# Lab 5: JavaFX GUI

## 1. Objective

1) Students are able to implement GUI using JavaFx.

## 2. Instruction

1) Create Java Project named **"2110215_Lab5"**.
2) Copy folders inside **"toStudent/src"** to your project directory src folder.
3) Copy **"toStudent/res"** folder to your project directory folder and make it a source folder.
4) You are to implement the following classes (detail for each class is given in section 3 and 4)

   a) **ControlGridPane**        (package gui)

   b) **ControlPane**        (package gui)

   c) **MineSweeperSquare**      (package gui)

   d) **MineSweeperPane** (package gui)

## 3. Problem Statement: Minesweeper
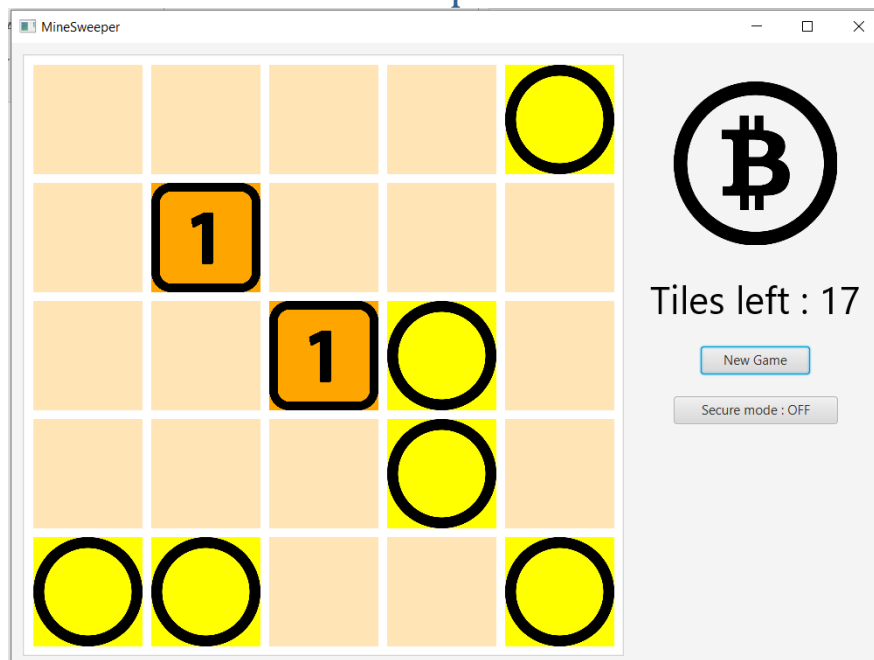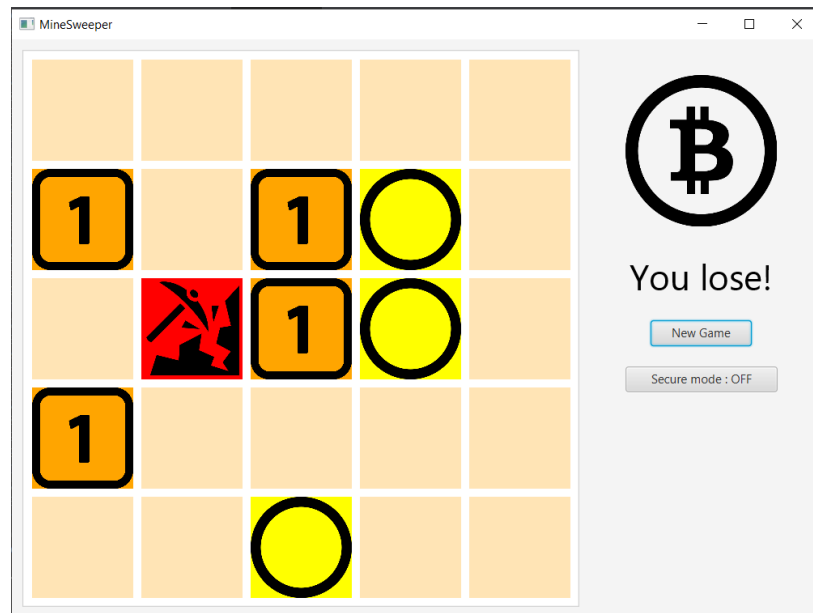


Sample screenshot of the application.

Minesweeper is a classic game, which player have to secure the tiles that mines exist and reveal all other tiles to win the game.
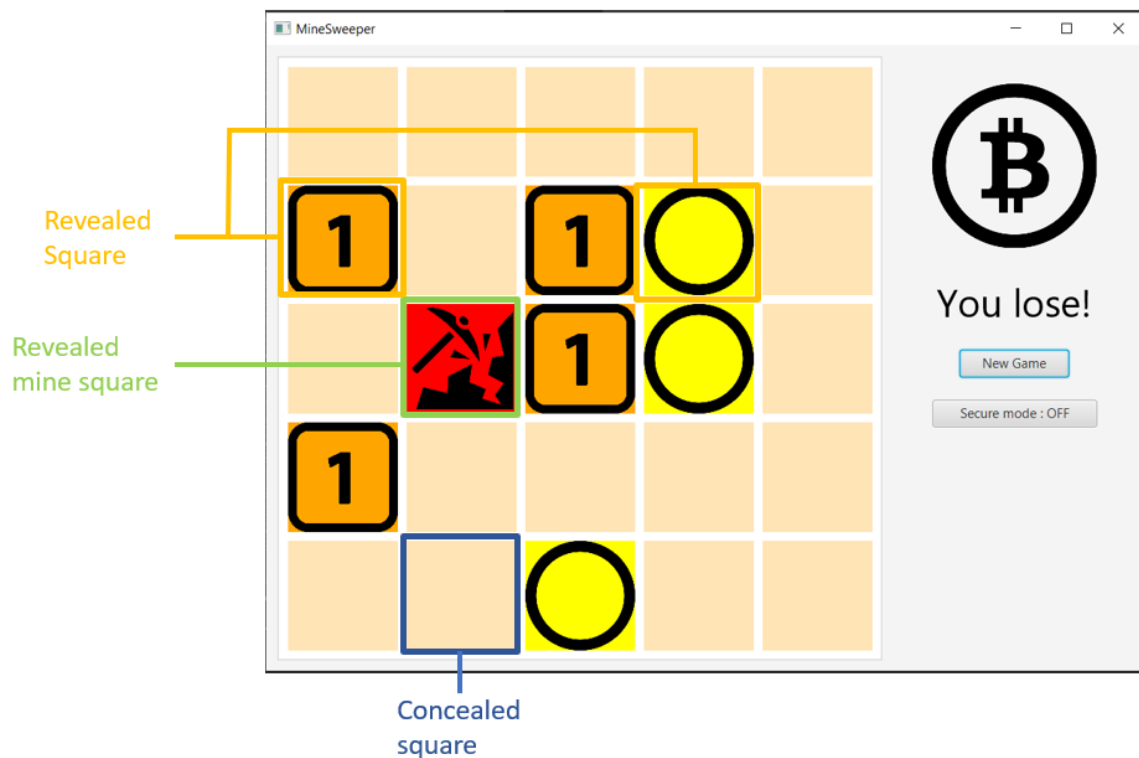
Types of squares in the game.

The player would click a square to reveal a tile. If it shows a yellow circle, it means there is not any mine diagonally adjacent to the revealed tile. In case that it shows a number, it means there is that number of mines diagonally adjacent to the revealed tile.

If player reveals a mine he will lose immediately and he will not be able to reveal any tile after losing.

Sample losing screenshot of the game.

The player has to reveal all tiles which do not contain a mine and secures mine square with secure mode to win the game.
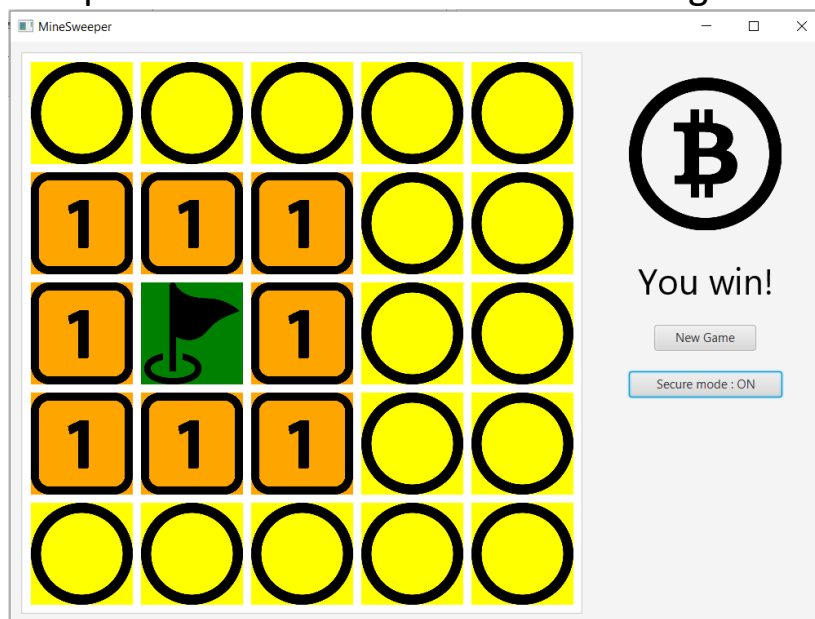


Sample winning screenshot of the game.

You have to finish the minesweeper game application that contains the game pane and the control pane which contains new game button, secure mode toggle button and gives the information about game states.

## 4. Implementation Detail
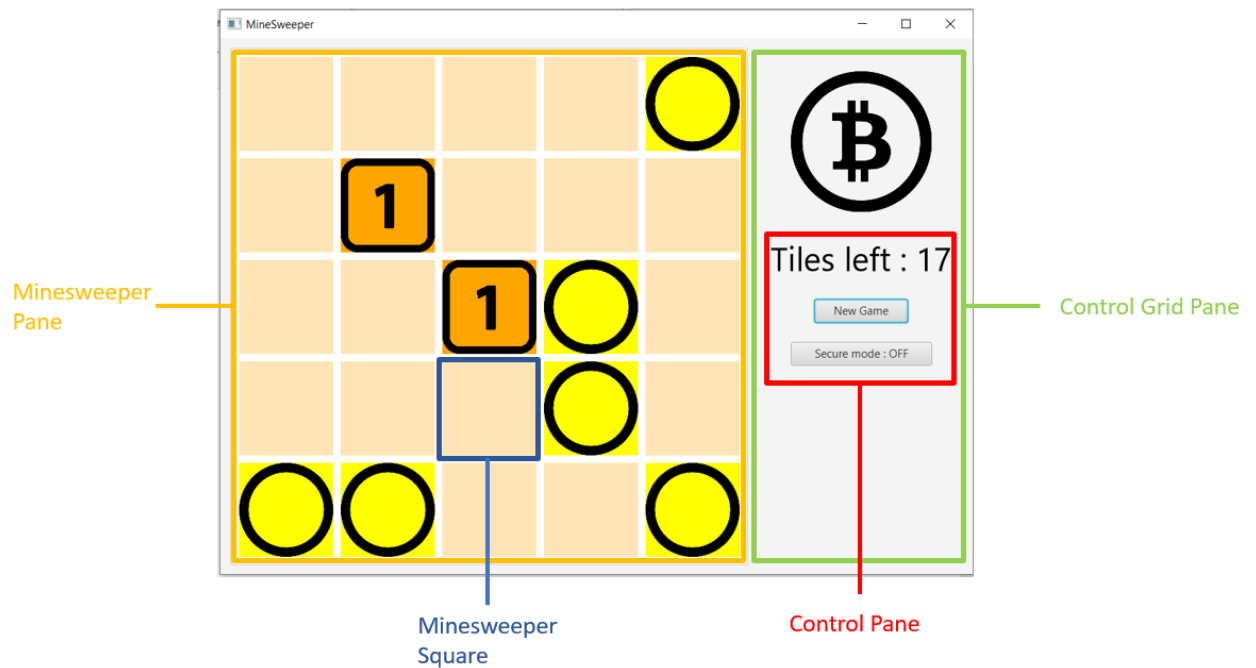
Detailed GUI of the Application.

The class package is summarized below.

**\* In the following class description, only details of IMPORTANT fields and methods are given. \***

## 4.1 Package gui

4.1.1. public class **MineSweeperSquare**: This class represents a pane that conceals circle, number or mine behind it.



*Field*

| Name | Description |
|---|---|
| - boolean isDrawn | Represent that the cell has been drawn or not. |

| - Color baseColor | The base color of the cell when it does not have anything drawn on it. |
|---|---|
| - int xPosition | Position of the cell in X-axis. As a default of position in JavaFX GridPane, X-axis is aligned from left to right. (You can see in Detailed GUI of the Application figure) |
| - int yPosition | Position of the cell in Y-axis. As a default of position in JavaFX GridPane, Y-axis is aligned from top to bottom. (You can see in Detailed GUI of the Application figure) |
| - final String oURL | URL of circle image. The value should be added in constructor as "o.png" |
| - final String oneURL | URL of number one image. The value should be added in constructor as "one.png" |
| - final String mineURL | URL of mine image. The value should be added in constructor as "mine.png" |
| - final String flagURL | URL of flag image. The value should be added in constructor as "flag.png" |

*Constructor*

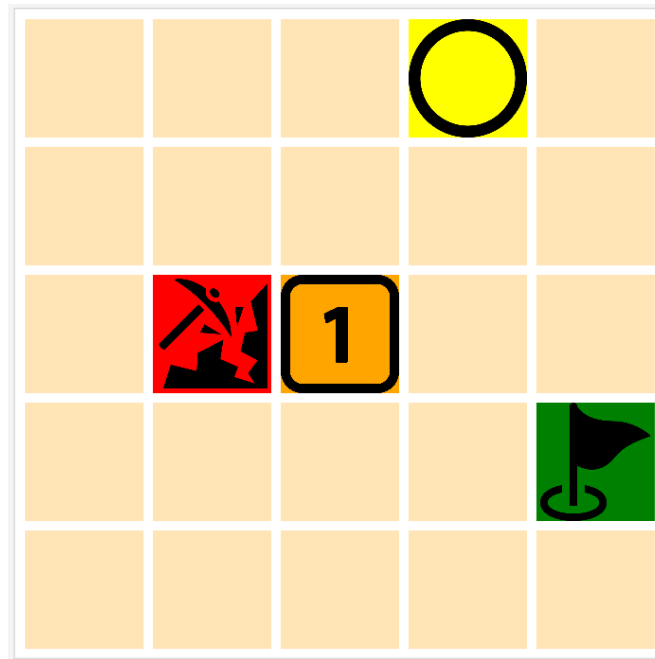| Name | Description |
|---|---|
| + MineSweeperSquare (int x, int y) | **/* FILL CODE */** Constructor method. Initializes with the following specifications: - assign oURL as "o.png", oneURL as "one.png", mineURL as "mine.png" and flagURL as "flag.png" - set xPostion as x and yPosition as y - set preferred width and height to 100. - set minimum width and height to 100. - set baseColor as MOCCASIN |

| | |
|---|---|
| | - call initializeCellColor() to initialize *cell color*<br>- add EventHandler on mouse clicked to the square which will call method onClickHandler when the square is clicked. |

*Method*

| Name | Description |
|---|---|
| - void onClickHandler() | **/* FILL CODE */**<br>This method is called when the cell is clicked. Does the following:<br>- check if the game has ended by using GameLogic.getInstance().isGameEnd()<br>- if the game has ended, do nothing.<br>- if the game has not ended, check if the game is in secure mode or not by using GameLogic.getInstance().isSecureMode() and<br>  if this square state is not revealed by using GameLogic.getInstance().getBoardState()[xPosition][yPosition]<br><ul><li>In this case that this square is not revealed and the game is not in secure mode draw a picture in this square using draw method by checking GameLogic.getInstance().getBoardMark() in this square position and resolve by this following case<ol><li>ONE: draw image from oneURL with orange background color</li><li>NOTHING: draw image from oURL with yellow background color</li><li>MINE: draw image from mineURL with red background color</li></ol></li></ul>Then using GameLogic.getInstance().updateState method to update this square state in game logic to revealed |

| | |
|---|---|
| | - if the game is in secure mode and this square has not been drawn<br>    ● draw image from flagURL with green background color<br>    ● Then using GameLogic.getInstance().updateState method to update this square state in game logic to secured<br><mark>**Caution: Even a square has been secured, if it is clicked while the secure mode is off, it will make the square revealed.**</mark> |
| - void draw(Image image, Color backgroundColor) | - set the Background with backgroundColor with image<br><mark>**/*You have to fill some code in this method*/**</mark><br>- then, set isDrawn as true |
| + void initializeCellColor( ) | <mark>**/* FILL CODE */**</mark><br>Set the **Background** to be **filled** with color as baseColor, and CornerRadii and Insets as empty. Set *isDrawn* to false.<br>This method is used for initializing and resetting the cell. |
| + getter/setter for each field. | <mark>**/* FILL CODE */**</mark> |

 

4.1.2. public class **MineSweeperPane**: This class represents a grid pane with

Minesweeper squares.

*Field*

| Name | Description |
|------|-------------|
| -<br>ArrayList<MineSweeperSquare> allCells | **/\* FILL CODE \*/**<br>List that contains MineSweeperSquare objects in the grid. |

*Constructor*

| Name | Description |
|------|-------------|
| +<br>MineSweeperPane() | **/\* FILL CODE \*/**<br>Constructor method. Initializes with the following specifications:<br>- initializing allCells<br>- set horizontal gap and vertical gap as 8<br>- set the inset padding of 8 and preferred width as 500<br>- set alignment as **CENTER**<br>- set border to **LIGHTGRAY** color, stroke style **SOLID**, corner radii **EMPTY**, with **DEFAULT** width.<br>- set background as **WHITE** color |

| | - initialize MineSweeperSquare and add them to allCells and to this pane in the form of 5*5 grid square |
|---|---|

*Method*

| Name | Description |
|---|---|
| + ArrayList<NumberSquare> getAllCells () | **/* FILL CODE */** Getter method for *allCells*. |

4.1.3. public class **ControlPane**: This class is the pane that contains game information text, new game button and secure mode button. Items in the pane is arranged vertically.
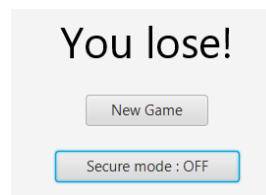


*Field*

| Name | Description |
|---|---|
| - Text gameText | The Text for displaying whose turn it is, or who wins. |
| - Button newGameButton | The button for beginning a new round. |
| - Button secureModeButton | The button for toggling secure mode of the game. |
| - MineSweeperPane mineSweeperPane | A MineSweeperPane that is updated by this ControlPane. |

*Constructor*

| Name | Description |
|---|---|

| + ControlPane (MineSweeperPane mineSweeperPane) | **/* FILL CODE */** Constructor method. Sets mineSweeperPane field to match the parameter. Then, initializes with the following specifications:<br>- set the alignment to **CENTER**.<br>- set preferred width to 300.<br>- set spacing to 20.<br>- call initializeGameText() to initialize *gameText.*<br>- call initializeNewGameButton() to initialize *newGameButton*.<br>- call initializeSecureModeButton() to initialize *secureModeButton*.<br>- add gameText, newGameButton and secureModeButton field to this pane's children in correct order. |
|---|---|

*Method*

| Name | Description |
|---|---|
| - void initializeGameText() | **/* FILL CODE */**<br>- Initializes *gameText* with text " Tiles left : " and the number from GameLogic.getInstance().getTileCount()<br>- set *gameText* font with size 35 |
| + void updateGameText(String text) | **/* FILL CODE */**<br>- set *gameText* with text *text* |
| - void initializeNewGameButton() | **/* FILL CODE */**<br>- initialize *newGameButton* with text "New Game".<br>- set the button preferred width to 100. |

| | - set onAction to handle with newGameButtonHandler() method. (See below) |
|---|---|
| - void initializeSecureModeButton() | **/* FILL CODE */**<br>- initialize *SecureModeButton* with text<br>"Secure mode : OFF".<br>- set the button preferred width to 150.<br>- set onAction to handle with secureModeButtonHandler() method. (See below) |
| - void newGameButtonHandler() | **/* FILL CODE */**<br>This method is the handler method for *newGameButton*.<br>Does the following:<br>- resetting game state using GameLogic.getInstance().newGame() method<br>- set secureModeButton text to "Secure mode : OFF"<br>- set *gameText* text to "Tiles left : " with GameLogic.getInstance().getTileCount()<br>- resetting all cell in mineSweeperPane by using initializeCellColor() |
| - void secureModeButtonHandler() | **/* FILL CODE */**<br>This method is the handler method for *secureModeButton*.<br>Does the following:<br>- toggle secureModeButton state using |

| | GameLogic.getInstance().toggleSecureMode() method<br>- Check if the is in secure mode using GameLogic.getInstance().isSecureMode()<br>- If the is not in secure mode, set secureModeButton text to "Secure mode : OFF"<br>- Otherwise, set secureModeButton text to "Secure mode : ON" |
|---|---|

4.1.3. public class **ControlGridPane**: This class is the vertical gird that contains bitcoin icon and control pane.
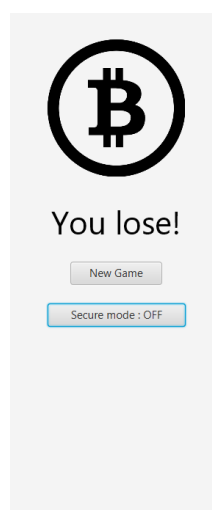


*Field*

| Name | Description |
|---|---|

| - final String miningImageURL | . URL of game icon image. The value should be added in constructor as "bitcoin.png" |
| - controlPane controlPane | A Control Pane that should be added to this grid. |

*Constructor*

| Name | Description |
|---|---|
| + ControlGridPane (ControlPane controlPane) | /* FILL CODE */ Constructor method. Sets controlPane field to match the parameter. - Assign miningImageURL as "bitcoin.png" - Create an imageView from miningImageURL - Set the image fit width and height to 150 - Create border pane with preferred width 150 and preferred height 200 - set the image to the center of the border pane - add the border pane and controlPane to this grid in vertically order. |

## 4.2 Package logic

4.2.1. public enum **SquareMark: This class is already provided.** This enum contains every type of marks you can find when the square is revealed.

| Name | Description |
|---|---|
| - ONE | represents number one |
| - NOTHING | represents blank tile |
| - MINE | represents a mine. |

4.2.2. public enum **SquareState: This class is already provided.** This enum contains every type of states squares can be.

| Name | Description |
|------|-------------|
| - CONCEALED | represents a concealed square |
| - REVEALED | represents a revealed square |
| - SECURED | represents a secured square |

4.2.3. public class **GameLogic**: **This class is already provided.** Only useful fields and methods are shown here.

*Field*

| Name | Description |
|------|-------------|
| - <u>GameLogic instance</u> | Instance that represents GameLogic class. This implementation confirms that we have only one GameLogic. |
| - boolean isGameEnd | A field that shows if the game has ended or not. |
| - boolean isGameWin | A field that shows if the player wins the game or not. |
| - int mineX | position where the mine exists in x-axis. Assigned as 1 |
| - int mineY | position where the mine exists in y-axis. Assigned as 2 |
| - int tileCount | Number counted the tile which have not been revealed |
| - ControlPane controlPane | ControlPane that will update when game state is changed. |
| - boolean secureMode | A field that shows if the game is in secure mode or not. |

| - SquareMark[][] boardMark | Array contains all marks on the board. |
|---|---|
| - SquareMark[][] boardMark | Array contains all marks on the board. |

*Method*

| Name | Description |
|---|---|
| + void updateState(int x, int y, SquareState state) | This method should be called when MineSweeperSquare has been drawn.<br><br>GameLogic will update the state of the square and check whether the game ends or not. Then, update the ControlPane information text.<br><br>If the game is not in secure mode, decrease tileCount and update the ControlPane information text. |
| - void checkGameEnd() | This method is used to check if the game has ended or not<br><br>If the mine is revealed, the game end and player lose the game<br><br>If the mine is secured and all other tiles are revealed, the game end and player win the game. |
| +void newGame() | Reset the game state to its initial state. |
| + static GameLogic getInstance() | Getter of instance.<br>Use this method whenever you want to use the method in GameLogic. |

## 4.2 Package main

4.2.1. public class **Main**: This class contains main method. It is an entry point of the application. <mark>This class is partly provided.</mark>

*Method*

| Name | Description |
|------|-------------|
| + void start(Stage primaryStage) throws Exception | <mark>/* FILL CODE */</mark><br>Start setting of the application<br>- Create horizontal box as a root of all componenets<br>- Set inset padding as 10<br>- Set spacing as 10<br>- Set preferred height as 400<br>- Set preferred width as 800<br>- Initialize Minesweeper Pane, Control pane with the Minesweeper pane, and Control grid pane with the Control pane<br>- Set the control pane to game logic using GameLogic.getInstance().setControlPane<br>- Add the Minesweeper Pane and the Control grid pane to the root<br>- Create the scene with the root<br>- Set primaryStage scene as the created scene<br>- Set primaryStage title as "MineSweeper"<br>- Show primaryStage |

## 5. Scoring Criteria (10 points)

### 5.1 Initializing program
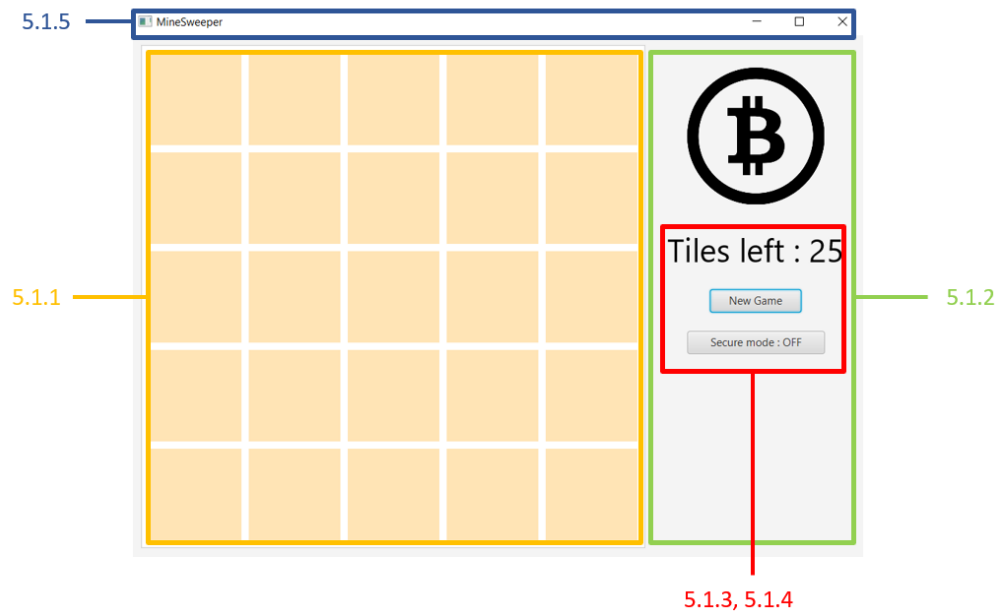


5.1.1 All cells in the MineSweeperPane are displayed properly and in right color. (1 point)
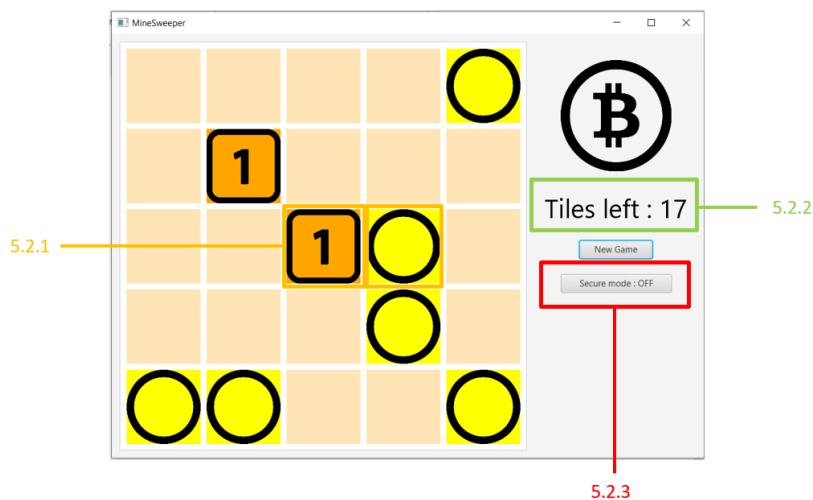
5.1.2 Image in Control grid pane is displayed in proper size and in correct order to the Control pane. (1 point)

5.1.3 Text in Control pane is displayed properly. (0.5 point)

5.1.4 Button in Control pane is displayed properly. (0.5 point)

5.1.5 Title of the program is displayed correctly. (0.5 point)

## 5.2 Game play when secure mode is off



5.2.1 While secure mode is off, clicking on the square which is not the mine square (even it is secured or not) should draw a number or a circle correctly following to the game logic board mark with the right background color. (2 point)

5.2.2 The information on Control pane updates correctly. It also should not change when a cell is clicked twice. (1 point)

5.2.3 Secure mode button shows secure mode state as off. (0.5 point)
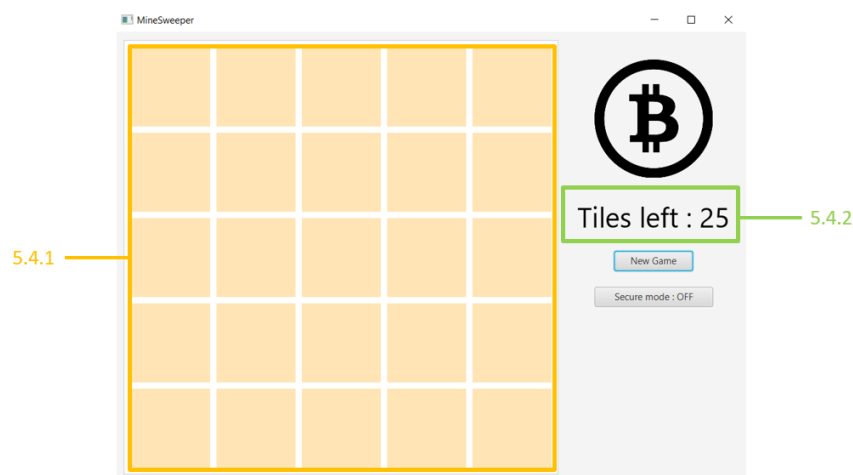
## 5.3 Losing the game
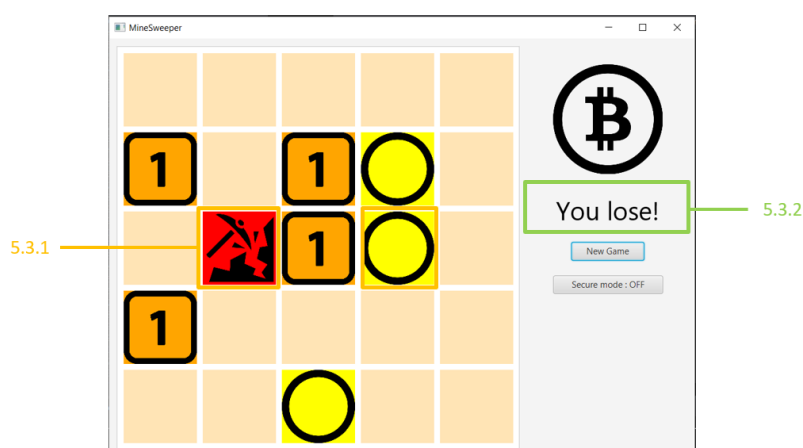


5.3.1 While secure mode is off, when the mine square (even it is secured or not) is clicked, the square displays the right image with the right background color. (1 point)

5.3.2 When you lose the game, the information on Control pane displays correctly. (1 point)

5.3.3 When game reaches the end, all cells cannot be clicked. (1 point)
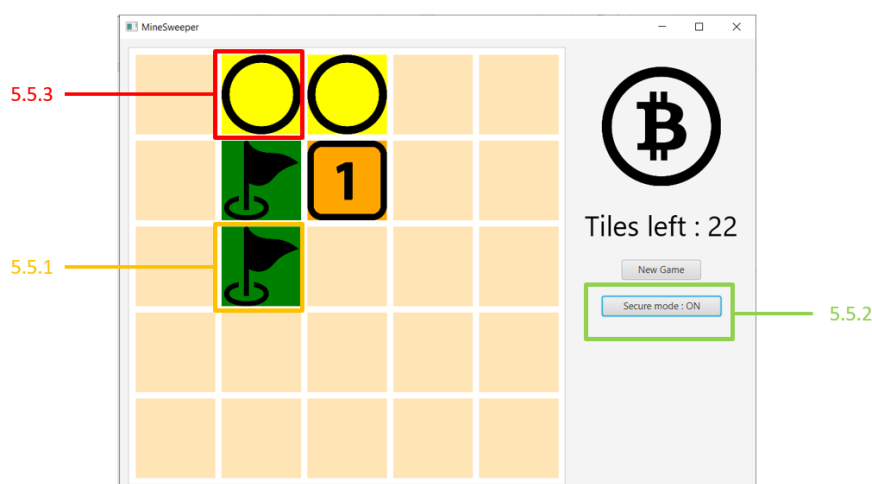
## 5.4 New game



5.4.1 When new game button is clicked, all cells in MineSweeperPane reset to its initial state.
(1 point)

5.4.2 When new game button is clicked, the text in control pane resets to its initial state.
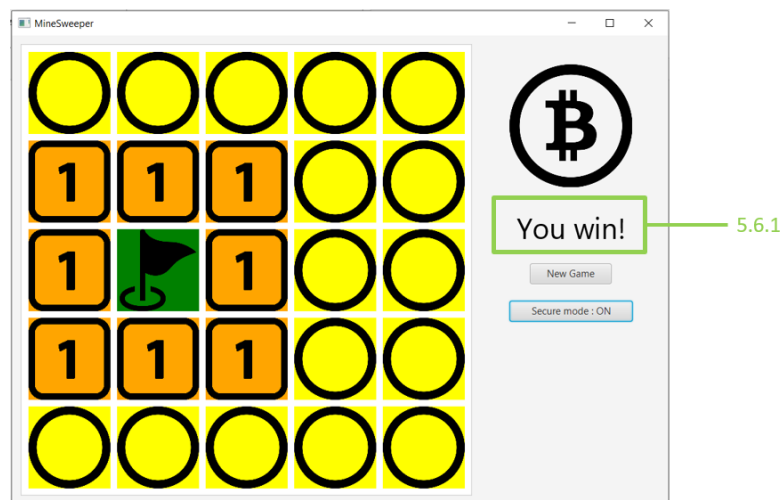(1 point)

## 5.5 Secure mode

5.5.1 While secure mode is on, when any concealed square is clicked, the square displays the flag image with the right background color. (1 point)

5.5.2 Secure mode button shows secure mode state as on. (0.5 point)

5.5.3 While secure mode is on, when any drawn square is clicked, it should not change anything. (1 point)

## 5.5 Winning the game



5.5.1 When you win the game, the information on Control pane displays correctly. (1 point)