```
/*********************************************************************
 *
 * File:        MagicSquareGUI.java
 *
 * Author:      Dan Gerstl
 *
 * Date:        05/19/2018
 *
 * Purpose:     Project 02
 *
 * Description: GUI which presents JTextFields to fill in as well as
 *              allowing the user to open from a file to fill the
 *              TextFields with a possible Magic Square.
 *
 * Comment:     NA
 *
 *********************************************************************/

 import java.util.*;
 import java.lang.*;
 import java.awt.*;
 import java.awt.event.*;
 import javax.swing.*;
 import java.io.*;

 public class MagicSquareGUI implements ActionListener
 {
     /*** Class Constants ***/

     private final int SMALLEST_SQUARE = 3;

     /*** Class Variables ***/

     private JButton jbuCheckSquare = null;
     private JButton jbuNextSquare  = null;
     private JButton jbuClear       = null;
     private JButton jbuExit        = null;
     private JButton jbuBrowse      = null;

     private JTextField jteFileName = null;

     private IntegerTextField[][] squareTextFields =
             new IntegerTextField[9][9];

     private JComboBox jcoSizeSelect = null;

     private JPanel jpaMagicSquare = null;

     private Scanner inStream = null;

     /*** Constructor ***/

     public MagicSquareGUI(String title)
     {
         /*** Local Variables ***/

         JFrame jfrMagicSquare = null;

         Container c = null;

         /*** Instantiate Objects ***/

         jfrMagicSquare = new JFrame(title);
```

```
        /*** Get JFrame Address ***/

        c = jfrMagicSquare.getContentPane();

        /*** Set attributes for containers ***/

        setJFrameAttributes(jfrMagicSquare);
        c.setBackground(Color.lightGray);

        /*** Build GUI ***/

        buildGUI(c);
    }

    /*** GUI Methods ***/

    private void setJFrameAttributes(JFrame myGUI)
    {
        /*** Set Size ***/

        myGUI.setSize(475, 500);

        /*** Set Colors ***/

        myGUI.setBackground(Color.lightGray);

        /*** Set Layout ***/

        myGUI.setLayout(new GridBagLayout());

        myGUI.setVisible(true);
    }

    private void buildGUI(Container c)
    {
        /*** Local Constants ***/

        final int DEFAULT_SQUARE_SIZE = SMALLEST_SQUARE;

        /*** Local Variables ***/

        JPanel jpaSelection   = null;
        JPanel jpaButtons     = null;
        JPanel jpaOutput      = null;

        GridBagConstraints constraints = null;

        /*** Instantiate Components ***/

        jpaSelection   = createSelectionPanel();
        jpaButtons     = createButtonPanel();
        jpaMagicSquare = createMagicSquarePanel(DEFAULT_SQUARE_SIZE);

        constraints = new GridBagConstraints();

        /*** Set Constraints ***/

        constraints.insets  = new Insets(2,5,2,5);
        constraints.anchor  = GridBagConstraints.FIRST_LINE_START;
        constraints.fill    = GridBagConstraints.BOTH;
        constraints.gridx   = 0;
        constraints.gridy   = 0;
        constraints.weightx = 0.5;
        constraints.weighty = 0.0;
```

```
        /*** Add components to container ***/

        c.add(jpaSelection, constraints);

        constraints.gridy = 1;
        c.add(jpaButtons, constraints);

        constraints.gridy   = 2;
        constraints.weighty = 1.0;
        c.add(jpaMagicSquare, constraints);
    }

    /*** JPanel Creation Methods ***/

    private JPanel createSelectionPanel()
    {
        /*** Local Constants ***/

        final String[] SQUARE_SIZES = {String.valueOf(SMALLEST_SQUARE), "4",
                                       "5", "6", "7", "8", "9"};

        /*** Local Variables ***/

        JPanel jpaSelection = null;

        JLabel jlaSquareSize = null;
        JLabel jlaFileSelect = null;

        ImageIcon browse = null;

        /*** Instantiate Container ***/

        jpaSelection = new JPanel();

        /*** Set Container Attributes ***/

        jpaSelection.setBackground(Color.lightGray);
        jpaSelection.setLayout(new FlowLayout());

        /*** Instantiate Components ***/

        jlaSquareSize = new JLabel("Square size:");
        jlaFileSelect = new JLabel("File:");

        jcoSizeSelect = new JComboBox<>(SQUARE_SIZES);
        jcoSizeSelect.setEditable(false);
        jcoSizeSelect.addActionListener(this);

        jteFileName = new JTextField(20);

        browse    = new ImageIcon("Open 16x16.png");
        jbuBrowse = new JButton(browse);
        jbuBrowse.addActionListener(this);

        /*** Add components to container ***/

        jpaSelection.add(jlaSquareSize);
        jpaSelection.add(jcoSizeSelect);

        jpaSelection.add(jlaFileSelect);
        jpaSelection.add(jteFileName);
        jpaSelection.add(jbuBrowse);
```

```
        return jpaSelection;
    }

    private JPanel createButtonPanel()
    {
        /*** Local Variables ***/

        JPanel jpaButtons = null;

        /*** Instantiate Container ***/

        jpaButtons = new JPanel();

        /*** Set Container Attributes ***/

        jpaButtons.setBackground(Color.lightGray);
        jpaButtons.setLayout(new FlowLayout());

        /*** Instantiate Components ***/

        jbuCheckSquare = new JButton("Check square");
        jbuCheckSquare.addActionListener(this);

        jbuNextSquare  = new JButton("Next square");
        jbuNextSquare.addActionListener(this);

        jbuClear       = new JButton("Clear");
        jbuClear.addActionListener(this);

        jbuExit        = new JButton("Exit");
        jbuExit.addActionListener(this);

        /*** Add components to container ***/

        jpaButtons.add(jbuCheckSquare);
        jpaButtons.add(jbuNextSquare);
        jpaButtons.add(jbuClear);
        jpaButtons.add(jbuExit);

        return jpaButtons;
    }

    private JPanel createMagicSquarePanel(int squareSize)
    {
        /*** Local Variables ***/

        JPanel jpaMagicSquare = null;

        Font bigFont = null;

        /*** Instantiate Container ***/

        jpaMagicSquare = new JPanel();

        /*** Set Container Attributes ***/

        jpaMagicSquare.setBackground(Color.lightGray);
        jpaMagicSquare.setLayout(new GridBagLayout());

        /*** Instantiate Components ***/

        bigFont = new Font("Courier New", Font.BOLD, 20);

        for (int i = 0; i < squareTextFields.length; i++)
```

```
        {
            for (int j = 0; j < squareTextFields.length; j++)
            {
                squareTextFields[i][j] = new IntegerTextField(3, 1,
                                                (squareSize * squareSize));

                squareTextFields[i][j].setHorizontalAlignment
                                        (JTextField.CENTER);

                squareTextFields[i][j].setFont(bigFont);
            }
        }

        /*** Create square ***/

        jpaMagicSquare = createMagicSquare(jpaMagicSquare, squareSize);

        return jpaMagicSquare;
    }

    private JPanel createMagicSquare(JPanel squarePanel, int squareSize)
    {
        /*** Local Variables ***/

        GridBagConstraints constraints = null;

        /*** Instantiate Objects ***/

        constraints = new GridBagConstraints();

        /*** Set Constraints ***/

        constraints.insets  = new Insets(5,3,5,3);
        constraints.anchor  = GridBagConstraints.FIRST_LINE_START;
        constraints.fill    = GridBagConstraints.BOTH;

        /*** Add components to container ***/

        for (int i = 0; i < squareSize; i++)
        {
            for (int j = 0; j < squareSize; j++)
            {
                /*** Update TextField maximum ***/

                squareTextFields[i][j].setMaximum(squareSize * squareSize);

                /*** Set component location on grid ***/

                constraints.gridy = i;
                constraints.gridx = j;

                /*** Add component to panel ***/

                squarePanel.add(squareTextFields[i][j], constraints);
            }
        }
        return squarePanel;
    }

    /*** Action Event ***/

    public void actionPerformed( ActionEvent e)
    {
        if (e.getSource() == jcoSizeSelect)
```

```
        {
            processSizeChange();
        }

        else if (e.getSource() == jbuBrowse)
        {
            processBrowse();
        }

        else if (e.getSource() == jbuCheckSquare)
        {
            processCheckSquare();
        }

        else if (e.getSource() == jbuNextSquare)
        {
            processNextSquare(jteFileName.getText());
        }

        else if (e.getSource() == jbuClear)
        {
            processClear();
        }

        else if (e.getSource() == jbuExit)
        {
            System.exit(0);
        }
    }

    /*** Action Event Methods ***/

    private void processSizeChange()
    {
        /*** Local Variables ***/

        int userSelection = 0;

        /*** Get user selection ***/

        userSelection = Integer.parseInt(jcoSizeSelect.getSelectedItem()
                        .toString());

        /*** Clear JPanel ***/

        jpaMagicSquare.removeAll();

        /*** Get new square ***/

        jpaMagicSquare = createMagicSquare(jpaMagicSquare, userSelection);

        /*** Update JPanel ***/

        jpaMagicSquare.revalidate();

        /*** Redraw JPanel ***/

        jpaMagicSquare.repaint();
    }

    private void processBrowse()
    {
        /*** Local Variables ***/
```

```
        File file = null;

        /*** Instantiate Objects ***/

        JFileChooser jfcChooser = new JFileChooser();

        /*** Set location ***/

        jfcChooser.setCurrentDirectory(new File("."));

        /*** Get button selection ***/

        int returnValue = jfcChooser.showOpenDialog(null);

        /*** Get chosen file if open was selected ***/

        if (returnValue == JFileChooser.APPROVE_OPTION)
        {
            /*** Get chosen file ***/

            file = jfcChooser.getSelectedFile();

            /*** Update filename in TextField ***/

            jteFileName.setText(file.getName());

            /*** Open file in Scanner ***/

            try
            {
                inStream = new Scanner(file);
            }

            catch (FileNotFoundException e)
            {
                displayMessage("File <" + file.getName() + "> not found. " +
                               "Please select a valid file.");
            }
        }
    }

    private void processCheckSquare()
    {
        /*** Local Variables ***/

        int squareSize = 0;

        /*** Reset TextField colors ***/

        MagicSquare.setTextFieldArrayColor(squareTextFields, Color.WHITE);

        /*** Get current square size ***/

        squareSize = Integer.parseInt(jcoSizeSelect.getSelectedItem()
                     .toString());
        try
        {
            MagicSquare.checkIfMagicSquare(squareTextFields, squareSize);
        }

        catch (IntegerUserInputException e)
        {
            displayMessage(e.getMessage());
        }
```

```
        catch (Exception e)
        {
            displayMessage(e.getMessage());
        }
    }

    private void processNextSquare(String fileName)
    {
        /*** Local Variables ***/

        String[] record = null;

        String temp = null;

        /*** Reset TextField colors ***/

        MagicSquare.setTextFieldArrayColor(squareTextFields, Color.WHITE);

        /*** Clear TextFields ***/

        for (int i = 0; i < squareTextFields.length; i++)
        {
            for (int j = 0; j < squareTextFields.length; j++)
            {
                squareTextFields[i][j].setText("");
            }
        }

        try
        {
            /*** Ensure has open file ***/

            getFileFromTextField();

            if (inStream.hasNextLine())
            {
                temp = inStream.nextLine();

                /*** Remove extra spaces ***/

                temp = temp.trim();

                while (temp.indexOf("  ") != -1)
                {
                    temp = temp.replace("  ", " ");
                }

                /*** Place string in array ***/

                record = temp.split(" ");

                /*** Echo string array into TextFields ***/

                stringToTextFieldArray(record, squareTextFields);

                /*** Check Square ***/

                processCheckSquare();
            }

            else if (!inStream.hasNextLine())
            {
                displayMessage("End of file reached. Selecting \"Next " +
```

```
                                "square\"" + " will restart at the " +
                                "beginning of the file.");

                /*** Clear Scanner ***/

                inStream = null;

                /*** Restart file ***/

                getFileFromTextField();
            }
        }

        catch (NullPointerException e)
        {
            displayMessage("No file selected. Please select a file to " +
                           "process.");

            processBrowse();
        }
    }

    private void processClear()
    {
        /*** Clear Scanner ***/

        inStream = null;

        /*** Clear TextFields ***/

        jteFileName.setText("");

        /*** Reset TextField Colors ***/

        MagicSquare.setTextFieldArrayColor(squareTextFields, Color.WHITE);

        /*** Clear Array ***/

        for (int i = 0; i < squareTextFields.length; i++)
        {
            for (int j = 0; j < squareTextFields.length; j++)
            {
                squareTextFields[i][j].setText("");
            }
        }
    }

    /*** Helper Methods ***/

    private void stringToTextFieldArray(String[] strArray,
                                        JTextField[][] jteArray)
    {
        /*** Local Variables ***/

        int counter = 0;

        int squareSize = 0;

        /*** Determine square size from array ***/

        squareSize = (int)Math.ceil(Math.sqrt(strArray.length));

        try
        {
```

```
            /*** Clear JPanel ***/

            jpaMagicSquare.removeAll();

            /*** Update square size from array ***/

            jpaMagicSquare = createMagicSquare(jpaMagicSquare, squareSize);

            /*** Update JPanel ***/

            jpaMagicSquare.revalidate();

            /*** Redraw JPanel ***/

            jpaMagicSquare.repaint();

            /*** Update ComboBox to reflect new square size ***/

            jcoSizeSelect.setSelectedIndex(squareSize - SMALLEST_SQUARE);

            /*** Trim string array elements ***/

            for (int i = 0; i < strArray.length; i++)
            {
                strArray[i] = strArray[i].trim();
            }

            try
            {
                /*** Place string array into JTextField array ***/

                while (counter < strArray.length)
                {
                    for (int i = 0; i < squareSize; i++)
                    {
                        for (int j = 0; j < squareSize; j++)
                        {
                            jteArray[i][j].setText(strArray[counter]);
                            counter++;
                        }
                    }
                }
            }

            catch (ArrayIndexOutOfBoundsException e)
            {
                displayMessage("Record does not contain a full square.");
            }
        }

        catch (ArrayIndexOutOfBoundsException e)
        {
            displayMessage("Provided file has too large of a record. " +
                           "Please select a new file.");

            /*** Clear GUI ***/

            processClear();

            /*** Open browse window for user ***/

            processBrowse();
        }
    }
```

```
    public void displayMessage(String message)
    {
        JOptionPane.showMessageDialog(null, message);
    }

    private void getFileFromTextField()
    {
        /*** Local Variables ***/

        File file = null;

        /*** Check if file has already been opened ***/

        if (inStream == null)
        {
            try
            {
                /*** Get file ***/

                file = new File(jteFileName.getText());

                /*** Open file ***/

                inStream = new Scanner(file);
            }

            catch (FileNotFoundException e)
            {
                displayMessage("File <" + jteFileName.getText() +
                               "> is not valid. Please select a valid " +
                               "file.");

                processBrowse();
            }
        }
    }
}
```