Table Of Content

Package base

Class Summary

HighScore

Structure for storing high scores.

Main

The main class containing the main method.

MinesweeperPreferences

Stores the preferences, high scores and the saved game.

MinesweeperPreferences.Difficulty

Difficulty enumeration.

base

Class HighScore

```
java.lang.Object
    |
    +--base.HighScore
```

All Implemented Interfaces:

java.io.Serializable

< Constructors > < Methods >

public class **HighScore**
extends java.lang.Object
implements java.io.Serializable

Structure for storing high scores.

Constructors

HighScore

```
public  HighScore(java.lang.String name,
                  int seconds)
```

Main constructor-

Parameters:

name - the name of the player
seconds - the score of the player

Methods

getName

```
public java.lang.String getName()
```

Returns:

the name fo the player

getSeconds

```
public int getSeconds()
```

Returns:

the score of the player

setName

```
public void setName(java.lang.String name)
```

Parameters:

name - the name to set

setSeconds

```
public void setSeconds(int seconds)
```

Parameters:

seconds - the score to set

base

Class Main

```
java.lang.Object
    |
    +--base.Main
```

< Constructors > < Methods >

public class **Main**

extends java.lang.Object

The main class containing the main method.

Constructors

Main

`public  Main()`

Methods

getPrefs

`public static MinesweeperPreferences getPrefs()`

Returns:

the preferences used by the program.

---

main

`public static void main(java.lang.String[] args)`

The entry point of the program.

Parameters:

args - command line arguments, unused

---

savePreferences

`public static void savePreferences()`

Writes the preferences into a file.

base

Class MinesweeperPreferences

```
java.lang.Object
   |
   +--base.MinesweeperPreferences
```

All Implemented Interfaces:

java.io.Serializable

< Constructors > < Methods >

---

public class **MinesweeperPreferences**
extends java.lang.Object
implements java.io.Serializable

Stores the preferences, high scores and the saved game.

Constructors

MinesweeperPreferences

`public  MinesweeperPreferences()`

Main constructor.

Methods

decrementBombs

`public int decrementBombs()`

Decrease the number of mines to be uncovered by one.

Returns:

the number of mines to be uncovered after decrementing

---

getBombsLeft

`public int getBombsLeft()`

Returns:

the number of mines to be uncovered

---

getDifficulty

```
public MinesweeperPreferences.Difficulty getDifficulty()
```

Returns:
the difficulty

---

getEasyHighScore

```
public HighScore getEasyHighScore()
```

Returns:
the high score in easy difficulty

---

getHardHighScore

```
public HighScore getHardHighScore()
```

Returns:
the high score in hard difficulty

---

getMediumHighScore

```
public HighScore getMediumHighScore()
```

Returns:
the high score in medium difficulty

---

getNumberOfBombs

```
public int getNumberOfBombs()
```

Returns:
the number of mines to find

---

getNumberOfColumns

```
public int getNumberOfColumns()
```

Returns:

the number of columns

---

getNumberOfRows

```
public int getNumberOfRows()
```

Returns:

the number of rows

---

getPlayerName

```
public java.lang.String getPlayerName()
```

Returns:

the name of the player

---

getSavedGame

```
public game.MineCell[][] getSavedGame()
```

Returns:

the saved game

---

getSavedTime

```
public int getSavedTime()
```

Returns:

the play tim of the saved game

---

incrementBombs

```
public int incrementBombs()
```

Increase the number of mines to be uncovered by one.

Returns:

the number of mines to be uncovered after incrementing

---

isShowTimer

```
public boolean isShowTimer()
```

Returns:

whether the player wants to see the timer

---

isUseQuestionMark

```
public boolean isUseQuestionMark()
```

Returns:

whether the player wants to use question marks

---

saveHighScore

```
public void saveHighScore(int time,
                          MinesweeperPreferences.Difficulty difficulty)
```

Saved the current score as the high score if better.

Parameters:

time - the time of the current score
difficulty - the difficulty of the game

---

setBombsLeft

```
public void setBombsLeft(int bombsLeft)
```

Parameters:

bombsLeft - number to set the remaining mines

---

setDifficulty

```
public void setDifficulty(MinesweeperPreferences.Difficulty difficulty)
```

Parameters:

difficulty - the difficulty to set

---

setDifficulty

```
public void setDifficulty(int rows,
                          int columns,
                          int bombs)
```

Sets the difficulty of the game.

Parameters:

rows - the number of rows
columns - the number of columns
bombs - the number of mines

---

setEasyHighScore

```
public void setEasyHighScore(HighScore easyHighScore)
```

Parameters:

easyHighScore - high score in easy difficulty to set

---

setHardHighScore

```
public void setHardHighScore(HighScore hardHighScore)
```

Parameters:

hardHighScore - high score in hard difficulty to set

---

setMediumHighScore

```
public void setMediumHighScore(HighScore mediumHighScore)
```

Parameters:

mediumHighScore - high score in medium difficulty to set

---

setPlayerName

```
public void setPlayerName(java.lang.String playerName)
```

Parameters:

playerName - the name to set the player's name

---

setSavedGame

```
public void setSavedGame(game.MineCell[][] savedGame)
```

Parameters:

savedGame - the game to save

---

setSavedTime

```
public void setSavedTime(int savedTime)
```

Parameters:

savedTime - play time to save

---

setShowTimer

```
public void setShowTimer(boolean showTimer)
```

Parameters:

showTimer - boolean to set the 'show timer' option

---

setUseQuestionMark

```
public void setUseQuestionMark(boolean useQuestionMark)
```

Parameters:

useQuestionMark - boolean to set the 'use question mark' option

---

base

Class MinesweeperPreferences.Difficulty

```
java.lang.Object
    |
    +--java.lang.Enum
        |
        +--base.MinesweeperPreferences.Difficulty
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

< Fields > < Methods >

public static final class **MinesweeperPreferences.Difficulty**
extends java.lang.Enum
implements java.io.Serializable

Difficulty enumeration.

Fields

CUSTOM

`public static final MinesweeperPreferences.Difficulty` **`CUSTOM`**

EASY

`public static final MinesweeperPreferences.Difficulty` **`EASY`**

HARD

`public static final MinesweeperPreferences.Difficulty` **`HARD`**

MEDIUM

`public static final MinesweeperPreferences.Difficulty` **`MEDIUM`**

Methods

valueOf

`public static MinesweeperPreferences.Difficulty` **`valueOf`**`(java.lang.String name)`

values

```
public static base.MinesweeperPreferences.Difficulty[] values()
```

Package game

Class Summary

MineCell

The class containing the properties of a mine cell.

MineCellContent

Enumeration used to represent the content of a mine cell

MineCellState

Enumeration used to represent the state of a mine cell,

MineField

The core class of the game.

Player

The class used to represent the player and its properties.

---

game

Class MineCell

```
java.lang.Object
    |
    +--game.MineCell
```

All Implemented Interfaces:

java.io.Serializable

< Constructors > < Methods >

---

public class **MineCell**
extends java.lang.Object
implements java.io.Serializable

The class containing the properties of a mine cell.

Constructors

MineCell

```
public  MineCell()
```

Default constructor.

MineCell

```
public  MineCell(MineCellContent content)
```

Constructor.

Parameters:

content - the content to be set

MineCell

```
public  MineCell(MineCellContent content,
                 MineCellState state)
```

Constructor

Parameters:

content - the content to be set
state - the state to be set

MineCell

```
public  MineCell(MineCellState state)
```

Constructor.

Parameters:

state - the state to be set

Methods

getContent

```
public MineCellContent getContent()
```

Returns:

the content of the mine cell

---

getContentValue

```
public int getContentValue()
```

Decodes to content of the mine cell to an integer.

Returns:

the decoded integer

getState

```
public MineCellState getState()
```

Returns:

the state of the mine cell

isBomb

```
public boolean isBomb()
```

Returns:

whether the cell contains a mine

isEmpty

```
public boolean isEmpty()
```

Returns:

whether the cell is empty

isFlagged

```
public boolean isFlagged()
```

Returns:

whether the cell is flagged

isProtected

```
public boolean isProtected()
```

Returns:

whether the cell is protected

isQuestionMarked

```
public boolean isQuestionMarked()
```

Returns:
whether the cell is marked with a question mark

---

isRevealed

```
public boolean isRevealed()
```

Returns:
whether the cell has been uncovered

---

isUnmarked

```
public boolean isUnmarked()
```

Returns:
whether the cell is unmarked

---

setContent

```
public void setContent(MineCellContent content)
```

Parameters:
content - the content to be set

---

setState

```
public void setState(MineCellState state)
```

Parameters:
state - the state to be set

---

game

Class MineCellContent

```
java.lang.Object
    |
    +--java.lang.Enum
        |
        +--game.MineCellContent
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

< Fields > < Methods >

public final class **MineCellContent**
extends java.lang.Enum
implements java.io.Serializable

Enumeration used to represent the content of a mine cell

Fields

BOMB

`public static final MineCellContent BOMB`

EIGHT

`public static final MineCellContent EIGHT`

EMPTY

`public static final MineCellContent EMPTY`

FIVE

`public static final MineCellContent FIVE`

FOUR

`public static final MineCellContent FOUR`

ONE

```
public static final MineCellContent ONE
```

PROTECTED

```
public static final MineCellContent PROTECTED
```

SEVEN

```
public static final MineCellContent SEVEN
```

SIX

```
public static final MineCellContent SIX
```

THREE

```
public static final MineCellContent THREE
```

TWO

```
public static final MineCellContent TWO
```

Methods

valueOf

```
public static MineCellContent valueOf(java.lang.String name)
```

---

values

```
public static game.MineCellContent[] values()
```

game

Class MineCellState

```
java.lang.Object
   |
   +--java.lang.Enum
         |
         +--game.MineCellState
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

< Fields > < Methods >

public final class **MineCellState**
extends java.lang.Enum
implements java.io.Serializable

Enumeration used to represent the state of a mine cell,

Fields

FLAGGED

`public static final MineCellState FLAGGED`

QUESTIONMARK

`public static final MineCellState QUESTIONMARK`

REVEALED

`public static final MineCellState REVEALED`

UNMARKED

`public static final MineCellState UNMARKED`

Methods

valueOf

`public static MineCellState valueOf(java.lang.String name)`

values

```
public static game.MineCellState[] values()
```

game

Class MineField

```
java.lang.Object
    |
    +--game.MineField
```

< Constructors > < Methods >

public class **MineField**
extends java.lang.Object

The core class of the game. Prepares the board for game play and handles game events.

Constructors

MineField

```
public  MineField(javax.swing.JLabel bombsLabel,
                  javax.swing.JLabel timeLabel,
                  javax.swing.JButton faceButton)
```

Class constructor. Prepares the board, loads the saved game if present.

Parameters:

bombsLabel - reference to a JLabel where the remaining number of mines is displayed
timeLabel - reference to a JLabel where the current elapsed time is displayed
faceButton - reference to a JButton for changing the face of the button

Methods

cancelTimer

```
public void cancelTimer()
```

Stops the timer if started.

getCellPanels

```
public gui.panel.MineCellPanel[][] getCellPanels()
```

Returns a 2 dimensional array of MineCellPanel objects.

Returns:

a MineCellPanel[][] object

---

getCells

```
public game.MineCell[][] getCells()
```

Returns a 2 dimensional array of MineCell objects.

Returns:

a MineCell[][] object

---

getColumns

```
public int getColumns()
```

Returns the number of columns in the current game.

Returns:

the number of columns

---

getRows

```
public int getRows()
```

Returns the number of rows of the current game.

Returns:

the number of rows

---

getTime

```
public int getTime()
```

Returns the elapsed game time.

Returns:

the time

---

onCellClick

```
public void onCellClick(int row,
                        int column)
```

Handles clicks on the cell in the given row and column.

Parameters:

row - the row the cell is in
column - the column the cell is in

---

onPreferenceChanged

```
public void onPreferenceChanged()
```

Handles preference modification events.

---

onTwoButtonCellClick

```
public void onTwoButtonCellClick(int row,
                                 int column)
```

Handles clicks made with both mouse buttons on uncovered numbered cells.

Parameters:

row - the row the cell is in
column - the column the cell is in

---

toggleFlag

```
public void toggleFlag(int row,
                       int column)
```

Handles right click on the covered cell in the given row and column.

Parameters:

row - the row the cell is in
column - the column the cell is in

---

game

Class Player

```
java.lang.Object
    |
    +--game.Player
```

< Constructors > < Methods >

---

public class **Player**
extends java.lang.Object

The class used to represent the player and its properties.

Constructors

Player

`public  Player()`

Methods

isAlive

`public static boolean isAlive()`

Returns:
whether the player is alive

---

isGameStarted

`public static boolean isGameStarted()`

Returns:
whether the player has started the game

---

setGameStarted

```
public static void setGameStarted(boolean gameStarted)
```

Parameters:

gameStarted - boolean to set whether the player has started to game

---

setIsAlive

```
public static void setIsAlive(boolean isAlive)
```

Parameters:

isAlive - boolean to set whether the player is alive

Package gui

Class Summary

MineFieldGUI

Class used for drawing the whole board.

MinesweeperGUI

The main frame of the game.

gui

Class MineFieldGUI

```
java.lang.Object
    |
    +--java.awt.Component
        |
        +--java.awt.Container
            |
            +--javax.swing.JComponent
                |
                +--javax.swing.JPanel
                    |
                    +--gui.MineFieldGUI
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.image.ImageObserver, java.io.Serializable, javax.accessibility.Accessible, javax.swing.TransferHandler.HasGetTransferHandler

< Constructors > < Methods >

public class **MineFieldGUI**
extends javax.swing.JPanel

Class used for drawing the whole board.

Constructors

MineFieldGUI

```
public  MineFieldGUI(javax.swing.JLabel bombsLabel,
                     javax.swing.JLabel timeLabel,
                     javax.swing.JButton faceButton)
```

Main constructor.

Parameters:

bombsLabel - reference to a JLabel for showing the number of mines left
timeLabel - reference to a JLabel for showing current play time
faceButton - reference to a JButton for changing the face of the button

Methods

cancelTimer

```
public void cancelTimer()
```

Stops the timer.

---

getMineField

```
public MineField getMineField()
```

Returns:

MineField object of the current game

---

saveGame

```
public void saveGame()
```

Saves the game.

---

gui

Class MinesweeperGUI

```
java.lang.Object
    |
    +--java.awt.Component
        |
        +--java.awt.Container
            |
            +--java.awt.Window
                |
                +--java.awt.Frame
                    |
                    +--javax.swing.JFrame
                        |
                        +--gui.MinesweeperGUI
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.image.ImageObserver, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.TransferHandler.HasGetTransferHandler, javax.swing.WindowConstants

---

< Constructors >

---

public class **MinesweeperGUI**
extends javax.swing.JFrame

The main frame of the game.

Constructors

MinesweeperGUI

`public  **MinesweeperGUI**()`

Main constructor. Creates every component of the game.

Package gui.dialog

Class Summary

CustomDifficultyDialog

A dialog for setting a custom difficulty.

HighScoresDialog

Dialog for showing the high scores

MinesweeperPreferencesDialog

Dialog for game preferences.

---

gui.dialog

Class CustomDifficultyDialog

```
java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--java.awt.Window
                    |
                    +--java.awt.Dialog
                          |
                          +--javax.swing.JDialog
                                |
                                +--gui.dialog.CustomDifficultyDialog
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.event.ActionListener, java.awt.image.ImageObserver, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.TransferHandler.HasGetTransferHandler, javax.swing.WindowConstants

---

< Constructors > < Methods >

---

public class **CustomDifficultyDialog**
extends javax.swing.JDialog
implements java.awt.event.ActionListener

A dialog for setting a custom difficulty.

Constructors

CustomDifficultyDialog

```
public  CustomDifficultyDialog(java.awt.Frame owner,
                               boolean modal)
```

Main constructor. It creates and opens the dialog.

Parameters:

owner - the frame from which the dialog is displayed
modal - specifies whether dialog blocks user input to other top-level windows when shown

Methods

actionPerformed

```
public void actionPerformed(java.awt.event.ActionEvent e)
```

This method is called when an action is performed on it's parent component.

Parameters:

e - the event

gui.dialog

Class HighScoresDialog

```
java.lang.Object
    |
    +--java.awt.Component
        |
        +--java.awt.Container
            |
            +--java.awt.Window
                |
                +--java.awt.Dialog
                    |
                    +--javax.swing.JDialog
                        |
                        +--gui.dialog.HighScoresDialog
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.event.ActionListener, java.awt.image.ImageObserver, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.TransferHandler.HasGetTransferHandler, javax.swing.WindowConstants

< Constructors > < Methods >

public class **HighScoresDialog**
extends javax.swing.JDialog
implements java.awt.event.ActionListener

Dialog for showing the high scores

Constructors

HighScoresDialog

```
public  HighScoresDialog(java.awt.Frame owner,
                         boolean modal)
```

Main constructor. It creates and opens the dialog.

Parameters:

owner - the frame from which the dialog is displayed
modal - specifies whether dialog blocks user input to other top-level windows when shown

Methods

actionPerformed

```
public void actionPerformed(java.awt.event.ActionEvent e)
```

This method is called when an action is performed on it's parent component.

Parameters:

e - the event

---

gui.dialog

Class MinesweeperPreferencesDialog

```
java.lang.Object
   |
   +--java.awt.Component
         |
         +--java.awt.Container
               |
               +--java.awt.Window
                     |
                     +--java.awt.Dialog
                           |
                           +--javax.swing.JDialog
                                 |
                                 +--gui.dialog.MinesweeperPreferencesDialog
```

All Implemented Interfaces:

java.awt.MenuContainer, java.awt.event.ActionListener, java.awt.image.ImageObserver, java.io.Serializable, javax.accessibility.Accessible, javax.swing.RootPaneContainer, javax.swing.TransferHandler.HasGetTransferHandler, javax.swing.WindowConstants

< Constructors > < Methods >

public class **MinesweeperPreferencesDialog**
extends javax.swing.JDialog

implements java.awt.event.ActionListener

Dialog for game preferences.

Constructors

MinesweeperPreferencesDialog

```
public  MinesweeperPreferencesDialog(javax.swing.JFrame owner,
                                     boolean modal,
                                     MineField mineField)
```

Main constructor. It creates and opens the dialog.

Parameters:

owner - the frame from which the dialog is displayed
modal - specifies whether dialog blocks user input to other top-level windows when shown
mineField - MineField object of the current game

Methods

actionPerformed

```
public void actionPerformed(java.awt.event.ActionEvent e)
```

This method is called when an action is performed on it's parent component.

Parameters:

e - the event

Package gui.panel

Class Summary

MineCellPanel

Class used for displaying mine cells.

---

gui.panel

Class MineCellPanel

```
java.lang.Object
    |
    +--gui.panel.MineCellPanel
```

All Implemented Interfaces:
java.awt.event.MouseListener

< Fields > < Constructors > < Methods >

---

public class **MineCellPanel**
extends java.lang.Object
implements java.awt.event.MouseListener

Class used for displaying mine cells.

Fields

SIZE

```
public static final int SIZE
```

Constructors

MineCellPanel

```
public  MineCellPanel(MineField mineField,
                      int row,
                      int column,
                      javax.swing.JButton faceButton)
```

Main constructor. Creates the necessary resources.

Parameters:

mineField - a MineField object this cell is loceted in
row - row index of this cell
column - column index of this cell
faceButton - reference to a JButton for changing the face of the button

Methods

flagCell

`public void flagCell()`

Flags the cell.

---

getButton

`public javax.swing.JButton getButton()`

Returns:

the button which uncovers the cell when pressed

---

getCellContent

`public javax.swing.JPanel getCellContent()`

Returns:

the panel representing the content of the cell

---

getCellPanel

`public javax.swing.JPanel getCellPanel()`

Returns:

the panel of the mine cell

---

getMineIconLabel

`public javax.swing.JLabel getMineIconLabel()`

Returns:

the label containing the mine icon

---

mouseClicked

```
public void mouseClicked(java.awt.event.MouseEvent e)
```

Handles mouse button clicks.

Parameters:

e - the mouse event

---

mouseEntered

```
public void mouseEntered(java.awt.event.MouseEvent e)
```

Handles the 'mouse entering the window' event.

Parameters:

e - the mouse event

---

mouseExited

```
public void mouseExited(java.awt.event.MouseEvent e)
```

Handles the 'mouse exiting the window' event.

Parameters:

e - the mouse event

---

mousePressed

```
public void mousePressed(java.awt.event.MouseEvent e)
```

Handles mouse button presses.

Parameters:

e - the mouse event

---

mouseReleased

```
public void mouseReleased(java.awt.event.MouseEvent e)
```

Handles mouse button releases.

Parameters:

e - the mouse event

---

questionMark

```
public void questionMark()
```

Marks the cell with a question mark.

---

reveal

```
public void reveal(MineCellContent content)
```

Reveals the mine cell.

Parameters:

content - the content of this cell

---

setContent

```
public void setContent(MineCellContent content)
```

Sets the content of the panel.

Parameters:

content - the content to be set

---

toggleFlag

```
public MineCellState toggleFlag(MineCellState state,
                                boolean usingQuestionMarks)
```

Toggles between the states of the covered mine cell.

Parameters:

state - the previous state
usingQuestionMarks - specifies whether the question mark option is set

Returns:

the next state

INDEX