ÖZYEĞİN UNIVERSITY

CS 102
Object Oriented Programming

Model-View-Controller

Reyyan Yeniterzi
reyyan.yeniterzi@ozyegin.edu.tr

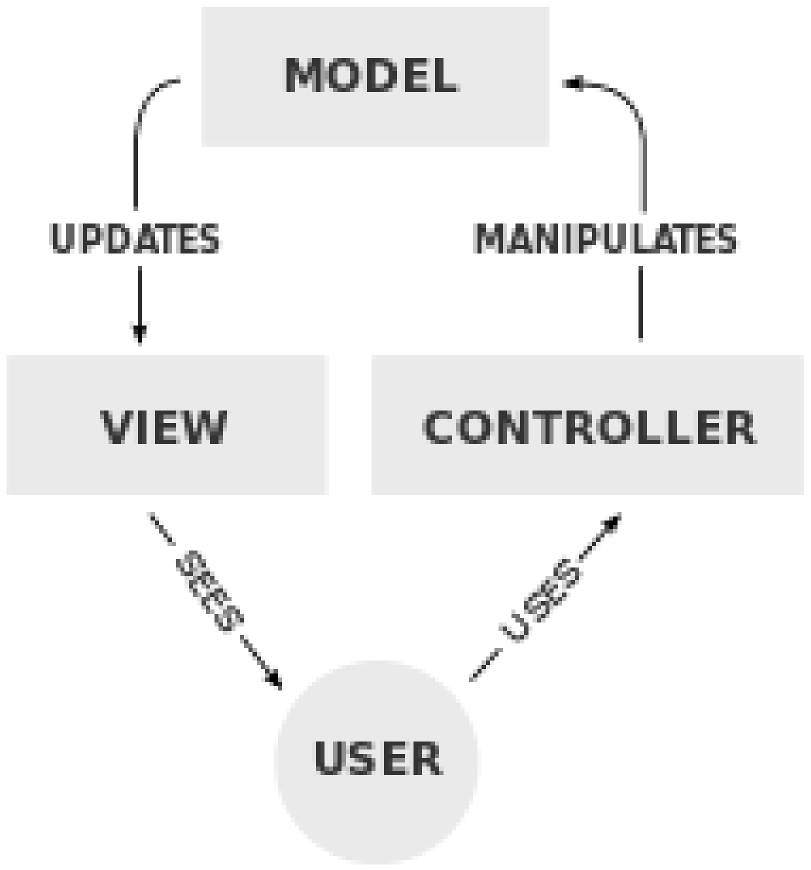
Model View Controller (MVC)

- MVC programming pattern
  - The idea is to keep the underlying logic seperate from the GUI.

  - **Model** defines the logic (the algorithm)
  - **View** defines the visualization (the GUI representation)
  - **Controller** defines how the view should be updated based on user interaction

Model View Controller (MVC)

- **Model** defines the logic (the algorithm)
- **View** defines the visualization (the GUI representation)
- **Controller** defines how the view should be updated based on user interaction



Source:https://en.wikipedia.org/wiki/Model-view-controller

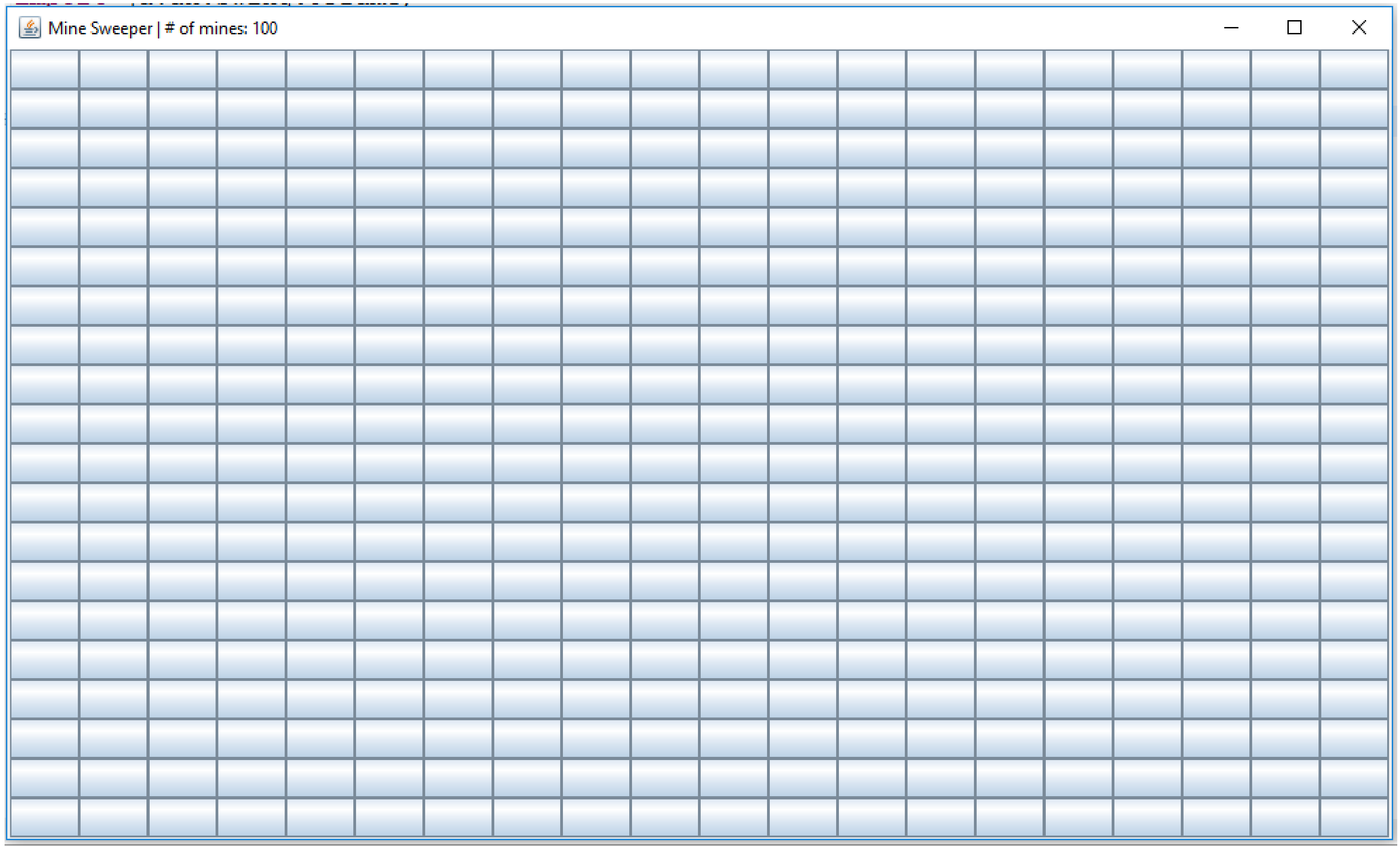
Model View Controller (MVC)

- MVC pattern is useful for simplifying complex applications.
  - One can focus on one aspect at a time

- The **model** does not need to know anything about the visualization. It is independent of the **view.**
  - **Controller** acts in between the **model** and **view.**
    - It seperates the **model** from the **view.**

A MVC Pattern Example

- MineSweeper example



MineSweeper

- Create a 20x20 grid
- Insert 100 mines

```
public class MineSweeper {
    private static final int NUM_MINES = 100;
    private static final int SIZE = 20;

    public static void main(String[] args) {
        JFrame frame = new JFrame("Mine Sweeper | # of mines: " + NUM_MINES);
        frame.add(new MineSweeperGUI(SIZE, SIZE, NUM_MINES));
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(1000, 600);
        frame.setVisible(true);
    }
}
```

MineSweeper

- Implement 3 classes
  - MineGrid is the Model.
    - The logic will be implemented in MineGrid class.
  - MineSweeperGUI is the View.
    - The visualization will be implemented in MineSweeperGUI class.
  - ButtonHandler is the Controller.
    - ButtonHandler class defines what action to take when a button is clicked and defines how the View should be updated.

MineSweeperGUI

- Start with the view: MineSweperGUI class

```
public class MineSweeper {
    private static final int NUM_MINES = 100;
    private static final int SIZE = 20;

    public static void main(String[] args) {
        JFrame frame = new JFrame("Mine Sweeper | # of mines: " + NUM_MINES);
        frame.add(new MineSweeperGUI(SIZE, SIZE, NUM_MINES));
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(1000, 600);
        frame.setVisible(true);
    }
}
```

- MineSweperGUI object is created with three parameters
  - # rows, # columns, # mines

MineSweeperGUI

```
class MineSweeperGUI extends JPanel {

    public MineSweeperGUI(int numRows, int numCols, int numMines) {

        /// ???
    }
}
```

- What should we do inside the constructor?

MineSweeperGUI

```
class MineSweeperGUI extends JPanel {

    public MineSweeperGUI(int numRows, int numCols, int numMines) {

        /// ???
    }
}
```

- What should we do inside the constructor?
- What should be the layout?

MineSweeperGUI

```
class MineSweeperGUI extends JPanel {

    public MineSweeperGUI(int numRows, int numCols, int numMines) {

        setLayout(new GridLayout(numRows, numCols));
    }
}
```

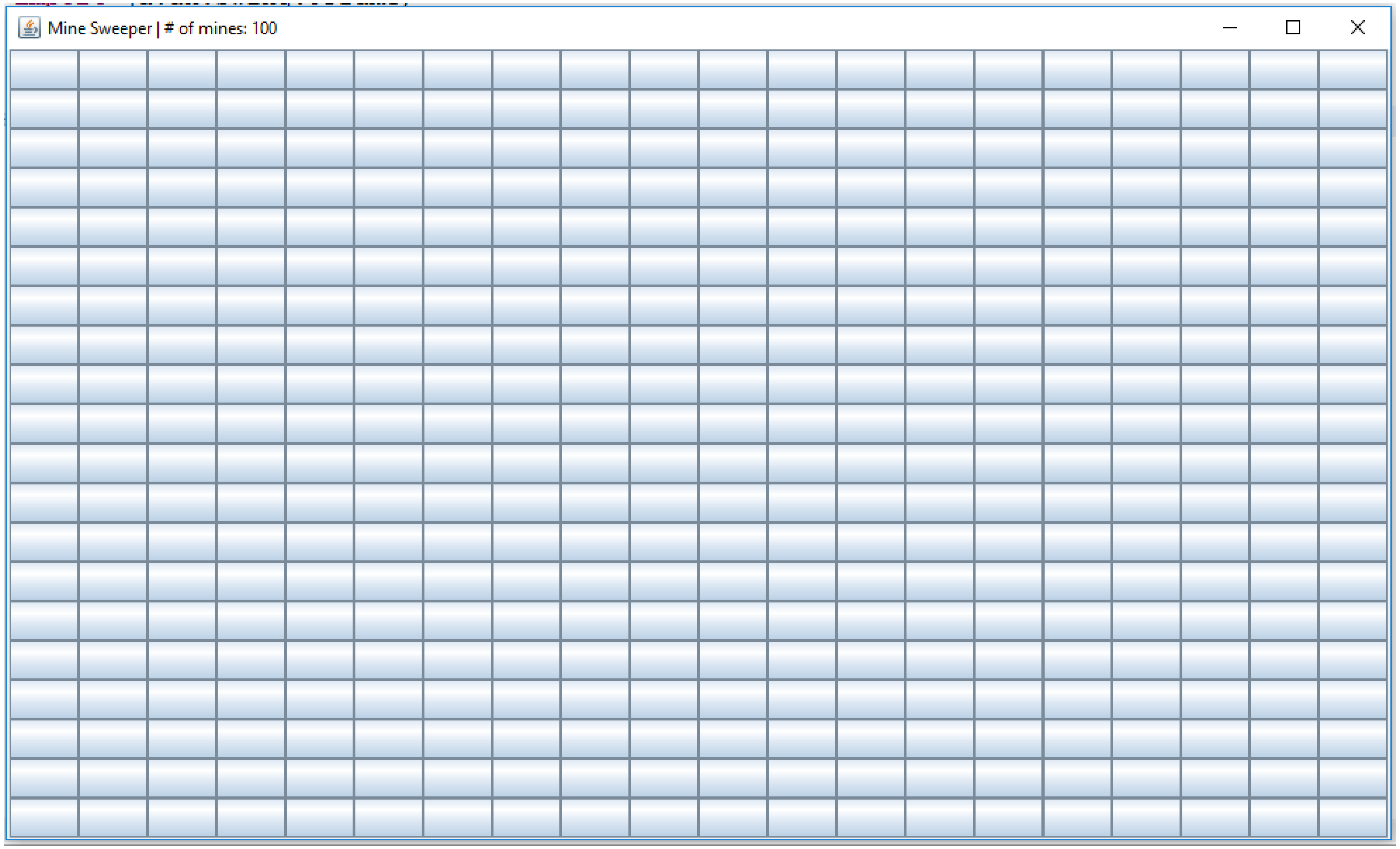
- We have created a grid. What now?
- How are we going to fill this?

MineSweeperGUI

```
class MineSweeperGUI extends JPanel {

    public MineSweeperGUI(int numRows, int numCols, int numMines) {

        setLayout(new GridLayout(numRows, numCols));
        for(int i = 0; i < numRows; i++) {
            for(int j = 0; j < numCols; j++) {
                JButton button = new JButton();
                add(button);
            }
        }
    }
}
```

- At this point, when we run this, we will see the minegrid without any functionality.

MineSweeperGUI



MineSweeper

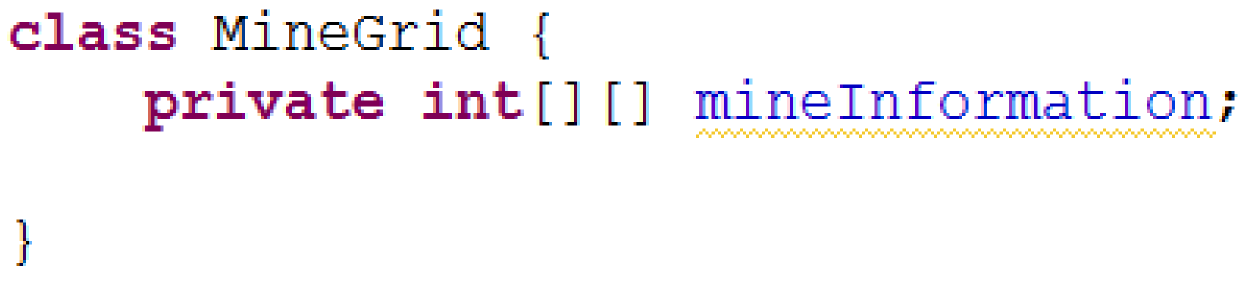
- Implement 3 classes
  - MineGrid is the Model.
    - The logic will be implemented in MineGrid class.
  - MineSweeperGUI is the View.
    - The visualization will be implemented in MineSweeperGUI class.
  - ButtonHandler is the Controller.
    - ButtonHandler class defines what action to take when a button is clicked and defines how the View should be updated.

MineGrid (Model)

- Lets implement the model.
- Any idea?
- Think about how should we keep the data…

MineGrid (Model)

- Create a two dimensional array.



```
class MineGrid {
    private int[][] mineInformation;

}
```

- What should be the # rows and # columns?

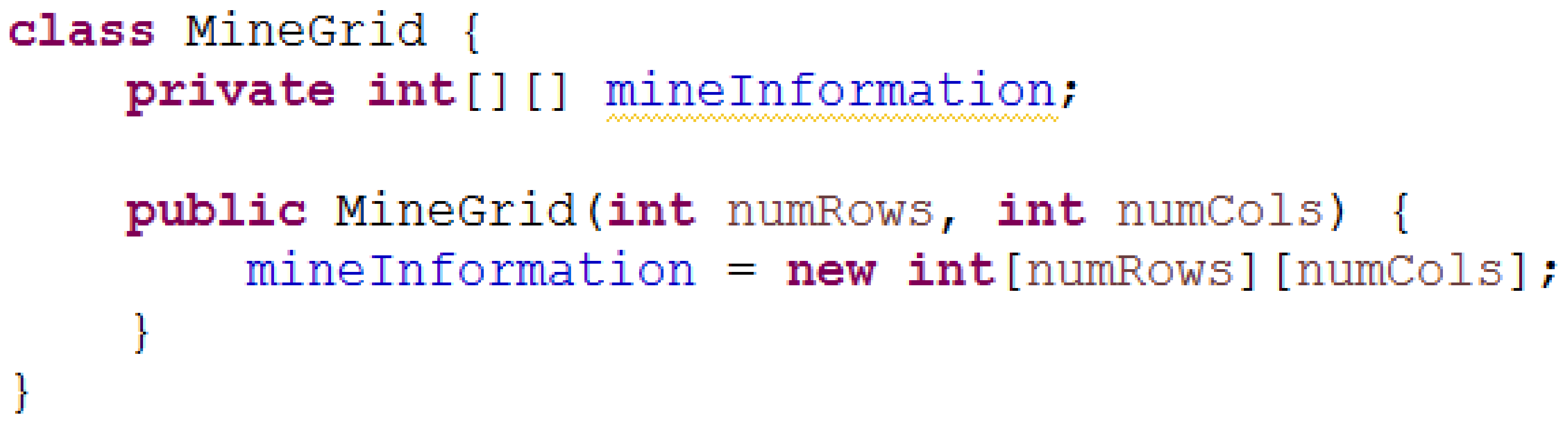
MineGrid (Model)

- Create a two dimensional array.

```
class MineGrid {
    private int[][] mineInformation;

}
```

- What should be the # rows and # columns?
- Get the # rows and # columns values at the constructor.

MineGrid (Model)



```
class MineGrid {
    private int[][] mineInformation;

    public MineGrid(int numRows, int numCols) {
        mineInformation = new int[numRows][numCols];
    }
}
```

MineGrid (Model)

```
class MineGrid {
    private int[][] mineInformation;

    public MineGrid(int numRows, int numCols) {
        mineInformation = new int[numRows][numCols];
    }
}
```

- At this point there is not any mine
- Set all cells to 0 initially.

MineGrid (Model)

```
class MineGrid {
    private int[][] mineInformation;

    public MineGrid(int numRows, int numCols) {
        mineInformation = new int[numRows][numCols];
        initializeCells();
    }

    private void initializeCells() {
        for(int i = 0; i < mineInformation.length; i++) {
            for(int j = 0; j < mineInformation[0].length; j++) {
                mineInformation[i][j] = 0;
            }
        }
    }
}
```

MineGrid (Model)

```
class MineGrid {
    private int[][] mineInformation;

    public MineGrid(int numRows, int numCols) {
        mineInformation = new int[numRows][numCols];
        initializeCells();
    }

    private void initializeCells() {
        for(int i = 0; i < mineInformation.length; i++) {
            for(int j = 0; j < mineInformation[0].length; j++) {
                mineInformation[i][j] = 0;
            }
        }
    }
}
```

- Now what?

MineGrid (Model)

```
class MineGrid {
    private int[][] mineInformation;

    public MineGrid(int numRows, int numCols) {
        mineInformation = new int[numRows][numCols];
        initializeCells();
    }

    private void initializeCells() {
        for(int i = 0; i < mineInformation.length; i++) {
            for(int j = 0; j < mineInformation[0].length; j++) {
                mineInformation[i][j] = 0;
            }
        }
    }
}
```

- Now what? Place the mines?

MineGrid (Model)

- Lets place the mines.
- Do we know how many mines we will place?

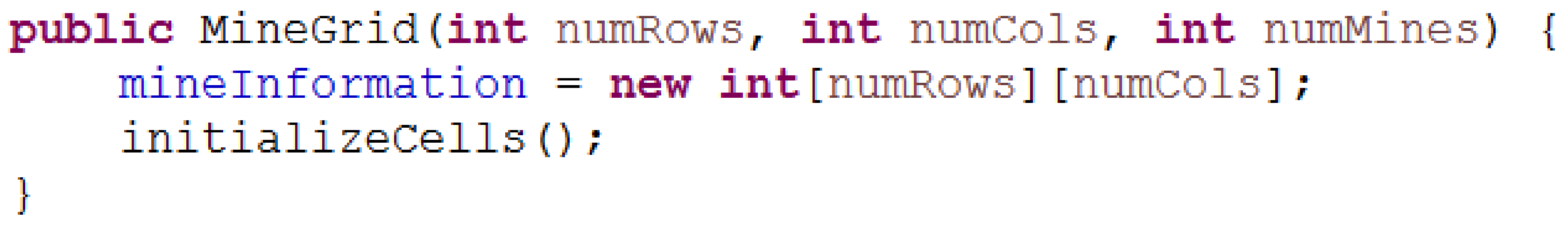
```
class MineGrid {
    private int[][] mineInformation;

    public MineGrid(int numRows, int numCols) {
        mineInformation = new int[numRows][numCols];
        initializeCells();
    }

    private void initializeCells() {
        for(int i = 0; i < mineInformation.length; i++) {
            for(int j = 0; j < mineInformation[0].length; j++) {
                mineInformation[i][j] = 0;
            }
        }
    }
}
```

MineGrid (Model)

- Right now, we don’t know.
- We need to get that number in the constructor.

MineGrid (Model)

- Right now, we don't know.
- We need to get that number in the constructor.



```
public MineGrid(int numRows, int numCols, int numMines) {
    mineInformation = new int[numRows][numCols];
    initializeCells();
}
```

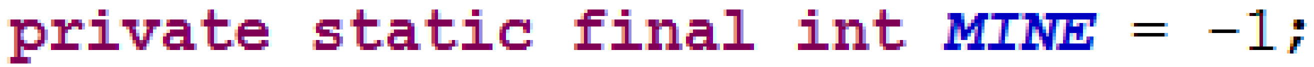
- We will randomly place `numMines` mines to the two dimensional array.
- How should we represent the mines?

MineGrid (Model)

- How should we represent the mines?
- We have an integer two dimensional array.

MineGrid (Model)

- How should we represent the mines?
- We have an integer two dimensional array.



```java
private static final int MINE = -1;
```

MineGrid (Model)

```
private static final int MINE = -1;
private int[][] mineInformation;

public MineGrid(int numRows, int numCols, int numMines) {
    mineInformation = new int[numRows][numCols];
    initializeCells();
    placeMines(numMines);
}

private void placeMines(int numMines) {
    Random random = new Random();
    for(int i = 0; i < numMines; i++) {
        int r = random.nextInt(mineInformation.length);
        int c = random.nextInt(mineInformation[0].length);
        mineInformation[r][c] = MINE;
    }
}
```

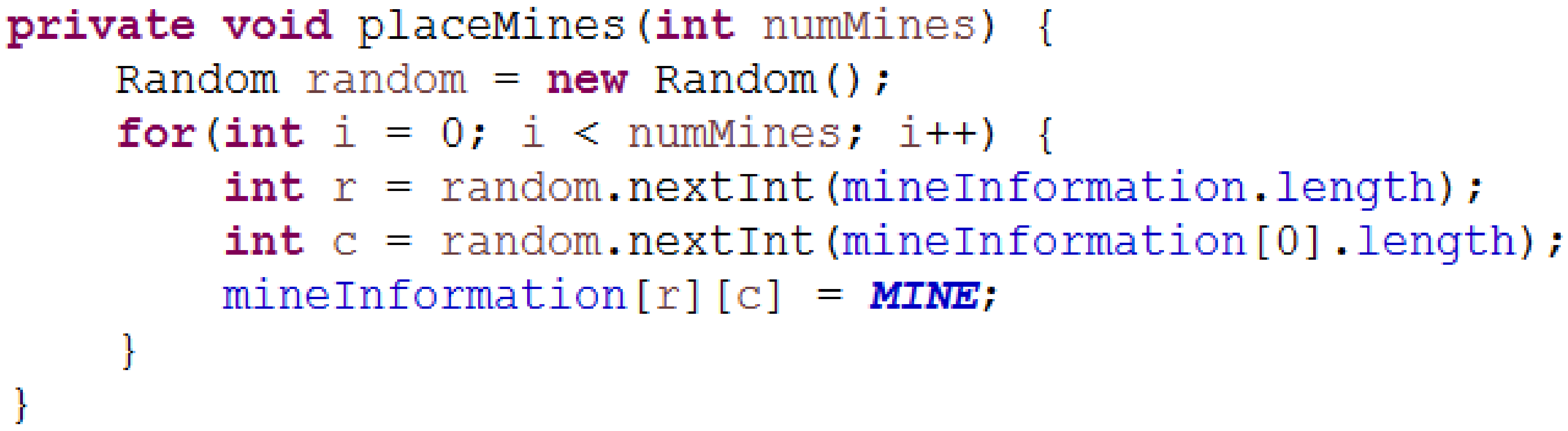
MineGrid (Model)

```
private static final int MINE = -1;
private int[][] mineInformation;

public MineGrid(int numRows, int numCols, int numMines) {
    mineInformation = new int[numRows][numCols];
    initializeCells();
    placeMines(numMines);
}

private void placeMines(int numMines) {
    Random random = new Random();
    for(int i = 0; i < numMines; i++) {
        int r = random.nextInt(mineInformation.length);
        int c = random.nextInt(mineInformation[0].length);
        mineInformation[r][c] = MINE;
    }
}
```

Any Problem?

MineGrid (Model)

- We may get the same (r, c) couple more than once.
- In that case, there will be less than `numMines` mines in the minegrid.



```java
private void placeMines(int numMines) {
    Random random = new Random();
    for(int i = 0; i < numMines; i++) {
        int r = random.nextInt(mineInformation.length);
        int c = random.nextInt(mineInformation[0].length);
        mineInformation[r][c] = MINE;
    }
}
```

MineGrid (Model)

```
private static final int MINE = -1;
private int[][] mineInformation;

public MineGrid(int numRows, int numCols, int numMines) {
    mineInformation = new int[numRows][numCols];
    initializeCells();
    placeMines(numMines);
}

private void placeMines(int numMines) {
    Random random = new Random();
    for(int i = 0; i < numMines; i++) {
        int r = random.nextInt(mineInformation.length);
        int c = random.nextInt(mineInformation[0].length);
        if (mineInformation[r][c] != MINE)
            mineInformation[r][c] = MINE;
        else
            i--;
    }
}
```

MineGrid (Model)

- Now the minegrid contains either 0s or the MINEs.
- What is next?

MineGrid (Model)

- Now the minegrid contains either 0s or the MINEs.
- What is next?

- If there is a mine in a cell, we need to update the cells surrounding the mine cell.
- How many cells do we need to update?

MineGrid (Model)

- Before the mines

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

MineGrid (Model)

- After the mines, we need to update the surrounding cells

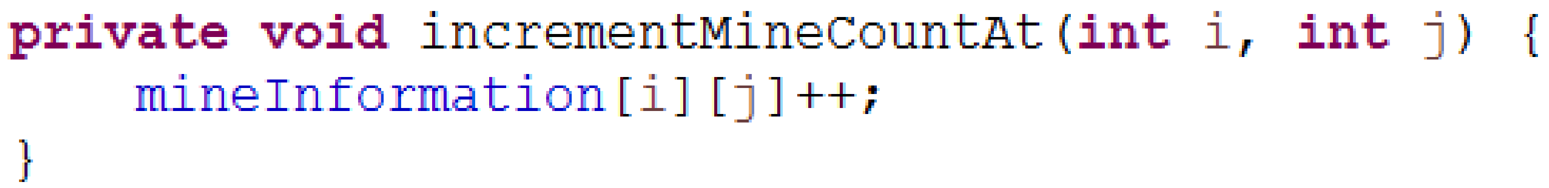
0	0	0	0	0
0	1	1	1	0
0	1	MINE	1	0
0	1	1	1	0
0	0	0	0	0

MineGrid (Model)

- Implement a method to increment the cell count.

MineGrid (Model)

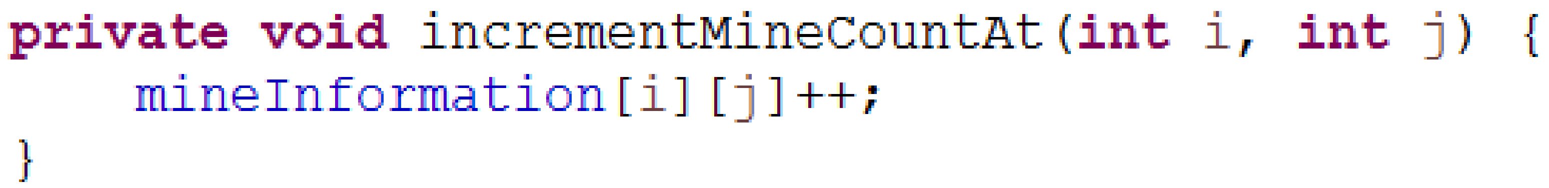
- Implement a method to increment the cell count.



```
private void incrementMineCountAt(int i, int j) {
    mineInformation[i][j]++;
}
```

MineGrid (Model)

- Implement a method to increment the cell count.



```java
private void incrementMineCountAt(int i, int j) {
    mineInformation[i][j]++;
}
```

- Now, we need to find the mines and update surrounding cells.

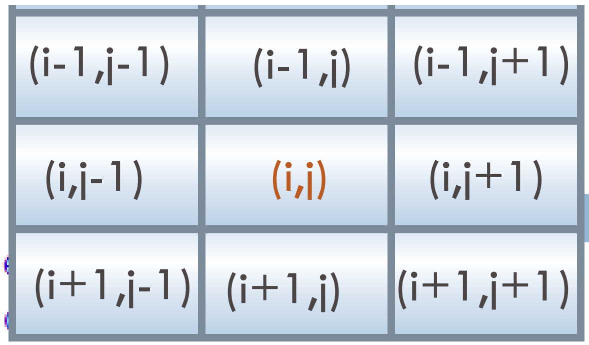
MineGrid (Model)

```java
private static final int MINE = -1;
private int[][] mineInformation;

public MineGrid(int numRows, int numCols, int numMines) {
    mineInformation = new int[numRows][numCols];
    initializeCells();
    placeMines(numMines);
    setMineInformation();
}

private void setMineInformation() {
    for(int i = 0; i < mineInformation.length; i++) {
        for(int j = 0; j < mineInformation[0].length; j++) {
            if(mineInformation[i][j] == MINE) {
                // ???
            }
        }
    }
}
```

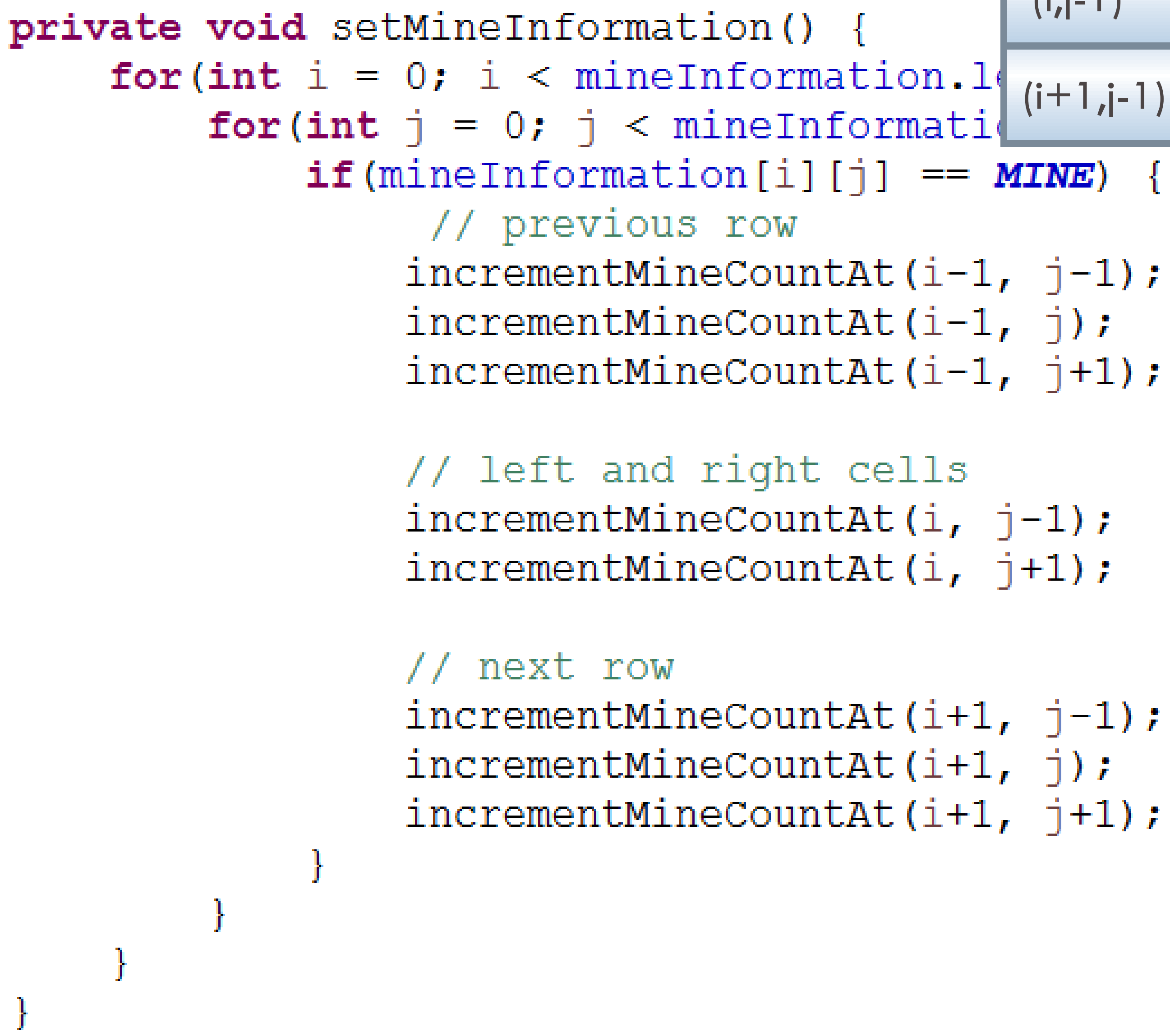
MineGrid (Model)

```
private void setMineInformation() {
    for(int i = 0; i < mineInformation.length; i++) {
        for(int j = 0; j < mineInformation[0].length; j++) {
            if(mineInformation[i][j] == MINE) {
                 // previous row
                incrementMineCountAt(i-1, j-1);
                incrementMineCountAt(i-1, j);
                incrementMineCountAt(i-1, j+1);

                // left and right cells
                incrementMineCountAt(i, j-1);
                incrementMineCountAt(i, j+1);

                // next row
                incrementMineCountAt(i+1, j-1);
                incrementMineCountAt(i+1, j);
                incrementMineCountAt(i+1, j+1);
            }
        }
    }
}
```

MineGrid (Model)



(i-1,j-1)	(i-1,j)	(i-1,j+1)
(i,j-1)	(i,j)	(i,j+1)
(i+1,j-1)	(i+1,j)	(i+1,j+1)



```
private void setMineInformation() {
    for(int i = 0; i < mineInformation.le
        for(int j = 0; j < mineInformati
            if(mineInformation[i][j] == MINE) {
                 // previous row
                incrementMineCountAt(i-1, j-1);
                incrementMineCountAt(i-1, j);
                incrementMineCountAt(i-1, j+1);

                // left and right cells
                incrementMineCountAt(i, j-1);
                incrementMineCountAt(i, j+1);

                // next row
                incrementMineCountAt(i+1, j-1);
                incrementMineCountAt(i+1, j);
                incrementMineCountAt(i+1, j+1);
            }
        }
    }
}
```

MineGrid (Model)

```
private void setMineInformation() {
    for(int i = 0; i < mineInformation.length; i++) {
        for(int j = 0; j < mineInformation[0].length; j++) {
            if(mineInformation[i][j] == MINE) {
                 // previous row
                incrementMineCountAt(i-1, j-1);
                incrementMineCountAt(i-1, j);
                incrementMineCountAt(i-1, j+1);

                // left and right cells
                incrementMineCountAt(i, j-1);
                incrementMineCountAt(i, j+1);

                // next row
                incrementMineCountAt(i+1, j-1);
                incrementMineCountAt(i+1, j);
                incrementMineCountAt(i+1, j+1);
            }
        }
    }
}
```

What happens if i=0 or i=mineInformation.length ?

MineGrid (Model)

```
private void setMineInformation() {
    for(int i = 0; i < mineInformation.length; i++) {
        for(int j = 0; j < mineInformation[0].length; j++) {
            if(mineInformation[i][j] == MINE) {
                 // previous row
                incrementMineCountAt(i-1, j-1);
                incrementMineCountAt(i-1, j);
                incrementMineCountAt(i-1, j+1);

                // left and right cells
                incrementMineCountAt(i, j-1);
                incrementMineCountAt(i, j+1);

                // next row
                incrementMineCountAt(i+1, j-1);
                incrementMineCountAt(i+1, j);
                incrementMineCountAt(i+1, j+1);
            }
        }
    }
}
```

Any idea how to fix this?

What happens if i=0 or i=mineInformation.length ?

MineGrid (Model)

- Implement a method which checks whether a (x,y) cell is within the limits of our grid.
- If (x,y) is within the limits, then the cell counter will be incremented.

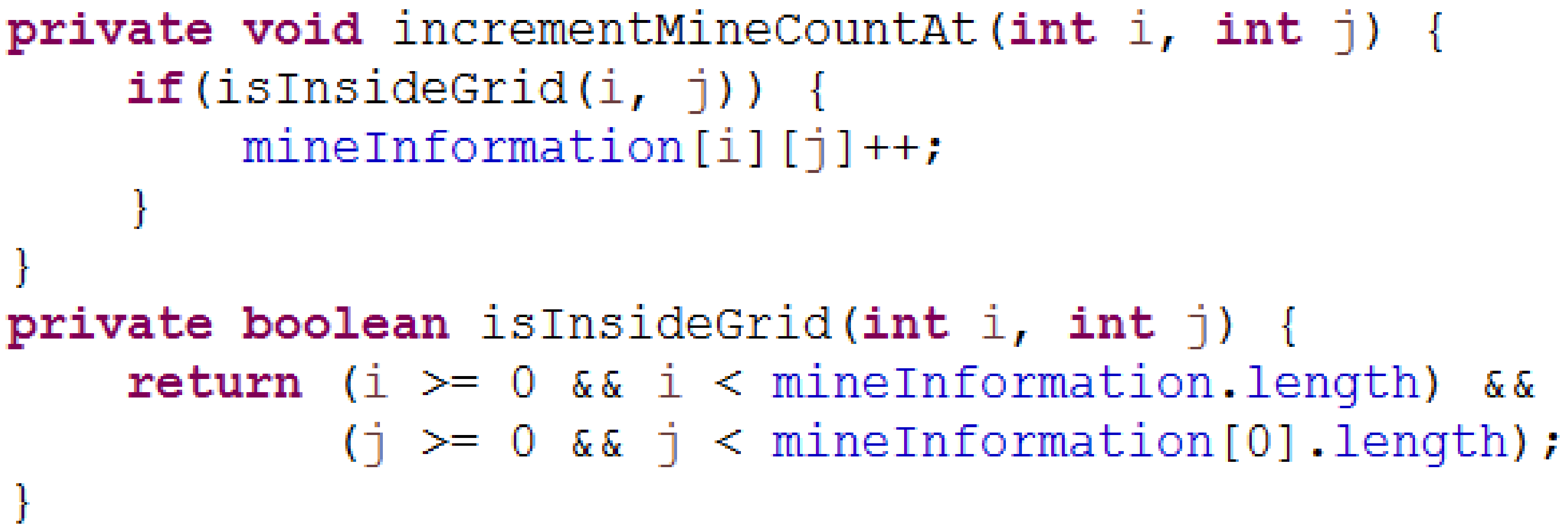
MineGrid (Model)

- Implement a method which checks whether a (x,y) cell is within the limits of our grid.
- If (x,y) is within the limits, then the cell counter will be incremented.

```java
private boolean isInsideGrid(int i, int j) {
    return (i >= 0 && i < mineInformation.length) &&
           (j >= 0 && j < mineInformation[0].length);
}
```

MineGrid (Model)

- Implement a method which checks whether a (x,y) cell is within the limits of our grid.
- If (x,y) is within the limits, then the cell counter will be incremented.



```
private void incrementMineCountAt(int i, int j) {
    if(isInsideGrid(i, j)) {
        mineInformation[i][j]++;
    }
}
private boolean isInsideGrid(int i, int j) {
    return (i >= 0 && i < mineInformation.length) &&
           (j >= 0 && j < mineInformation[0].length);
}
```

MineGrid (Model)

```
private void setMineInformation() {
    for(int i = 0; i < mineInformation.length; i++) {
        for(int j = 0; j < mineInformation[0].length; j++) {
            if(mineInformation[i][j] == MINE) {
                 // previous row
                incrementMineCountAt(i-1, j-1);
                incrementMineCountAt(i-1, j);
                incrementMineCountAt(i-1, j+1);

                // left and right cells
                incrementMineCountAt(i, j-1);
                incrementMineCountAt(i, j+1);

                // next row
                incrementMineCountAt(i+1, j-1);
                incrementMineCountAt(i+1, j);
                incrementMineCountAt(i+1, j+1);
            }
        }
    }
}
```

After those fixes, are there any other problems can you see?

MineGrid (Model)

```
private void setMineInformation() {
    for(int i = 0; i < mineInformation.length; i++) {
        for(int j = 0; j < mineInformation[0].length; j++) {
            if(mineInformation[i][j] == MINE) {
                 // previous row
                incrementMineCountAt(i-1, j-1);
                incrementMineCountAt(i-1, j);
                incrementMineCountAt(i-1, j+1);

                // left and right cells
                incrementMineCountAt(i, j-1);
                incrementMineCountAt(i, j+1);

                // next row
                incrementMineCountAt(i+1, j-1);
                incrementMineCountAt(i+1, j);
                incrementMineCountAt(i+1, j+1);
            }
        }
    }
}
```

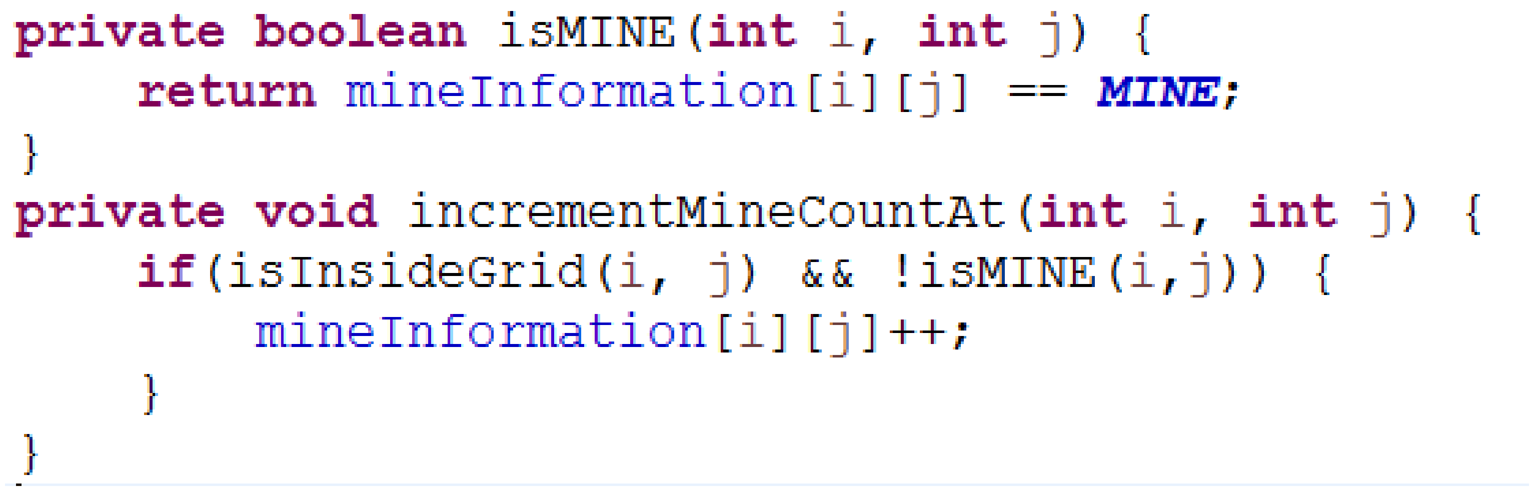
What if one of these cells contain a MINE? Remember MINE is represented with -1.

MineGrid (Model)

- Before incrementing a cell, make sure that it is does not contain a MINE.
- Implement a method to check whether a cell contains a MINE or not.

MineGrid (Model)

- Before incrementing a cell, make sure that it is does not contain a MINE.
- Implement a method to check whether a cell contains a MINE or not.



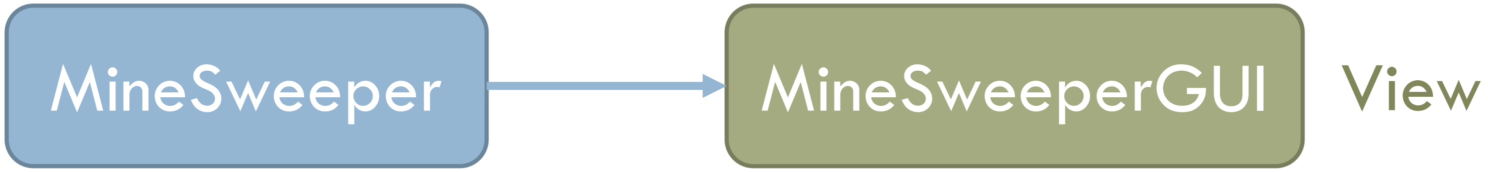
```java
private boolean isMINE(int i, int j) {
    return mineInformation[i][j] == MINE;
}
private void incrementMineCountAt(int i, int j) {
    if(isInsideGrid(i, j) && !isMINE(i,j)) {
        mineInformation[i][j]++;
    }
}
```

```
class MineGrid {
    private static final int MINE = -1;
    private int[][] mineInformation;
    public MineGrid(int numRows, int numCols, int numMines) {
        mineInformation = new int[numRows][numCols];
        initializeCells();
        placeMines(numMines);
        setMineInformation();
    }
    private void initializeCells() {
        for(int i = 0; i < mineInformation.length; i++) {
            for(int j = 0; j < mineInformation[0].length; j++) {
                mineInformation[i][j] = 0;
            }
        }
    }
    private void placeMines(int numMines) {
        Random random = new Random();
        for(int i = 0; i < numMines; i++) {
            int r = random.nextInt(mineInformation.length);
            int c = random.nextInt(mineInformation[0].length);
            if (mineInformation[r][c] != MINE)
                mineInformation[r][c] = MINE;
            else
                i--;
        }
    }
    private boolean isMINE(int i, int j) {
        return mineInformation[i][j] == MINE;
    }
```

```
private void setMineInformation() {
    for(int i = 0; i < mineInformation.length; i++) {
        for(int j = 0; j < mineInformation[0].length; j++) {
            if(mineInformation[i][j] == MINE) {
                 // previous row
                incrementMineCountAt(i-1, j-1);
                incrementMineCountAt(i-1, j);
                incrementMineCountAt(i-1, j+1);

                // left and right cells
                incrementMineCountAt(i, j-1);
                incrementMineCountAt(i, j+1);

                // next row
                incrementMineCountAt(i+1, j-1);
                incrementMineCountAt(i+1, j);
                incrementMineCountAt(i+1, j+1);
            }
        }
    }
}
private void incrementMineCountAt(int i, int j) {
    if(isInsideGrid(i, j) && !isMINE(i,j)) {
        mineInformation[i][j]++;
    }
}
private boolean isInsideGrid(int i, int j) {
    return (i >= 0 && i < mineInformation.length) &&
           (j >= 0 && j < mineInformation[0].length);
}
```

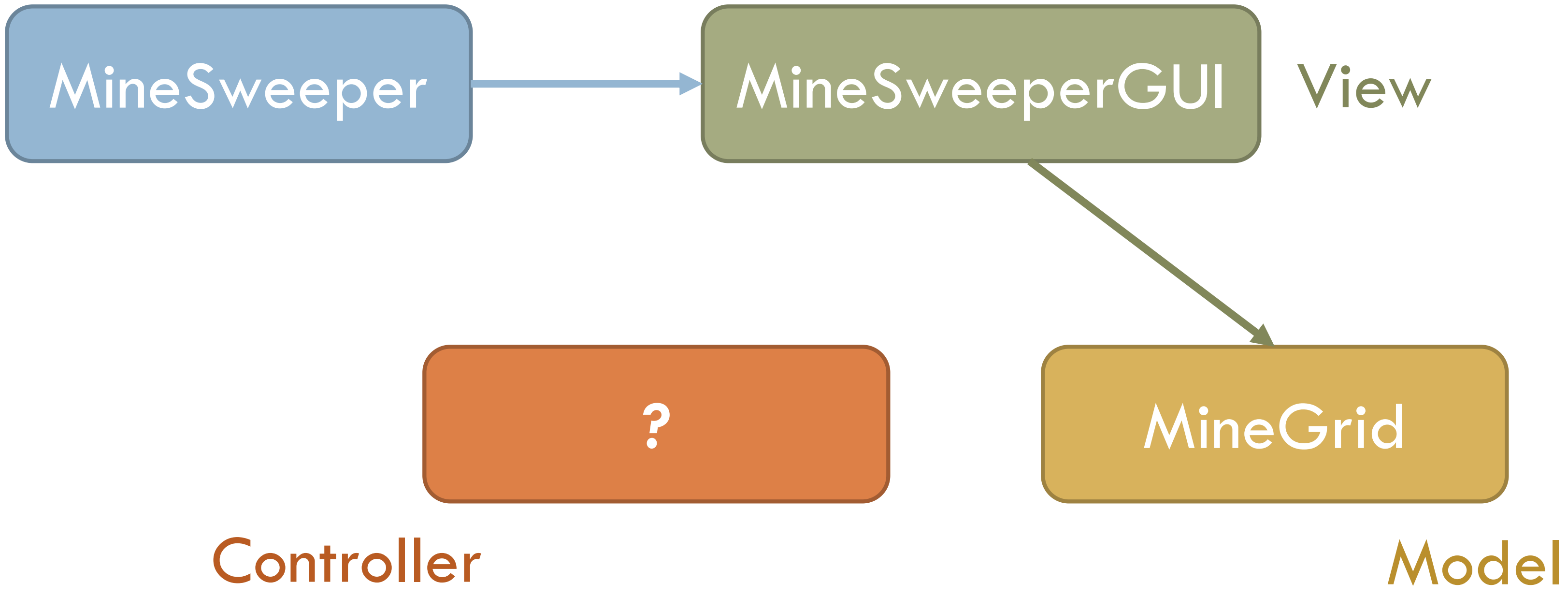
MineSweeper

- MineGrid class (the model) is implemented.
- Where are we going to instantinate it?

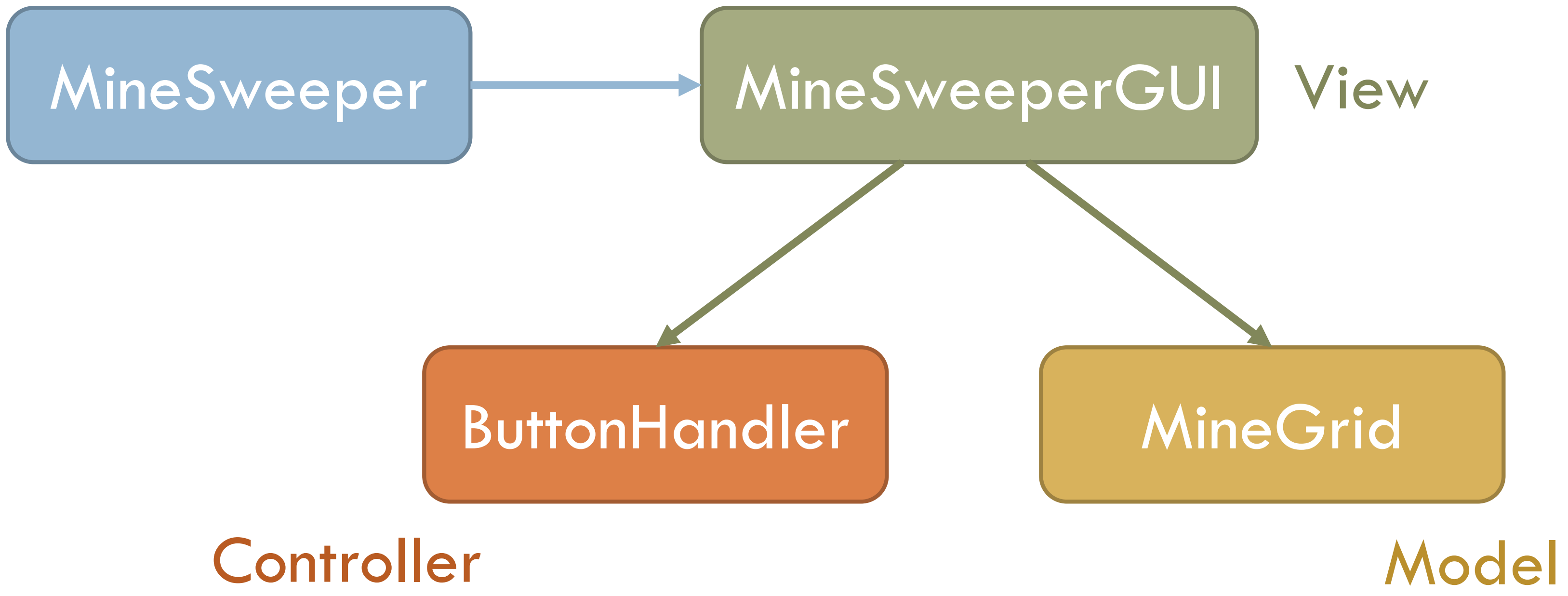


MineSweeper

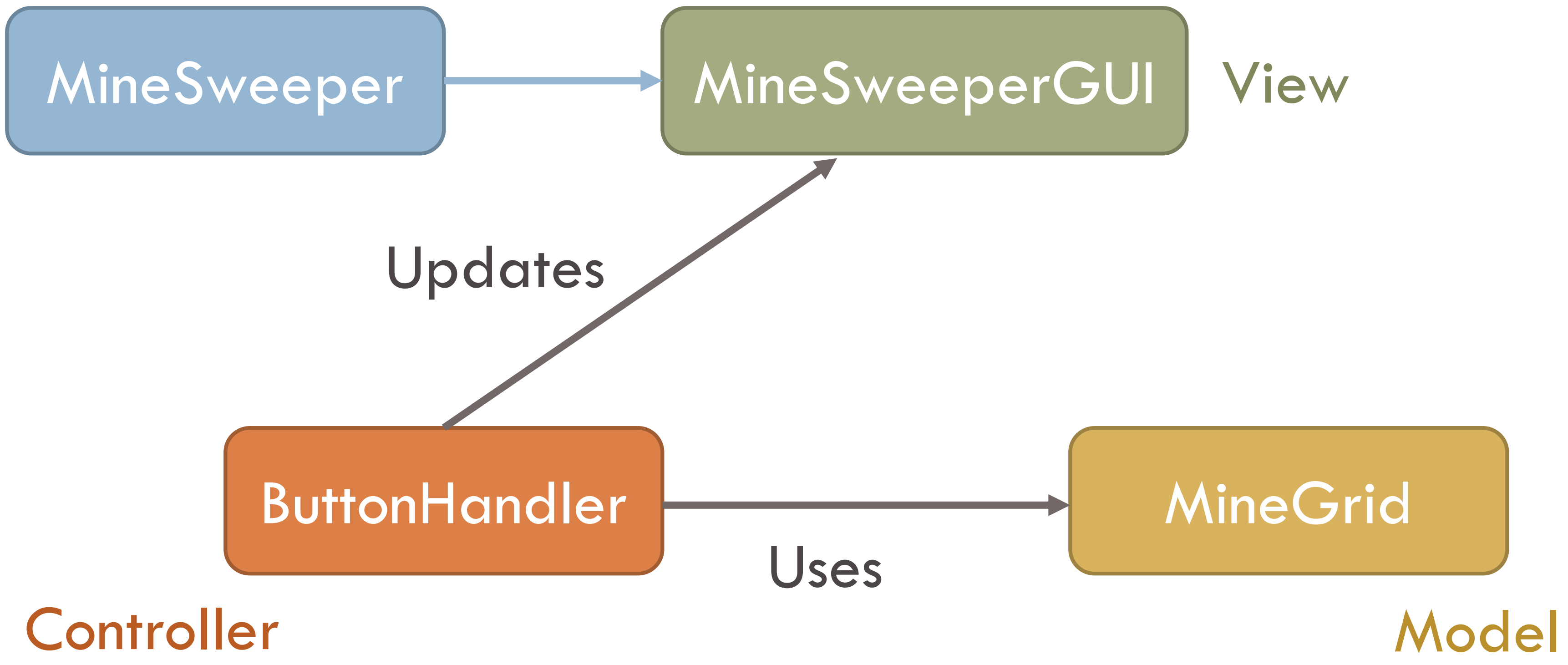
- MineGrid class (the model) is implemented.
- Where are we going to instantinate it?



MineSweeper



MineSweeper



MineSweeper

```
class MineSweeperGUI extends JPanel {
    private MineGrid grid;

    public MineSweeperGUI(int numRows, int numCols, int numMines) {

        grid = new MineGrid(numRows, numCols, numMines);

        setLayout(new GridLayout(numRows, numCols));
        for(int i = 0; i < numRows; i++) {
            for(int j = 0; j < numCols; j++) {
                JButton button = new JButton();
                add(button);
            }
        }
    }
}
```

MineSweeper

```
class MineSweeperGUI extends JPanel {
    private MineGrid grid;

    public MineSweeperGUI(int numRows, int numCols, int numMines) {

        grid = new MineGrid(numRows, numCols, numMines);

        setLayout(new GridLayout(numRows, numCols));
        for(int i = 0; i < numRows; i++) {
            for(int j = 0; j < numCols; j++) {
                JButton button = new JButton();
                add(button);
            }
        }
    }
}
```

- What is missing?

MineSweeper

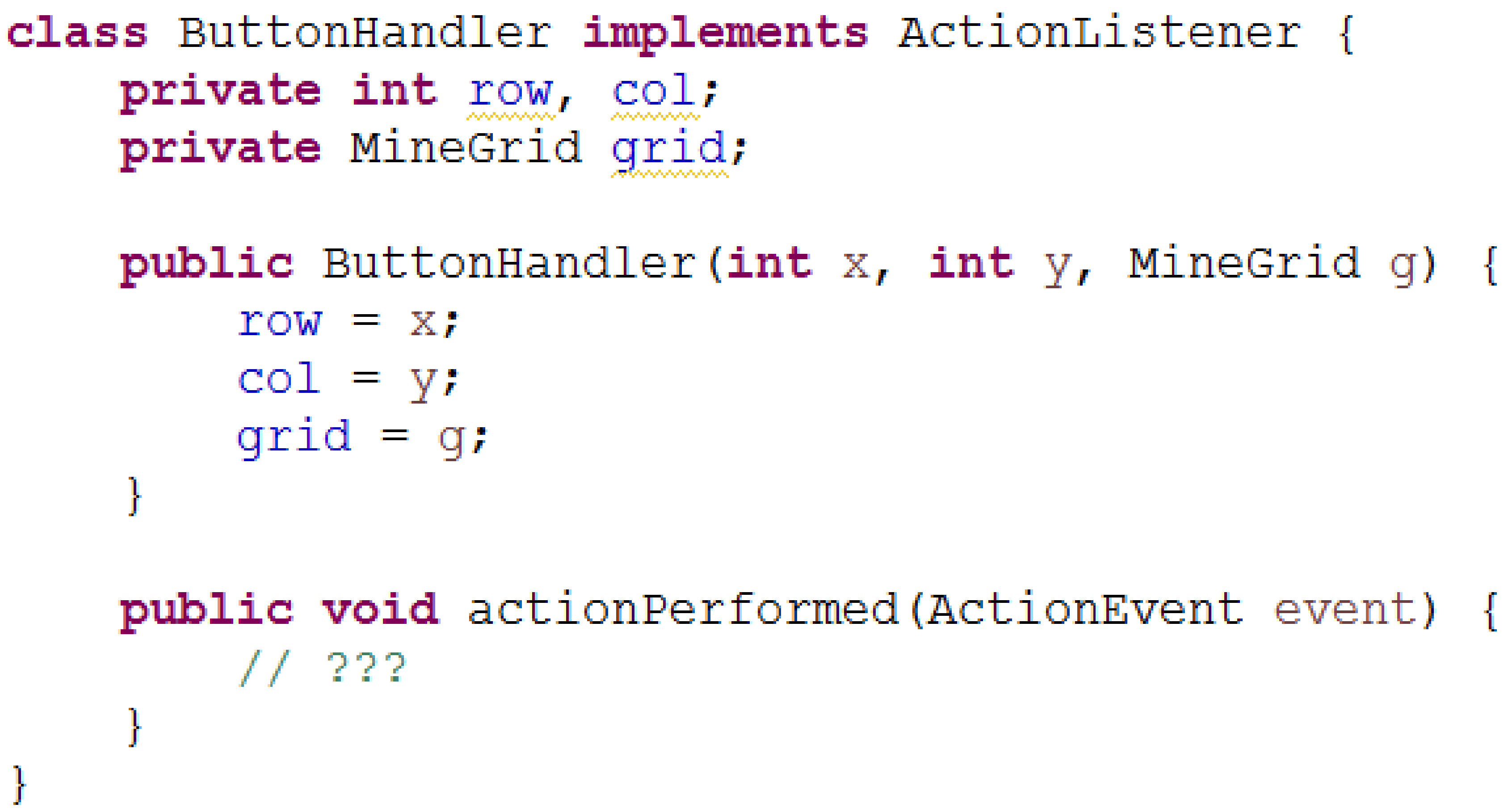
```
class MineSweeperGUI extends JPanel {
    private MineGrid grid;

    public MineSweeperGUI(int numRows, int numCols, int numMines) {

        grid = new MineGrid(numRows, numCols, numMines);

        setLayout(new GridLayout(numRows, numCols));
        for(int i = 0; i < numRows; i++) {
            for(int j = 0; j < numCols; j++) {
                JButton button = new JButton();
                add(button);
                button.addActionListener(new ButtonHandler(i,j, grid));
            }
        }
    }
}
```

MineSweeper

```
class MineSweeperGUI extends JPanel {
    private MineGrid grid;

    public MineSweeperGUI(int numRows, int numCols, int numMines) {

        grid = new MineGrid(numRows, numCols, numMines);

        setLayout(new GridLayout(numRows, numCols));
        for(int i = 0; i < numRows; i++) {
            for(int j = 0; j < numCols; j++) {
                JButton button = new JButton();
                add(button);
                button.addActionListener(new ButtonHandler(i,j, grid));
            }
        }
    }
}
```

- For each button, we need its row and column as well as the model, why?

ButtonHandler



```
class ButtonHandler implements ActionListener {
    private int row, col;
    private MineGrid grid;

    public ButtonHandler(int x, int y, MineGrid g) {
        row = x;
        col = y;
        grid = g;
    }

    public void actionPerformed(ActionEvent event) {
        // ???
    }
}
```

ButtonHandler

- What happens if user clicks a cell with mine?
- What happens otherwise?

ButtonHandler

- What happens if user clicks a cell with mine?
  - Game over.
  - Exit the system.
- What happens otherwise?
  - Display the content of the cell.
  - # mines surrounding

ButtonHandler

- What happens if user clicks a cell with mine?
  - Game over.
  - Exit the system.
- What happens otherwise?
  - Display the content of the cell.
  - # mines surrounding
- So we need to check whether a cell is a mine or not?
  - Do we have a function for this?

ButtonHandler

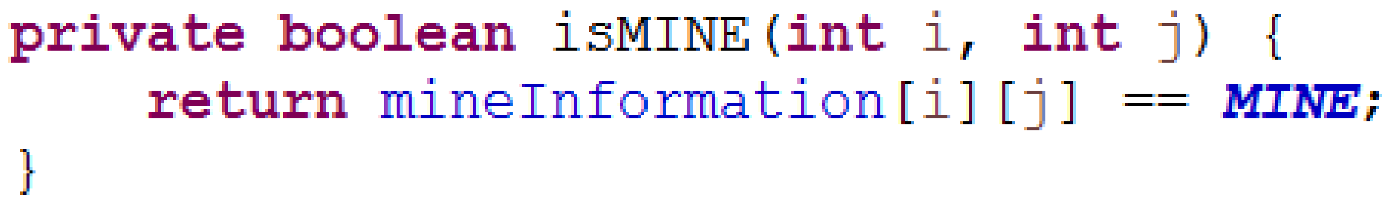
- So we need to check whether a cell is a mine or not?
  - Do we have a function for this?
  - Remember this one:

```
private boolean isMINE(int i, int j) {
    return mineInformation[i][j] == MINE;
}
```

  - Can we use this in ButtonHandler?

ButtonHandler

- So we need to check whether a cell is a mine or not?
  - Do we have a function for this?
  - Remember this one:



```
private boolean isMINE(int i, int j) {
    return mineInformation[i][j] == MINE;
}
```

  - Can we use this in ButtonHandler?
    - Make it public

ButtonHandler

- MineGrid

```
public boolean isMINE(int i, int j) {
    return mineInformation[i][j] == MINE;
}
```

- ButtonHandler

```
public void actionPerformed(ActionEvent event) {
    if(grid.isMINE(row, col)) {

    }
    else {

    }
}
```

ButtonHandler

- MineGrid

```
public boolean isMINE(int i, int j) {
    return mineInformation[i][j] == MINE;
}
```

- ButtonHandler

```
public void actionPerformed(ActionEvent event) {
    if(grid.isMINE(row, col)) {
        JOptionPane.showMessageDialog(null, "OOOPS!!");
        System.exit(0);
    }
    else {

    }
}
```

ButtonHandler

- MineGrid

```
public boolean isMINE(int i, int j) {
    return mineInformation[i][j] == MINE;
}
```

- ButtonHandler

```
public void actionPerformed(ActionEvent event) {
    if(grid.isMINE(row, col)) {
        JOptionPane.showMessageDialog(null, "OOOPS!!");
        System.exit(0);
    }
    else {
        // get the number from MineGrid(row,col)
        // display is on the button
    }
}
```

MineGrid

- We need a function in MineGrid class which returns the number inside the cell.

```
public int getCellContent(int i, int j) {
    return mineInformation[i][j];
}
```

ButtonHandler

- MineGrid

```
public int getCellContent(int i, int j) {
    return mineInformation[i][j];
}
```

- ButtonHandler

```
public void actionPerformed(ActionEvent event) {
    if(grid.isMINE(row, col)) {
        JOptionPane.showMessageDialog(null, "OOOPS!!");
        System.exit(0);
    }
    else {
        if (event.getSource() instanceof JButton) {
            JButton button = (JButton)event.getSource();
            button.setText(String.valueOf(grid.getCellContent(row, col)));
        }
    }
}
```

ButtonHandler

```
class ButtonHandler implements ActionListener {
    private int row, col;
    private MineGrid grid;

    public ButtonHandler(int x, int y, MineGrid g) {
        row = x;
        col = y;
        grid = g;
    }

    public void actionPerformed(ActionEvent event) {
        if(grid.isMINE(row, col)) {
            JOptionPane.showMessageDialog(null, "OOOPS!!");
            System.exit(0);
        }
        else {
            if (event.getSource() instanceof JButton) {
                JButton button = (JButton)event.getSource();
                button.setText(String.valueOf(grid.getCellContent(row, col)));
            }
        }
    }
}
```

- Check MineSweeper.java

Any Questions ?