**import** java.awt.GridLayout;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.text.SimpleDateFormat;

**import** javax.swing.JButton;

**import** javax.swing.JLabel;

**import** javax.swing.JMenuItem;

**import** javax.swing.JOptionPane;

**import** javax.swing.JPanel;

**public** **class** SudokuUI **extends** javax.swing.JFrame {

**private** JButton[][] buttons;

**private** ActionListener[][] actionListener;

**private** JPanel[][] blocks;

**private** Sudoku sudoku;

**private** **int** gameMode;

**private** **int** grid;

**private** **boolean** paused;

**private** **final** StopWatch stopWatch;

**private** JMenuItem expert;

**public** SudokuUI() {

sudoku = **new** Sudoku();

gameMode = Sudoku.***GAME\_MODE\_MEDIUM***;

grid = Sudoku.***GRID\_9X9***;

stopWatch = **new** StopWatch();

paused = **false**;

initComponents();

initialize();

startTimer();

}

**private** **void** initComponents() {

base = **new** javax.swing.JPanel();

options = **new** javax.swing.JPanel();

newGameBut = **new** javax.swing.JButton();

resetGameBut = **new** javax.swing.JButton();

pause = **new** javax.swing.JButton();

timeLabel = **new** javax.swing.JLabel();

resume = **new** javax.swing.JButton();

submit = **new** javax.swing.JButton();

holder = **new** javax.swing.JPanel();

board = **new** javax.swing.JPanel();

menue = **new** javax.swing.JMenuBar();

game = **new** javax.swing.JMenu();

newGame = **new** javax.swing.JMenuItem();

resetGame = **new** javax.swing.JMenuItem();

jSeparator1 = **new** javax.swing.JPopupMenu.Separator();

exit = **new** javax.swing.JMenuItem();

jMenu2 = **new** javax.swing.JMenu();

beginner = **new** javax.swing.JRadioButtonMenuItem();

intermediator = **new** javax.swing.JRadioButtonMenuItem();

expert = **new** javax.swing.JRadioButtonMenuItem();

help = **new** javax.swing.JMenu();

about = **new** javax.swing.JMenuItem();

JMenuItem h = **new** javax.swing.JMenuItem();

setDefaultCloseOperation(javax.swing.WindowConstants.***EXIT\_ON\_CLOSE***);

setTitle("Sudoku");

setBounds(**new** java.awt.Rectangle(0, 0, 0, 0));

setCursor(**new** java.awt.Cursor(java.awt.Cursor.***DEFAULT\_CURSOR***));

setMinimumSize(**new** java.awt.Dimension(500, 500));

base.setBackground(**new** java.awt.Color(204, 204, 204));

base.setAlignmentX(0.0F);

base.setAlignmentY(0.0F);

options.setBackground(**new** java.awt.Color(255, 255, 255));

newGameBut.setBackground(**new** java.awt.Color(255, 255, 255));

newGameBut.setFont(**new** java.awt.Font("Tahoma", 0, 12));

newGameBut.setText("New Game");

newGameBut.setMargin(**new** java.awt.Insets(0, 0, 0, 0));

newGameBut.setMaximumSize(**new** java.awt.Dimension(63, 19));

newGameBut.setMinimumSize(**new** java.awt.Dimension(63, 19));

newGameBut.setPreferredSize(**new** java.awt.Dimension(63, 19));

newGameBut.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

newGameButActionPerformed(evt);

}

});

resetGameBut.setBackground(**new** java.awt.Color(255, 255, 255));

resetGameBut.setFont(**new** java.awt.Font("Tahoma", 0, 12));

resetGameBut.setText("Reset Game");

resetGameBut.setMargin(**new** java.awt.Insets(0, 0, 0, 0));

resetGameBut.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

resetGameButActionPerformed(evt);

}

});

pause.setBackground(**new** java.awt.Color(255, 255, 255));

pause.setFont(**new** java.awt.Font("Tahoma", 0, 12));

pause.setText("Pause");

pause.setMargin(**new** java.awt.Insets(0, 0, 0, 0));

pause.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

pauseActionPerformed(evt);

}

});

timeLabel.setFont(**new** java.awt.Font("Tahoma", 0, 24));

timeLabel.setForeground(**new** java.awt.Color(51, 51, 51));

timeLabel.setHorizontalAlignment(javax.swing.SwingConstants.***RIGHT***);

timeLabel.setText("00:00:000");

timeLabel.setIconTextGap(0);

resume.setBackground(**new** java.awt.Color(255, 255, 255));

resume.setFont(**new** java.awt.Font("Tahoma", 0, 12));

resume.setText("Resume");

resume.setMargin(**new** java.awt.Insets(0, 0, 0, 0));

resume.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

resumeActionPerformed(evt);

}

});

submit.setBackground(**new** java.awt.Color(255, 255, 255));

submit.setFont(**new** java.awt.Font("Tahoma", 0, 12));

submit.setText("Submit");

submit.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

submitActionPerformed(evt);

}

});

javax.swing.GroupLayout optionsLayout = **new** javax.swing.GroupLayout(options);

options.setLayout(optionsLayout);

optionsLayout.setHorizontalGroup(optionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.***LEADING***)

.addGroup(optionsLayout.createSequentialGroup()

.addComponent(newGameBut, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 91, Short.***MAX\_VALUE***)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.***RELATED***)

.addComponent(resetGameBut, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 99, Short.***MAX\_VALUE***)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.***RELATED***)

.addComponent(pause, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 66, Short.***MAX\_VALUE***)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.***RELATED***)

.addComponent(resume, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 75, Short.***MAX\_VALUE***)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.***RELATED***)

.addComponent(submit, javax.swing.GroupLayout.***PREFERRED\_SIZE***, 82,

javax.swing.GroupLayout.***PREFERRED\_SIZE***)

.addGap(18, 18, 18).addComponent(timeLabel).addContainerGap()));

optionsLayout.setVerticalGroup(optionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.***LEADING***)

.addGroup(optionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.***BASELINE***)

.addComponent(newGameBut, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 25, Short.***MAX\_VALUE***)

.addComponent(resetGameBut, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 25, Short.***MAX\_VALUE***)

.addComponent(pause, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 25, Short.***MAX\_VALUE***)

.addComponent(resume, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 25, Short.***MAX\_VALUE***)

.addComponent(submit, javax.swing.GroupLayout.***PREFERRED\_SIZE***, 25,

javax.swing.GroupLayout.***PREFERRED\_SIZE***))

.addComponent(timeLabel, javax.swing.GroupLayout.Alignment.***TRAILING***,

javax.swing.GroupLayout.***PREFERRED\_SIZE***, 25, Short.***MAX\_VALUE***));

holder.setBackground(**new** java.awt.Color(255, 255, 255));

holder.setAlignmentX(0.0F);

holder.setAlignmentY(0.0F);

holder.setLayout(**new** java.awt.GridLayout(1, 1));

board.setBackground(**new** java.awt.Color(255, 255, 255));

board.setAlignmentX(0.0F);

board.setAlignmentY(0.0F);

board.setMinimumSize(**new** java.awt.Dimension(100, 100));

board.setPreferredSize(**new** java.awt.Dimension(100, 100));

board.setLayout(**new** java.awt.GridLayout(9, 9));

holder.add(board);

javax.swing.GroupLayout baseLayout = **new** javax.swing.GroupLayout(base);

base.setLayout(baseLayout);

baseLayout.setHorizontalGroup(baseLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.***LEADING***)

.addGroup(javax.swing.GroupLayout.Alignment.***TRAILING***,

baseLayout.createSequentialGroup().addContainerGap()

.addGroup(baseLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.***TRAILING***)

.addComponent(holder, javax.swing.GroupLayout.Alignment.***LEADING***,

javax.swing.GroupLayout.***DEFAULT\_SIZE***, 572, Short.***MAX\_VALUE***)

.addComponent(options, javax.swing.GroupLayout.Alignment.***LEADING***,

javax.swing.GroupLayout.***DEFAULT\_SIZE***,

javax.swing.GroupLayout.***DEFAULT\_SIZE***, Short.***MAX\_VALUE***))

.addContainerGap()));

baseLayout.setVerticalGroup(baseLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.***LEADING***)

.addGroup(baseLayout.createSequentialGroup().addContainerGap()

.addComponent(options, javax.swing.GroupLayout.***PREFERRED\_SIZE***,

javax.swing.GroupLayout.***DEFAULT\_SIZE***, javax.swing.GroupLayout.***PREFERRED\_SIZE***)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.***UNRELATED***)

.addComponent(holder, javax.swing.GroupLayout.***DEFAULT\_SIZE***, 534, Short.***MAX\_VALUE***)

.addContainerGap()));

game.setText("Game");

newGame.setText("New Game 9 X 9");

newGame.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

newGameActionPerformed(evt);

}

});

exit.setText("Exit");

exit.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

exitActionPerformed(evt);

}

});

game.add(newGame);

game.add(exit);

menue.add(game);

jMenu2.setText("Level");

beginner.setAccelerator(

javax.swing.KeyStroke.*getKeyStroke*(java.awt.event.KeyEvent.***VK\_B***, java.awt.event.InputEvent.***CTRL\_MASK***));

beginner.setText("Beginner");

beginner.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

beginnerActionPerformed(evt);

}

});

jMenu2.add(beginner);

intermediator.setAccelerator(

javax.swing.KeyStroke.*getKeyStroke*(java.awt.event.KeyEvent.***VK\_I***, java.awt.event.InputEvent.***CTRL\_MASK***));

intermediator.setSelected(**true**);

intermediator.setText("Intermediator");

intermediator.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

intermediatorActionPerformed(evt);

}

});

jMenu2.add(intermediator);

expert.setAccelerator(

javax.swing.KeyStroke.*getKeyStroke*(java.awt.event.KeyEvent.***VK\_E***, java.awt.event.InputEvent.***CTRL\_MASK***));

expert.setText("expert");

expert.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

expertActionPerformed(evt);

}

});

jMenu2.add(expert);

menue.add(jMenu2);

help.setText("Help");

h.setText("How to play");

about.setText("Authors");

h.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

helpActionPerformed(evt);

}

});

about.addActionListener(**new** java.awt.event.ActionListener() {

**public** **void** actionPerformed(java.awt.event.ActionEvent evt) {

aboutActionPerformed(evt);

}

});

help.add(h);

help.add(about);

menue.add(help);

setJMenuBar(menue);

javax.swing.GroupLayout layout = **new** javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.***LEADING***).addComponent(

base, javax.swing.GroupLayout.***DEFAULT\_SIZE***, javax.swing.GroupLayout.***DEFAULT\_SIZE***, Short.***MAX\_VALUE***));

layout.setVerticalGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.***LEADING***).addComponent(base,

javax.swing.GroupLayout.***DEFAULT\_SIZE***, javax.swing.GroupLayout.***DEFAULT\_SIZE***, Short.***MAX\_VALUE***));

pack();

}

**private** **void** helpActionPerformed(java.awt.event.ActionEvent evt) {

stopWatch.pause();

JOptionPane.*showMessageDialog*(**this**,

"A standard Sudoku puzzle consists of a grid of 9 blocks. Each block contains 9 boxes arranged in 3 rows and 3 columns.\n There is only one valid solution to each Sudoku puzzle. The only way the puzzle can be considered solved correctly is when all 81 boxes contain numbers and the other Sudoku rules have been followed.\r\n"

+ "\r\n"

+ "· When you start a game of Sudoku, some blocks will be pre-filled for you. You can change these numbers in the course of the game.\r\n"

+ "\r\n"

+ "·Each column must contain all of the numbers 1 through 9 and no two numbers in the same column of a Sudoku puzzle can be the same.\r\n"

+ "\r\n"

+ "· Each row must contain all of the numbers 1 through 9 and no two numbers in the same row of a Sudoku puzzle can be the same.\r\n"

+ "\r\n"

+ "· Each block must contain all of the numbers 1 through 9 and no two numbers in the same block of a Sudoku puzzle can be the same. ");

stopWatch.resume();

}

**private** **void** newGameActionPerformed(java.awt.event.ActionEvent evt) {

grid = Sudoku.***GRID\_9X9***;

**int**[][] puzzle = sudoku.getNewPuzzle(grid, gameMode);

createBoard(puzzle);

}

**private** **void** resetGameActionPerformed(java.awt.event.ActionEvent evt) {

**int**[][] puzzle = sudoku.resetPuzzle();

createBoard(puzzle);

}

**private** **void** intermediatorActionPerformed(java.awt.event.ActionEvent evt) {

gameMode = Sudoku.***GAME\_MODE\_MEDIUM***;

**int**[][] puzzle = sudoku.getNewPuzzle(grid, gameMode);

createBoard(puzzle);

}

**private** **void** expertActionPerformed(java.awt.event.ActionEvent evt) {

gameMode = Sudoku.***GAME\_MODE\_EXPERT***;

**int**[][] puzzle = sudoku.getNewPuzzle(grid, gameMode);

createBoard(puzzle);

}

**private** **void** exitActionPerformed(java.awt.event.ActionEvent evt) {

System.*exit*(0);

}

**private** **void** beginnerActionPerformed(java.awt.event.ActionEvent evt) {

gameMode = Sudoku.***GAME\_MODE\_EASY***;

**int**[][] puzzle = sudoku.getNewPuzzle(grid, gameMode);

createBoard(puzzle);

}

**private** **void** pauseActionPerformed(java.awt.event.ActionEvent evt) {

stopWatch.pause();

showMessage("Paused");

}

**private** **void** resumeActionPerformed(java.awt.event.ActionEvent evt) {

stopWatch.resume();

holder.removeAll();

holder.add(board);

holder.repaint();

**this**.setVisible(**true**);

}

**private** **void** resetGameButActionPerformed(java.awt.event.ActionEvent evt) {

**int**[][] puzzle = sudoku.resetPuzzle();

createBoard(puzzle);

}

**private** **void** newGameButActionPerformed(java.awt.event.ActionEvent evt) {

**int**[][] puzzle = sudoku.getNewPuzzle(grid, gameMode);

createBoard(puzzle);

}

**private** **void** submitActionPerformed(java.awt.event.ActionEvent evt) {

**if** (!isAnsComplete()) {

JOptionPane.*showMessageDialog*(**this**, "Please complete your answer.You can see rules in the help section.");

} **else** {

stopWatch.stop();

**boolean** isAnsCorrect = sudoku.check(getAns());

String messageStr = "";

**if** (isAnsCorrect) {

messageStr = "Congratulation You have won the Game in " + timeLabel.getText();

} **else** {

messageStr = "Sorry You have failed.You can see rules in the help section. ";

}

showMessage(messageStr);

}

}

**private** **void** aboutActionPerformed(java.awt.event.ActionEvent evt) {

stopWatch.pause();

JOptionPane.*showMessageDialog*(**this**, "Version: 1.0.0\nGame made by: Rohan,Ravi,Rushabh,Tushar\nInstitute: VJTI,Matunga ");

stopWatch.resume();

}

**private** **void** createBoard(**int**[][] puzzle) {

board.removeAll();

grid = puzzle.length;

blocks = **new** JPanel[grid][grid];

buttons = **new** JButton[grid][grid];

actionListener = **new** ActionListener[grid][grid];

board.setLayout(**new** GridLayout(grid, grid, 3, 3));

**int** rowsInGrid = grid == 9 ? 3 : 2;

**for** (**int** i = 0; i < grid; i++) {

**for** (**int** j = 0; j < grid; j++) {

blocks[i][j] = **new** JPanel();

buttons[i][j] = **new** JButton();

String text = "";

**if** (0 < puzzle[i][j] && puzzle[i][j] <= grid) {

text += puzzle[i][j];

} **else** {

**final** JButton tempbutton = buttons[i][j];

**final** JPanel tempBlock = blocks[i][j];

actionListener[i][j] = **new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

viewInputs(tempBlock, tempbutton, grid);

}

};

buttons[i][j].addActionListener(actionListener[i][j]);

}

buttons[i][j].setText(text);

buttons[i][j].setFont(**new** java.awt.Font("Tahoma", 0, 24));

**if** (((0 <= i && i < rowsInGrid) || (rowsInGrid \* 2 <= i && i < grid)) && (3 <= j && j < 6)) {

buttons[i][j].setBackground(**new** java.awt.Color(204, 204, 204));

} **else** **if** ((rowsInGrid <= i && i < rowsInGrid \* 2) && ((0 <= j && j < 3) || (6 <= j && j < 9))) {

buttons[i][j].setBackground(**new** java.awt.Color(204, 204, 204));

} **else** {

buttons[i][j].setBackground(**new** java.awt.Color(255, 255, 255));

}

blocks[i][j].setLayout(**new** GridLayout(1, 1));

blocks[i][j].add(buttons[i][j]);

board.add(blocks[i][j]);

}

}

holder.removeAll();

holder.add(board);

board.repaint();

holder.repaint();

**this**.setVisible(**true**);

stopWatch.start();

}

**private** **void** initialize() {

**int**[][] puzzle = sudoku.getNewPuzzle(grid, gameMode);

createBoard(puzzle);

}

**private** **void** viewInputs(JPanel block, JButton inputButtton, **int** numOfInput) {

JPanel inputs = **new** Inputs(**this**, block, inputButtton, numOfInput);

block.remove(inputButtton);

block.add(inputs);

**this**.setVisible(**true**);

}

**public** **void** setInput(String ans, JPanel block, JButton inputButtton) {

block.removeAll();

inputButtton.setText(ans);

inputButtton.setFont(**new** java.awt.Font("Tahoma", 1, 24));

block.add(inputButtton);

**this**.repaint();

}

**private** **int**[][] getAns() {

**int** ans[][] = **new** **int**[grid][grid];

**for** (**int** i = 0; i < grid; i++) {

**for** (**int** j = 0; j < grid; j++) {

**try** {

ans[i][j] = Integer.*parseInt*(buttons[i][j].getText());

} **catch** (NumberFormatException e) {

ans[i][j] = 0;

}

}

}

**return** ans;

}

**private** **boolean** isAnsComplete() {

**boolean** isAnsComplete = **true**;

**for** (**int** i = 0; i < grid; i++) {

**for** (**int** j = 0; j < grid; j++) {

**try** {

Integer.*parseInt*(buttons[i][j].getText());

} **catch** (NumberFormatException e) {

isAnsComplete = **false**;

**break**;

}

}

}

**return** isAnsComplete;

}

**private** **void** showMessage(String message) {

JLabel messageLabel = **new** JLabel();

messageLabel.setFont(**new** java.awt.Font("Tahoma", 1, 20));

messageLabel.setHorizontalAlignment(javax.swing.SwingConstants.***CENTER***);

messageLabel.setText(message);

holder.removeAll();

holder.add(messageLabel);

holder.repaint();

**this**.setVisible(**true**);

}

**private** **void** startTimer() {

Thread thread = **new** Thread(**new** Runnable() {

**public** **void** run() {

stopWatch.start();

**while** (**true**) {

**if** (!paused) {

**final** String timeString = **new** SimpleDateFormat("mm:ss:SSS").format(stopWatch.getElapsedTime());

timeLabel.setText("" + timeString);

}

}

}

});

thread.start();

}

**private** javax.swing.JMenuItem about;

**private** javax.swing.JPanel base;

**private** javax.swing.JRadioButtonMenuItem beginner;

**private** javax.swing.JPanel board;

**private** javax.swing.JMenuItem exit;

**private** javax.swing.JMenu game;

**private** javax.swing.JMenu help;

**private** javax.swing.JPanel holder;

**private** javax.swing.JRadioButtonMenuItem intermediator;

**private** javax.swing.JMenu jMenu2;

**private** javax.swing.JPopupMenu.Separator jSeparator1;

**private** javax.swing.JMenuBar menue;

**private** javax.swing.JMenuItem newGame;

**private** javax.swing.JMenuItem newGame6X6;

**private** javax.swing.JButton newGameBut;

**private** javax.swing.JPanel options;

**private** javax.swing.JButton pause;

**private** javax.swing.JMenuItem resetGame;

**private** javax.swing.JButton resetGameBut;

**private** javax.swing.JButton resume;

**private** javax.swing.JButton submit;

**private** javax.swing.JLabel timeLabel;

}