Final project: Minesweeper

Eva Cristina Beltrán Reyes, A01114495

Introduction

Minesweeper is a video game that was originally created by Robert Donner in 1989. Its objective is to find all the places that have no mines in a field with a fixed number of them. Now, the because of its nature, it will be used as a class exercise for practicing object oriented analysis, design and programming principles. Also, for using GUI elements and relating them with a logic package.

Project description

The program was made in NetBeans in 2019, using JAVA language. It models a variant of the known game "Minesweeper" and its objective is to find all the squares that do not have a bomb as a content.

How to play: The user will click in the initial squares trying to unlock all the white squares and numbers and, if desired, he or she will mark the bombs with flags.

How to win: When all the white squares and numbers are displayed and no bomb was clicked on.

List of features

Levels:

There will be two levels: the first one, the "easy" level, will be a square grid with 7 rows and 7 columns of smaller squares and will have 10 bombs. The second one, the "hard" level will be a square grid with 10 rows and 10 columns of smaller squares; also, it will have 20 bombs.

Grid elements:

The game has different elements that make it work, these are:

Initial squares: These are the first ones to appear to the user, the initial grid is formed by this squares.

Flag: This element can be used by the user to represent a place were a bomb is hidden. Represented with a blue square with a P.

White square: This element represents that there is nothing hidden in that square. Represented with a pink square.

Number: This element represents the number of bombs that are touching (around) that square. Represented with a yellow square and the corresponding number. s

Bomb: This element represents that there is a bomb hidden in that square, if it appears to the user it will explode. Represented with a red square and a *.

Grid actions:

This part will describe what action can be done with each element and what is its result.

Initial squares: Two different actions can be done to this type of element: left click with the mouse with and activated or deactivated flag.

The left click with deactivated flag can have the next results:

Flag: the flag blocks the left click with activated flag (it will not do any change to that specific square) and can be taken off with the reset button.

The left click with activated flag can result in three different elements: white square, number, bomb. After the click no more action can be done to this element. The appearance of the elements is the action itself. Also, when a white square is clicked on (unlocked) it will unlock all the other white squares that are on its surroundings to all directions an it will stop when it finds a number in each direction (but it will also display the number). When a number is unlocked only the number will show. On the other hand, if the bomb is clicked, it will show all the other bombs and the user will loose the game.

Menu elements:

Flag: This button will activate or deactivate the flag. If it is true, when the user clicks a button it will become a flag. When it is false, the hidden property will be shown (white square, bomb or number). The state of the button is shown below it.

Save game: This button will save the current movements in a file with a name given by the user inside a folder in the game using serialization. This file can be opened and the game is resumed.

Restart: Makes the grid go to its initial state and does not change the hidden elements.

New game: Changes all the hidden elements and makes all the squares gray once more (initial squares).

Initial look elements:

Easy button: Opens a game that employs an easy level.

Hard button: Opens a game that employs a hard level.

Load game: Opens a file with a name given by the user. It is for resuming a previously saved game. If the file does not exist, the program will inform the user.

Win or loose:

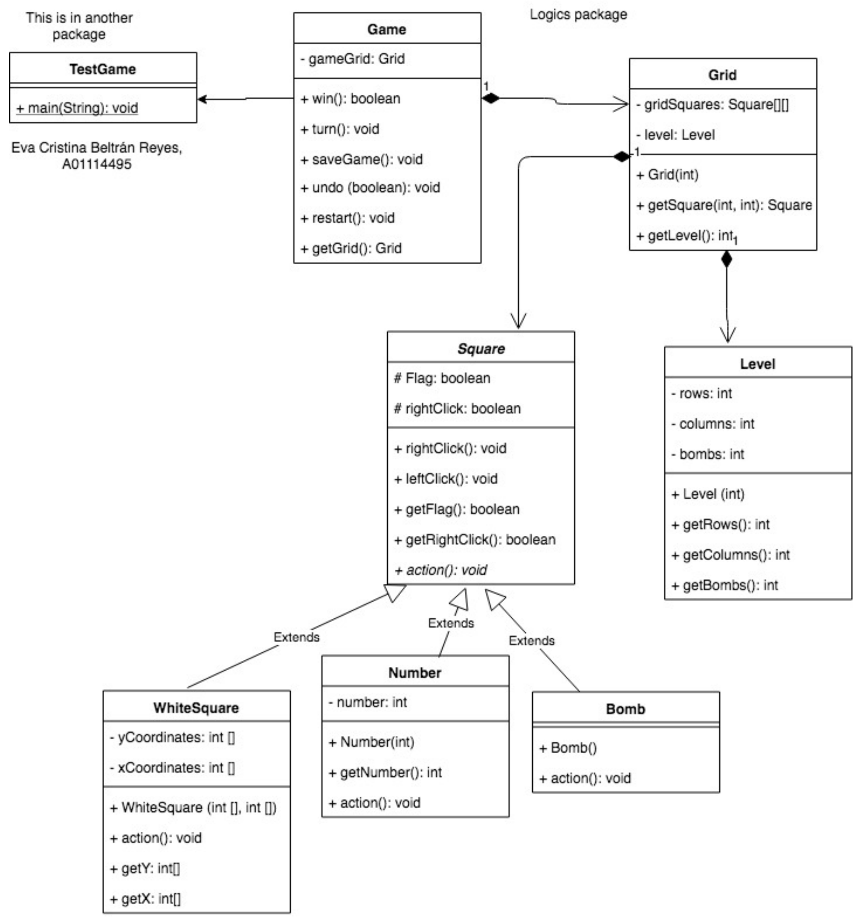
Win: The user will win if all the squares that are not bombs have been discovered.

Loose: The user will loose if he or she clicks in a bomb. The program will inform the user the game has been lost and will give a new game.

Diagrams: OOAD process:

Find diagram here: https://drive.google.com/file/d/1AIVD6IwwQm-hj-byei-tSvTHIuvdFeWD/view?usp=sharing

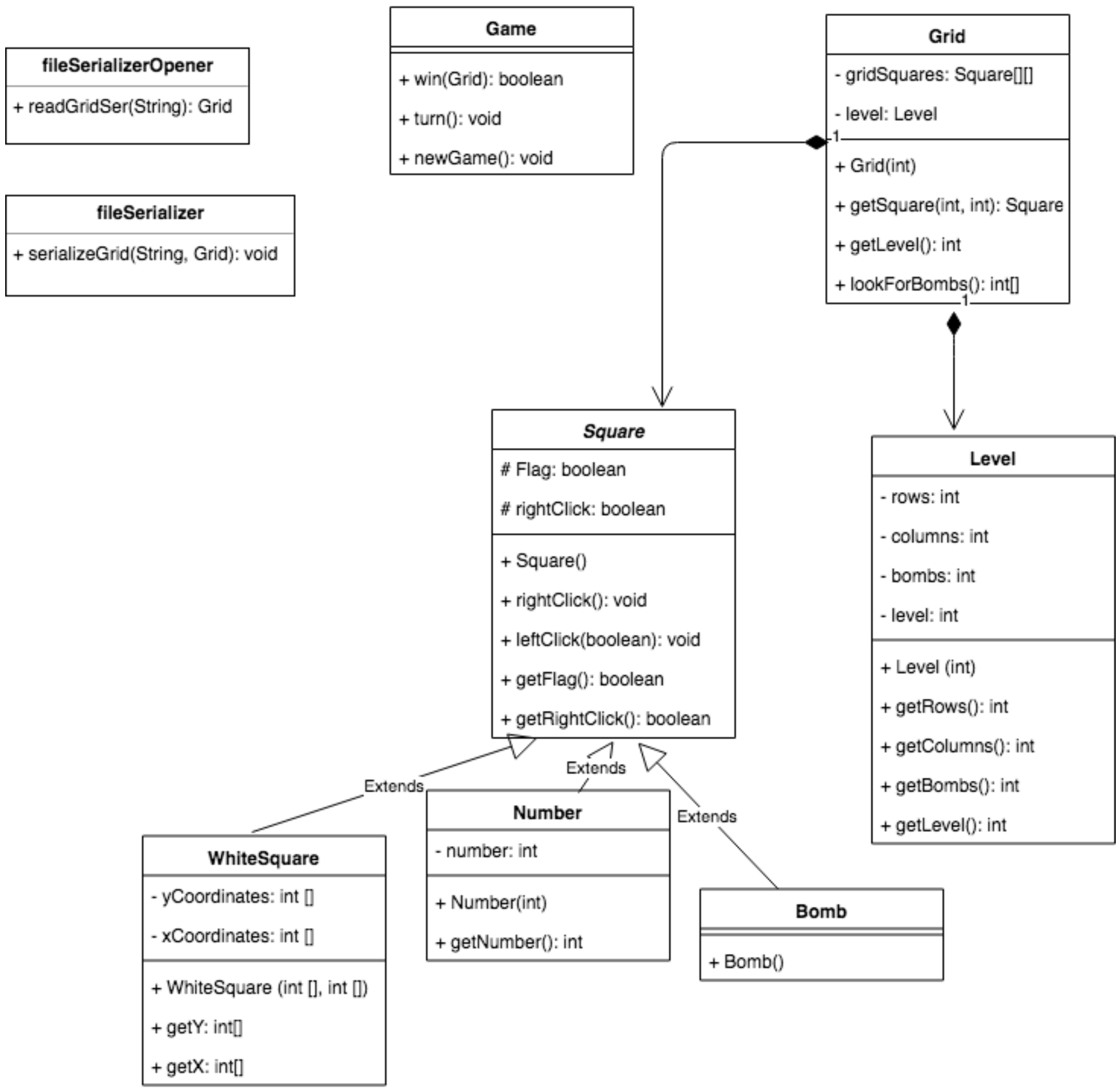
Initial diagram:



Final diagrams:
The diagram was changed, because while coding, I realized I could make the program more efficient or the methods could process different things so it would be easier for the GUI to handle the grid of squares.

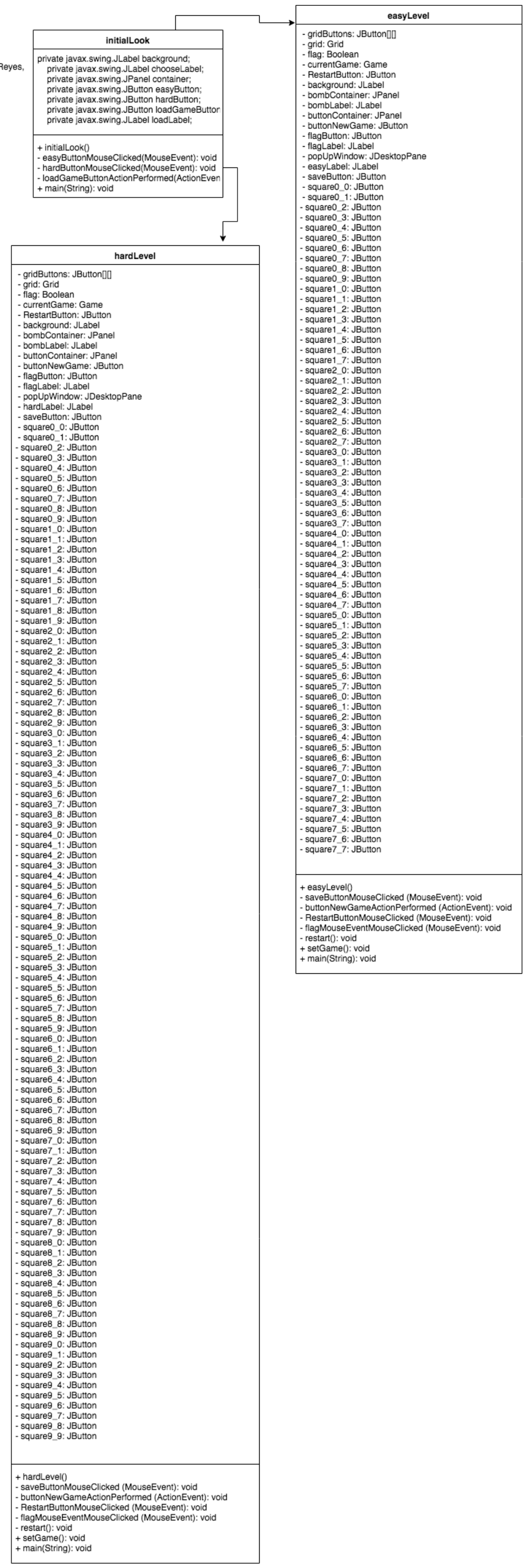
Eva Cristina Beltrán Reyes,
A01114495

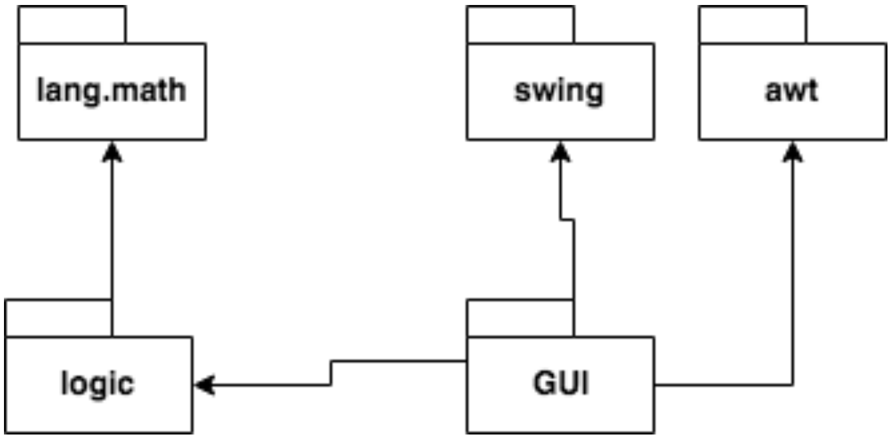
Logics package



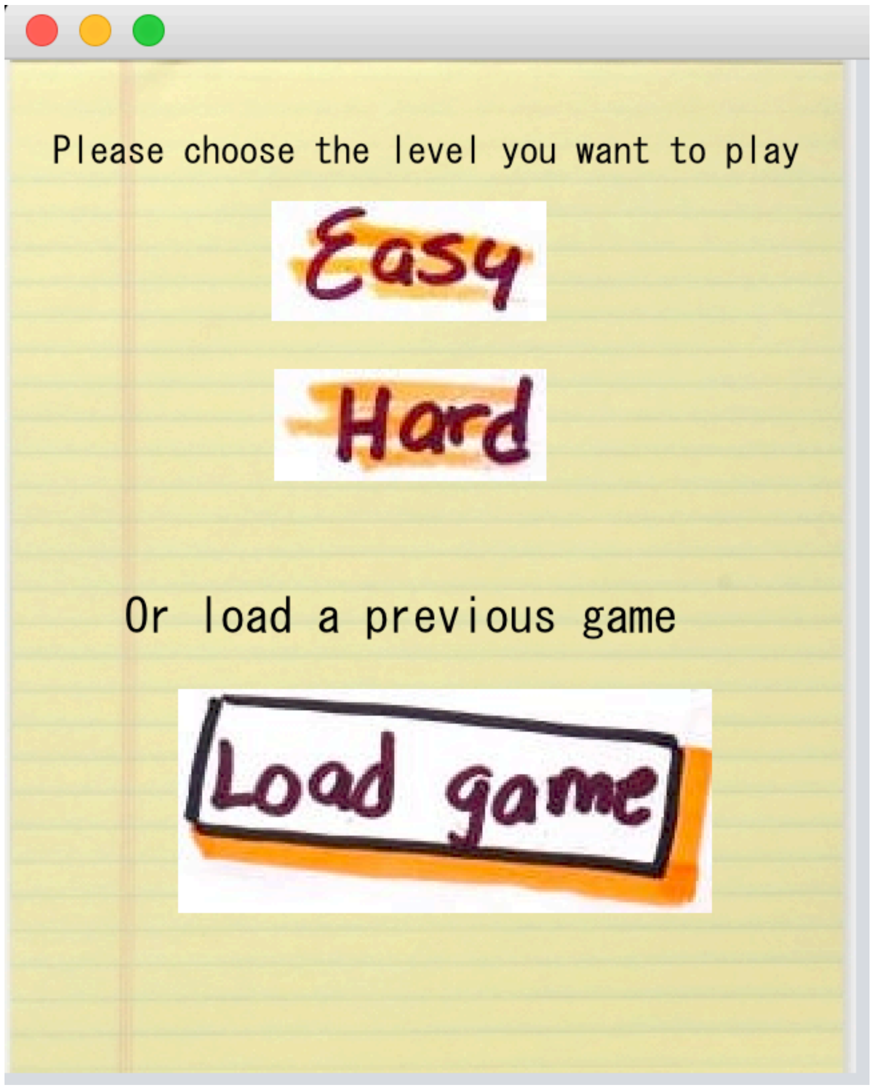
GUI
package

Eva Cristina Beltrán Reyes,
A01114495

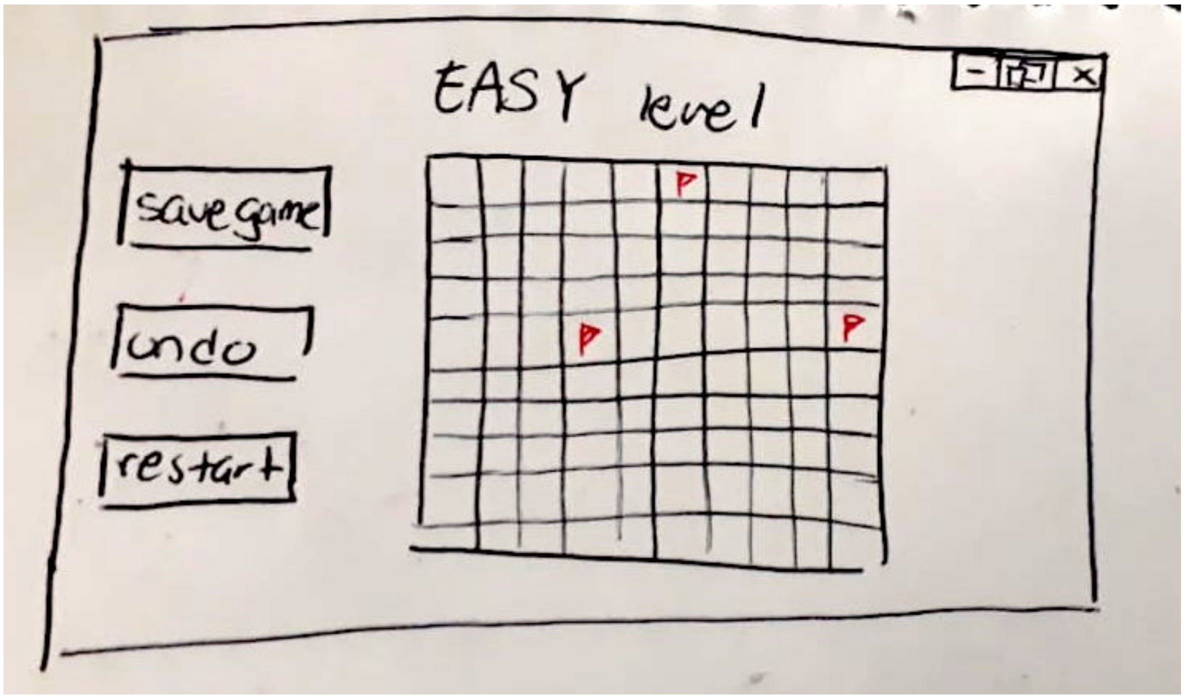


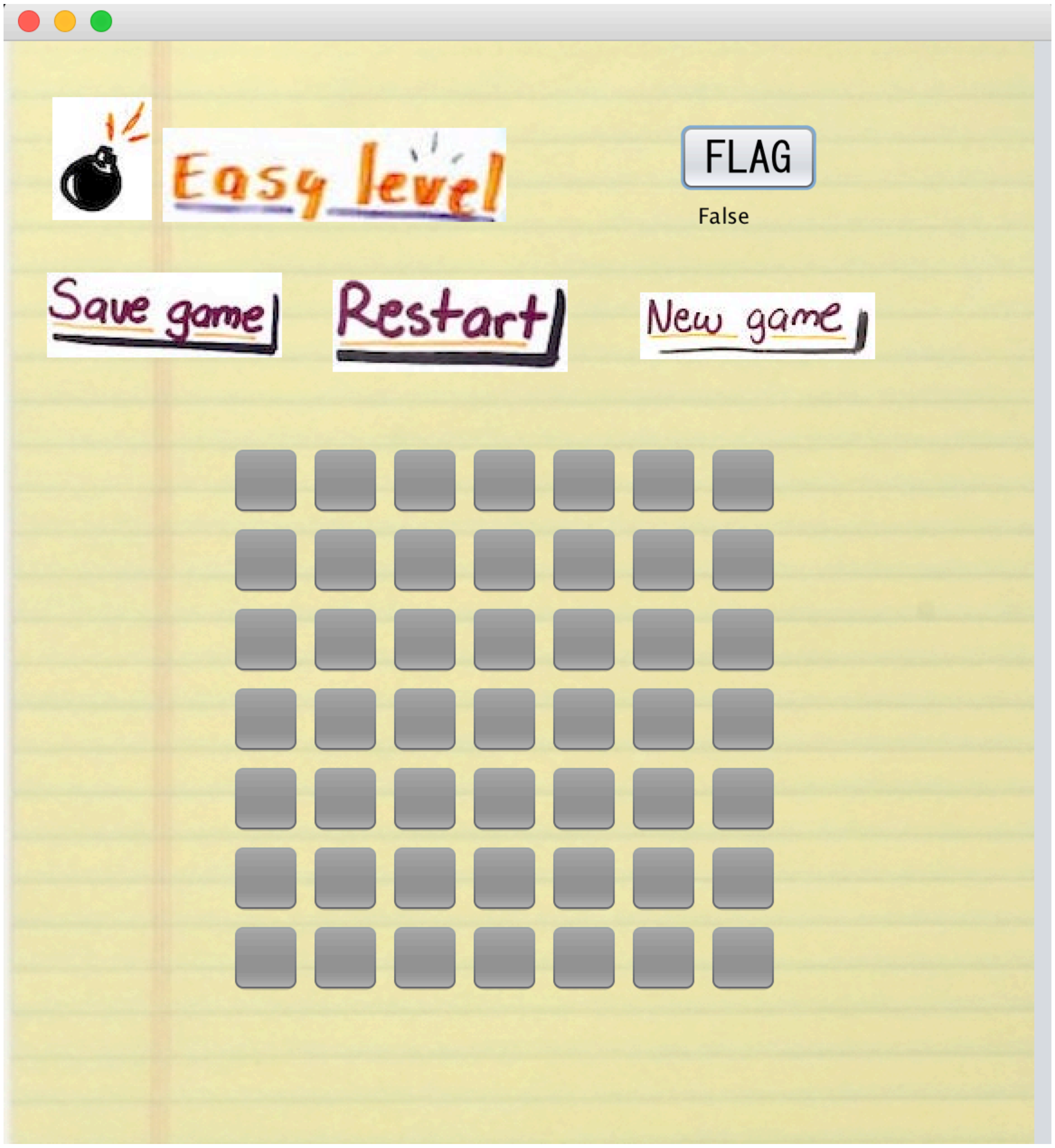


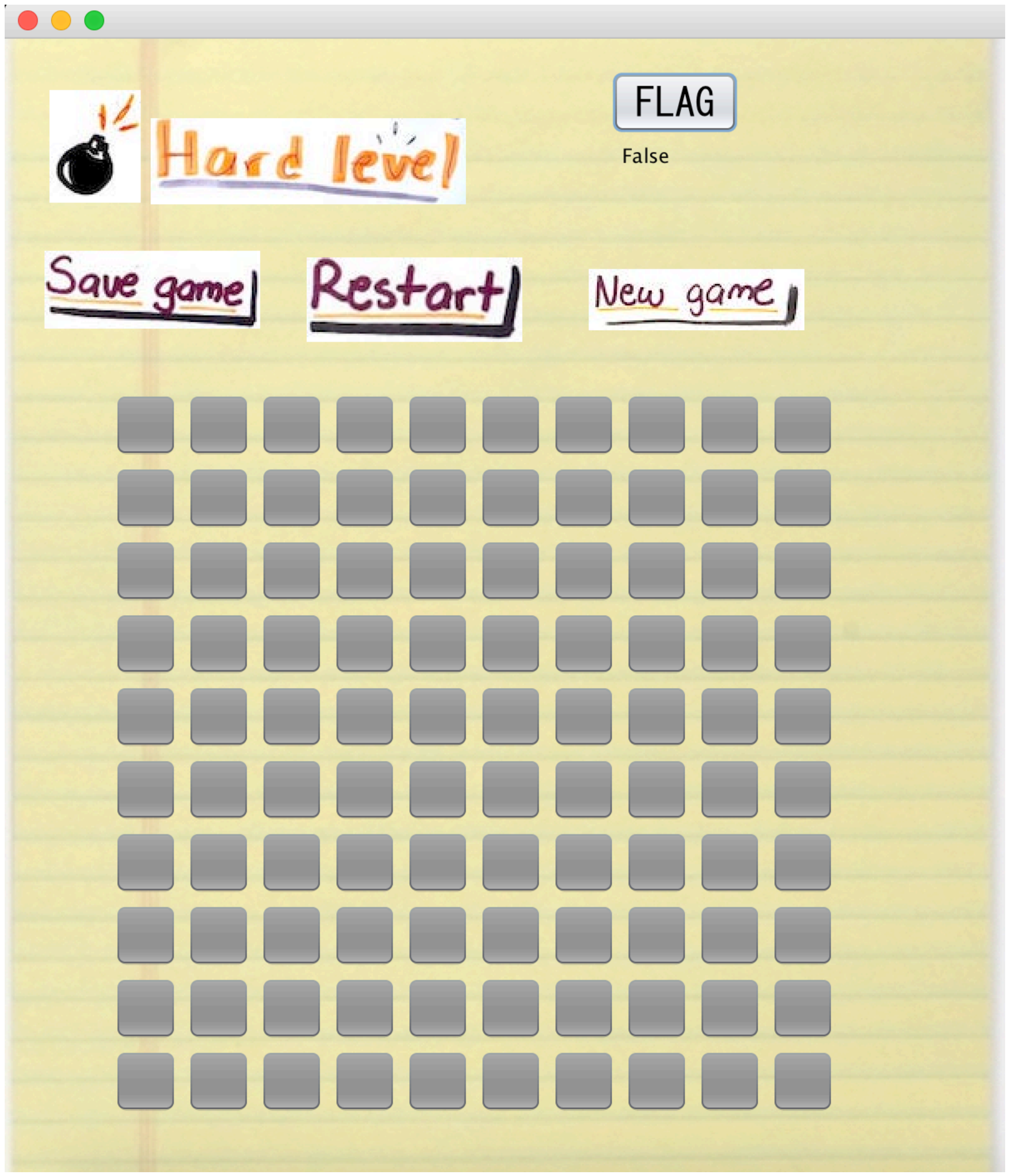
Mock-ups:
The game initially starts with a screen that lets the user choose between the two levels or continuing a previously saved game.



After clicking an option, the game looks like a grid of squares (its size its given by the chosen level) and in each square there is a hidden element which the user can discover by clicking on it. Above the grid, there is a menu with 4 options: save game (saves the current movements and actions of the user), restart (erases all the current movements but it doesn’t change the hidden elements), new game (erases all the current movements, rearranges the bombs and starts over the game) and flag (for placing flags).


EASY level
save game
undo
restart
-
x


Easy level
FLAG
False
Save game
Restart
New game



Total time spent:
The time for analysis was: 70 minutes (1 hour and 10 minutes)
The time for design was: 100 minutes (1 hour and 40 minutes)
The time for programming and testing was: 1437 minutes (23 hours and 57 minutes)
The total time for the project was: 1607 minutes (26 hours and 47 minutes)