);

        // Link key presses with the actions
        kl = new KeyboardListener(mc, this);
        gd.registerKeyListener(kl);

        setNewMaze(mc);
    }

    /**
     * Dynamically changes the maze that is displayed
     *
     * @param mc
     */
    public void setNewMaze MazeContainer mc) {
        maze = mc.maze;
        width = mc.nCellsX;
        height = mc.nCellsY;
        this.mc = mc;

        /**
         * Update graphical maze
         * FIXME : this should allow mazes of different sizes,
         * which is not the case now
         */
        gd.setNewMaze(mc);
    }

    /**
     * Displays the solution on the screen for a player during a whole second
     *
     * @param p Which player's solution do you want
     */
    public void displaySolution() {
        Point p1 = findPlayer(player);

        int[][] solution = AStar.solve(mc, p1.x, p1.y);
        gd.setSolution(solution);

        Timer timer = new Timer(1000, new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                gd.clearSolution();
            }
        });

        timer.setRepeats(false);
        timer.start();
    }
```

```java

    /**
     * Call this when you want a new game
     * @param mazeID
     */
    public void generateNewMaze(int mazeID){
        this.setNewMaze(new MazeContainer(width, height, mazeID));
    }

    /**
     * Gives us the location of player inside the maze
     *
     * @param p The player we want
     * @return The location of the player
     */
    private Point findPlayer(Player p) {
        /**
         * We go through all the elements
         */
        for (int i = 0; i < height; i++) {
            for (int j = 0; j < width; j++) {

                // TODO Complete this
                Point point = new Point(0,0);
                return point;

            }
        }

        // This means that the player hasn't been found
        // which can happen for instance in single player games
        return null;
    }

    /**
     * Check if some player has reached the exit of the maze
     */
    public boolean checkWinner() {

        // TODO Complete this

        return false;
    }

    /**
     * Method used to move a player inside the maze
     *
     * @param d Which direction do you want to go to ?
     */
    public void movePlayer(Direction d) {

        // TODO Complete this

    }

    public static void main(String[] args) {
        MazeContainer mc = new MazeContainer(10, 10);
        MazeGame mg = new MazeGame(mc);

        // TODO Students should implement next line
        // mg.movePlayer(Direction.DOWN);

        // This shows a nice message window
        // JOptionPane.showMessageDialog(null, "Title of the window", "Text of the window !",
JOptionPane.INFORMATION_MESSAGE);
    }
}

```