Project

-Image Puzzle Game -

제작자 : 이원희

2021.08

INDEX

🙌프로젝트 개요

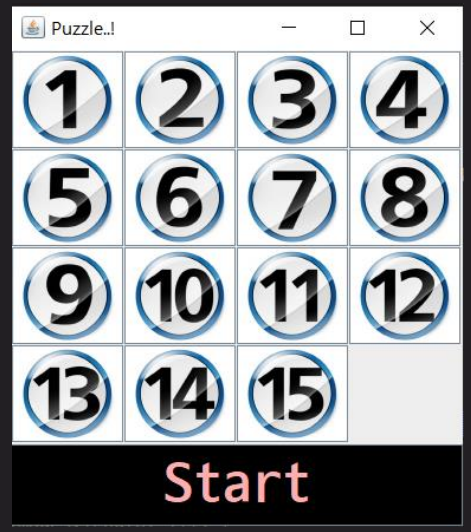
- JAVA GUI 를 활용.
- Swing 패키지를 사용.
- Jframe를 상속받아서 생성자를 선언시 윈도우창이 뜨도록 설계.

- Jlist 클래스를 사용.
- ActionListener, MouseListener 을 구현한다.
- 원하는 이미지를 선택 후 해당 이미지 퍼즐을 푸는 프로그램

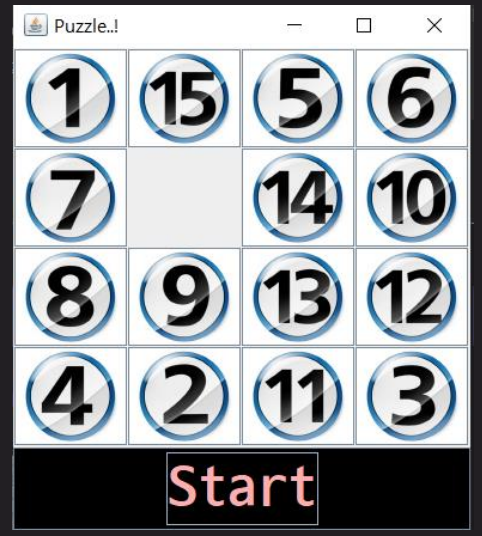
프로그램 순서

1 <원하는 퍼즐 이미지 선택 시작>

2 <퍼즐게임 화면 생성>



3 <시작버튼 클릭 후 게임시작>



화면구성_이미지 선택페이지

JOPtionPane -> showInputDialog 로 구성

```
String imageName = (String)JOptionPane.showInputDialog(null, "퍼즐 이미지 선택", "퍼즐선택" , JOptionPane.PLAIN_MESSAGE,
        new ImageIcon("./src/puzzleImage/mini/original.jpg"),puzzleName, "미니언즈");
```

이미지 제목을 배열에 담아서
showInputDialog 의 인수로 넘겨준다.

```
String [] puzzleName = {"뽀로로" ,
                        "보노보노",
                        "코난",
                        "인형",
                        "영웅",
                        "미니언즈",
                        "나루토",
                        "숫자",
                        "원피스",
                        "피카츄",
                        "푸",
                        "호랑이형님"};
```

화면구성_퍼즐 게임 페이지

1. ActionListener =>버튼 클릭 후 퍼즐이 랜덤으로 섞이는 기능을 구현하기 위해 ActionListener 인터페이스 구현

2. 마우스로 퍼즐을 맞추기 위해 MouseListener 인터페이스 구현

```
implements ActionListener, MouseListener
```

전체 Frame => BoderLayOut 으로 구현

- 퍼즐 부분 패널 4 * 4 GridLayot 으로 구성
- 전체 프레임의 CENTER에 배치

- Start 버튼 이 클릭되면 퍼즐 무작위로 섞임 기능 구현
- 전체 프레임의 SOUTH 에 배치

프로젝트 동작구동 및 코드해석

💥프로그램 동작 KEY POINT ❗
1. 퍼즐 구현기능 => 4*4 = 16개의 버튼을 만들어서 배치한다.

```
String[] numbers = { "1", "2", "3", "4", "5", "6", "7", "8", "9",
        "10", "11", "12", "13", "14", "15", "16" };

public void viewPuzzle() {
    for (int i = 0; i < puzzleButtons.length; i++) {
        int index = Integer.parseInt(numbers[i]) - 1;
        puzzleButtons[i] = new JButton(new ImageIcon(images[index]));

        puzzlePanel.add(puzzleButtons[i]);
        puzzleButtons[i].addMouseListener(this);

        puzzleButtons[i].setName(numbers[i]);

        if (puzzleButtons[i].getName().equals("16")) {
            puzzleButtons[i].setVisible(false);
        }
    }
}
```

1. 16개의 버튼 배열(puzzleButtons) 를 만든다.

2. 버튼의 순번을 기억하는 문자열 배열을 만든다
   * 문자열 배열을 만드는 이유?
   퍼즐들을 섞는 것은 의미가 없고 사진배열의 index 를 섞어야 한다. 하지만 배열은 인덱스 순서는 바꿀 수 없고 퍼즐에 사진을 넣어줄 때의 사진 순서를 바꿔주면 되기 때문에 강제적으로 숫자문자열 배열을 만들어서 섞어 줘야한다.
3. 또한 퍼즐들을 마우스로 조작하기 위해 퍼즐들을 MouseListener 으로 제어한다.

프로젝트 동작구동 및 코드해석

💥프로그램 동작 KEY POINT❗
2. Start 버튼클릭=> 사진들이 무작위로 섞임

```java
@Override
public void actionPerformed(ActionEvent e) {

    if (e.getActionCommand().equals("Start")) {
        for (int i = 0; i < 100000; i++) {
            int r = random.nextInt(15) + 1;
            String temp = numbers[0];
            numbers[0] = numbers[r];
            numbers[r] = temp;
        }
    }
    resetPuzzleTemplet();

}

//  패널에 버튼을 올려서 BoderLayout의 CENTER에 올려주는 메소드
public void viewPuzzle() {
    for (int i = 0; i < puzzleButtons.length; i++) {
        int index = Integer.parseInt(numbers[i]) - 1;
        puzzleButtons[i] = new JButton(new ImageIcon(images[index]));

        puzzlePanel.add(puzzleButtons[i]);
        puzzleButtons[i].addMouseListener(this);

        puzzleButtons[i].setName(numbers[i]);

        if (puzzleButtons[i].getName().equals("16")) {
            puzzleButtons[i].setVisible(false);
        }
    }
}

private void resetPuzzleTemplet() {
    for (int i = 0; i < puzzleButtons.length; i++) {
        puzzleButtons[i].setVisible(true);
        puzzlePanel.remove(puzzleButtons[i]);
    }
    viewPuzzle();
    add(puzzlePanel);
    revalidate();
}
```

1. 시작 버튼이 클릭되면 1~16 까지 문자가 담긴 배열의 순서를 섞어준다

2. 패널에 배치한 퍼즐 들을 삭제 시켜준 뒤
   2-1. viewPuzzle() 메소드로 무작위로 생성된 퍼즐들을 배치한다.
   2-2. 지워진 퍼즐들을 패널에 다시 추가한다
   2-3. revalidate() 메소드로 화면을 새로고침 해준다.
3. viewPuzzle() 메소드
   기존에 패널에 배치된 퍼즐들을 지우고 바뀐 숫자 문자열을 사진 배열의 index로 활용해서 순서가 무작위로 바뀐 퍼즐 사진들을 0~15 번째 퍼즐 배열에 순서대로 담아주고 패널에 다시 퍼즐 배열을 추가한다.

프로젝트 동작구동 및 코드해석

💥프로그램 동작 KEY POINT ❗
3-1 . 마우스 커서를 해당 퍼즐에 올려놓는다 => 빈 공간쪽으로 퍼즐이 이동!

```java
@Override
public void mouseEntered(MouseEvent e) {

    int selectIndex = 0; // 클릭한 index
    JButton button = (JButton) e.getSource();
    for (int i = 0; i < puzzleButtons.length; i++) {
        if (button.getName().equals(puzzleButtons[i].getName())) {
            selectIndex = i;
            break;
        }
    }
```

1. 마우스 커서를 올려놓은 퍼즐의 객체를 기억해준다.
   (JButton button = (JButton) e.getSource();
2. 퍼즐 배열 중에서 해당 주소가 일치하는 퍼즐의 index를 기억해둔다.
   (int selectIndex)

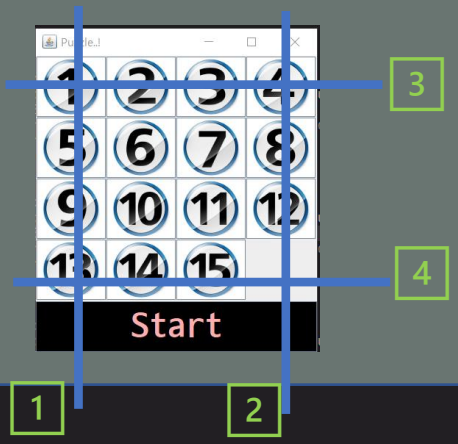
프로젝트 동작구동 및 코드해석

💥프로그램 동작 KEY POINT ❗
3-2 . 마우스 커서를 해당 퍼즐에 올려놓는다 => 빈 공간쪽으로 퍼즐이 이동!

```
            # 왼쪽
if (selectIndex % 4 != 0) {
    if (puzzleButtons[selectIndex - 1].getName().equals("16")) {
        String temp = numbers[selectIndex - 1];
        numbers[selectIndex - 1] = numbers[selectIndex];
        numbers[selectIndex] = temp;
    }
}
            # 오른쪽
if (selectIndex % 4 != 3) {
    if (puzzleButtons[selectIndex + 1].getName().equals("16")) {
        String temp = numbers[selectIndex + 1];
        numbers[selectIndex + 1] = numbers[selectIndex];
        numbers[selectIndex] = temp;
    }
}
            # 위쪽
if (selectIndex / 4 != 0) {
    if (puzzleButtons[selectIndex - 4].getName().equals("16")) {
        String temp = numbers[selectIndex - 4];
        numbers[selectIndex - 4] = numbers[selectIndex];
        numbers[selectIndex] = temp;
    }
}
            # 아래쪽
if (selectIndex / 4 != 3) {
    if (puzzleButtons[selectIndex + 4].getName().equals("16")) {
        String temp = numbers[selectIndex + 4];
        numbers[selectIndex + 4] = numbers[selectIndex];
        numbers[selectIndex] = temp;
    }
}
```

1. 선택한 버튼의 이미지 인덱스로 사용한 숫자 문자열 배열(numbers) 의 값과 빈공간의 numbers 값을 바꿔준다.

- 고려할 사항
 - 4행 4열 의 퍼즐 중에서 ActionListener가 실행된 버튼 중 아래와 같은 경우는 해당 방향으로 퍼즐이 움직여선 안되며, 공통 사항으로 빈공백(16번) 을 건드려도 움직이게 하면 안된다.
  1. 첫번째 열 왼쪽 방향
  2. 네번째 열 오른쪽 방향
  3. 첫번째 행 위쪽 방향
  4. 네번째 행 아래쪽 방향



프로젝트 동작구동 및 코드해석

💥프로그램 동작 KEY POINT ❗
4. 정답여부 검사!

```
        good: // 레이블
        while (true) {
            for (int i = 0; i < puzzleButtons.length - 1; i++) {
                if (i + 1 != Integer.parseInt(puzzleButtons[i].getName())) {
                    break good; // good 라는 레이블의 지정된 반복(while)을 탈출시킨다.
                }
            }
            JOptionPane.showMessageDialog(puzzlePanel, "정답 축하해요~~");
            System.exit(0);
        }
    }
```

퍼즐 배열의 인덱스(0부터 시작이니..+1식한 값) 와 퍼즐의 name(숫자 문자열 배열의 값) 이 모두 일치하면 무한 반복을 종료한 뒤 JOptionPane 클래스로 정답이라는 문자메세지 창을 출력한뒤 프로그램을 종료한다.

프로젝트 후기

- JAVA GUI 를 확실히 이해할 수 있는 계기가 됐다.
- 회원가입 하는 기능을 적용하고 DB 와 연동시켜서 회원별 정답을 맞춘 랭킹을 보여주는 기능 구현해보기.
- 난이도 별로 퍼즐의 복잡도를 조절해줄 수 있게 무작위로 섞어주는 반복을 조절하는 기능을 구현해보기

Q & A

Thank You!