

# Swing Component

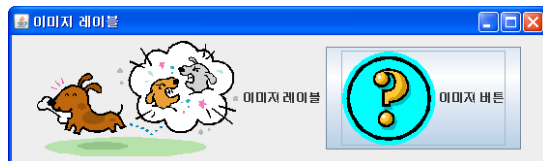
514760  
2020년 봄학기  
6/2/2020  
박경신

## Overview

- JLabel, ImageIcon
- JTextField, JTextArea
- JButton, JCheckBox, JRadioButton
- JSlider, JComboBox
- JPanel, JScrollPane, JOptionPane
- JTable
- Timer
- JFileChooser, JMenu
- JDialog
- 사용자 정의 컴포넌트

## 스윙 컴포넌트에 이미지 표시하기

- 모든 스윙 컴포넌트에는 텍스트 옆에 이미지를 추가로 표시할 수 있다.



```
ImageIcon image = new ImageIcon("image.gif");  
  
JLabel label = new JLabel("이미지 레이블");  
label.setIcon(image);
```

## 이미지 표시하기

```
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) {  
        ImageIcon dog = new ImageIcon("dog.gif");  
        label.setIcon(dog);  
        label.setText(null);  
    }  
}
```



## 체크 박스

- 체크 박스(check box)란 사용자가 클릭하여서 체크된 상태와 체크되지 않은 상태 중의 하나로 만들 수 있는 컨트롤



## JCheckBox의 메소드

	이름	설명
생성자	JCheckBox()	레이블이 없는 Checkbox를 생성한다.
	JCheckbox(String label)	지정된 레이블의 Checkbox를 생성한다.
	JCheckbox(String label, boolean selected)	지정된 상태와 레이블을 가지는 Checkbox를 생성한다.
메소드	String getText()	체크 박스에 표시되는 텍스트를 가져온다.
	Boolean isSelected()	만약 체크 박스가 선택되었으면 true를 반환한다.
	void setSelected(boolean value)	매개 변수가 true이면 체크 박스를 체크 상태로 만든다.
	void setText(String text)	체크 박스 텍스트를 설정한다.

```

public class CheckBoxPanel extends JPanel implements ItemListener {
    JCheckBox[] buttons = new JCheckBox[3]; String[] fruits = { "apple", "grape", "orange" };
    JLabel[] pictureLabel = new JLabel[3]; ImageIcon[] icon = new ImageIcon[3];
    public CheckBoxDemo() {
        super(new GridLayout(0, 4));
        for (int i = 0; i < 3; i++) {
            buttons[i] = new JCheckBox(fruits[i]);
            buttons[i].addItemListener(this);
            pictureLabel[i] = new JLabel(fruits[i] + ".gif");
            icon[i] = new ImageIcon(fruits[i] + ".gif");
        }
        JPanel checkPanel = new JPanel(new GridLayout(0, 1));
        for (int i = 0; i < 3; i++)
            checkPanel.add(buttons[i]);
        add(checkPanel);
        add(pictureLabel[0]);
        add(pictureLabel[1]);
        add(pictureLabel[2]);
    }
    public void ItemStateChanged(ItemEvent e) {
        ImageIcon image = null;
        Object source = e.getItemSelectable();
        for (int i = 0; i < 3; i++) {
            if (source == buttons[i]) {
                if (e.getStateChange() == ItemEvent.DESELECTED)
                    pictureLabel[i].setIcon(null);
                else
                    pictureLabel[i].setIcon(icon[i]);
            }
        }
    }
}

public static void main(String[] args) {
    JFrame frame = new JFrame("CheckBoxDemo");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    CheckBoxPanel panel = new CheckBoxPanel();
    panel.setOpaque(true);
    frame.add(panel);
    frame.setSize(500, 200);
    frame.setVisible(true);
}
    
```

## 라디오 버튼

- 라디오 버튼은 체크 박스와 비슷하지만 하나의 그룹 안에서는 한 개의 버튼만 선택할 수 있다는 점이 다르다



## 라디오 버튼

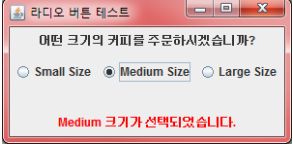
- 라디오 버튼을 생성한다.
  - `JRadioButton radio1 = new JRadioButton("선택 1");`
  - `JRadioButton radio2 = new JRadioButton("선택 2");`
  - `JRadioButton radio3 = new JRadioButton("선택 3");`
- `ButtonGroup` 객체를 생성한다.
  - `ButtonGroup group = new ButtonGroup();`
- 라디오 버튼들을 `ButtonGroup` 객체에 추가한다.
  - `group.add(radio1);`
  - `group.add(radio2);`
  - `group.add(radio3);`

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

class RadioButtonFrame extends JFrame implements ActionListener {
    private JRadioButton small, medium, large; private JLabel text;
    private JPanel topPanel, sizePanel, resultPanel;
    public RadioButtonFrame() {
        setTitle("라디오 버튼 테스트");
        setSize(300, 150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        topPanel = new JPanel();
        JLabel label = new JLabel("어떤 크기의 커피를 주문하시겠습니까?");
        topPanel.add(label);
        add(topPanel, BorderLayout.NORTH);
        sizePanel = new JPanel();
        small = new JRadioButton("Small Size");
        medium = new JRadioButton("Medium Size");
        large = new JRadioButton("Large Size");
        ButtonGroup size = new ButtonGroup();
        size.add(small);
        size.add(medium);
        size.add(large);
        small.addActionListener(this);
        medium.addActionListener(this);
        large.addActionListener(this);
        sizePanel.add(small);
        sizePanel.add(medium);
        sizePanel.add(large);
        add(sizePanel, BorderLayout.CENTER);
        resultPanel = new JPanel();
        text = new JLabel("크기가 선택되지 않았습니다.");
        text.setForeground(Color.red);
        resultPanel.add(text);
        add(resultPanel, BorderLayout.SOUTH);
        setVisible(true);
    }

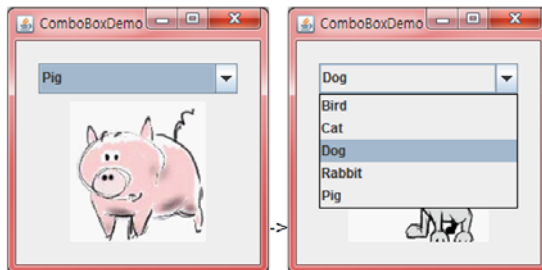
    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == small) {
            text.setText("Small 크기가 선택되었습니다.");
        }
        if (e.getSource() == medium) {
            text.setText("Medium 크기가 선택되었습니다.");
        }
        if (e.getSource() == large) {
            text.setText("Large 크기가 선택되었습니다.");
        }
    }
}

public class RadioButtonTest extends JFrame {
    public static void main(String[] args) {
        new RadioButtonFrame();
    }
}
```



## 콤보박스

- 콤보 박스(combo box)도 여러 항목 중에서 하나를 선택하는데 사용할 수 있다.



## 콤보박스 메소드

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

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

- 여기에 항목을 추가하려면 `addItem()` 메소드를 사용한다.

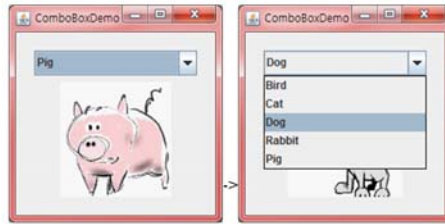
```
combo.addItem("dog");
combo.addItem("lion");
combo.addItem("tiger");
```

## 경계 만들기

- **경계(border)**란 시각적으로 컴포넌트들을 그룹핑할 때 사용하는 장식적인 요소
  - `Border border = BorderFactory.createTitledBorder("크기");`
  - `sizePanel.setBorder(border);`

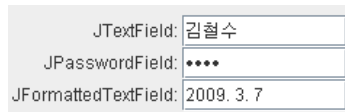


```
public class ComboBoxFrame extends JFrame implements ActionListener {
    JLabel label;
    public ComboBoxFrame() {
        setTitle("콤보 박스");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 200);
        String[] animals = {"Bird", "Cat", "Dog", "Rabbit", "Pig"};
        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) {
        ComboBoxFrame frame = new ComboBoxFrame();
    }
}
```



## 텍스트 필드

- 텍스트 필드(text field)는 입력이 가능한 한 줄의 텍스트 필드를 만드는 데 사용된다.
- 패스워드 필드에 사용자가 암호를 입력하면 글자들이 모두 \* 문자로 표시된다.



```
JTextField textfield = new JTextField(30); // 30자 크기의 텍스트 필드를 만든다.
JTextField textfield = new JTextField("Initial String"); // 초기화 문자열
```

```
System.out.println(textField.getText());
```

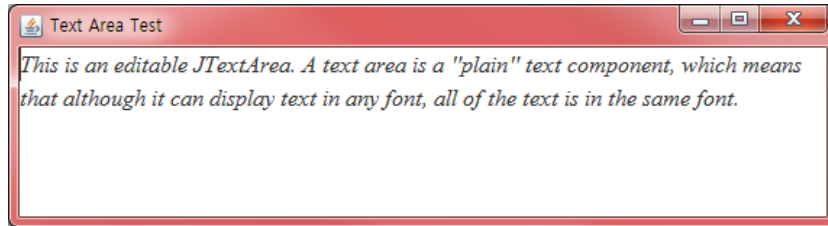
```
textField.setText("Seoul");
```

## 패스워드 필드

- 패스워드 필드에 사용자가 암호를 입력하면 글자들이 모두 \* 문자로 표시된다.

## 텍스트 영역

- 텍스트 영역(TextArea): 여러 줄의 텍스트가 들어 갈 수 있는 컴포넌트

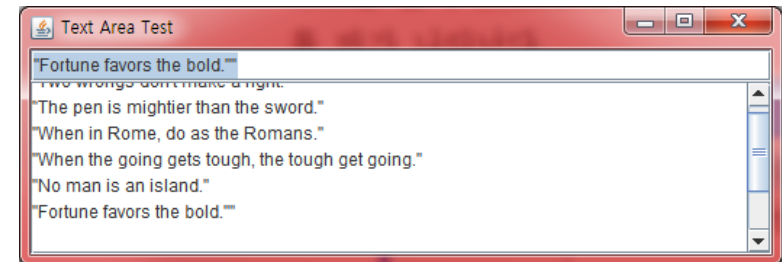


생성자	설명
JTextArea()	비어있는 새로운 JTextArea를 생성한다.
JTextArea(int rows, int columns)	지정된 줄수와 칸수의 비어있는 JTextArea를 생성한다.
JTextArea(String text)	지정된 Text가 입력된 JTextArea를 생성한다.

## 스크롤 페인

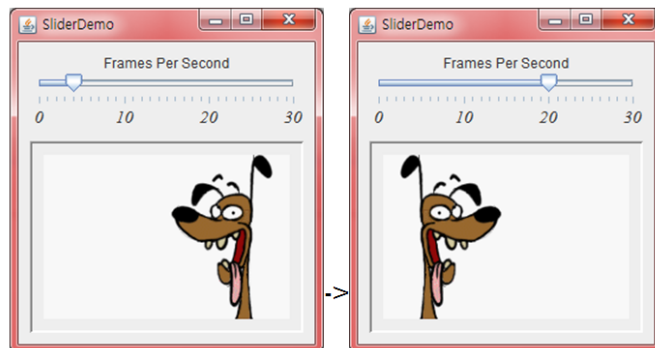
- 텍스트 영역에 스크롤바를 만들려면 스크롤 페인에 텍스트 영역을 넣어야 한다.

```
textArea = new JTextArea(10, 30); // 텍스트 영역을 생성한다.
JScrollPane scrollPane = new JScrollPane(textArea); //
```



## 슬라이더

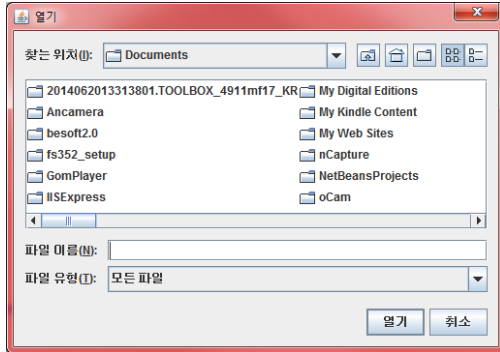
- 슬라이더(slider)는 사용자가 특정한 범위 안에서 하나의 값을 선택할 수 있는 컴포넌트이다.



```
public class SliderFrame extends JFrame implements ChangeListener {
    static final int INIT_VALUE = 15; private JButton buttonOK; private JSlider slider; private JButton button;
    public SliderFrame() {
        JPanel panel;
        setTitle("슬라이더 테스트");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        panel = new JPanel();
        JLabel label = new JLabel("슬라이더를 움직여보세요", JLabel.CENTER);
        label.setAlignmentX(Component.CENTER_ALIGNMENT);
        panel.add(label);
        slider = new JSlider(0, 30, INIT_VALUE);
        slider.setMajorTickSpacing(10); // 큰 눈금 간격
        slider.setMinorTickSpacing(1); // 작은 눈금 간격
        slider.setPaintTicks(true); // 눈금을 표시한다.
        slider.setPaintLabels(true); // 값을 레이블로 표시한다.
        slider.addChangeListener(this); // 이벤트 리스너를 붙인다.
        panel.add(slider);
        button = new JButton("");
        ImageIcon icon = new ImageIcon("dog.gif");
        button.setIcon(icon);
        button.setSize(INIT_VALUE * 10, INIT_VALUE * 10);
        panel.add(button);
        add(panel);
        setSize(300, 300);
        setVisible(true);
    }
    public void stateChanged(ChangeEvent e) {
        JSlider source = (JSlider) e.getSource();
        if (!source.getValueIsAdjusting()) {
            int value = (int) source.getValue();
            button.setSize(value * 10, value * 10);
        }
    }
    public static void main(String[] args) {
        new SliderFrame();
    }
}
```

## 파일 선택기

- 파일 선택기(File Chooser)는 파일 시스템을 탐색하여 파일이나 디렉토리를 선택하는 GUI를 제공한다. 파일 선택기를 표시하려면 JFileChooser 클래스를 사용한다.



```
public void actionPerformed(ActionEvent e) {
    // "파일 오픈"버튼에 대한 액션 이벤트 처리
    if (e.getSource() == openButton) {
        int returnVal = fc.showOpenDialog(this);
        if (returnVal == JFileChooser.APPROVE_OPTION) {
            File file = fc.getSelectedFile();
            // 실제 파일을 오픈한다.
        } else {
            // 사용자 취소
        }

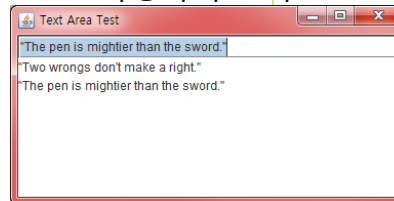
        // "파일 저장"버튼에 대한 액션 이벤트 처리
    } else if (e.getSource() == saveButton) {
        int returnVal = fc.showSaveDialog(this);
        if (returnVal == JFileChooser.APPROVE_OPTION) {
            File file = fc.getSelectedFile();
            // 실제 파일에 저장한다.
        } else {
            // 사용자 취소
        }
    }
}

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

```
class TextAreaFrame extends JFrame implements ActionListener {
    protected JTextField textField;
    protected JTextArea textArea;
    public TextAreaFrame() {
        setTitle("Text Area Test");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        textField = new JTextField(30);
        textField.addActionListener(this);
        textArea = new JTextArea(10, 30);
        textArea.setEditable(false);
        add(textField, BorderLayout.NORTH);
        add(textArea, BorderLayout.CENTER);
        pack();
        setVisible(true);
    }
    public void actionPerformed(ActionEvent evt) {
        String text = textField.getText();
        textArea.append(text + "\n");
        textField.selectAll();
        textArea.setCaretPosition(textArea.getDocument().getLength());
    }
}

public class TextAreaTest extends JFrame {
    public static void main(String[] args) {
        new TextAreaFrame();
    }
}
```

- 사용자가 텍스트 필드에 텍스트를 입력하고 엔터키를 누르면 이것을 텍스트 영역에 추가하는 프로그램을 작성하여 보자.



## Menu

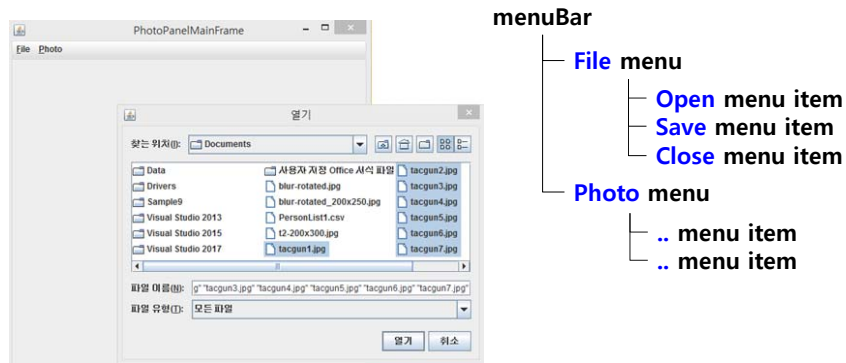
- 메뉴
  - 프로그램에서 수행할 수 있는 명령집합
  - 계층적요소로 구성된 user interface
- JMenuBar (최상위 메뉴바)
 

```
JMenuBar menuBar = new JMenuBar();
this.setJMenuBar(menuBar); // this 프레임에 메뉴바 추가
```
- JMenu
 

```
JMenu fileMenu = new JMenu("File");
fileMenu.setMnemonic(KeyEvent.VK_F);
menuBar.add(fileMenu);
```
- JMenuItem 명령을 수행하거나 서브메뉴를 가지는 항목
 

```
JMenuItem openMenuItem = new JMenuItem("Open", KeyEvent.VK_O);
fileMenu.add(openMenuItem);
openMenuItem.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) { ..... }
});
```

## Menu

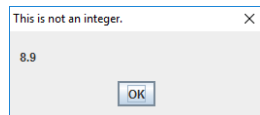
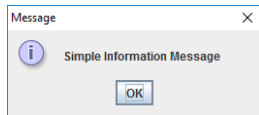


## Dialog

- 대화상자 (JDialog)
  - 다양한 컨트롤을 포함하고 있는 일종의 윈도우
  - 사용자로부터 입력을 받거나 정보를 출력
  - 정적, 버튼, 편집 등 다양한 컨트롤들을 배치하고 관리하는 윈도우

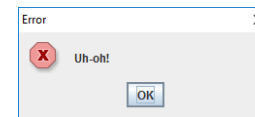
## Message Dialog

- void JOptionPane.showMessageDialog(Component, Object, String, int, Icon)
  - JOptionPane.showMessageDialog(null, "Simple Information Message");
  - JOptionPane.showMessageDialog(null, 8.9, "This is not an integer.", JOptionPane.PLAIN\_MESSAGE);



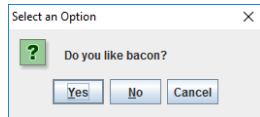
## Message Dialog

- JOptionPane.showMessageDialog(null, "Uh-oh!", "Error", JOptionPane.ERROR\_MESSAGE);
- ImageIcon icon = new ImageIcon("src/images/turtle64.png");
- JOptionPane.showMessageDialog(null, "I like turtles.", "Customized Dialog", JOptionPane.INFORMATION\_MESSAGE, icon);

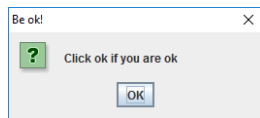


## Message Dialog

- int JOptionPane.showConfirmDialog(Component, Object, String, int, int, Icon)
  - int input = JOptionPane.showConfirmDialog(null, "Do you like bacon?"); // 0:yes 1:no 2:cancel

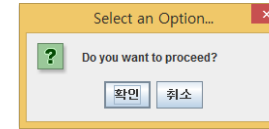


- int input = JOptionPane.showConfirmDialog(null, "Click ok if you are ok", "Be ok!", JOptionPane.DEFAULT\_OPTION); // 0:ok

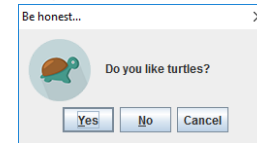


## Message Dialog

- int input = JOptionPane.showConfirmDialog(null, "Do you want to proceed?", "Select an Option...", JOptionPane.OK\_CANCEL\_OPTION); // 0:ok 2:cancel

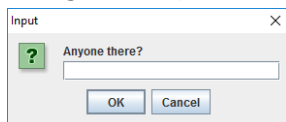


- ImageIcon icon = new ImageIcon("src/images/turtle64.png");
- int input = JOptionPane.showConfirmDialog(null, "Do you like turtles?", "Be honest...", JOptionPane.YES\_NO\_CANCEL\_OPTION, JOptionPane.QUESTION\_MESSAGE, icon);

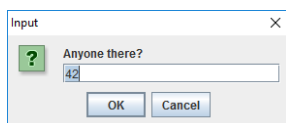


## Message Dialog

- Object JOptionPane.showInputDialog(Component, Object, String, int, Icon, Object[], Object)
  - String m = JOptionPane.showInputDialog("Anyone there?");

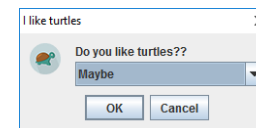


- String m = JOptionPane.showInputDialog("Anyone there?", 42);
- if(m.isEmpty() || m.equals("42")){ frame.close(); }



## Message Dialog

- String[] options = {"I adore turtles", "Yes", "Maybe", "Urm...", "No", "Hate them"};
- ImageIcon icon = new ImageIcon("src/images/turtle32.png");
- String m = (String)JOptionPane.showInputDialog(null, "Do you like turtles??", "I like turtles", JOptionPane.QUESTION\_MESSAGE, icon, options, options[2]);

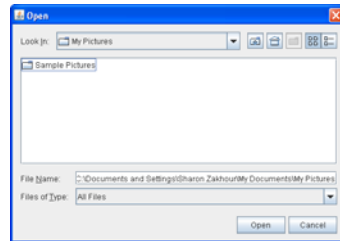




## JFileChooser

### □ int JFileChooser.showOpenDialog(Component)

```
public File[] openFileDialog() {
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setMultiSelectionEnabled(true);
    int returnValue = fileChooser.showOpenDialog(null);
    if (returnValue == JFileChooser.APPROVE_OPTION) {
        System.out.println("openFile");
        File[] selectedFiles = fileChooser.getSelectedFiles();
        return selectedFiles;
    }
    else {
        System.out.println("file not opened");
        return null;
    }
}
```



## JFileChooser

### □ int JFileChooser.showSaveDialog(Component)

```
public String saveFileDialog() {
    JFileChooser fileChooser = new JFileChooser();
    int returnValue = fileChooser.showSaveDialog(null);
    if (returnValue == JFileChooser.APPROVE_OPTION) {
        System.out.println("saveFile");
        String selectedFile = fileChooser.getSelectedFile().getAbsolutePath().toString();
        return selectedFile;
    }
    else {
        System.out.println("file not opened");
        return null;
    }
}
```

## JFileChooser

주요 속성	설명
File getSelectedFile() File[] getSelectedFiles()	대화상자에서 선택한 모든 파일(들) 반환
setCurrentDirectory(File)	시작 디렉터리 지정
setFileFilter(FileFilter)	대화상자에서 "파일 형식으로 저장" 또는 "파일 형식" 상자에 표시되는 선택 옵션을 결정하는 현재 파일 이름필터 문자열을 가져오거나 설정 "Text Document(*.txt)*.txt" + "All Files*.*" "All Image Files*.bmp*.gif*.jpg*.png"
setFileSelectionMode(int)	FILES_ONLY 파일만 선택가능 DIRECTORIES_ONLY 디렉토리만 선택가능 FILES_AND_DIRECTORIES 파일과 디렉토리 선택가능
setMultiSelectionEnabled(boolean)	대화상자에서 여러 개의 파일을 선택가능 여부 지정

## Custom Dialog

### □ 사용자정의 대화상자

- **Modeless** — modeless dialog를 닫지 않더라도 다른 윈도우 사용을 막지 않음
- **Document-modal** — document-modal dialog를 닫지 않으면 같은 document 안의 (자기 자식 윈도우만 제외하고) 모든 윈도우 사용을 막음.
- **Application-modal** — application-modal dialog를 닫지 않으면 같은 application 안의 (자기 자식 윈도우만 제외하고) 모든 윈도우 사용을 막음
- **Toolkit-modal type** — toolkit-modal dialog 를 닫지 않으면 같은 toolkit 안의 (자기 자식 윈도우만 제외하고) 모든 윈도우 사용을 막음.

## Custom Component

- 사용자 정의 컴포넌트 작성
  - 기존 Swing Component를 상속받아 확장
- Number만 입력하는 TextField를 작성 (Custom TextField)

- NumberTextField는 기존 JTextField를 상속받아 확장

```
public void processKeyEvent(KeyEvent e) {
    char c = e.getKeyChar();
    if (!Character.isDigit(c) && !isValidSign(c) && c != '.' && c != '\b') {
        e.consume();
    }
    super.processKeyEvent(e);
    return;
}
private boolean isValidSign(char c) {
    if (getText() == null || "".equals(getText().trim()) && c == '-')
        return true;
    return false;
}
```

## AnimatedImagePanel (Custom Panel)

```
public class ImagePanel extends JPanel implements ActionListener {
    public ImagePanel() {
        loadImages();
        timer = new Timer(500, this);
    }
    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        Graphics2D g2 = (Graphics2D) g;
        if (images[currentImage] != null)
            g2.drawImage(images[currentImage], 0, 0, this);
    }
    public void actionPerformed(ActionEvent e) { // timer tick
        currentImage = (currentImage + 1) % totalImages;
        repaint();
    }
}
```

