**JAVA LAB 실습 12**

20234014 컴퓨터 공학과 이은정

1.   
**import** java.awt.Color;

**import** java.awt.GridLayout;

**import** javax.swing.ImageIcon;

**import** javax.swing.JFrame;

**import** javax.swing.JLabel;

**import** javax.swing.JPanel;

**class** MyFrame **extends** JFrame {

MyFrame() {

**this**.setTitle("Swing 이미지");

**this**.setSize(1000, 600);

JPanel panel = **new** JPanel();

panel.setLayout(**new** GridLayout(1, 3));

JLabel dog = **new** JLabel("");

JLabel cat = **new** JLabel("");

JLabel bird = **new** JLabel("");

ImageIcon dogImg = **new** ImageIcon("src/Image/dog.png");

ImageIcon catImg = **new** ImageIcon("src/Image/cat.png");

ImageIcon birdImg = **new** ImageIcon("src/Image/bird.png");

dog.setIcon(dogImg);

cat.setIcon(catImg);

bird.setIcon(birdImg);

panel.add(cat);

panel.add(dog);

panel.add(bird);

**this**.add(panel);

**this**.pack();

**this**.setDefaultCloseOperation(***EXIT\_ON\_CLOSE***);

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

}

}

**public** **class** labelImage {

**public** **static** **void** main(String[] args) {

MyFrame myframe = **new** MyFrame();

}

}



위 코드는 이미지 레이블을 배치하는 코드이다. 우선 cat.png, dog.png, bird.png라는 이름의 파일을 다운받는다. 이 그림들은 하나의 패널에 gridLayout을 사용해 나타낼 수 있다.   
panel이라는 이름의 JPanel을 하나 만들고, 레이아웃을 panel.setLayout(**new** GridLayout(1, 3)); 로 설정한다. 따라서 그림이 1행 3열을 이루며 창에 표시될 수 있도록 한다. 그 위에 JLabel 3개를 배치한다. 해당 라벨은 이미지 아이콘을 가지고 있어야 하기 때문에 ImageIcon dogImg = **new** ImageIcon("src/Image/dog.png"); 처럼 ImageIcon을 통해 이미지를 가져와 dog.setIcon(dogImg); 를 통해 넣어준다. 고양이와 새 그림도 위와 같은 과정을 따른다. 그 이후, **this**.pack(); 메서드를 통해 창의 크기를 맞춰 조절한다. 해당 클래스를 실행하게 되면 위와 같이 그림 3개가 연속으로 표시된다.

2.   
**import** java.awt.BorderLayout;

**import** java.awt.FlowLayout;

**import** javax.swing.ImageIcon;

**import** javax.swing.JButton;

**import** javax.swing.JFrame;

**import** javax.swing.JLabel;

**import** javax.swing.JPanel;

**class** MyCar **extends** JFrame {

MyCar() {

setSize(500, 500);

**this**.setLayout(**new** BorderLayout());

JPanel panel1 = **new** JPanel();

JPanel panel2 = **new** JPanel();

panel2.setLayout(**new** FlowLayout());

**this**.setLayout(**new** BorderLayout());

ImageIcon carImg = **new** ImageIcon("src/Image/car.jpg");

JLabel car = **new** JLabel("");

car.setIcon(carImg);

JButton left = **new** JButton("LEFT");

left.addActionListener(e -> {

car.setLocation(car.getX() - 10, car.getY());

});

JButton right = **new** JButton("RIGHT");

right.addActionListener(e -> {

car.setLocation(car.getX() + 10, car.getY());

});

panel1.add(car);

panel2.add(left);

panel2.add(right);

**this**.add(panel1, BorderLayout.***CENTER***);

**this**.add(panel2, BorderLayout.***SOUTH***);

**this**.pack();

**this**.setDefaultCloseOperation(***EXIT\_ON\_CLOSE***);

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

}

}

**public** **class** Car {

**public** **static** **void** main(String[] args) {

MyCar mycar = **new** MyCar();

}

}



다음은 버튼을 누르면 자동차 그림이 움직이도록 만든 코드이다. 

을 통해 이미지를 라벨에 적용한다. 이 라벨은 panel1에 위치한다. 또한 panel2에 left, right 라는 이름의 버튼을 생성한다. 해당 panel들은 큰 이미지 아래 버튼이 가운데 위치한 모습이어야 하기 때문에 panel은 borderLayout, panel들 내부는 flowLayout을 사용한다. 버튼을 누르면 car.getX()을 통해 CAR 라벨 객체의 X 값을 가져온다. 그리고, 10을 더하거나 뺀 값을 다시 setLocation(car.getX() + 10, car.getY()) 에 넣어줌으로써 위치를 지정해준다.

3.   
**import** java.awt.BorderLayout;

**import** java.awt.GridLayout;

**import** javax.swing.ButtonGroup;

**import** javax.swing.ImageIcon;

**import** javax.swing.JFrame;

**import** javax.swing.JLabel;

**import** javax.swing.JPanel;

**import** javax.swing.JRadioButton;

**class** RadioClass **extends** JFrame {

JRadioButton dog, cat, bird, rabbit, pig;

RadioClass() {

JPanel panel1 = **new** JPanel();

panel1.setLayout(**new** BorderLayout());

JPanel panel2 = **new** JPanel();

panel2.setLayout(**new** GridLayout(0, 1));

JLabel label = **new** JLabel("");

ImageIcon img = **new** ImageIcon("src/Image/pig.png");

label.setIcon(img);

panel1.add(label, BorderLayout.***CENTER***);

bird = **new** JRadioButton("BIRD");

bird.addActionListener(e -> {

ImageIcon changeImg = **new** ImageIcon("src/Image/bird.png");

label.setIcon(changeImg);

});

panel2.add(bird, BorderLayout.***WEST***);

cat = **new** JRadioButton("CAT");

cat.addActionListener(e -> {

ImageIcon changeImg = **new** ImageIcon("src/Image/cat.png");

label.setIcon(changeImg);

});

panel2.add(cat, BorderLayout.***WEST***);

dog = **new** JRadioButton("DOG");

dog.addActionListener(e -> {

ImageIcon changeImg = **new** ImageIcon("src/Image/dog.png");

label.setIcon(changeImg);

});

panel2.add(dog, BorderLayout.***WEST***);

rabbit = **new** JRadioButton("RABBIT");

rabbit.addActionListener(e -> {

ImageIcon changeImg = **new** ImageIcon("src/Image/rabbit.png");

label.setIcon(changeImg);

});

panel2.add(rabbit, BorderLayout.***WEST***);

pig = **new** JRadioButton("PIG");

pig.addActionListener(e -> {

ImageIcon changeImg = **new** ImageIcon("src/Image/pig.png");

label.setIcon(changeImg);

});

panel2.add(pig, BorderLayout.***WEST***);

ButtonGroup group = **new** ButtonGroup();

group.add(bird);

group.add(cat);

group.add(dog);

group.add(rabbit);

group.add(pig);

**this**.add(panel1, BorderLayout.***CENTER***);

**this**.add(panel2, BorderLayout.***WEST***);

**this**.pack();

**this**.setDefaultCloseOperation(***EXIT\_ON\_CLOSE***);

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

}

}

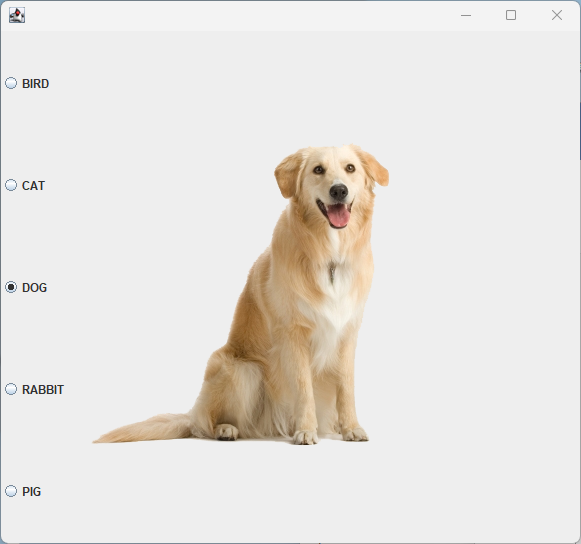
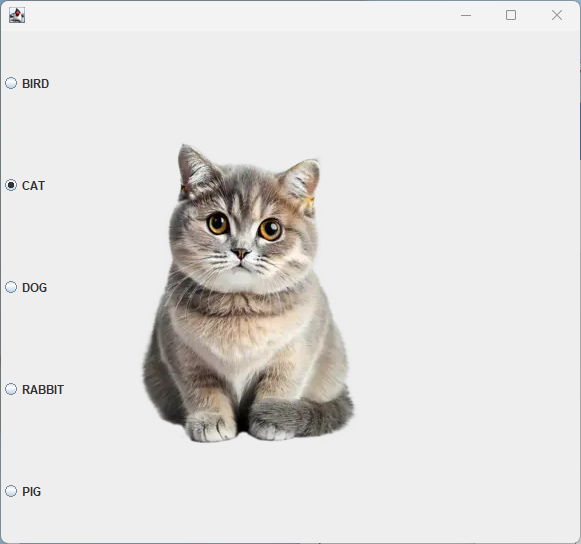
**public** **class** Radio {

**public** **static** **void** main(String[] args) {

RadioClass radio = **new** RadioClass();

}

}



해당 코드는 이미지 라벨을 라디오 버튼을 누름으로써 바꾸는 코드이다. bird = **new** JRadioButton("BIRD"); 처럼 동물 이름을 나타내는 라디오 버튼들을 만든다. 추후 버튼들은 하나의 그룹 안에서 작동해야 하기 때문에 아래 그룹 처럼 하나의 그룹으로 묶는다.

각각의 라디오버튼을 누르는 기능은 addActionListener을 통해 구현할 수 있다. 해당 버튼을 누르면, 기존의 이미지 객체를 해당 이미지 객체로 바꾼다. 기존의 이미지는 아래 label에서 구현되어 있었다. 그러나 버튼을 누를 시, 



위 코드를 통해 라벨 객체에 해당 이미지 객체를 다시 배정함으로써 이미지를 바꾼다.

4.

**class** Calculator **extends** JFrame {

JPanel panel1, panel2;

JTextField text;

String result;

ArrayList<Float> numList;

ArrayList<String> operator;

**public** Calculator() {

**this**.setTitle("Calculator");

panel1 = **new** JPanel();

panel2 = **new** JPanel();

numList = **new** ArrayList<Float>();

operator = **new** ArrayList<>();

result = ""; // Initialize result to an empty string

**this**.setName("Calculator");

**this**.setSize(500, 500);

**this**.setLayout(**new** BorderLayout());

text = **new** JTextField();

text.setText(" ");

panel2.add(text);

// First row

panel1.setLayout(**new** GridLayout(4, 4));

**for** (**int** i = 7; i <= 9; i++) {

**int** number = i;

JButton button = **new** JButton(Integer.*toString*(i));

button.addActionListener(**new** ActionListener() {

@Override

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

result += Integer.*toString*(number);

text.setText(result);

}

});

panel1.add(button);

}

JButton slash = **new** JButton("/");

slash.addActionListener(e -> {

numList.add(Float.*parseFloat*(result));

result = "";

operator.add("/");

text.setText(result);

});

panel1.add(slash);

// Second row

**for** (**int** i = 4; i <= 6; i++) {

**int** number = i;

JButton button = **new** JButton(Integer.*toString*(i));

button.addActionListener(**new** ActionListener() {

@Override

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

result += Integer.*toString*(number);

text.setText(result);

}

});

panel1.add(button);

}

JButton mul = **new** JButton("\*");

mul.addActionListener(e -> {

numList.add(Float.*parseFloat*(result));

result = "";

operator.add("\*");

text.setText(result);

});

panel1.add(mul);

// Third row

**for** (**int** i = 1; i <= 3; i++) {

**int** number = i;

JButton button = **new** JButton(Integer.*toString*(i));

button.addActionListener(**new** ActionListener() {

@Override

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

result += Integer.*toString*(number);

text.setText(result);

}

});

panel1.add(button);

}

JButton min = **new** JButton("-");

min.addActionListener(e -> {

numList.add(Float.*parseFloat*(result));

result = "";

operator.add("-");

text.setText(result);

});

panel1.add(min);

// Fourth row

JButton zero = **new** JButton("0");

zero.addActionListener(e -> {

result += '0';

text.setText(result);

});

panel1.add(zero);

JButton dot = **new** JButton(".");

dot.addActionListener(e -> {

result += '.';

text.setText(result);

});

panel1.add(dot);

JButton endCal = **new** JButton("=");

endCal.addActionListener(e -> {

numList.add(Float.*parseFloat*(result));

text.setText(Float.*toString*(calculate(operator, numList)));

result = ""; // Clear the result after calculation

numList.clear(); // Clear the lists after calculation

operator.clear();

});

panel1.add(endCal);

JButton plus = **new** JButton("+");

plus.addActionListener(e -> {

numList.add(Float.*parseFloat*(result));

result = "";

operator.add("+");

text.setText(result);

});

panel1.add(plus);

**this**.add(panel1, "Center");

**this**.add(panel2, "North");

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

**this**.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

}

**float** calculate(ArrayList<String> list, ArrayList<Float> num) {

**float** res = num.get(0);

**for** (**int** i = 0; i < list.size(); i++) {

String operation = list.get(i);

**float** operand = num.get(i + 1);

**switch** (operation) {

**case** "+":

res += operand;

**break**;

**case** "-":

res -= operand;

**break**;

**case** "\*":

res \*= operand;

**break**;

**case** "/":

res /= operand;

**break**;

}

}

**return** res;

}

}

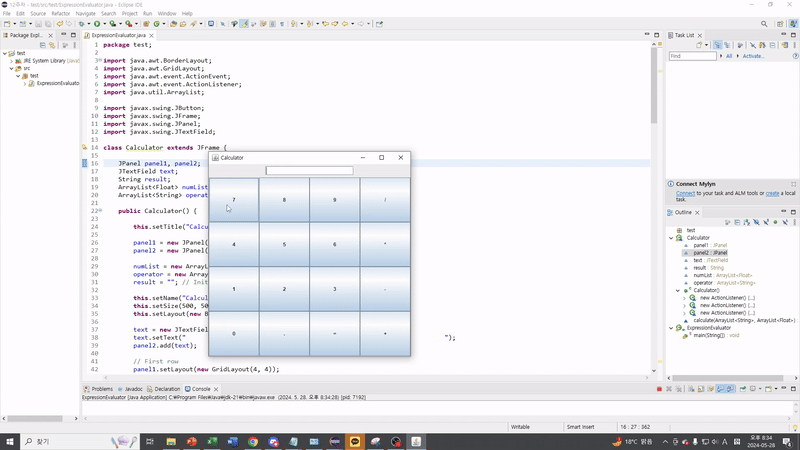
**public** **class** ExpressionEvaluator {

**public** **static** **void** main(String[] args) {

Calculator calculator = **new** Calculator();

}

}



위 코드는 계산기를 나타내는 코드이다. 해당 코드는 숫자를 입력하면, 텍스트 필드에 값을 입력하고, 입력 값을 계산하는 코드이다. 먼저 위 코드가 어떻게 동작하는지 살펴보자. 해당 코드는 borderLayout을 사용해 이루어져 있다. 버튼 객체들이 center에 모여있고, textField가 North에 위치해 있다. 센터에는 gridLayout을 통해 여러 버튼을 배치한다. 이 버튼들은 누를 시 해당 수에 해당하는 숫자를 입력할 수 있다.

**for** (**int** i = 4; i <= 6; i++) {

**int** number = i;

JButton button = **new** JButton(Integer.*toString*(i));

button.addActionListener(**new** ActionListener() {

@Override

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

result += Integer.*toString*(number);

text.setText(result);

}

});

panel1.add(button);

}

위 코드처럼, 특정 숫자를 for문으로 돌릴 시, 해당 버튼에 해당하는 버튼을 Integer.toString(i)를 통해 String으로 바꾸어 버튼으로 만든다. 해당 버튼을 누를시, result에 해당 i를 추가하고 textField에 다시 그 숫자를 세팅한다. 숫자가 아닌 연산자를 살펴보자.

JButton mul = **new** JButton("\*");

mul.addActionListener(e -> {

numList.add(Float.*parseFloat*(result));

result = "";

operator.add("\*");

text.setText(result);

});

panel1.add(mul);

위 코드는 \* 버튼이다. 해당 버튼을 누를 시, result를 숫자화 하여 numList에 추가하고, result를 초기화 한 이후, textField도 비워준다. 최종적으로 =을 누르게 되면 계산값을 반환한다.

**float** calculate(ArrayList<String> list, ArrayList<Float> num) {

**float** res = num.get(0);

**for** (**int** i = 0; i < list.size(); i++) {

String operation = list.get(i);

**float** operand = num.get(i + 1);

**switch** (operation) {

**case** "+":

res += operand;

**break**;

**case** "-":

res -= operand;

**break**;

**case** "\*":

res \*= operand;

**break**;

**case** "/":

res /= operand;

**break**;

}

}

**return** res;

}

숫자와 연산자가 저장된 리스트를 받아 계산한다. 해당 연산자에 따라 각기 다른 계산을 행한 후, 그 결과값을 반환하고, 이를 textField에 넣어준다.

5.

**package** Menu;

**import** javax.swing.JFrame;

**import** javax.swing.JMenu;

**import** javax.swing.JMenuBar;

**import** javax.swing.JMenuItem;

**import** javax.swing.JPanel;

**import** javax.swing.JTextField;

**class** MyFrame **extends** JFrame {

JTextField text;

String word;

MyFrame() {

**this**.setTitle("Menu Example");

JPanel panel = **new** JPanel();

**this**.setSize(400, 200);

word = "";

text = **new** JTextField(20);

panel.add(text);

JMenuBar menuBar = **new** JMenuBar();

JMenu fileMenu = **new** JMenu("파일");

menuBar.add(fileMenu);

JMenu open = **new** JMenu("열기");

fileMenu.add(open);

JMenuItem projectItem = **new** JMenuItem("저장하기");

projectItem.addActionListener(e -> {

text.setText(projectItem.getText());

});

JMenuItem moduleItem = **new** JMenuItem("다른이름으로 저장하기");

moduleItem.addActionListener(e -> {

text.setText(moduleItem.getText());

});

fileMenu.add(projectItem);

fileMenu.add(moduleItem);

JMenu printMenu = **new** JMenu("출력");

fileMenu.add(printMenu);

JMenuItem printerPrint = **new** JMenuItem("프린터로 출력하기");

printerPrint.addActionListener(e -> {

text.setText(printerPrint.getText());

});

JMenuItem pdfPrint = **new** JMenuItem("pdf로 출력하기");

pdfPrint.addActionListener(e -> {

text.setText(pdfPrint.getText());

});

printMenu.add(printerPrint);

printMenu.add(pdfPrint);

**this**.setJMenuBar(menuBar);

**this**.add(panel);

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

**this**.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

}

}

**public** **class** menu {

**public** **static** **void** main(String[] args) {

MyFrame f = **new** MyFrame();

}

}

menu바에 파일을 하나 추가한다. JMenuBar menuBar = new JMenuBar(); 을 통해 메뉴바를 추가하고, 각각의 요소를 추가한다. JMenu open = new JMenu("열기"); fileMenu.add(open); 처럼 각각의 요소를 메뉴바에 추가한다. 출력 파트에서는 JMenuItem pdfPrint = new JMenuItem("pdf로 출력하기");

pdfPrint.addActionListener(e -> {

text.setText(pdfPrint.getText());

});

위처럼 메뉴 아이템 하위의 메뉴아이템이기 때문에 메뉴 아이템을 만든 이후, 메뉴아이템 print에 추가한다. 이후, 각각의 요소들에 textField에 text.setText(pdfPrint.getText()); 를 통해 필드 텍스트를 바꾼다.

