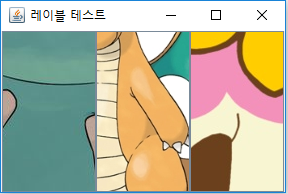
< JAVA PROGRAMMING LAB -9장: 이벤트 처리 심화>

◇ 문제 및 요구하는 것

◆ 프로그램 실행결과



**1. 3개의 레이블을 윈도우에 배치하고 여기에 이미지를 추가해라**

◇ 제목

◇ 제목

◇ 제목

◇ 제목

◇ 제목

◇ 제목

**import** java.awt.GridLayout;

**import** javax.swing.\*;

**public** **class** Book{

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

JFrame f=**new** JFrame();

f.setSize(300,200);

f.setTitle("레이블 테스트");

f.setLayout(**new** GridLayout(0,3));

JLabel la1=**new** JLabel();

JLabel la2=**new** JLabel ();

JLabel la3=**new** JLabel ();

ImageIcon icon1=**new** ImageIcon("c:\\p1.jpg");

ImageIcon icon2=**new** ImageIcon("c:\\p2.jpg");

ImageIcon icon3=**new** ImageIcon("c:\\p3.jpg");

la1.setIcon(icon1);

la2.setIcon(icon2);

la3.setIcon(icon3);

f.add(la1);

f.add(la2);

f.add(la3);

f.setVisible(**true**);

}

}

◇ 제목

◇ 제목

◇ 프로그램 소스

일단 윈도우 창을 만들기 위해 JFeame객체를 만들었다. 윈도우의 사이즈 및 타이틀을 지정하고 배치 관리자를 지정했다. 그리드 레이아웃을 이용하여 윈도우에 위젯이 올 때 격자 형식(3열)으로 배치되게 설정하였다. 이후 라벨 객체를 세 개 만들고 이미지 아이콘 객체를 세 개 만들었다. 이미지 아이콘 객체는 c드라이브에 있는 이미지를 갖고 있는 객체로, 위젯의 아이콘으로 쓸 수 있게 만들어졌다. 즉 앞서 만든 라벨 객체에 setIcon메소드(아이콘 이미지 객체)를 하면 라벨의 아이콘이 이미지가 되겠다. 버튼이라면 모를까, 라벨이 아이콘이 있다는 것은 말이 안 된다. 이 부분은 그냥, 라벨 위젯 옆에 이미지를 넣는다고 생각하자. 마지막으로 윈도우에 라벨들을 추가하면 이미지 세 개가 격자 형태(1행 3열)로 나타날 것이다.

◇ 제목

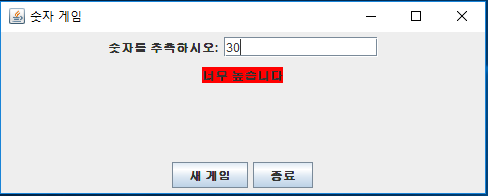
◇ 제목

◇ 프로그램 설명

< JAVA PROGRAMMING LAB -9장: 이벤트 처리 심화>

◇ 문제 및 요구하는 것

◆ 프로그램 실행결과



**2. 프로그램은 1부터 100사이의 랜덤 수를 가지고 있다. 입력해서 랜덤 수보다 높으면 높다고, 낮으면 낮다고, 같으면 맞다고 텍스트를 보여라**

◇ 제목

◇ 제목

◇ 제목

◇ 제목

◇ 제목

◇ 제목

**import** java.awt.BorderLayout;

**import** java.awt.Color;

**import** java.awt.FlowLayout;

**import** java.awt.GridLayout;

**import** javax.swing.\*;

**public** **class** Book{

**static** **int** *num*=(**int**)(Math.*random*()\*100)+1;

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

JFrame f=**new** JFrame();

f.setSize(500,200);

f.setTitle("숫자 게임");

JPanel[] jp=**new** JPanel[3];

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

jp[i]=**new** JPanel();

}

JLabel text=**new** JLabel("숫자를 추측하시오:");

JTextField txthome= **new** JTextField(15);

jp[0].add(text);

jp[0].add(txthome);

f.add(jp[0],BorderLayout.***NORTH***);

JLabel text2=**new** JLabel("게임 시작");

jp[1].add(text2);

f.add(jp[1],BorderLayout.***CENTER***);

JButton bt1=**new** JButton("새 게임");

JButton bt2=**new** JButton("종료");

jp[2].add(bt1);

jp[2].add(bt2);

f.add(jp[2],BorderLayout.***SOUTH***);

txthome.addActionListener(e->{

**int** com=Integer.*parseInt*(txthome.getText());

**if**(com>*num*)

text2.setText("너무 높습니다");

**else** **if**(com<*num*)

text2.setText("너무 낮습니다");

**else**

text2.setText("맞았습니다");

text2.setBackground(Color.***red***);

text2.setOpaque(**true**);

});

bt1.addActionListener(e->{

text2.setText("새 게임");

text2.setOpaque(**false**);

*num*=(**int**)(Math.*random*()\*100)+1;

});

bt2.addActionListener(e->{

System.*exit*(0);

});

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

f.setVisible(**true**);

}

}

◇ 제목

◇ 제목

◇ 프로그램 소스

◇ 제목

◇ 제목

◇ 프로그램 소스

main함수에서도 쓰일 수 있기 위해 정적 변수 num을 만들었다. 이 num은 1부터 100까지 랜덤 수가 저장된다. 다음은 GUI설명이다. 크기가 3인 패널 객체 배열을 만들고, 모든 요소에 패널 객체를 넣었다. 다음은 숫자를 추측해보라는 텍스트를 만들고 글자 수 15개가 들어가는 텍스트 필드를 만들었다. 이후 패널 배열 첫 번째 요소에 텍스트와 텍스트 필드를 추가하였다 다음은 게임 시작이라는 텍스트를 만들고 패널 배열 두 번째 요소에 추가하였다. 마지막으로 "새 게임", "종료" 텍스트를 포함한 버튼 2개를 만들고 패널 배열 세 번째 요소에 추가하였다. 이렇게 컴포넌트들을 가진 패널 세 개가 만들어졌다. 이 패널들은 BorderLayout에 의해 윈도우의 북쪽,센터,남쪽에 배치시켰다. 다음은 텍스트 필드의 람다식이다. addActionListener메소드에 있는 일종의 객체로, 사용자가 텍스트 필드에 입력하고 엔터를 치면 동작한다. 일단 com이라는 정수형 변수에 텍스트필드에 입력된 텍스트를 정수형으로 바꿔서 저장하였다. 그리고 com과 num을 비교해서 너무 높은지,낮은지,같은지를 비교하고 세 번째 패널 안의 텍스트로 알려주었다. 텍스트의 배경 색도 바꾸었다. setOparque를 True해야 색이 바뀐다는 점을 기억해야 한다. 다음은 "새 게임"버튼에 대한 람다식이다. 앞서 말한 텍스트의 글자를 바꾸고, oparque를 false해서 색을 없앤 다음에 num값을 다시 세팅하였다. 마지막으로 "종료" 버튼에 대한 람다식인데 그냥 시스템을 종료한다는 소스가 들어 있다.

◇ 제목

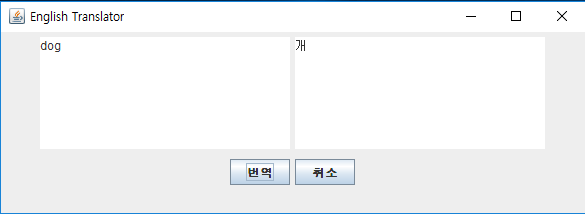
◇ 제목

◇ 프로그램 설명

< JAVA PROGRAMMING LAB -9장: 이벤트 처리 심화>

◇ 문제 및 요구하는 것

◆ 프로그램 실행결과



**3. ArrayList에 미리 입력된 영어 단어가 오면, 번역하는 프로그램을 만들어라**

◇ 제목

◇ 제목

◇ 제목

◇ 제목

◇ 제목

◇ 제목

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** java.util.ArrayList;

**import** javax.swing.\*;

**class** Word{

String han\_word;

String eng\_word;

**public** Word(String s1,String s2){

han\_word=s1;

eng\_word=s2;

}

}

**public** **class** Book{

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

ArrayList<Word> list=**new** ArrayList<>();

list.add(**new** Word("개","dog"));

list.add(**new** Word("고양이","cat"));

list.add(**new** Word("소","cow"));

JFrame f=**new** JFrame();

f.setSize(600,220);

f.setTitle("English Translator");

JPanel[] jp=**new** JPanel[2];

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

jp[i]=**new** JPanel();

}

JTextArea txthome= **new** JTextArea(7,25);

JTextArea txthome2= **new** JTextArea(7,25);

jp[0].add(txthome);

jp[0].add(txthome2);

f.add(jp[0],BorderLayout.***NORTH***);

JButton bt1=**new** JButton("번역");

JButton bt2=**new** JButton("취소");

jp[1].add(bt1);

jp[1].add(bt2);

f.add(jp[1],BorderLayout.***CENTER***);

**class** MyListener **implements** ActionListener{

**private** Word word;

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

**if**(e.getSource()==bt1) {

String s=txthome.getText();

**for**(Word word:list) {

**if**(s.equals(word.eng\_word))

txthome2.setText(word.han\_word);

}

}

**else** {

txthome.setText("");

txthome2.setText("");

}

}

}

bt1.addActionListener(**new** MyListener());

bt2.addActionListener(**new** MyListener());

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

f.setVisible(**true**);

}

}

◇ 제목

◇ 제목

◇ 프로그램 소스

◇ 제목

◇ 제목

◇ 프로그램 소스

main함수부터 살펴보자. 어레이 리스트가 쓰였는데, <>안에 Word라고 하는 객체가 들어있다. 즉 이 리스트에는 Word객체들을 담을 것이라는 말이다. Word클래스는 두 개의 필드와 생성자 한 개가 있다. 생성자에서는 문자열을 받아 필드에 값을 넘겨주는 역할을 한다. 예를 들어 생성자에 개와 dog를 넘겨주면 개,dog가 필드로 담길 것이다. 이렇게 만들어진 Word객체를 리스트 요소에 추가한다. 이러한 작업을 필자는 세 번 하였다. 각기 다른 필드들을 지닌 Word객체들을 리스트 요소들이 가지게 되겠다. 다음은 윈도우 생성을 하고 패널 객체 배열을 만든 후 배열 요소에 패널 객체 두 개를 추가하였다. 이번엔 새로운 컴포넌트가 나왔다. JTextArea라는 위젯으로 JTextField의 확장판이다. 이 위젯을 만들 땐, 생성자에 수 두 개를 전달하는데, 행과 열에 해당한다. 즉 7,25라는건 사용자가 7행25열만큼 텍스트를 입력할 수 있다는 말이다. 이런 위젯을 2개 만들었다. 그리고 첫 번째 패널에 두 텍스트아레아를 추가하였다. 이 패널은 윈도우의 북쪽에 삽입된다. 이후엔 번역 버튼,취소 버튼을 만들고 두 번째 패널에 추가하고, 패널을 윈도우 센터에 삽입하였다. 다음에 나오는 Mylistener을 보기 전에 클래스 다음 나오는 addActionListener을 살펴보자. 이 메소드의 인수를 보면 MyListener객체가 있는데, 이 객체가 번역,취소 버튼의 이벤트 처리를 담당하는 객체라고 등록하는 것이라 볼 수 있다. 이제 중요한 MyListener클래스를 봐보자. 이 클래스는 ActionListener을 구현한다. 그리고 actionPerformed메소드를 구현할 필요가 있다. e.getSource()는 이벤트를 준 객체의 참조변수가 반환되는데 이게 bt1(번역 버튼)일 때, 텍스트아레아에 있는 글자가 스트링 변수에 담긴다. 그리고 for-each문을 통해 어레이 리스트의 요소들을 하나씩 꺼내서 word라는 (Word객체를 가리킬 수 있는)참조변수가 가리키게 하였다. 이후, 스트링 변수와 word가 가리키는 객체의 eng\_word를 비교해서 같으면, word가 가리키는 객체의 han\_word(한글)을 다른 텍스트 아레아에 나타내기로 하였다. 이런 식으로 미리 어레이 리스트에 문자들을 등록한 뒤 찾아서 쓸 수 있다.

◇ 제목

◇ 제목

◇ 프로그램 설명