19장 스윙과 이벤트 처리

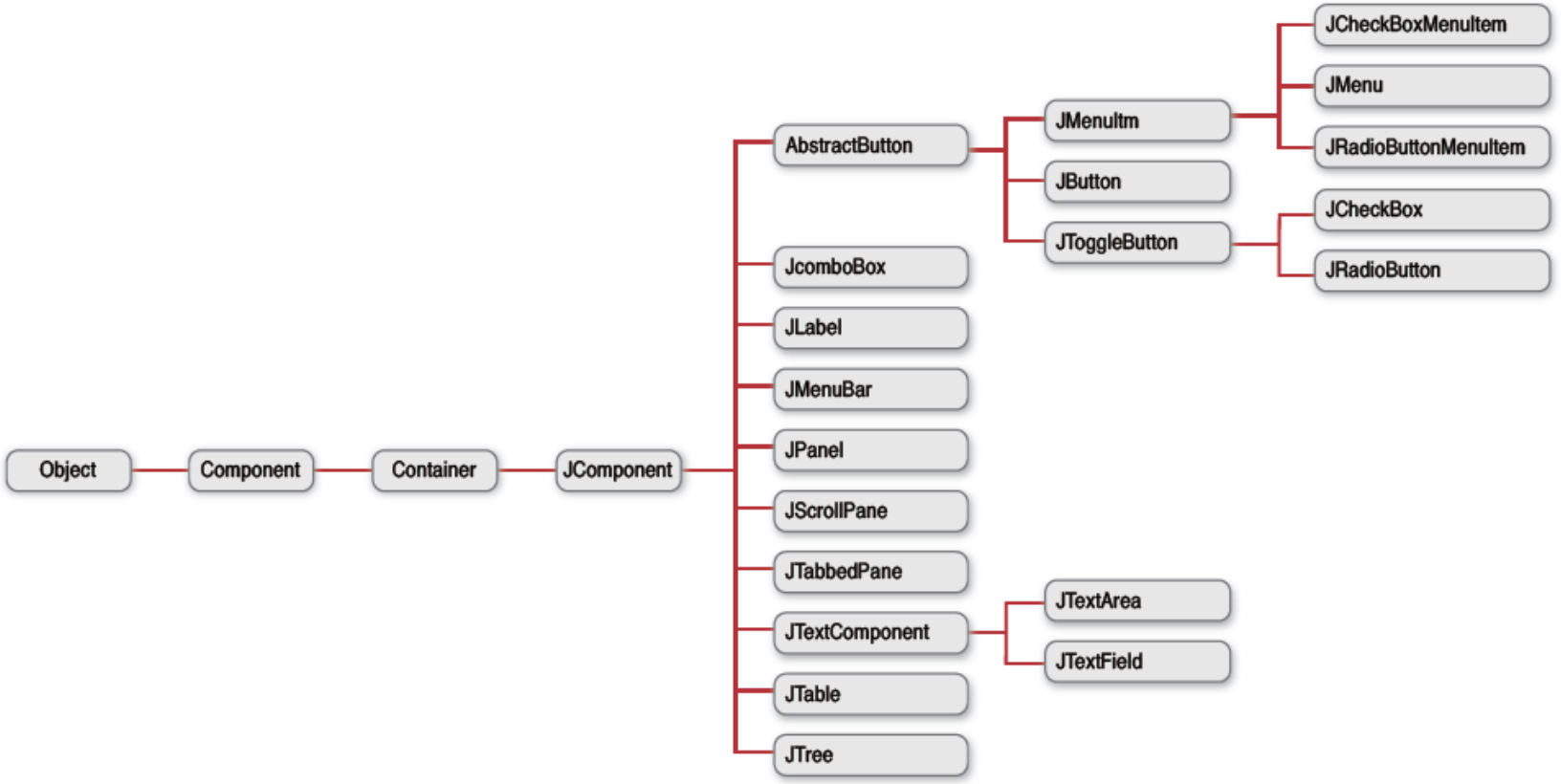
• 학습 목표

- 사용자 인터페이스인 스윙 컴포넌트의 사용에 관해 학습합니다.
- 스윙의 다양한 컴포넌를 사용하여 GUI를 구축에 관해 학습합니다.
- 컴포넌트에서 이벤트 처리를 반복 학습합니다.

1-1 스윙(javax.swing) 패키지의 개요

- **스윙(Swing)**
  - 선(sun)사가 넷스케이프사와 공동으로 개발한 새로운 그래픽 툴킷(toolkit)
  - 그래픽을 JVM이 자체적으로 처리
  - 컴포넌트를 그리기 위해 실행 중인 컴퓨터에서 하나의 컴포넌트(예를 들면 프레임 윈도)만을 사용
  - 스윙과 관련된 모든 클래스들은 javax.swing 패키지에 포함

1-1 스윙(javax.swing) 패키지의 개요



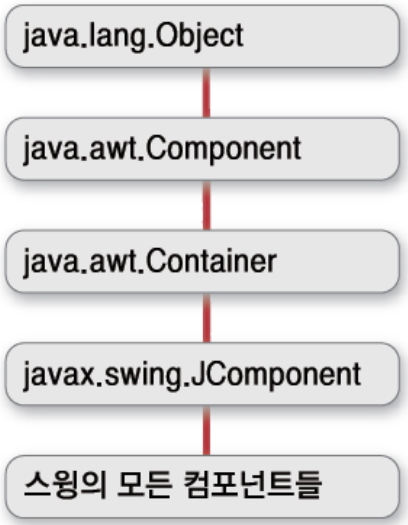
1-1 스윙(javax.swing) 패키지의 개요

클래스	설명
AbstractButton	버튼과 연관된 클래스들의 상위 추상 클래스
ButtonGroup	버튼을 그룹화하기 위한 클래스
ImageIcon	이미지를 아이콘으로 캡슐화하여 제공하는 클래스
JApplet	애플릿(applet)의 스윙 버전으로 Applet 클래스로부터 상속된 클래스
JButton	스윙에서 사용하는 버튼 클래스
JCheckBox	스윙에서 사용하는 체크박스 클래스
JCheckBoxMenuItem	스윙의 메뉴에서 사용하는 체크박스 클래스
JComboBox	풀다운 리스트를 보여줄 수 있는 콤보박스 클래스
JComponent	모든 스윙 컴포넌트의 상위 클래스
JLabel	스윙에서 사용하는 라벨 클래스
JMenu	스윙의 메뉴에서 사용하는 메뉴 클래스
JMenuBar	스윙의 메뉴바를 만들기 위한 클래스
JMenuItem	스윙에서 사용하는 메뉴 관련 클래스의 상위 클래스
JMenuItem	스윙에서 사용하는 메뉴 관련 클래스의 상위 클래스
JPanel	스윙에서 사용하는 패널 클래스
JRadioButton	스윙에서 사용하는 라디오 버튼 클래스
JRadioButtonMenuItem	스윙의 메뉴에서 사용하는 라디오 버튼 클래스
JScrollPane	스크롤 윈도를 나타내는 클래스
JTabbedPane	그룹을 홀더 형태로 제공하는 윈도를 나타내는 클래스
JTable	데이터를 행과 열로 나타내는 클래스
JTextArea	스윙에서 사용하는 텍스트 에리어 클래스
JTextComponent	스윙에서 사용하는 텍스트 관련 클래스들의 상위 클래스
JTextFeild	스윙에서 사용하는 텍스트 필드 클래스
JToggleButton	JCheckBox 클래스와 JRadioButton 클래스의 상위 클래스
JTree	트리 형태를 나타내는 클래스

1-2 JComponent 클래스

Jcomponent 클래스

- AWT 패키지의 Component 클래스와 Container 클래스의 하위 클래스
- 상위 클래스에서 제공되는 기본적인 기능(메소드)들을 그대로 상속받아 사용할 수 있다는 의미



ImageIcon 클래스

- 이미지를 아이콘화하여 제공하는 클래스

[형식] ImageIcon 클래스의 생성자

```
ImageIcon(String filname)
ImageIcon(URL url)
```

filename : 이미지 파일을 지정하여 아이콘을 생성

url : 웹 주소를 지정하여 아이콘을 생성

메소드 이름	설명
int getIconHeight()	아이콘의 높이를 픽셀 값으로 반환
int getIconWidth()	아이콘의 폭을 픽셀 값으로 반환
void setImage(Image i)	i로 지정된 파일을 이미지로 설정

● 스윙에서 라벨은 JLabel 클래스를 사용

[형식] JLabel 클래스의 생성자

```
JLabel(ImageIcon i)
JLabel(String s)
JLabel(String s, ImageIcon i, int align)
```

i : ImageIcon 객체로 라벨 생성

s : 문자열을 사용하여 라벨 생성

align : 아이콘과 문자열의 위치를 의미. SwingConstants 인터페이스의 상수를 사용
(SwingConstants.LEFT, SwingConstants.RIGHT, SwingConstants.CENTER,
SwingConstants.LEADING, SwingConstants.TRAILING 상수)

3 이미지 아이콘(ImageIcon)과 라벨(Label)

메소드	설명
Icon getIcon()	JLabel 객체가 가지고 있는 Icon 객체를 반환
String getText()	JLabel 객체가 가지고 있는 문자열을 문자열 객체로 반환
void setIcon(Icon i)	JLabel 객체에 i로 지정된 아이콘을 설정
void setText(String s)	JLabel 객체에 s로 지정된 문자열을 설정

3 이미지 아이콘(ImageIcon)과 라벨(Label)

예제 15.1 JLabelTest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04:
05: class JLabel1 extends JFrame implements ActionListener {
06:   private JLabel result = new JLabel();
07:   public ImageIcon i1,i2;
08:   public JLabel1() {
09:     i1 = new ImageIcon("apple.jpg");
10:     i2 = new ImageIcon("pear.jpg");
11:     JButton apple = new JButton("사과");
12:     JButton pear = new JButton("배");
13:     Container ct = getContentPane();
14:     ct.setLayout(new FlowLayout());
15:     ct.add(apple);
16:     ct.add(pear);
17:     ct.add(result);
18:     apple.addActionListener(this);
19:     pear.addActionListener(this);
```

이벤트 처리를 위해 속성으로 선언

이미지 아이콘 객체 생성

버튼 객체에 리스너 등록

3 이미지 아이콘(ImageIcon)과 라벨(Label)

```
20:     setTitle("JLabel Test1");
21:     setSize(300,250);
22:     setVisible(true);
23:   }
24:   public void actionPerformed(ActionEvent ae) {
25:     if(ae.getActionCommand()=="사과") {
26:             result.setText("맛있는 사과입니다");
27:             result.setIcon(i1);
28:     }else {
29:             result.setText("맛있는 배입니다");
30:             result.setIcon(i2);
31:     }
32:   }
33: }
34: public class JLabelTest1 {
35:   public static void main(String[] args) {
36:     new JLabel1();
37:   }
38: }
```

라벨의 아이콘을 설정

버튼에 따라 결과를 출력

- 한 줄의 텍스트 : JTextField
- 여러 줄의 텍스트 : JTextArea

[형식] JTextField, JTextArea 클래스 생성자

```
JTextField()
JTextField(int cols)
JTextField(Sting str, int cols)
JTextField(Sting str)
```

```
JTextArea()
JTextArea(String str)
JTextArea(int rows, int cols)
JTextArea(String str, int rows, int cols)
```

str : 생성될 때 나타낼 문자열

cols, rows : 텍스트 필드 또는 텍스트 에리어의 화면에서의 크기. 입력되는 문자의 수와는 상관 없음

3 텍스트 필드와 텍스트 에리어

메소드 이름	설명
int getColumns(int c)	텍스트 필드의 폭을 반환
String getText()	입력된 내용을 문자열로 반환
void setColumns(int c)	텍스트 필드의 폭을 설정
void setEditable(boolean b)	수정가능 여부를 설정

3 텍스트 필드와 텍스트 에리어

메소드 이름	설명
void append(String str)	str을 새로운 라인에 추가
int getColumns(int c)	텍스트 필드의 폭을 반환
int getLineCount()	현재까지 입력된 라인 수를 반환
int getRows()	텍스트 에리어의 초기 라인 수를 반환
String getText()	입력된 내용을 문자열로 반환
void setColumns(int c)	텍스트 필드의 폭을 설정
void setEditable(boolean b)	수정가능 여부를 설정

3 텍스트 필드와 텍스트 에리어

예제 15.2 JTFJTATest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04:
05: class JTFJTA1 extends JFrame implements  ActionListener {
06:   private JTextField jtf;
07:   private JTextArea jta;
08:   private JLabel jl = new JLabel("입력하십시요");
09:   public JTFJTA1() {
10:     jtf = new JTextField(20);
11:     jta = new JTextArea(10,20);
```

텍스트 객체 선언

라벨 객체 생성

20자 텍스트 객체 생성

10줄 20자 텍스트 객체 생성

3 텍스트 필드와 텍스트 에리어

```
12:     jta.setEditable(false);
13:     Container ct = getContentPane();
14:     ct.setLayout(new FlowLayout());
15:     JPanel p1 = new JPanel();
16:     p1.add(jtf);
17:     p1.add(jta);
18:     ct.add(p1);
19:     ct.add(jl);
20:     jtf.addActionListener(this);
21:     setTitle("JTextField,JTextArea Test1");
22:     setSize(500,300);
23:     setVisible(true);
24:   }
25:   public void actionPerformed(ActionEvent ae) {
26:     if(jta.getLineCount() <= jta.getRows()) {
27:             jta.append(ae.getActionCommand()+"\n");
28:             jtf.setText("");
29:     } else {
30:             jl.setText("입력이 종료되었습니다");
31:             jtf.setEditable(false);
32:     }
33:   }
34: }
35: public class JTFJTATest1 {
36:   public static void main(String[] args) {
37:     new JTFJTA1();
38:   }
39: }
```

텍스트 에리어 수정 불가 설정

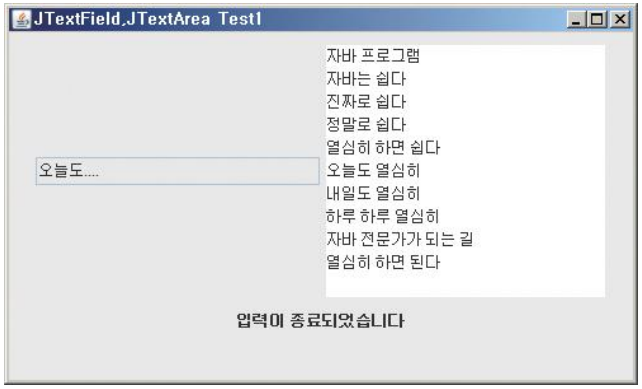
패널 객체 생성

패널에 추가

컨테이너에 패널 추가

텍스트의 크기(줄)보다 작을 때만 추가

크기를 넘어서면 라벨을 출력하고, 텍스트 필드 수정 불가 설정



- **AbstractButton 클래스 : 버튼(버튼, 체크박스 버튼, 라디오 버튼 등)들의 작동을 제어할 수있는 많은 메소들을 제공**

표 19-6 AbstractButton 클래스의 주요 메소드

메소드	설명
String getActionCommand()	버튼이 실행한 명령을 반환
void setActionCommand(String s)	버튼에 대해 새로운 실행 명령을 s로 설정
String getText()	버튼의 텍스트(버튼의 이름으로 지정) 반환
void setText(String text)	버튼의 텍스트를 text로 설정
Icon getIcon()	버튼의 묵시적 아이콘을 반환
void setIcon(Icon icon)	버튼의 묵시적 아이콘을 icon으로 설정
Icon getPressedIcon()	버튼이 눌러졌을 때 나타낼 아이콘을 반환
void setPressedIcon(Icon icon)	버튼이 눌러졌을 때 나타낼 아이콘을 icon으로 설정
Icon getRolloverIcon()	버튼 위로 마우스가 지나갈 때 나타낼 아이콘을 반환
void setRolloverIcon(Icon icon)	버튼 위로 마우스가 지나갈 때 나타낼 아이콘을 icon으로 설정
Boolean isSelected()	버튼의 상태(선택 : true, 비선택 : false)를 반환
void setSelected(Boolean b)	버튼의 선택 상태를 b로 설정

JButton 클래스

[형식] JButton 클래스의 생성자

```
JButton(ImageIcon i)
JButton(String s)
JButton(String s, ImageIcon i)
```

i : ImageIcon 파일

s : 버튼 위에 나타낼 문자열

4 버튼

예제 15.3 JButtonTest1.java

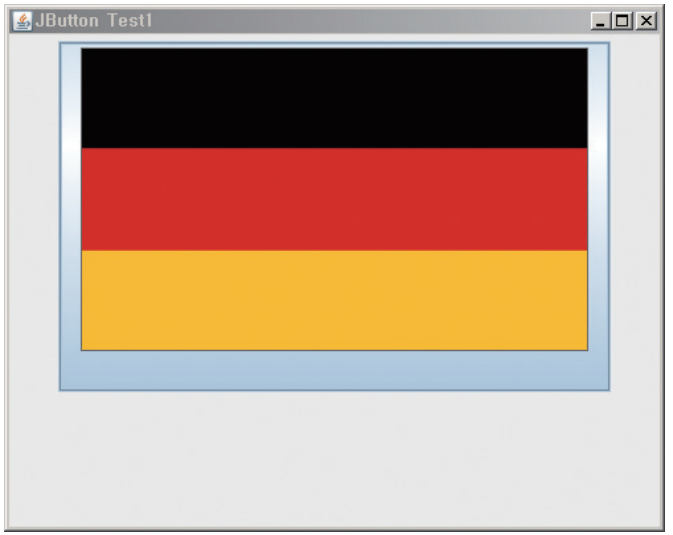
```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04:
05: class JButton1 extends JFrame {
06:   public JButton1() {
07:     ImageIcon korea = new ImageIcon("korea1.gif");
08:     ImageIcon usa = new ImageIcon("usa.gif");
09:     ImageIcon germany = new ImageIcon("germany.gif");
10:     JButton nat = new JButton(korea);
11:     nat.setPressedIcon(usa);
12:     nat.setRolloverIcon(germany);
13:     Container ct = getContentPane();
14:     ct.setLayout(new FlowLayout());
15:     ct.add(nat);
16:     setTitle("JButton Test1");
17:     setSize(500,400);
18:     setVisible(true);
19:   }
20: }
21: public class JButtonTest1 {
22:   public static void main(String[] args) {
23:     new JButton1();
24:   }
25: }
```

이미지 아이콘 객체 생성

이미지를 가지는 버튼 생성

버튼이 눌려졌을 때 나타낼 이미지 설정

버튼위로 마우스가 지나갈 때 나타낼 이미지 설정



예제 15.4 JButtonTest2.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04:
05: class JButton2 extends JFrame implements ActionListener {
06:   JTextField jtf;
07:   public JButton2() {
08:     jtf = new JTextField(10);
09:     ImageIcon apple = new ImageIcon("apple.jpg");
10:     ImageIcon banana = new ImageIcon("banana.jpg");
11:     ImageIcon persimmom = new ImageIcon("persimmom.jpg");
12:     ImageIcon pear = new ImageIcon("pear.jpg");
13:     ImageIcon grape = new ImageIcon("grape.jpg");
14:     JButton jb1 = new JButton("사과",apple);
15:     JButton jb2 = new JButton("바나나",banana);
16:     JButton jb3 = new JButton("감",persimmom);
17:     JButton jb4 = new JButton("배",pear);
18:     JButton jb5 = new JButton("포도",grape);
19:     Container ct = getContentPane();
20:     ct.setLayout(new GridLayout(3,2));
```

텍스트 객체 생성

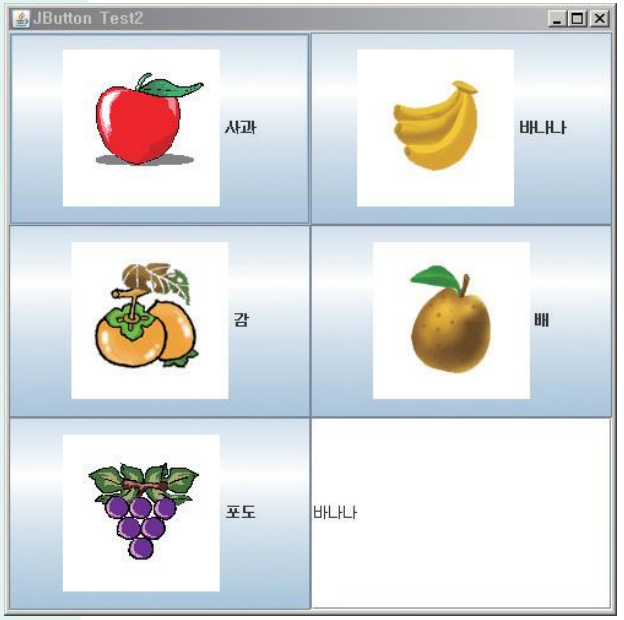
5개의 이미지 객체 생성

이미지와 문자열을 가지는 버튼 객체 생성

컨테이너를 Grid 배치 관리자로 설정

```
21:     ct.add(jb1);
22:     ct.add(jb2);
23:     ct.add(jb3);
24:     ct.add(jb4);
25:     ct.add(jb5);
26:     ct.add(jtf);
27:     jb1.addActionListener(this);
28:     jb2.addActionListener(this);
29:     jb3.addActionListener(this);
30:     jb4.addActionListener(this);
31:     jb5.addActionListener(this);
32:     setTitle("JButton Test2");
33:     setSize(250,300);
34:     setVisible(true);
35:   }
36:   public void actionPerformed(ActionEvent ae) {
37:     jtf.setText(ae.getActionCommand());
38:   }
39: }
40: public class JButtonTest2 {
41:   public static void main(String[] args) {
42:     new JButton2();
43:   }
44: }
```

눌려진 버튼 객체의 라벨을 출력

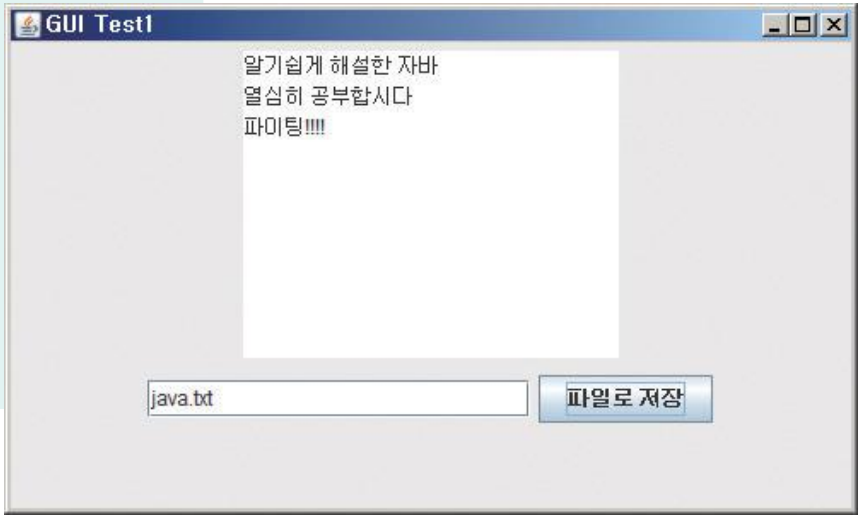


예제 15.5 GUITest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04: import java.io.*; <---------- 입출력 패키지 포함
05:
06: class GUI1 extends JFrame implements  ActionListener {
07:   private JTextField jtf;
08:   private JTextArea jta;
09:
10:   public GUI1() {
11:     JButton jb = new JButton("파일로 저장");
12:     jtf = new JTextField("파일이름을 입력하세요",20);
13:     jta = new JTextArea("파일의 내용을 입력하세요", 10,20);
14:     Container ct = getContentPane();
15:     ct.setLayout(new FlowLayout());
16:     JPanel p1 = new JPanel();
17:     ct.add(jta);
18:     p1.add(jtf);
19:     p1.add(jb);
20:     ct.add(p1);
21:     jb.addActionListener(this);
22:     setTitle("GUI Test1");
23:     setSize(500,300);
24:     setVisible(true);
25:   }
26:   public void actionPerformed(ActionEvent ae) {
```

4 버튼

```
27:     try {                                          ← 입출력을 위한 예외 처리
28:             String s = jtf.getText();
29:             FileOutputStream fos = new FileOutputStream(s);   ← 파일 이름으로
30:             DataOutputStream dos = new DataOutputStream(fos); ← 출력 객체 생성
31:             dos.writeUTF(jta.getText());           ← 텍스트 에리어의 내용을 파일로 출력
32:             fos.close();
33:             System.out.print(s + "파일이 생성되었습니다");
34:             }
35:     catch(Exception e) {}                          ← 입출력을 위한 예외 처리
36:   }
37: }
38: public class GUITest1 {
39:   public static void main(String[] args) {
40:     new GUI1();
41:   }
42: }
```



예제 15 6 GUITest2.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04: import java.io.*;
05:
06: class GUI2 extends JFrame implements  ActionListener {
07:   private JTextField jtf;
08:   private JTextArea jta;
09:
10:   public GUI2() {
11:     JButton jb = new JButton("파일 읽어오기");
12:     jtf = new JTextField("파일이름을 입력하세요",20);
13:     jta = new JTextArea(10,20);
14:     jta.setEditable(false);
15:     Container ct = getContentPane();
16:     ct.setLayout(new FlowLayout());
17:     JPanel p1 = new JPanel();
18:     p1.add(jtf);
19:     p1.add(jb);
20:     ct.add(p1);
21:     ct.add(jta);
22:     jb.addActionListener(this);
23:     setTitle("GUI Test2");
24:     setSize(400,400);
25:     setVisible(true);
26:   }
```

4 버튼

```
29:             String s = jtf.getText();
30:             FileInputStream fis = new FileInputStream(s);
31:             DataInputStream dis = new DataInputStream(fis);
32:             jta.setText(dis.readUTF());
33:             fis.close();
34:             System.out.print(s + "파일을 읽었습니다");
35:             }
36:     catch(Exception e) {
37:             System.out.print("파일이 존재하지 않습니다");
38:     }
39:   }
40: }
41: public class GUITest2 {
42:   public static void main(String[] args) {
43:     new GUI2();
44:   }
45: }
```

파일 이름으로 입력 객체 생성

파일의 내용을 텍스트 에리어의 내용으로 설정

JCheck 클래스

[형식] JCheckBox 클래스의 생성자

```
JCheckBox(Icon I)
JCheckBox(Icon i, boolean state)
JCheckBox(String s)
JCheckBox(String s, boolean state)
JCheckBox(String s, Icon I)
JCheckBox(String s, Icon i, boolean state)
```

i,s : 체크박스 버튼에 나타낼 아이콘과 문자열

state : state가 참이면 선택된 상태로 체크박스를 표시

표 19-7 JCheckBox 클래스의 메소드

메소드 이름	설명
void setSelected(boolean state)	체크박스의 상태를 설정
Object[] getSelectedObjects()	선택된 요소들을 Object의 배열로 반환

5 체크 박스

예제 15.7 JCheckBoxTest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04: class JCheckBox1 extends JFrame implements ItemListener {
05:   JTextField jtf;
06:   public JCheckBox1() {
07:     jtf = new JTextField(10);
08:     JCheckBox jc1 = new JCheckBox("JSP");
09:     JCheckBox jc2 = new JCheckBox("PHP");
10:     JCheckBox jc3 = new JCheckBox("ASP");
11:     JCheckBox jc4 = new JCheckBox("Servlet");
12:     Container ct = getContentPane();
13:     ct.setLayout(new FlowLayout());
14:     ct.add(jc1);
15:     ct.add(jc2);
16:     ct.add(jc3);
17:     ct.add(jc4);
18:     ct.add(jtf);
```

Item 리스너를 포함하여 작성

체크박스 객체 생성

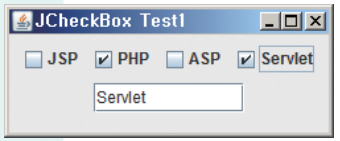
5 체크 박스

```
19:     jc1.addItemListener(this);
20:     jc2.addItemListener(this);
21:     jc3.addItemListener(this);
22:     jc4.addItemListener(this);
23:     setTitle("JCheckBox Test1");
24:     setSize(250,100);
25:     setVisible(true);
26:   }
27:   public void itemStateChanged(ItemEvent ie) {
28:     jtf.setText(((JCheckBox)ie.getItem()).getText());
29:   }
30: }
31: public class JCheckBoxTest1 {
32:   public static void main(String[] args) {
33:     new JCheckBox1();
34:   }
35: }
```

체크박스에 리스너 등록

이벤트 처리 루틴 작성

getItem() 메소드를 사용하여 이벤트 객체 판별



JRadioButton 클래스

[형식] JRadioButton 클래스의 생성자

```
JRadioButton(Icon i)
JRadioButton(Icon i, boolean state)
JRadioButton(String s)
JRadioButton(String s, boolean state)
JRadioButton(String s, Icon I)
JRadioButton(String s, Icon i, boolean state)
```

i,s : 체크박스 버튼에 나타낼 아이콘과 문자열

state : state가 참이면 선택된 상태로 라디오 버튼을 표시

표 19-8 JRadioButton 클래스의 메소드

메소드 이름	설명
void setSelected(boolean state)	라디오 버튼의 상태를 설정
Object[] getSelectedObjects()	선택된 요소들을 Object의 배열로 반환

● 버튼을 그룹으로 묶는 기능 : ButtonGroup 클래스

[형식] ButtonGroup 클래스의 생성자

```
ButtonGroup()
```

표 19-9 ButtonGroup 클래스의 메소드

메소드 이름	설명
void add()	버튼 그룹에 버튼을 추가. 버튼 그룹에 속한 버튼은 하나만 선택 가능

6 라디오 버튼

예제 15 8 JRadioButtonTest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04:
05: class JRadioButton1 extends JFrame implements ActionListener {
06:   JLabel jl;
07:   JRadioButton[] jr = new JRadioButton[5];
08:   String[] hobby = {"걷기","등산","골프","스킨스쿠버","페러글라이딩"};
09:   public JRadioButton1() {
10:     jl = new JLabel();
11:     JLabel jl1 = new JLabel("당신의 취미는?");
12:     JLabel jl2 = new JLabel("당신의 나이는?");
13:     JCheckBox[] jc = new JCheckBox[5];
14:     String[] age = {"20대","30대","40대","50대","60대"};
15:     JPanel hobbypanel = new JPanel();
16:     JPanel agepanel = new JPanel();
17:     ButtonGroup gb = new ButtonGroup();
18:     for(int i = 0 ; i < 5 ; i++) {
19:             jr[i] = new JRadioButton(hobby[i]);
20:             jc[i] = new JCheckBox(age[i]);
21:             hobbypanel.add(jr[i]);
22:             agepanel.add(jc[i]);
```

이벤트 처리루틴에서 사용할 요소를 속성으로 설정

나이를 나타내는 체크박스 버튼을 버튼 그룹으로 설정

취미를 나타내는 라디오 버튼 객체를 배열로 생성

나이를 나타내는 체크박스 객체를 배열로 생성

취미 판넬에 객체 추가

나이 판넬에 객체 추가

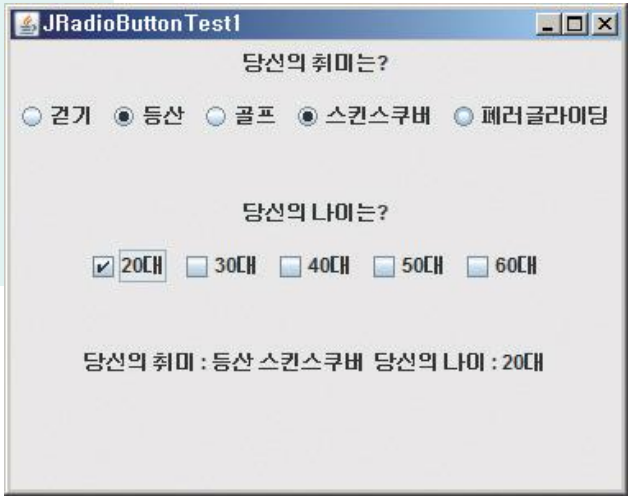
```
23:             jc[i].addActionListener(this);  <-- 체크박스 배열 객체에 리스너 등록
24:             gb.add(jc[i]);  <-- 버튼그룹에 객체 추가
25:     }
26:
27:     Container ct = getContentPane();
28:     ct.setLayout(new GridLayout(3,1));
29:     JPanel jp1 = new JPanel();
30:     jp1.add(jl1);
31:     jp1.add(hobbypanel);
32:     JPanel jp2 = new JPanel();
33:     jp2.add(jl2);
34:     jp2.add(agepanel);
35:     JPanel jp3 = new JPanel();
36:     jp3.add(jl);
37:     ct.add(jp1);
38:     ct.add(jp2);
39:     ct.add(jp3);
40:     setTitle("JRadioButtonTest1");
41:     setSize(350,300);
42:     setVisible(true);
43: }
```

6 라디오 버튼

```
44:   public void actionPerformed(ActionEvent ae) {
45:      String s = "당신의 취미 : " ;
46:      for(int i = 0 ; i < 5 ; i++){
47:               if(jr[i].isSelected()==true)
48:                        s=s+hobby[i]+" ";
49:      }
50:      s=s+" 당신의 나이 : ";
51:      jl.setText(s+ae.getActionCommand());
52:   }
53: }
54: public class JRadioButtonTest1 {
55:   public static void main(String[] args) {
56:      new JRadioButton1();
57:   }
58: }
```

선택된 라디오 버튼을 문자열로 구성

라벨에 결과를 설정



JComboBox 클래스

[형식] JComboBox 클래스의 생성자

```
JComboBox()
JComboBox(Vector v)
```

v : v로 지정된 벡터 객체로 콤보 박스 객체 생성

표 19-10 JComboBox 클래스의 메소드

메소드 이름	설명
void addItem(Object obj)	콤보박스에 obj로 지정된 객체 항목을 추가
Object[] getSelectedObjects()	선택된 요소들을 Object의 배열로 반환
int getSelectedIndex()	선택된 객체의 인덱스를 반환
Object getSelectedItem()	선택된 객체를 반환

7 콤보박스

예제 5.9 JComboBoxTest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04:
05: class JComboBox1 extends JFrame implements ItemListener {
06:   JLabel jl;
07:   public JComboBox1() {
08:     jl = new JLabel();
09:     JComboBox jcb = new JComboBox(); <-- 콤보박스 객체 생성
10:     String fr[] = {"persimmom","banana","pear","apple","cherry","grape"};
11:     Container ct = getContentPane();
12:     ct.setLayout(new FlowLayout());
13:     for(int i = 0 ; i < 6 ; i++){
14:             jcb.addItem(fr[i]); <-- 콤보박스의 리스트에 추가
15:     }
16:     ct.add(jcb);
17:     ct.add(jl);
18:     jcb.addItemListener(this); <-- 콤보박스에 Item 리스너 등록
19:     setTitle("JComboBox Test1");
20:     setSize(200,200);
21:     setVisible(true);
22:   }
23:   public void itemStateChanged(ItemEvent ie) { <-- 이벤트 처리 루틴 작성
24:     String fruit = (String)ie.getItem(); <-- 선택된 항목 추출(표18-10 참조)
25:     jl.setIcon(new ImageIcon(fruit+".jpg")); <-- 라벨의 아이콘으로 선택된 항목 설정
26:   }
27: }
28: public class JComboBoxTest1 {
29:   public static void main(String[] args) {
30:     new JComboBox1();
31:   }
32: }
```

JList 클래스

[형식] JList 클래스의 생성자

```
JList()
JList(Object[] items)
```

items : 리스트로 생성할 배열

표 19-11 JList 클래스의 메소드

메소드	설명
int getSelectedIndex()	리스트 중에서 첫 번째 선택된 아이템의 인덱스를 반환
void setSelectedIndex(int index)	index로 지정된 아이템을 선택
int[] getSelectedIndices()	리스트 중에서 선택된 모든 아이템의 인덱스를 배열로 반환
void setSelectedIndices(int[] indices)	indices 배열에서 지정된 아이템을 모두 선택
Object getSelectedValue()	리스트 중에서 첫 번째 선택된 아이템을 Object 객체로 반환
Object[] getSelectedValues()	리스트에서 선택된 모든 아이템을 Object 객체의 배열로 반환(인덱스 순서)

예제 15 10 JListTest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import javax.swing.event.*;  <-- 스윙 이벤트 패키지 포함
04:
05: class JList1 extends JFrame implements ListSelectionListener {
06:   private String[] fruitlist = {"persimmom","banana","apple","grape",
    "cherry","pear"};  <-- 과일 이름 배열 생성
07:   private JList jlst = new JList(fruitlist);  <-- 이름 배열로 리스트 리스트 객체 생성
08:   private ImageIcon[] fruiticons = {  <-- 아이콘 객체 배열 생성
09:             new ImageIcon("persimmom.jpg"),
10:             new ImageIcon("banana.jpg"),
11:             new ImageIcon("apple.jpg"),
12:             new ImageIcon("grape.jpg"),
13:             new ImageIcon("cherry.jpg"),
14:             new ImageIcon("pear.jpg")
15:   };
16:   private JLabel[] jlicon = new JLabel[6];  <-- 아이콘을 위한 라벨 객체 배열 생성
17:   private JLabel jlname = new JLabel();  <-- 라벨을 저장할 객체 생성
18:
19:   public JList1() {
20:     Container ct = getContentPane();
21:     JPanel p1 = new JPanel();
22:     p1.setLayout(new GridLayout(3,2,5,5));  <-- 아이콘 패널 Grid 배치관리자 설정
23:     for (int i = 0 ; i < 6 ; i++ ){
24:             p1.add(jlicon[i] = new JLabel());
25:     }
26:     JPanel p2 = new JPanel();
27:     p2.add(jlname);
28:     ct.add(jlst, BorderLayout.WEST);  <-- Border 배치관리자로 설정
29:     ct.add(p1, BorderLayout.CENTER);
30:     ct.add(p2, BorderLayout.EAST);
```

```
31:     jlst.addListSelectionListener(this);
32:     setTitle("JList Test1");
33:     setSize(700,300);
34:     setVisible(true);
35:   }
36:   public void valueChanged(ListSelectionEvent lse) {
37:     for (int i=1 ; i < 6 ; i++ ){
38:             jlicon[i].setIcon(null);
39:     }
40:     int[] indices = jlst.getSelectedIndices();
41:     String s = "당신이 선택한 항목은 : ";
42:     for (int j = 0 ; j < indices.length ; j++ ){
43:             jlicon[j].setIcon(fruiticons[indices[j]]);
44:             s = s + fruitlist[indices[j]]+" ";
45:     }
46:     jlname.setText(s);
47:   }
48: }
49: public class JListTest1 {
50:   public static void main(String[] args) {
51:     new JList1();
52:   }
53: }
```

리스트에 리스너 등록

아이콘을 모두 삭제

선택된 항목을 배열로 추출

선택된 아이콘을 표시

선택된 아이콘의 문자열을 생성

선택한 아이콘의 문자열 표시

- **JScrollPane 클래스**

[형식] JScrollPane 클래스의 생성자

```
JScrollPane(Component comp)
JScrollPane(int vsb, int hsb)
JScrollPane(Component comp, int vsb, int hsb)
```

comp : 스크롤바가 추가될 컴포넌트

vsb, hsb : 수직, 수평 스크롤바를 설정하기 위해 사용되는 상수

- **vsb와 hsb는 수직, 수평 스크롤 바를 설정하기 위해 사용되는 상수로서 다음과 같은 상수가 사용됨**
  - HORIZONTAL_SCROLLBAR_ALWAYS : 항상 수평 스크롤 바를 제공
  - HORIZONTAL_SCROLLBAR_AS_NEEDED : 필요한 경우(패널의 크기를 넘어서는 경우)에만 수평 스크롤 바 제공
  - VERTICAL_SCROLLBAR_ALWAYS : 항상 수직 스크롤 바를 제공
  - VERTICAL_SCROLLBARAS_NEEDED : 필요한 경우에만 수직 스크롤 바 제공

예제 15 11 JScrollPaneTest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: class JScrollPane1 extends JFrame {
04:   public JScrollPane1() {
05:     Container ct = getContentPane();
06:     JPanel jp = new JPanel();
07:     jp.setLayout(new GridLayout(20,5));
08:     for(int i = 1 ; i <=100 ; i++ )
09:             jp.add(new JButton(i + "번"));
10:     int v = ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED;
11:     int h = ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED;
12:     JScrollPane jsp = new JScrollPane(jp, v, h);
13:     ct.add(jsp, BorderLayout.CENTER);
14:     setTitle("JScrollPane Test1");
15:     setSize(300,300);
16:     setVisible(true);
17:   }
18: }
19: public class JScrollPaneTest1 {
20:   public static void main(String[] args) {
21:     new JScrollPane1();
22:   }
23: }
```

패널에 100개의 버튼 추가

상수 설정

패널과 상수로 JScrollPane 객체 생성

JTabbedPane 클래스

[형식] JTabbedPane 클래스의 생성자

```
JTabbedPane()
JTabbedPane(int tabPlacement)
```

tabPlacement : 방향과 위치를 지정하는 상수(JTabbedPane.TOP, JTabbedPane.BOTTOM, JTabbedPane.LEFT, or JTabbedPane.RIGHT)

표 19-12 JTabbedPane 클래스의 메소드

메소드 이름	설명
void addTab(String str, Component c)	그룹 홀더에 그룹을 추가. str은 홀더의 제목. c는 폴더의 내용을 가지고 있는 객체

다음은 그룹 홀더를 만들기 위한 일반적인 과정입니다.

① 그룹 홀더에 추가할 각각의 홀더 객체를 JPanel 클래스를 이용하여 모두 작성

② JTabbedPane 클래스를 이용하여 그룹 홀더 객체를 생성

③ 작성된 JPanel 객체를 addTab() 메소드를 이용하여 JTabbedPane 객체에 추가

④ 완성된 그룹 홀더(JTabbedPane 객체)를 프레임에 추가

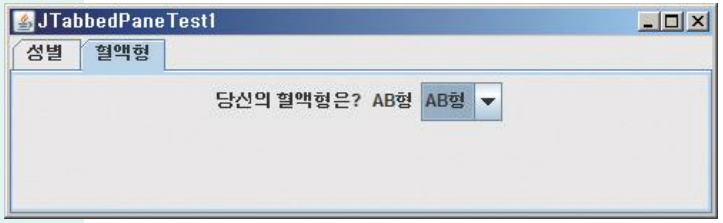
예제 1 5 12 JTabbedPaneTest1.java

```
01: import javax.swing.*;
02: import java.awt.*;
03: import java.awt.event.*;
04:
05: class RadioPanel extends JPanel implements ActionListener {
06:   JLabel jl1 = new JLabel("당신의 성별은? ");
07:   JLabel jl2 = new JLabel("");
08:   public RadioPanel() {
09:     JRadioButton jrb1 = new JRadioButton("남자");
10:     JRadioButton jrb2 = new JRadioButton("여자");
11:     add(jl1);
12:     add(jl2);
13:     add(jrb1);
14:     add(jrb2);
15:     jrb1.addActionListener(this);
16:     jrb2.addActionListener(this);
17:   }
18:   public void actionPerformed(ActionEvent ae) {
19:     String s = ae.getActionCommand();
20:     jl2.setText(s);
21:   }
22: }
23:
24: class ComboPanel extends JPanel implements ItemListener {
25:   JLabel jl1 = new JLabel("당신의 혈액형은? ");
26:   JLabel jl2 = new JLabel("");
27:   public ComboPanel() {
28:     JComboBox jcb = new JComboBox();
```

JPanel 클래스를 상속하여 클래스 작성

JPanel 클래스를 상속하여 클래스 작성

```
29:     jcb.addItem("A형");
30:     jcb.addItem("B형");
31:     jcb.addItem("AB형");
32:     jcb.addItem("O형");
33:     add(jl1);
34:     add(jl2);
35:     add(jcb);
36:     jcb.addItemListener(this);
37:   }
38:   public void itemStateChanged(ItemEvent ie) {
39:     Object s = ie.getItem();
40:     jl2.setText(s.toString());
41:   }
42: }
43:
44: class JTabbedPane1 extends JFrame {          <-- 그룹 폴더를 위한 클래스 생성
45:   public JTabbedPane1() {
46:     JTabbedPane jtp = new JTabbedPane();     <-- 그룹 폴더 객체 생성
47:     RadioPanel rp = new RadioPanel();        <-- 각 패널의 객체를 생성
48:     ComboPanel cp = new ComboPanel();
49:     jtp.addTab("성별", rp);                  <-- 그룹 폴더에 패널 추가
50:     jtp.addTab("혈액형", cp);
51:     getContentPane().add(jtp);               <-- 컨테이너에 그룹 폴더 추가
52:     setTitle("JTabbedPaneTest1");
53:     setSize(500,150);
54:     setVisible(true);
55:   }
56: }
57:
58: public class JTabbedPaneTest1 {
59:   public static void main(String[] args) {
60:     new JTabbedPane1();
61:   }
62: }
```



메뉴 관련 클래스

- JMenuBar, JMenu, JMenuItem, JCheckBoxMenuItem, JradioButtonMenuItem 등

스윙의 메뉴는 다음과 같은 순서로 작성됩니다.

① JMenuBar 객체를 생성

② 메뉴바에 나타낼 JMenu 객체를 생성

③ 각각의 메뉴에 연결되는 서브 메뉴를 JMenuItem 객체로 생성하여 메뉴에 추가

④ 메뉴가 완성될 때까지 2–3번을 반복

⑤ 메뉴가 완성되면 모든 메뉴를 JMenuBar에 추가

⑥ setJMenuBar() 메소드를 이용하여 메뉴바를 프레임에 추가

예제 15 13 JMenuTest1.java

```
01: import javax.swing.*;
02: import java.awt.event.*;
03: import java.awt.*;
04:
05: class JMenu1 extends JFrame implements ActionListener {
06:   JTextField jtf;
07:   public JMenu1() {
08:     Container c = getContentPane();
09:     c.setLayout(new BorderLayout());
10:     JMenuBar jmb = new JMenuBar();
11:     JMenu menu1 = new JMenu("파일");
12:     JMenu menu2 = new JMenu("편집");
13:     JMenuItem jmi = new JMenuItem("새파일");
14:     jmi.addActionListener(this);
15:     menu1.add(jmi);
16:     jmi = new JMenuItem("저장하기");
17:     jmi.addActionListener(this);
18:     menu1.add(jmi);
19:     jmb.add(menu1);
20:     jmi = new JMenuItem("열기");
21:     jmi.addActionListener(this);
22:     menu2.add(jmi);
```

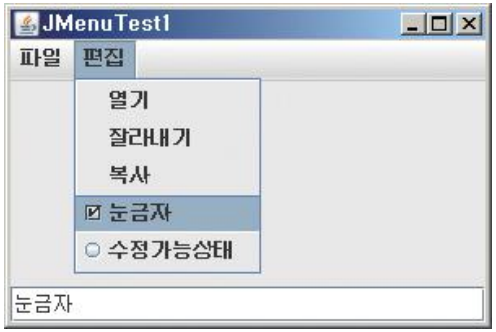
메뉴바 객체 생성

메뉴 객체 생성

메뉴아이템을 생성하여 리스너를 등록하고 메뉴에 추가

메뉴바에 메뉴를 추가

```
23:     jmi = new JMenuItem("잘라내기");
24:     jmi.addActionListener(this);
25:     menu2.add(jmi);
26:     jmi = new JMenuItem("복사");
27:     jmi.addActionListener(this);
28:     menu2.add(jmi);
29:     JCheckBoxMenuItem jcbmi = new JCheckBoxMenuItem("눈금자");
30:     jcbmi.addActionListener(this);
31:     menu2.add(jcbmi);
32:     JRadioButtonMenuItem jrbmb = new JRadioButtonMenuItem("수정가능상태");
33:     jrbmb.addActionListener(this);
34:     menu2.add(jrbmb);
35:     jmb.add(menu2); <-------------------------- 메뉴바에 메뉴를 추가
36:     setJMenuBar(jmb); <-------------------------- 프레임에 메뉴바 추가(표18-4 참조)
37:     jtf = new JTextField();
38:     c.add(jtf, BorderLayout.SOUTH);
39:     setTitle("JMenuTest1");
40:     setSize(300,200);
41:     setVisible(true);
42:   }
43:   public void actionPerformed(ActionEvent ae) {
44:     jtf.setText(ae.getActionCommand());
45:   }
46: }
47: public class JMenuTest1 {
48:   public static void main(String[] args) {
49:     new JMenu1();
50:   }
51: }
```



▪ 학습정리

● 스윙(Swing) 컴포넌트

① javax.swing 패키지는 가장 상위 클래스로 JComponent 클래스를 가진다. 그 의미는 JComponent 클래스에 제공되는 모든 메소드를 사용할 수 있다는 의미이다.

② JComponent 클래스는 java.awt 패키지의 Component, Container 클래스를 상위 클래스로 가진다.

● 컴포넌트에서의 이벤트 처리

컴포넌트에서 전형적인 이벤트 처리는 다음과 같은 순서로 이루어진다.

① 받아들일 이벤트와 연관된 인터페이스를 사용하여 클래스 생성

② GUI 구축을 위한 컴포넌트 객체를 생성

③ 이벤트를 받아들인 컴포넌트 객체(여기서는 버튼 객체)에 리스너를 등록

④ add() 메소드를 사용하여 컴포넌트를 패널(또는 프레임)에 추가

⑤ 이벤트를 처리할 이벤트 처리 루틴 작성(예 : action 이벤트)

▪ 학습정리

● 스윙 컴포넌트의 사용과 이벤트 처리

① 다양한 컴포넌트를 사용하여 사용자 인터페이스를 구축할 수 있다.

② 대부분의 컴포넌트에서 이벤트 처리는 같은 형태로 처리된다.

③ 이벤트를 처리하기 위해서는 우선 컴포넌트에서 발생하는 이벤트가 어떤 종류가 있는지를 판별해야 하고, 그 종류의 이벤트를 처리하기 위해서 사용하는 이벤트 리스너를 선택해야 한다.

④ 이벤트를 처리할 클래스를 작성할 때에는 선별된 이벤트 리스너를 포함하여 작성하여야 하며, 리스너 인터페이스에서 선언된 모든 메소드를 오버라이딩하여 이벤트 처리 루틴을 작성하여야 한다.