|  |
| --- |
| **개발자 매뉴얼** |
|  |

**1.제목**

|  |
| --- |
| 캐릭터 조각 퍼즐 |

**2 Image Panel**

|  |
| --- |
| **import** javax.swing.\*;  **import** java.awt.\*;  **class** ImagePanel **extends** JPanel {  **private** **int**[][] \_cells = { //타일의 초기 상태  {0, 4, 5},  {2, 6, 1},  {7, 3, 8}  };  **private** **int**[][] \_precells = {  {0, 1, 2},  {3, 4, 5},  {6, 7, 8}  }; //완성된 타일의 상태  **private** Image \_image;  **private** Toolkit toolkit;  // 타일 이미지의 크기  **private** **int** cellSize;  //마지막 타일 초기값  **private** **int** blankCell;  // 빈 타일의 X 좌표값  **private** **int** blankCellCol;  // 빈 타일의 Y 좌표값  **private** **int** blankCellRow;  // 점수 횟수  **private** **int** counter;  // 타일들의 완성 값  //트루면 퍼즐 완성  **private** **boolean** bingoState;  ImagePanel() {  toolkit = getToolkit();  \_image = toolkit.getImage("image1.png");  cellSize = 200; // 타일 크기  blankCellCol = 2; // 초기 빈 타일 X 위치  blankCellRow = 2; // 초기 빈 타일 Y 위치  blankCell = \_cells[blankCellRow][blankCellCol];  counter = 0; // 점수(클릭 횟수)  bingoState = checkBingoState(); // 퍼즐 완성 검사  }  **private** **int** getCellSize() { **return** cellSize; } //타일의 크기  **private** **boolean** getBingoState() { **return** bingoState; }  **private** **boolean** checkBingoState() { //퍼즐완성상태  **for** (**int** row = 0; row < 3; ++row) {  **for** (**int** col = 0; col < 3; ++col) {  **if** (\_cells[row][col] != \_precells[row][col]) {  **return** **false**;  }  }  }  **return** **true**;  }  **private** **boolean** isNeighborToBlankCell(**int** row, **int** col) { //타일의 움직임  **if** (col + 1 == blankCellCol && row == blankCellRow) {  // 우측 타일이 빈 타일  **return** **true**;  }  **if** (col - 1 == blankCellCol && row == blankCellRow) {  // 좌측 타일이 빈 타일  **return** **true**;  }  **if** (col == blankCellCol && row + 1 == blankCellRow) {  // 아래 타일이 빈 타일  **return** **true**;  }  **if** (col == blankCellCol && row - 1 == blankCellRow) {  // 윗 타일이 빈 타일  **return** **true**;  }  **return** **false**;  }  **private** **void** moveToBlank(**int** row, **int** col) { //타일을 빈 타일로 이동  **int** oldBlankCell = \_cells[blankCellRow][blankCellCol];  **int** newBlankCell = \_cells[row][col];  \_cells[blankCellRow][blankCellCol] = newBlankCell;  \_cells[row][col] = oldBlankCell;  blankCellRow = row;  blankCellCol = col;  bingoState = checkBingoState();  }  // 점수횟수 1증가  **private** **void** increaseCounter() {  counter++;  }  **public** **boolean** processInput(**int** x, **int** y) { //완성인지 검토  **if** (!getBingoState()) {  **int** cellSize = getCellSize();  **int** col = x / cellSize;  **int** row = y / cellSize;  // 움직이면 트루  **if** (isNeighborToBlankCell(row, col))  moveToBlank(row, col);  // 타일을 이동 후 화면을 갱신  repaint();  //이동하면 1증가  increaseCounter();  **return** **true**; //이동하면 true  }  }  **return** **false**;  }  **public** **void** paint(Graphics g) {  //바탕은 흰색  g.clearRect(0, 0, getWidth(), getHeight());  **for** (**int** row = 0; row < 3; ++row) {  **for** (**int** col = 0; col < 3; ++col) {  **int** x = col \* cellSize;  **int** y = row \* cellSize;  //빈타일이면 그리지않고 완성이면 빈타일그림  **if** (\_cells[row][col] != blankCell || bingoState) {  **int** imageRow = \_cells[row][col] / 3;  **int** imageCol = \_cells[row][col] % 3;  **int** ix1 = imageCol \* \_image.getWidth(**this**) / 3;  **int** iy1 = imageRow \* \_image.getHeight(**this**) / 3;  **int** ix2 = ix1 + \_image.getWidth(**this**) / 3;  **int** iy2 = iy1 + \_image.getHeight(**this**) / 3;  g.drawImage(\_image, x, y, x + cellSize, y + cellSize, ix1, iy1, ix2, iy2, **this**);  }  }  }  //점수횟수  {  //문자열 폰트색상  g.setFont(**new** Font("TimesRoman", Font.***BOLD***, 70));  g.setColor(Color.***MAGENTA***);  // 점수 문자열 출력  g.drawString("점수", 600 + 20, 100);  // 점수 출력  String counterString = String.*format*("%d", counter);  g.drawString(counterString, 600 + 20, 200);  }  .  **if** (bingoState) { //퍼즐완성하면 완성출력  g.setFont(**new** Font("TimesRoman", Font.***BOLD***, 70));  g.setColor(Color.***GREEN***);  g.drawString("완성", 600 + 20, 300);  }  }  } |
|  |

**3. Mouse Handler**

|  |
| --- |
| import javax.sound.sampled.AudioInputStream;  import javax.sound.sampled.AudioSystem;  import javax.sound.sampled.Clip;  import java.awt.event.MouseEvent;  import java.awt.event.MouseListener;  import java.io.File;  class MouseHandler implements MouseListener {  ImagePanel \_imagePanel;  public MouseHandler(ImagePanel imagePanel) {  \_imagePanel = imagePanel;  }  public void mouseEntered(MouseEvent e) {  }  public void mouseExited(MouseEvent e) {  }  public void mousePressed(MouseEvent e) { //마우스 이동하면 타일이동  if (\_imagePanel.processInput(e.getX(), e.getY())) {  playClickSound();  }  }  public void mouseClicked(MouseEvent e) {  }  public void mouseReleased(MouseEvent e) {  }  private void playClickSound() { //오디오  try  {  String audioClickFileName = "do.wav";  AudioInputStream audioInputStream = AudioSystem.getAudioInputStream(new File(audioClickFileName).getAbsoluteFile());  Clip audioClick = AudioSystem.getClip();  audioClick.open(audioInputStream);  audioClick.start();  } catch (Exception e) {}  }  } |
|  |

**4. Puzzle**

|  |
| --- |
| import javax.swing.\*;  import java.awt.\*;  public class Puzzle {  public static void main(String[] args) {  JFrame frame = new JFrame("루피 퍼즐 맞추기");  frame.setLocation(500, 200);  frame.setPreferredSize(new Dimension(600 + 200, 600));  Container contentPane = frame.getContentPane();  ImagePanel imagePanel = new ImagePanel();  imagePanel.addMouseListener(new MouseHandler(imagePanel));  contentPane.add(imagePanel, BorderLayout.CENTER);  frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  frame.pack();  frame.setVisible(true);  }  } |