Power Java

제21장 스윙 컴포넌트 II

이번 장에서 학습할 내용

- 레이블과 버튼에 이미지 표시
- 리스트
- 콤보 박스
- 스피너
- 슬라이더
- 트리
- 색상선택기및 폰트선택기
- 메뉴
- 대화상자
- 테이블

텍스트와 레이블에 이미지 추가

텍스트와 레이블에 이미지 추가

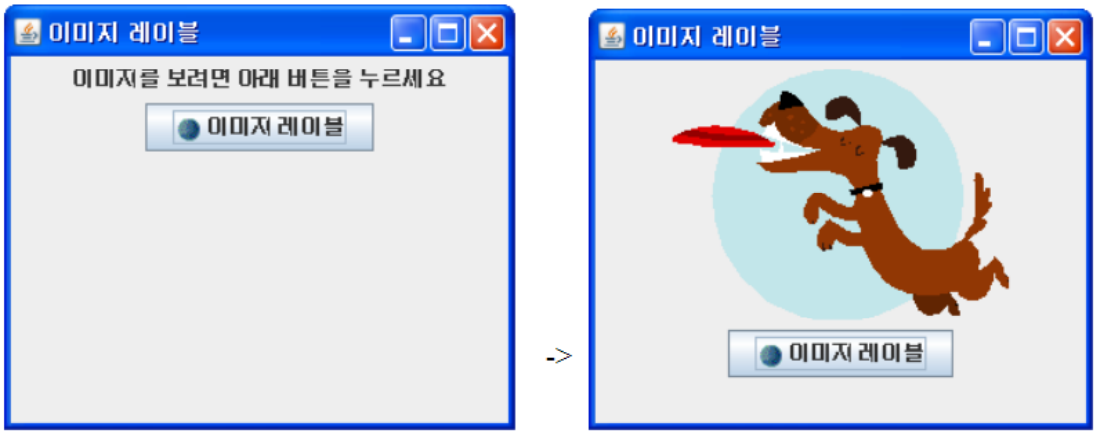
1. ImageIcon 인스턴스를 생성하여야 한다.
   - ImageIcon image = **new** ImageIcon("image.gif");

2. setIcon() 메소드를 사용
   - JLabel label = **new** JLabel("이미지 레이블");
   - label.setIcon(image);

```
public class ImageLabelTest extends JFrame implements ActionListener {
    private JPanel panel;
    private JLabel label;
    private JButton button;

    public ImageLabelTest() {
        setTitle("이미지 레이블");
        setSize(300,250);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        panel = new JPanel();
        label = new JLabel("이미지를 보려면 아래 버튼을 누르세요");

        button = new JButton("이미지 레이블");
        ImageIcon icon = new ImageIcon("icon.gif");
        button.setIcon(icon);
        button.addActionListener(this);
        panel.add(label);
        panel.add(button);

        add(panel);
        setVisible(true);
    }
```

```
    public static void main(String[] args) {
        ImageLabelTest t=new ImageLabelTest();
    }

    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        ImageIcon dog = new ImageIcon("dog.gif");
        label.setIcon(dog);
        label.setText(null);
    }
}
```

실행 결과

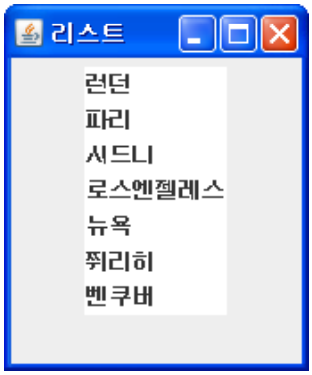


중간점검

1. 버튼에 이미지를 표시하는 절차를 설명하라. 어떻게 텍스트와 이미지를 동시에 표시할 수 있는가?
2. 기존에 존재하는 레이블이나 버튼에 이미지를 추가하는 메소드는?

리스트

- 리스트(List)는 여러 개의 선택 항목 중에서 하나를 선택하기 위한 컴포넌트이다.
- 리스트는 한 줄에 하나씩 선택 항목을 나타내며 이 영역은 스크롤이 가능하다.
- 일반적으로 사용자는 마우스 클릭에 의하여 항목을 선택
- 더블 클릭이나 엔터 키를 치면 액션 이벤트가 발생한다.



리스트의 메소드

■ 생성자

생성자	설 명
JList()	비어 있는 리스트를 생성한다.
JList(Object[] items)	배열에 있는 값들을 가지고 선택 항목을 생성한다.
JList(ListModel list)	지정된 리스트 모델을 사용하는 리스트를 생성한다.
JList(Vector[] items)	벡터에 있는 값들을 가지고 선택 항목을 생성한다.

리스트를 생성하는 가장 일반적인 방법은 배열을 JList의 생성자에 전달하는 것이다. JList의 생성자를 이용하여서 리스트를 생성하여 보면 다음과 같다.

```
private String[] names = { "하나", "둘", "셋", "넷", "다섯", "여섯" };
list = new JList(names);
```


■ 선택 모드

리스트에서는 항목을 선택할 수 있는 모드에는 다음의 3가지가 있다.

표 22.1 리스트에서의 선택 모드

모드	그림	설명
단일 선택 (SINGLE_SELECTION)		한 번에 하나의 항목만이 선택된다. 사용자가 새로운 항목을 클릭하면 이전의 선택은 지워진다.
단일 구간 선택 (SINGLE_INTERVAL_SELECTION)		여러 개의 연속적인 항목들이 선택될 수 있다. 일정 범위의 항목을 선택하려면 첫 번째 항목을 클릭하고 쉬프트 키를 누른 채로 마지막 항목을 클릭한다.
다중 구간 선택 (MULTIPLE_INTERVAL_SELECTION)		디폴트로 항목들이 자유롭게 선택될 수 있다. 컨트롤 키를 누른 채로 마우스로 클릭을 하면 여러 개를 선택할 수 있다.

선택 모드 변경

```
list.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
```

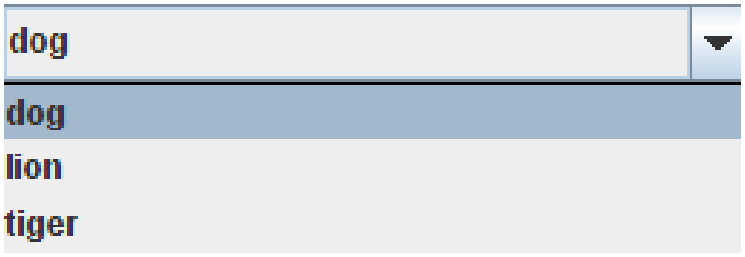

리스트이 이벤트 처리

■ 이벤트 처리

리스트의 항목이 선택되면 리스트 선택 이벤트(ListSelectionEvent)를 발생한다. 이 이벤트는 리스트 선택 리스너를 가지고 처리할 수 있다. 리스트 선택 리스너에는 하나의 메소드만 구현되어 있으면 되는데 바로 valueChanged() 메소드이다.

```
private class ListListener implements ListSelectionListener {
    public void valueChanged(ListSelectionEvent e) {
            if (list.getSelectedIndex() == -1) {
                //선택이 되지 않았음
            } else {
                //선택이 되었음
            }
    }
}
```

콤보박스



콤보 박스의 메소드

■ 생성자

콤보 박스를 생성하기 위해서는 먼저 생성자 중에서 하나를 골라서 호출하여야 한다. 첫 번째 생성자는 비어 있는 콤보 박스를 생성한다.

```
JComboBox combo = new JComboBox();
```

여기에 항목을 추가하려면 addItem() 메소드를 사용한다.

```
combo.addItem("dog");
combo.addItem("lion");
combo.addItem("tiger");
```

콤보 박스의 이벤트 처리

```
private class ComboListener  implements ActionListener
        {
         public void actionPerformed(ActionEvent e)
         {
               if (e.getSource() == combo)
               {
                     String s = (String)combo.getSelectedItem();
                     if (s.equals("dog"))
                           System.out.println("강아지가 선택되었습니다");
               }
         }
        }
```


ComboBoxTest.java

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class ComboBoxTest extends JFrame implements ActionListener {
        JLabel label;

        public ComboBoxTest() {
                setTitle("콤보 박스");
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setSize(300, 200);

                String[] animals = { "dog", "lion", "tiger" };
                JComboBox animalList = new JComboBox(animals);
                animalList.setSelectedIndex(0);
                animalList.addActionListener(this);

                label = new JLabel();
                label.setHorizontalAlignment(JLabel.CENTER);
                changePicture(animals[animalList.getSelectedIndex()]);
                add(animalList, BorderLayout.PAGE_START);
                add(label, BorderLayout.PAGE_END);
                setVisible(true);
        }
```

```
        public void actionPerformed(ActionEvent e) {
                JComboBox cb = (JComboBox) e.getSource();
                String name = (String) cb.getSelectedItem();
                changePicture(name);
        }

        protected void changePicture(String name) {
                ImageIcon icon = new ImageIcon(name + ".gif");
                label.setIcon(icon);
                if (icon != null) {
                        label.setText(null);
                } else {
                        label.setText("이미지가 발견되지 않았습니다.");
                }
        }

        public static void main(String[] args) {
                ComboBoxTest frame=new ComboBoxTest();
        }
}
```

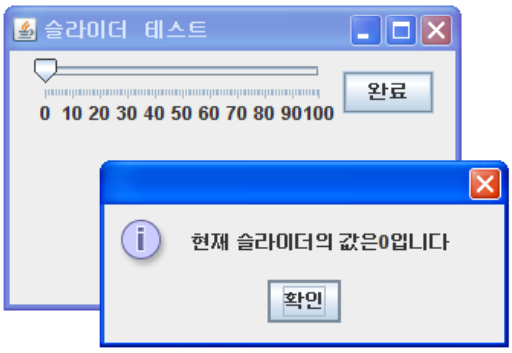

실행 결과

스피너

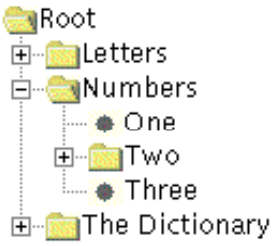

```
JSpinner spinner;

// List Model 테스트
String[] items = { "소설", "잡지", "전공서적" ,"취미"};
SpinnerListModel listModel = new SpinnerListModel(items);
spinner = new JSpinner(listModel);
panel.add(spinner);
```


슬라이더

```
public SliderTest() {
        JPanel panel;

        panel = new JPanel();
        slider = new JSlider(0, 100, 0);
        slider.setMajorTickSpacing(10);
        slider.setMinorTickSpacing(1);
        slider.setPaintTicks(true);
        slider.setPaintLabels(true);
        panel.add(slider);
        buttonOK = new JButton("완료");
        panel.add(buttonOK);
        add(panel);
        buttonOK.addActionListener(this);

}
```



트리



트리의 생성 순서

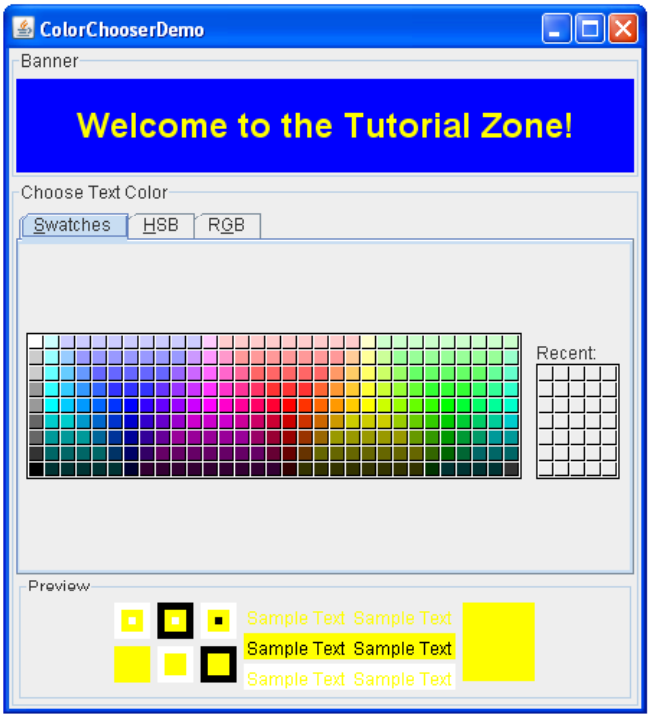
첫번째 단계로 생성자를 이용하여서 노드들을 생성한다.

```
DefaultMutableTreeNode root = new DefaultMutableTreeNode("루트 노드");
DefaultMutableTreeNode child1 = new DefaultMutableTreeNode("첫번째 자식 노드");
DefaultMutableTreeNode child2 = new DefaultMutableTreeNode("두번째 자식 노드");
```

다음 단계는 생성된 노드들을 서로 부모 자식 관계로 연결하는 것이다.

```
root.add(child1);
root.add(child2);
```

색상 선택기



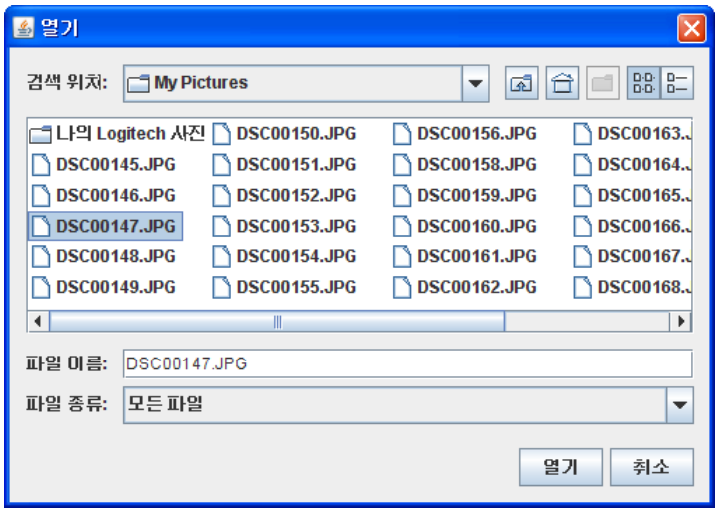

```
JColorChooser  colorChooser = new JColorChooser(Color.RED);
```


```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.colorchooser.*;

public class ColorChooserTest extends JFrame implements ChangeListener {

        protected JColorChooser color;

        public ColorChooserTest() {
                setTitle("색상 선택기  테스트");
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

                color = new JColorChooser();
                color.getSelectionModel().addChangeListener(this);
                color.setBorder(BorderFactory.createTitledBorder("색상 선택"));

                JPanel panel = new JPanel();
                panel.add(color);
                add(panel);
                pack();
                this.setVisible(true);

        }
```

예제

```
    public void stateChanged(ChangeEvent e) {
        Color newColor = color.getColor();
    }

    public static void main(String[] args) {
        new ColorChooserTest();
    }
}
```


파일 선택기



파일 선택기

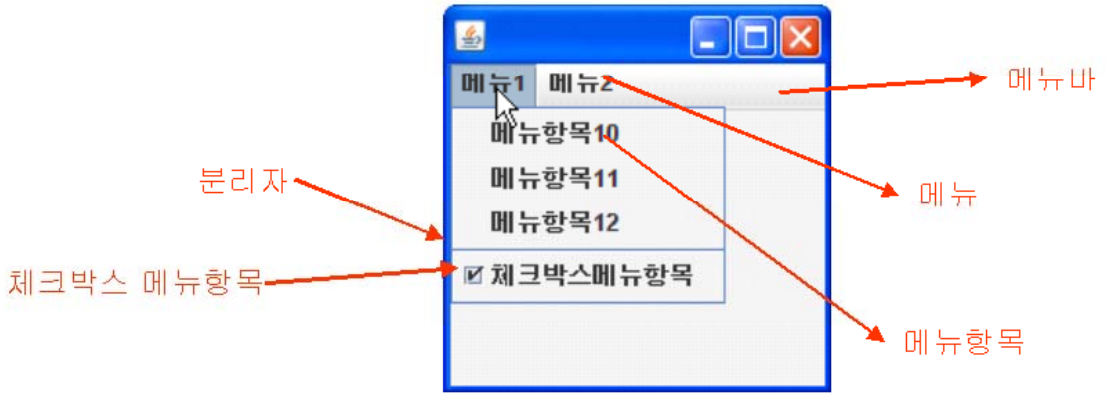
- 표준 열기 대화상자를 만드는 것은 다음의 문장으로 한다.

```
//파일 선택기를 생성한다.
final JFileChooser fc = new JFileChooser();
...
//버튼이 클릭되면 반환된다.
int returnVal = fc.showOpenDialog(aComponent);
```


파일 선택기

```
public void actionPerformed(ActionEvent e) {
    //Handle open button action.
    if (e.getSource() == openButton) {
        int returnVal = fc.showOpenDialog(FileChooserTest.this);

        if (returnVal == JFileChooser.APPROVE_OPTION) {
            File file = fc.getSelectedFile();
                // 파일에 대한 처리를 한다.
        } else {
                // 파일 선택이 사용자에 의하여 취소되었음.
        }
    } ...
}
```

메뉴



메뉴 클래스의 상속 계층도

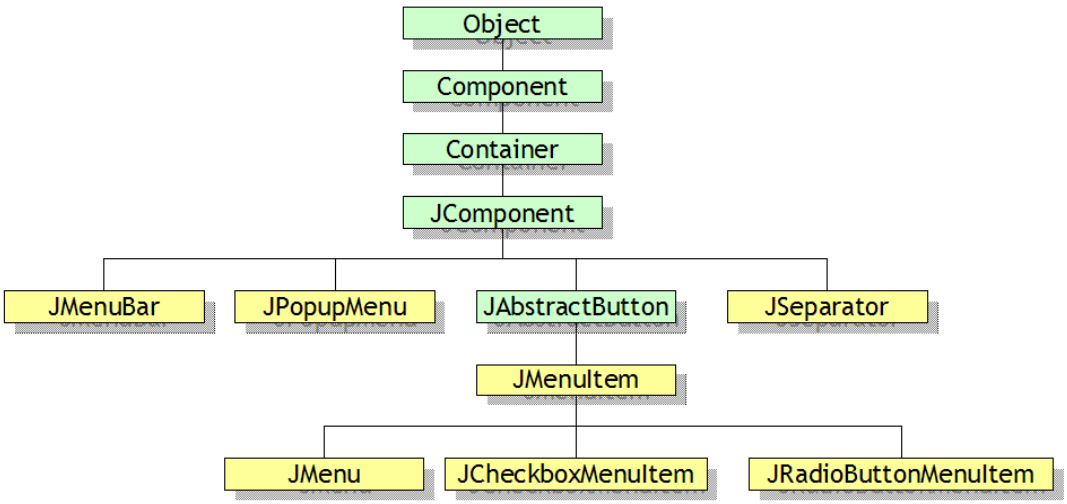


그림 22.11 메뉴 컴포넌트 계층도

메뉴 생성 순서

```
// ① 메뉴바 관련 변수를 선언한다.
JMenuBar menuBar;              // 메뉴바
JMenu menu;           // 메뉴
JMenuItem menuItem;            // 메뉴 항목

// ② 메뉴바를 생성한다.
menuBar = new JMenuBar();

// ③ 메뉴를 생성하여 메뉴바에 추가한다.
menu = new  JMenu("메뉴1");
menuBar.add(menu);

// ④ 메뉴 항목을 생성하여 메뉴에 추가한다.
menuItem = new JMenuItem("메뉴항목1",  KeyEvent.VK_T);
menu.add(menuItem);

// ⑤ 프레임에 메뉴바를 설정한다.
frame.setJMenuBar(mb);
```


대화 상자

- 대화 상자 윈도우는 임시 정보를 나타내는데 사용되는 독립적인 서브 윈도우이다.
- 몇 개의 미리 정해진 표준 대화 상자는 아주 간단히 만들 수 있다.

```
JOptionPane.showMessageDialog(frame, "표준 대화 상자는 간단히 만들 수 있습니다.");
```

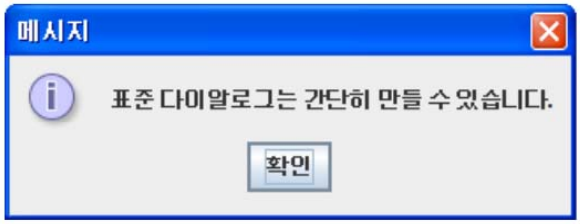


그림 22.14 간단한 대화 상자

showMessageDialog()

```
JOptionPane.showMessageDialog(this, "범위를 초과하였습니다.", "경고",
                                    JOptionPane.WARNING_MESSAGE);
```

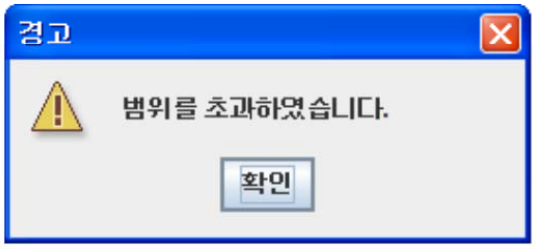


그림 22.15 showMessageDialog()를 사용한 대화 상자

showOptionDialog()

```
Object[] options = {"Well-Done", "Medium", "Rare"};
int n = JOptionPane.showOptionDialog(this,
                                "스테이크를 어떻게 요리할까요?",
                                "스테이크 주문",
                                JOptionPane.YES_NO_CANCEL_OPTION,
                                JOptionPane.QUESTION_MESSAGE,    null,
                                options,    options[2]);
```

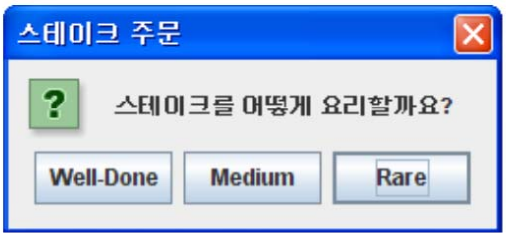


그림 22.18 showOptionDialog()를 이용한 대화 상자

테이블

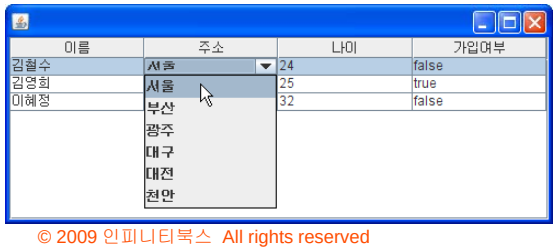
- 테이블은 스프레드 쉬트처럼 데이터를 테이블 형식으로 표시하고 사용자가 값을 편집할 수 있는 컴포넌트이다

그림 22.21 테이블에서의 용어

```
import javax.swing.*;
import java.awt.*;
import javax.swing.event.*;
import javax.swing.table.TableColumn;
import javax.swing.table.TableModel;

public class SimpleTableTest extends JFrame implements TableModelListener {

    public SimpleTableTest() {

        String[] columnNames = {"이름", "주소", "나이", "가입여부"};

        Object[][] data = {
            {"김철수", "서울", new Integer(24), new Boolean(false)},
            {"김영희", "부산", new Integer(25), new Boolean(true)},
            {"이혜정", "천안", new Integer(32), new Boolean(false)}
        };

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(500,200);
        final JTable table = new JTable(data, columnNames);

        table.setPreferredScrollableViewportSize(new Dimension(500, 150));
        table.setFillsViewportHeight(true);
```

```
        // 셀이 변경되면 이벤트 처리
        table.getModel().addTableModelListener(this);
        // 컬럼 헤더를 클릭하면 자동 정렬
        table.setAutoCreateRowSorter(true);

        // 주소를 입력할 때 지정된 도시만 입력이 가능하도록 편집기 설정
        TableColumn      cityColumn = table.getColumnModel().getColumn(1);
        JComboBox comboBox = new JComboBox();
        comboBox.addItem("서울");
        comboBox.addItem("부산");
        comboBox.addItem("광주");
        comboBox.addItem("대구");
        comboBox.addItem("대전");
        comboBox.addItem("천안");
        cityColumn.setCellEditor(new DefaultCellEditor(comboBox));

        //스크롤 페인을 설정하고 테이블을 추가
        JScrollPane scrollPane = new JScrollPane(table);

        //패널을 스크롤 페인에 추가
        add(scrollPane);
        setVisible(true);
    }
```


```
public void tableChanged(TableModelEvent e) {
        int row = e.getFirstRow();
        int column = e.getColumn();
        if( column == 2 ){
            TableModel model = (TableModel)e.getSource();
            String columnName=model.getColumnName(column);
            Object data = model.getValueAt(row, column);
            String s = (String)data;
            if( Integer.parseInt(s) > 100)
                JOptionPane.showMessageDialog(this, "범위를 초과하였습니다: " + s,  "경
고", JOptionPane.WARNING_MESSAGE);
        }
    }

    public static void main(String[] args) {
        new SimpleTableTest();
    }
}
```



Q & A