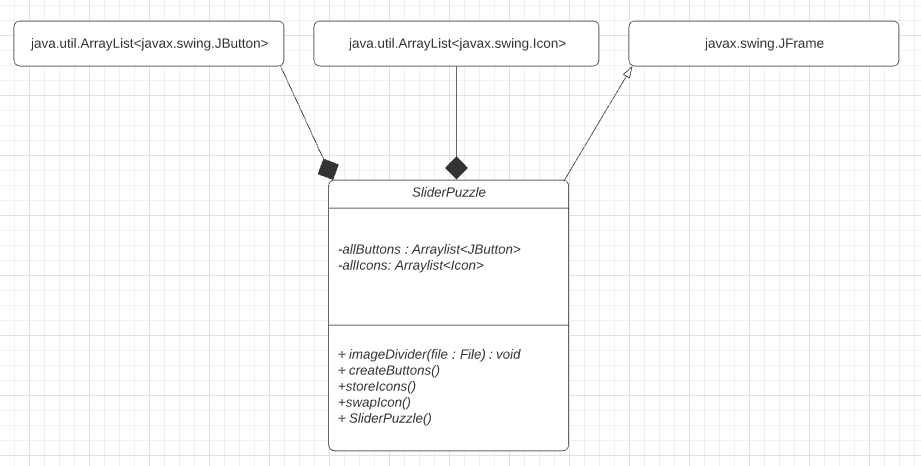
**Slider Puzzle Class Diagram:**



**Slider Puzzle Design Pseudocode:**

public class SliderPuzzle extends JFrame {

Initialize "allicons" as new arraylist to store icons

Initialize "allButtons" as new arraylist to store buttons

// Constructor for SliderPuzzle class

public SliderPuzzle(){

// call method that splits the main image into 12 jpg files

imageDivider(new File("PuzzleName.jpg"));

// call method that stores icons

storeButtons();

//call method that creates buttons

createButtons();

//Initialize 11 image icons that will be used as tiles

Icon image1 through image12 = new ImageIcon("TileName.jpg");

// Initialize final image icons that will be used for hints

Icon finalPictureIcon0 = new ImageIcon("Puzzle1.jpg");

Icon finalPictureIcon1 = new ImageIcon("Puzzle2.jpg");

Icon finalPictureIcon2 = new ImageIcon("Puzzle3.jpg");

// Create main and sub panels that will be added to frame

JPanel backGroundPanel = new JPanel();

JPanel subPanel = new JPanel();

JPanel mainPanel = new JPanel();

JPanel puzzleChooserPanel = new JPanel();

JPanel finalImageSubPanel = new JPanel();

// Create JButtons that will display image icons to form puzzle

JButton jButton1 through jButton12 = new JButton(image1 through 12);

// JLabel used for choosing puzzle

JLabel newPuzzleJLabel = new JLabel("Choose a Puzzle");

// JButtons for choosing new puzzle

JButton newPuzzleButton1 = new JButton("Puzzle 1");

JButton newPuzzleButton2 = new JButton("Puzzle 2");

JButton newPuzzleButton3 = new JButton("Puzzle 3");

// JLabel and JButtons that displays puzzle hints when selected

JLabel finalPictureLabel = new JLabel("Puzzle Hints:");

JButton finalImageJButton = new JButton("Final Image 1");

JButton finalImageJButton1 = new JButton("Final Image 2");

JButton finalImageJButton2 = new JButton("Final Image 3");

// JLabels used to show which icon has the power to swap with adjacent icon

JLabel jLabelPowerIcon = new JLabel(image12);

JLabel powerIconDefinitionLabel = new JLabel("This image will swap with the adjacent icon:");

// Set layouts for panels

subPanel.setLayout(new BoxLayout(subPanel, BoxLayout.X\_AXIS));

backGroundPanel.setLayout(new BoxLayout(backGroundPanel, BoxLayout.Y\_AXIS));

mainPanel.setPreferredSize(new Dimension(524, 522));

mainPanel.setLayout(new GridLayout(4,3,0,0));

setSize(new Dimension(524,522));

// Add image tiles to main panel

mainpanel.add(jButton1 through jButton12);

// Add power icon definition labels

subPanel.add(powerIconDefinitionLabel);

subPanel.add(jLabelPowerIcon);

// Add JLabel and JButtons to puzzle chooser panel

puzzleChooserPanel.add(newPuzzleJLabel);

puzzleChooserPanel.add(newPuzzleButton1);

puzzleChooserPanel.add(newPuzzleButton2);

puzzleChooserPanel.add(newPuzzleButton3);

// Add JLabel and JButtons to final image sub panel

finalImageSubPanel.add(finalPictureLabel);

finalImageSubPanel.add(finalImageJButton);

finalImageSubPanel.add(finalImageJButton1);

finalImageSubPanel.add(finalImageJButton2);

// Add panels

backGroundPanel.add(mainPanel);

backGroundPanel.add(subPanel);

backGroundPanel.add(puzzleChooserPanel);

backGroundPanel.add(finalImageSubPanel);

add(backGroundPanel);

}

// Add action listeners to puzzle hint JButtons

finalImageJButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

JOptionPane.showMessageDialog(null,"", "Hint",

JOptionPane.INFORMATION\_MESSAGE, finalPictureIcon0);

}

}) ;

finalImageJButton1.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

JOptionPane.showMessageDialog(null,"","Hint",

JOptionPane.INFORMATION\_MESSAGE, finalPictureIcon1);

}

});

finalImageJButton2.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

JOptionPane.showMessageDialog(null,"","Hint",

JOptionPane.INFORMATION\_MESSAGE, finalPictureIcon2);

}

});

pack();

setResizable(false);

setLocationRelativeTo(null);

setVisible(true);

/\*\*

\* Stores all icons for puzzle in an arraylist

\*/

public void storeIcons(){

for(int i = 0; i < 12; i++){

Icon icon = new ImageIcon("Name" + i + ".jpg");

allIcons.add(icon);

}

}

/\*\*

\* Generates 12 JButtons

\*/

public void createButtons(){

for (int i = 0; i < 12; i++){

new JButton(allIcons.get(i));

}

Collections.shuffle(allIcons);

for(int j = 0; j < 12; j++){

mainPanel.add(allButtons.get(j));

}

}

/\*\*

\* Splits the complete image into separate tiles

\* @param file to split

\*/

public void imageDivider(File file) {

try {

FileInputStream fileInputStream = new FileInputStream(file);

BufferedImage image = ImageIO.read(fileInputStream);

int rows = 4;

int cols = 3;

int tiles = rows \* cols;

// Calculate tile width and height

int tileWidth = image.getWidth() / cols;

int tileHeight = image.getHeight() / rows;

int count = 0;

// This array stores the image tiles

BufferedImage images[] = new BufferedImage[tiles];

for (int x = 0; x < rows; x++) {

for (int y = 0; y < cols; y++) {

images[count] = new BufferedImage(tileWidth, tileHeight, image.getType());

Graphics2D gr = images[count++].createGraphics();

gr.drawImage(image, 0, 0, tileWidth, tileHeight, tileWidth \* y, tileHeight \* x,

tileWidth \* y + tileWidth, tileHeight \* x + tileHeight, null);

gr.dispose();

}

}

// Create new files for each puzzle piece 1 through 12

for (int i = 0; i < images.length; i++) {

ImageIO.write(images[i], "jpg", new File("Puzzle Piece Names" + i + ".jpg"));

}

// Display message if file is not found in the working directory

} catch (IOException e) {

JOptionPane.showMessageDialog(null, e.getMessage());

}

}

public static void main(String[] args) {

ImagePuzzle imagePuzzle = new ImagePuzzle();

imagePuzzle.setDefaultCloseOperation(EXIT\_ON\_CLOSE);

}

}