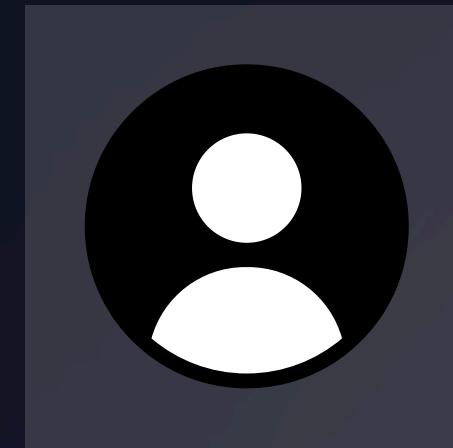




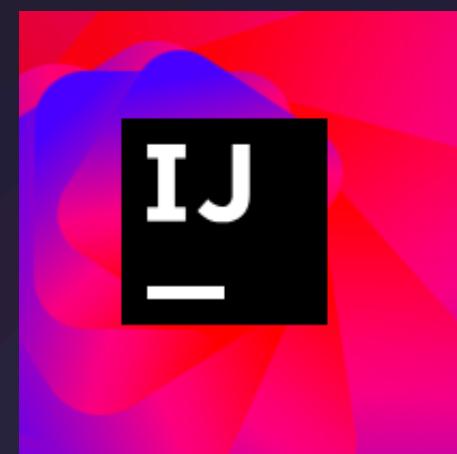
DSA Project

MINESWEEPER

CONTRIBUTOR

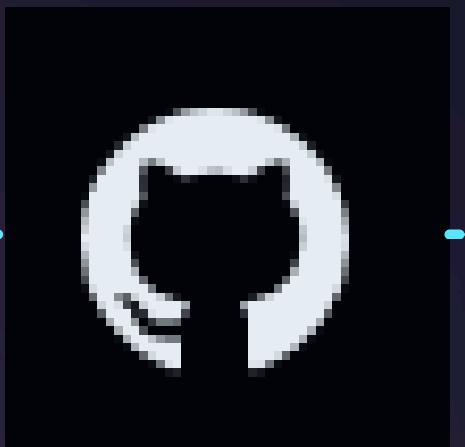


NGUYỄN ĐỨC NGUYỄN PHÚC
ITDSIU21108



IntelliJ IDEA

Tools



GITHUB



ECLIPSE IDE

TABLE OF CONTENTS

○ Introduction	01
○ Game Rules	02
○ UML & Git	03
○ Data Structure & Algorithm	04
○ Features & Design Pattern	05
○ Demo	06
○ Q&A	07



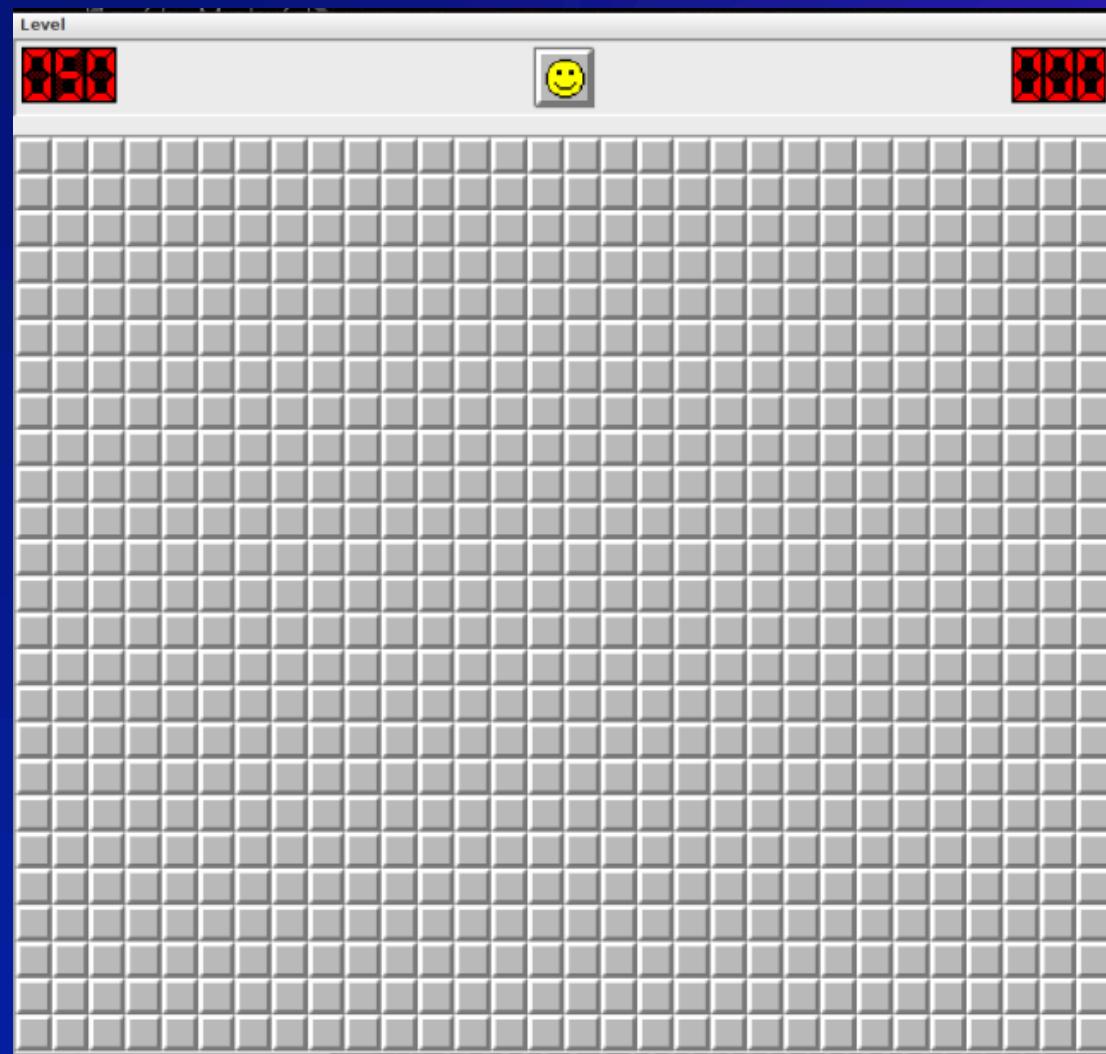
INTRODUCTION

CODED BY JAVA

- A game of logic.
- Make use of your intelligence to locate the mines.



GAME RULES

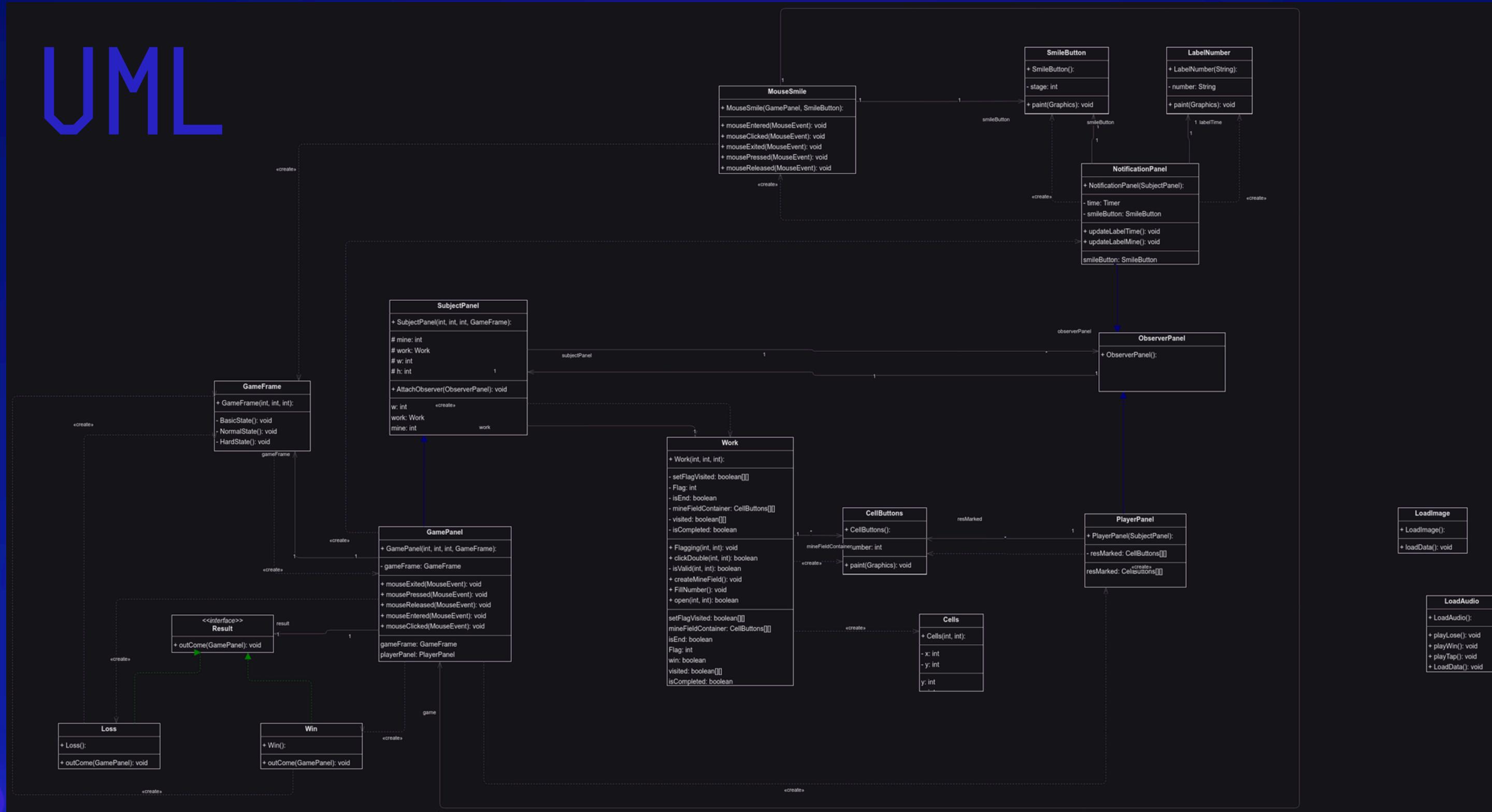


- Right Click to reveal a cell
- Left Click to place a flagging or unflagging
- Click on the bomb -> LOSE !
- 3 Levels: Beginner, Intermediate and Hard

GIT CONTRIBUTION



UML



DATA STRUCTURE & ALGORITHM

Applying Advanced Graph (BFS)

- The algorithm in class work
- click [work.java](#) to see

Time complexity : $O(N^2)$

DATA STRUCTURE & ALGORITHM

Applying Recursion

```
// Apply BFS Algorithm and matrix to open when click to cell "0" using recursion
public boolean open(int i, int j) { 3 usages ± phucnguyen140502*
    if (!isCompleted && !isEnd) {
        if (visited[i][j]) {
            if (mineField[i][j] == 0) { // open around until the cells have mines around
                visited[i][j] = false;
                mineFieldContainer[i][j].setNumber(0);
                mineFieldContainer[i][j].repaint();
                if (isWin()) {
                    isEnd = true;
                    return false;
                }
                for (int k = 0; k < 8; k++) {
                    int x = i + drow[k];
                    int y = j + dcol[k];

                    if (isValid(x, y) && visited[x][y]) {
                        open(x, y);
                    }
                }
            }
        }
    }
}
```

Time Complexity : $O(N^2)$

DATA STRUCTURE & ALGORITHM

Applying HashTable

```
public class LoadAudio { 11 usages ± phucnguyen140502  
    public static HashTable<String, Clip> listAudio; 7 usages
```

[Explain](#) | [Test](#) | [Document](#) | [Fix](#) | [Ask](#)

```
public static void LoadData() { 4 usages ± phucnguyen140502
```

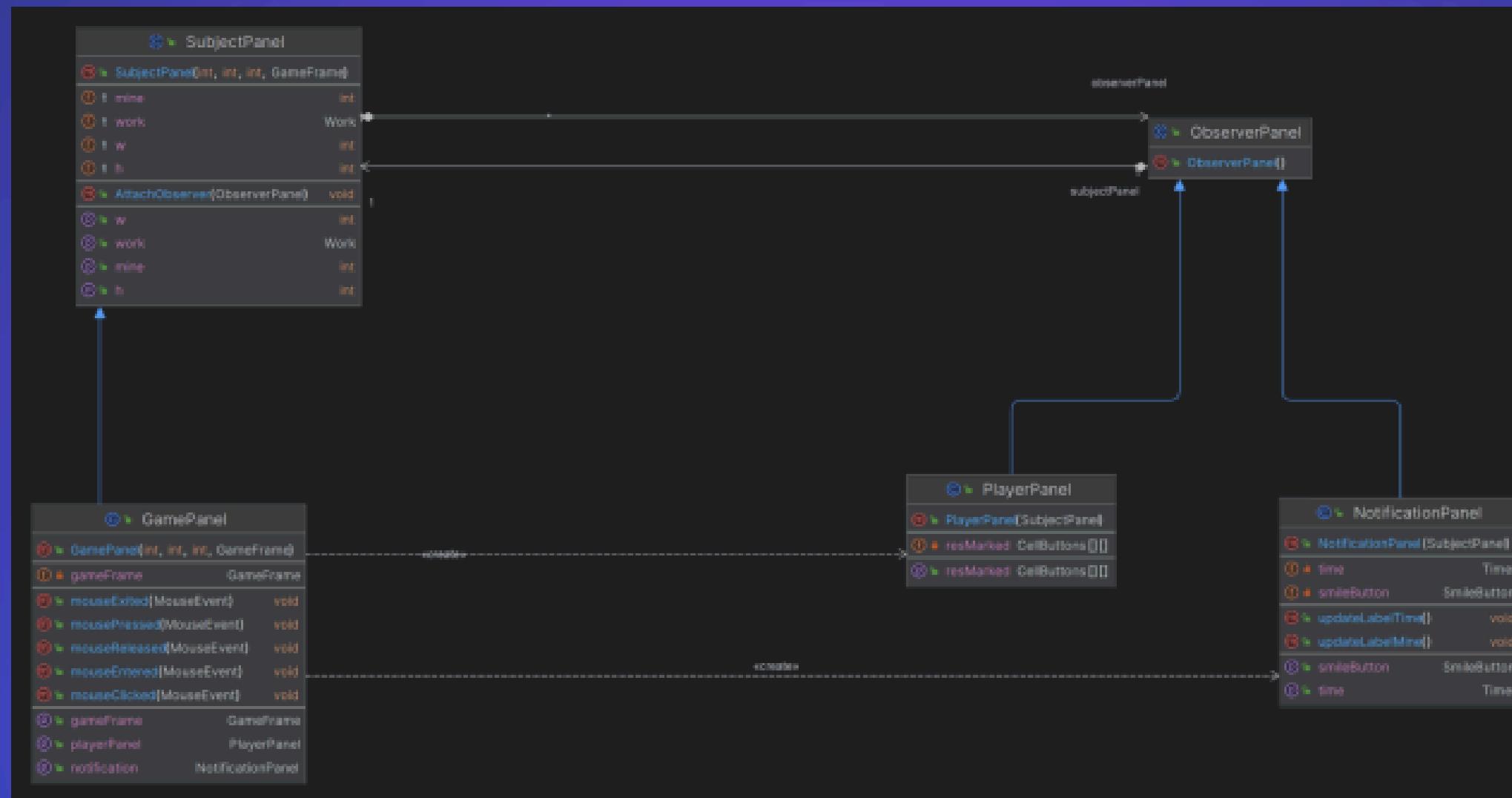
[Explain](#) | [Test](#) | [Document](#) | [FIX](#) | [ASK](#)

```
public static void loadData() { 3 usages ± phucnguyen140502  
    listImage = new HashTable<>( capacity: 35);
```

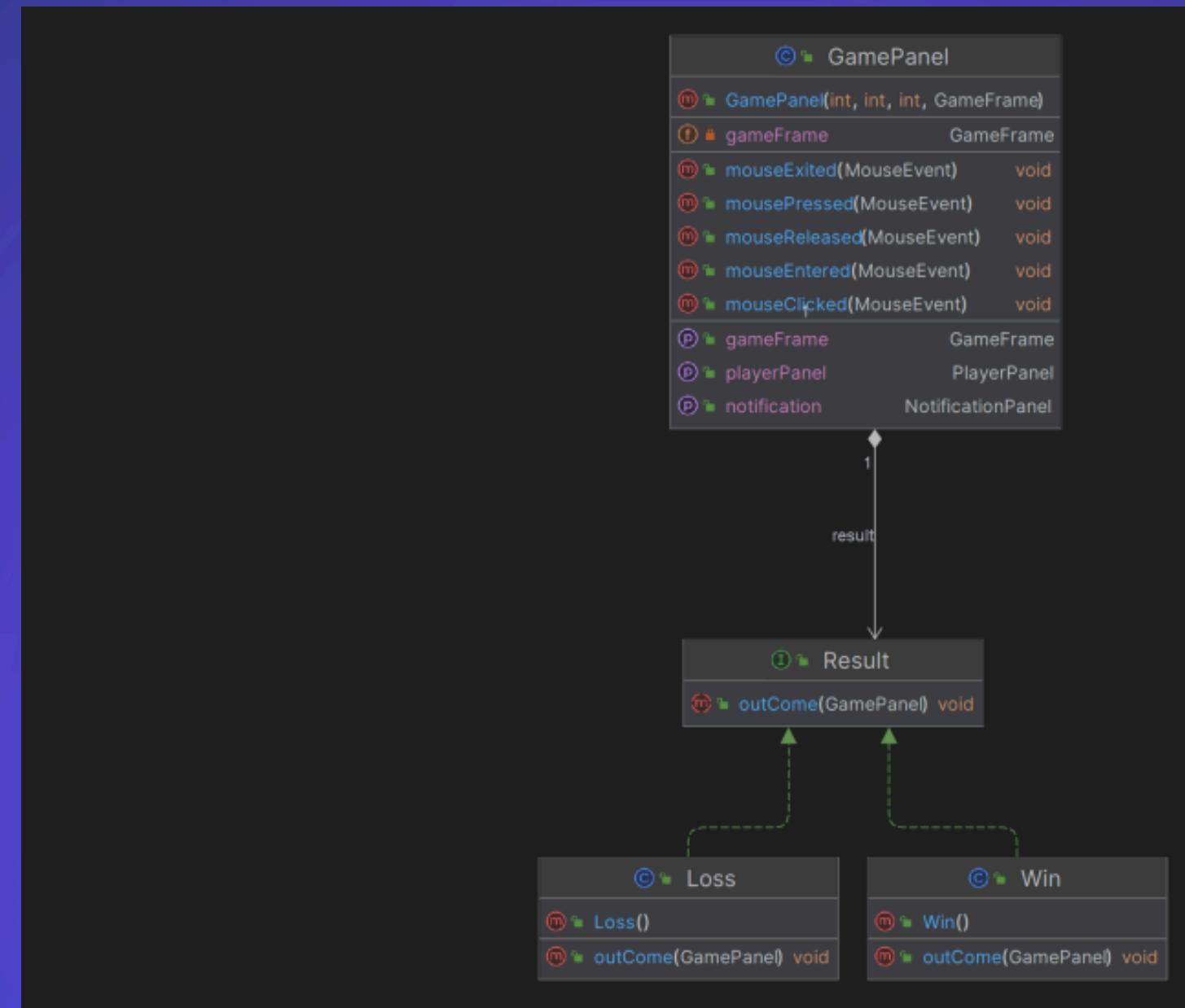
Create the board

```
| }  
| }  
  
Explain | Test | Document | Fix | Ask  
public void createMineField(){ 1 usage phucnguyen140502  
    int count;  
    do{  
        int locationX = random.nextInt(w);  
        int locationY = random.nextInt(h);  
        if (mineField[locationX][locationY] == -1) {  
            break;  
        }  
        mineField[locationX][locationY] = -1;  
        visited[locationX][locationY] = true;  
  
        count = 0;  
        for (int i = 0; i < w; i++) {  
            for (int j = 0; j < h; j++) {  
                if (mineField[i][j] == -1) {  
                    count++;  
                }  
            }  
        }  
    }while (count != mine);  
}
```

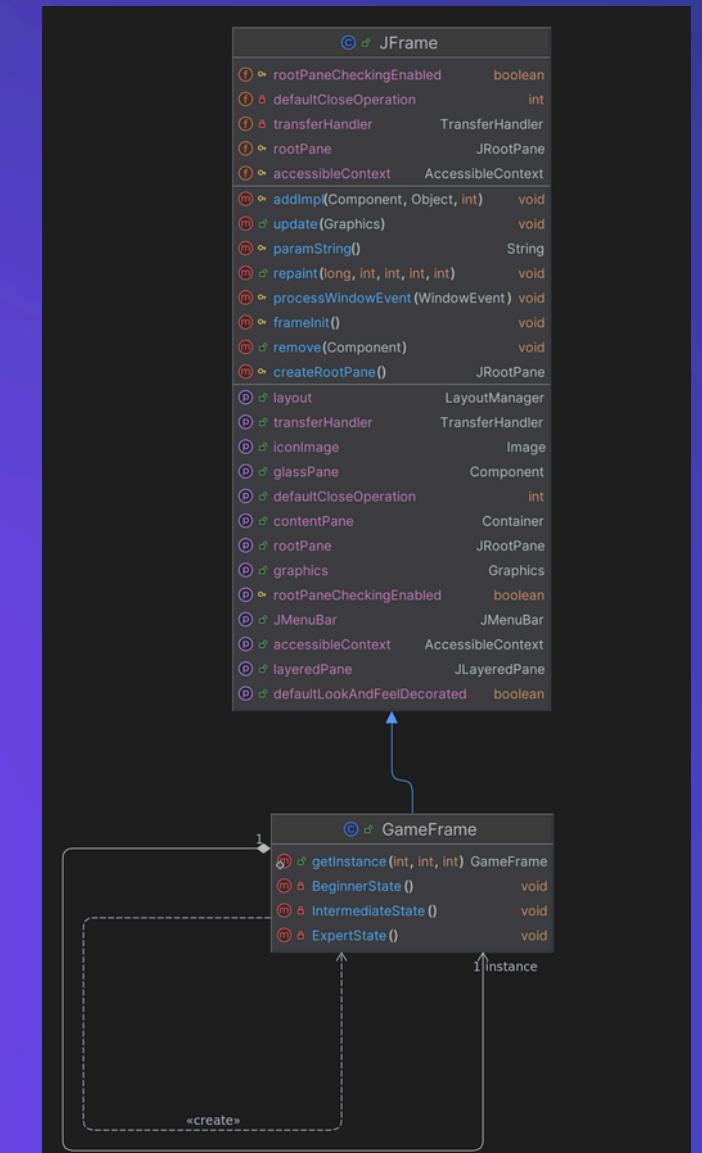
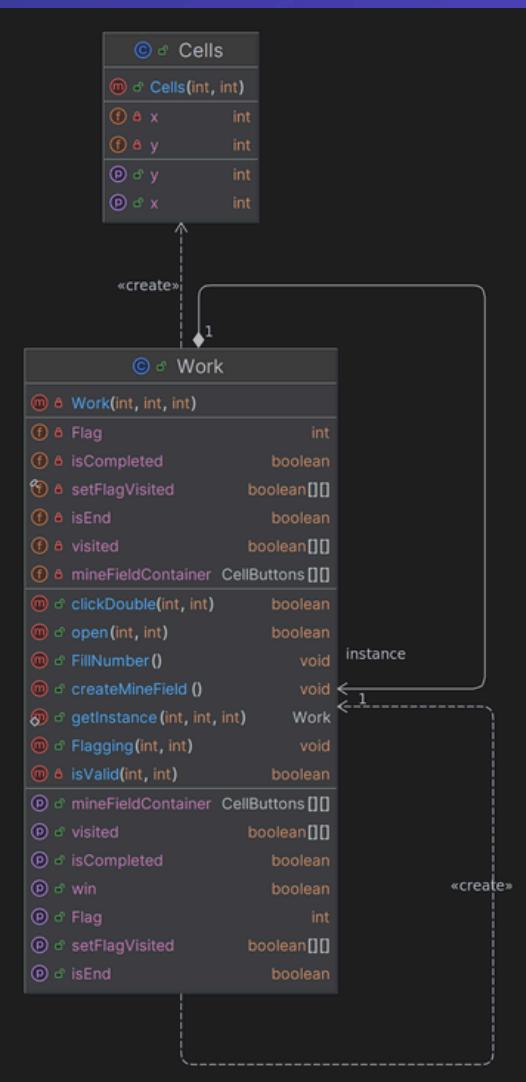
Observer Design Pattern



Strategy Design Pattern



Singleton Pattern



GAME DEMO



Thank you
for
listening



RESOURCE PAGE

