

# 자바 Swing

---

이것이 자바다  
(<http://cafe.naver.com/thisisjava>)

신용권  
2015.01.15

무단 복제 및 인터넷에 배포할 수 없습니다.

# 1 절. 목차

---

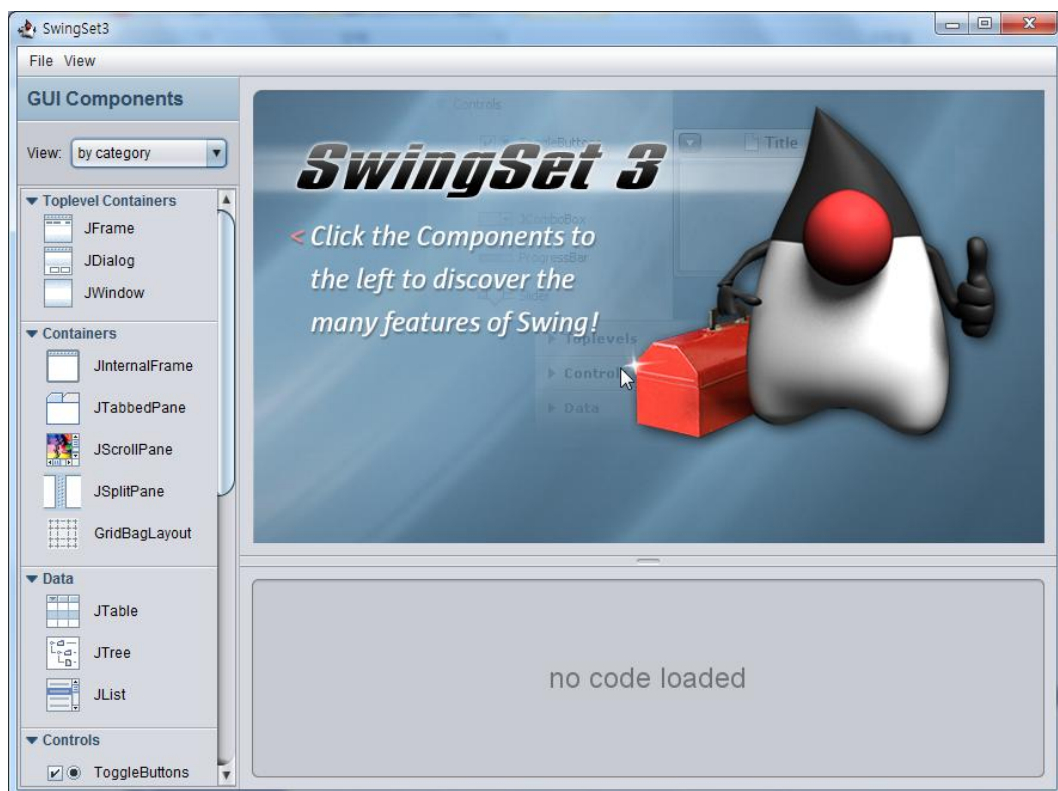
- 1 절 .AWT 와 Swing
- 2 절. Swing 애플리케이션 실행
- 3 절. Swin 컨테이너
- 4 절. 컴포넌트 배치
- 5 절. 이벤트 처리
- 6 절. 버튼 컴포넌트
- 7 절. 텍스트 컴포넌트
- 8 절. 리스트 컴포넌트
- 9 절. 테이블 컴포넌트
- 10 절. 트리 컴포넌트
- 11 절. 메뉴 컴포넌트
- 12 절. 툴바 컴포넌트
- 13 절. 다이얼로그
- 14 절. Java 2D
- [확인 문제]

## 2 절.AWT 와 Swing

UI(User Interface) 프로그램은 윈도우, 메뉴, 버튼, 라디오, 리스트등 시각적인 컴포넌트를 제공해서 사용자와 상호작용하게 된다. 자바는 이러한 UI 프로그램을 개발할 수 있도록 JDK 에서 JFC(Java Foundation Classes)를 제공하고 있다. JFC 는 UI 프로그램을 만들기 위한 클래스들의 모음으로 AWT(추상윈도우툴킷: Abstract Window Toolkit)와 Swing(스윙) 을 제공하고 있다. AWT 는 java.awt 패키지에서, Swing 은 javax.swing 패키지에서 제공되는데, AWT 와 Swing 의 차이점은 AWT 는 운영체제가 가지고 있는 컴포넌트를 그대로 이용하도록 만들어 졌고, Swing 은 자바에서 직접 컴포넌트를 만든다. AWT 는 운영 체제들이 공통적으로 가지고 있는 컴포넌트만 사용하므로 컴포넌트 수가 제한적이지만, Swing 은 자바에서 직접 제공하는 컴포넌트이기 때문에 다양하다. Swing 의 단점은 자바가 직접 컴포넌트를 생성하기 때문에 AWT 에 비해 CPU 와 메모리를 상대적으로 많이 사용한다. Swing 이 제공하는 컴포넌트를 미리 보려면, 오라클이 제공하는 다음 URL 을 웹브라우저에서 요청해 보면 된다.

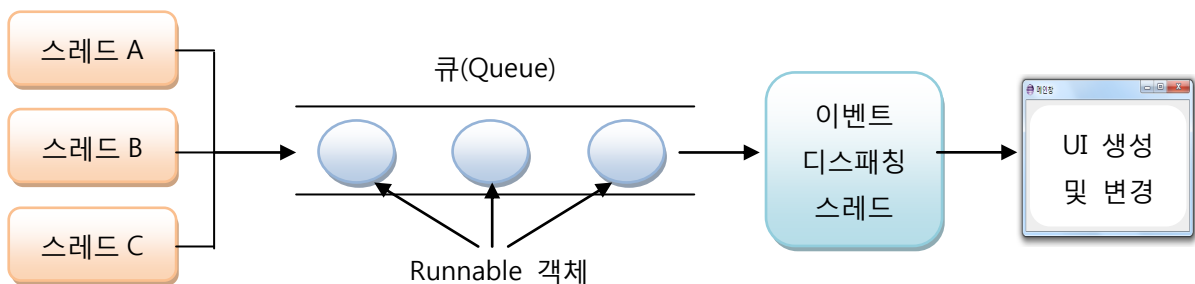
<http://download.java.net/javadesktop/swingset3/SwingSet3.jnlp>

인터넷 익스플로러는 바로 SwingSet3 이 다운로드되고 자동 실행되지만, 크롬은 SwingSet3.jnlp 파일을 먼저 다운로드 받고, 더블 클릭하면 SwingSet3 이 다운로드되고 자동 실행된다. JNLP 파일은 자바 웹 스타트(Java Web Start) 파일로서 웹을 통해 프로그램을 동적으로 다운로드 받아 실행할 수 있도록 해준다.



### 3 절. Swing 애플리케이션 실행

Swing 컨테이너와 컴포넌트는 스레드에 안전하지 않기 때문에 Swing 의 컴포넌트를 여러 스레드가 동시에 접근해서 변경하게 되면 문제가 발생할 수 있다. 그래서 Swing 은 이벤트 디스패칭 스레드(event-dispatching thread)에 의해 순차적으로 UI 변경 작업을 진행하도록 설계되어있다. 다른 스레드에서 UI 를 생성하거나 변경하는 작업이 필요할 때에는 작업해야할 내용을 Runnable 객체로 생성한 뒤, 큐(Queue)에 저장해 놓으면 이벤트 디스패칭 스레드가 순차적으로 큐에 있는 Runnable 객체를 꺼내어 UI 를 생성 및 변경시킨다.



이렇게 하면 여러 스레드가 동시에 UI 를 변경하는 문제가 해결 된다. 스레드가 큐(Queue)에 Runnable 객체를 저장하는 방법은 `javax.swing.SwingUtilities.invokeLater()`를 이용하면 된다. 메소드 이름이 `invokeLater` 인 이유는 먼저 큐에 있는 작업을 처리하고 나중에 실행된다는 의미이다. 다음은 `javax.swing.SwingUtilities.invokeLater()` 메소드를 호출하는 방법을 보여준다.

```
javax.swing.SwingUtilities.invokeLater(new Runnable() {
    public void run() {
        //UI 생성 및 변경 작업
    }
});
```

다음 예제는 JFrame 을 사용하여 메인 윈도우 창을 만들었다. JFrame 은 다음 절에서 자세히 살펴볼 것이다. `main()` 메소드를 실행하는 메인 스레드도 역시 큐에 Runnable 객체를 저장하기 위해 `SwingUtilities.invokeLater()` 메소드를 호출하였다. 이벤트 디스패칭 스레드는 메인 스레드가 큐에 넣어준 Runnable 객체를 꺼내어 JFrame 을 생성하고 보여준다.

#### 【App.java】Swing 애플리케이션 실행

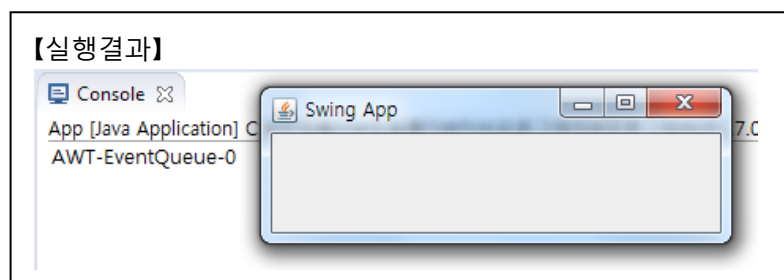
```
1 public class App extends JFrame {
2     public App() {
3         setTitle("Swing App");
4         setSize(300, 100);
```

```

5   }
6
7   public static void main(String[] args) {
8       javax.swing.SwingUtilities.invokeLater(new Runnable() {
9           public void run() {
10              App app = new App();
11              app.setVisible(true);
12              System.out.println(Thread.currentThread().getName());
13          }
14      });
15  }
16  }

```

12 라인에서 Runnable 객체를 실행하는 실제 스레드가 무엇인지 스레드의 이름을 출력하게 했는데, AWT-EventQueue-0 이라는 이름을 가지고 있는 이벤트 디스패칭 스레드임을 알 수 있다.



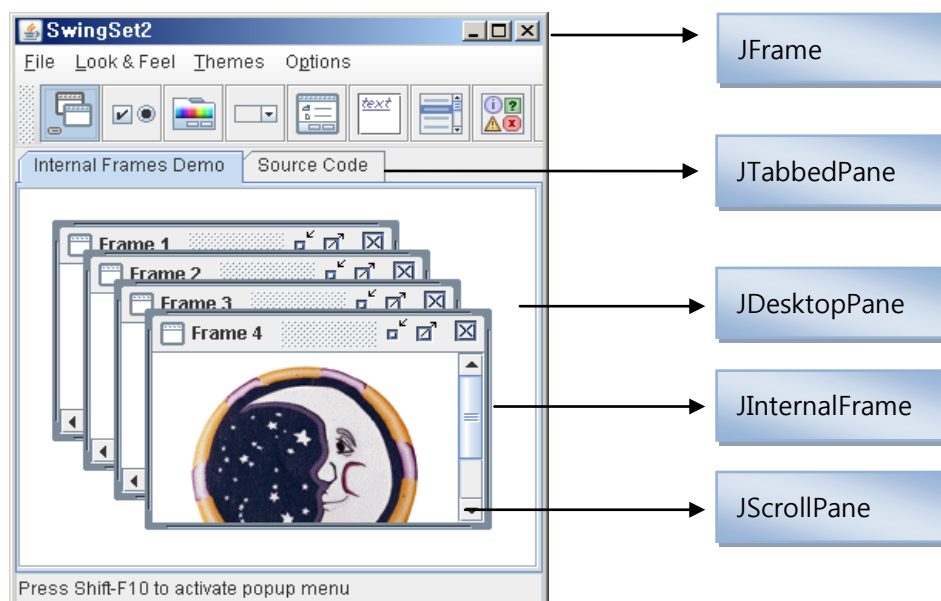
## 4 절. Swing 컨테이너

### 3.1 Swing 컨테이너 종류

윈도우 창과 같이 버튼, 라디오, 체크박스, 텍스트 필드등의 컴포넌트를 배치할 수 있는 클래스를 컨테이너라고 한다. Swing 은 기능에 따라 컨테이너 클래스들을 다음과 같이 제공하고 있다.

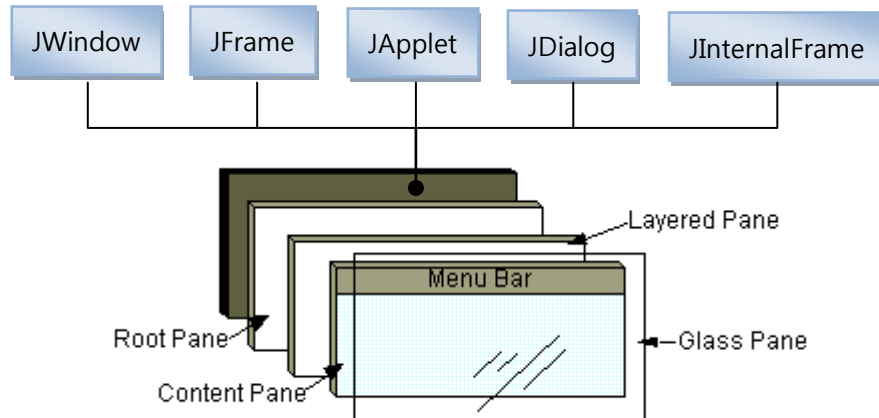
컨테이너 클래스	용도
JApplet	웹 브라우저에서 실행되는 프로그램을 만들 때 사용
JDesktopPane	내부 윈도우를 여러 개 포함할 수 있는 MDI 프로그램을 만들 때 사용
JDialog	다이얼로그 윈도우를 만들 때 사용
JFrame	작업 표시줄, 메뉴가 제공되는 윈도우를 만들 때 사용
JInternalFrame	JDesktopPane 에 포함되는 내부 윈도우를 만들 때 사용
JPanel	컴포넌트들을 배치할 때 사용
JScrollPane	수직 또는 수평 스크롤이 필요할 때 사용
JSplitPane	수직 또는 수평으로 보여주는 크기를 조절할 때 사용
JTabbedPane	여러가지 탭을 제공할 때 사용
JWindow	작업 표시줄, 메뉴가 없는 윈도우를 만들 때 사용

이들 컨테이너 클래스 중에서 JWindow, JFrame, JDialog 는 완전한 윈도우 창 형태를 갖고 있는 최상위 레벨 컨테이너(TopLevel Containers)이고, JApplet 은 웹 브라우저에서 실행되는 UI 프로그램을 개발할 때 사용되는 컨테이너이다. 이들을 제외한 나머지는 최상위 레벨 컨테이너나, JApplet 내부에서 사용되는 보조 컨테이너들이다.



## 3.2 Swing 컨테이너 구조

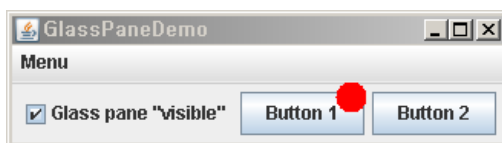
JWindow, JFrame, JApplet, JDialog, JInternalFrame 컨테이너는 다른 컨테이너와 달리 JRootPane 객체를 이용해서 컴포넌트를 관리한다. JRootPane 은 내부적으로 몇 개의 세부 판(pane)을 가지는 구조로 되어 있다.



### 1) GlassPane

GlassPane 은 다른 패널 위에 존재하면서 기본적으로 숨겨져 있는 투명한 판이다. GlassPane 에 대한 이해를 돕기 위해 책과 함께 제공되는 소스에서 GlassPaneDemo.java 를 찾아 이클립스에서 실행하거나, 다음 URL 을 브라우저에서 요청하면 GlassPaneDemo 프로그램이 실행될 것이다.

<http://docs.oracle.com/javase/tutorial/JWS/samples/uiswing/GlassPaneDemoProject/GlassPaneDemo.jnlp>



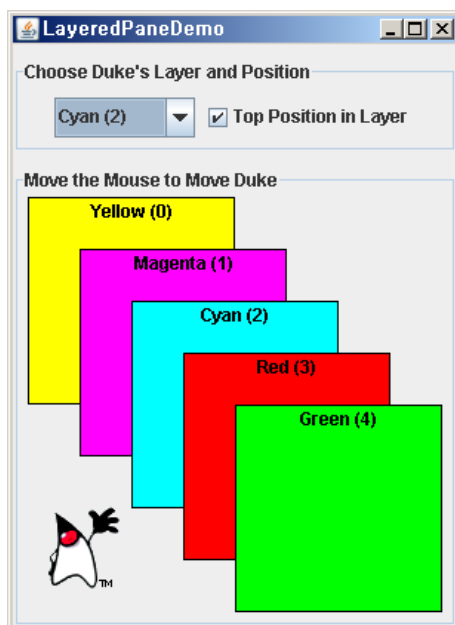
Glass pane "visible" 체크 박스를 체크하지 않은 상태에서 Menu 와 Button1, Button2 를 클릭하면 정상적인 컴포넌트 동작을 하지만, 체크 박스를 체크하게 되면 GlassPane 이 최상판이 된다. GlassPane 이 최상판이 되면 바로 밑에 있는 버튼과 메뉴들을 사용할 수 없다. 마치 유리로 덮여있는 진열대에서 유리 속의 물건을 만질 수 없는 이치와 같다. 대신 GlassPane 에 그려지는 동그란 빨간 점만 볼 수 있다.

## 2) LayeredPane

여러 컴포넌트들이 겹쳐질 때 각 컴포넌트의 상하 위치를 결정한다. GlassPane 에 대한 이해를 돕기 위해 책과 함께 제공되는 소스에서 LayeredPaneDemo.java 를 찾아 이클립스에서 실행하거나, 다음 URL 을 브라우저에서 요청하면 LayeredPaneDemo 프로그램이 실행될 것이다.

<http://docs.oracle.com/javase/tutorial/JWS/samples/uiswing/LayeredPaneDemoProject/LayeredPaneDemo.jnlp>

우측 하단의 손 흔드는 듀크(duke)에 마우스를 올려 놓으면 듀크가 마우스를 따라 움직인다. 듀크를 Yellow, Magenta, Cyan 영역에 갖다 놓으면 잘 보이지만, Red, Green 영역에 갖다 놓으면 가려진다. 듀크는 Cyan 과 Red 중간 위치의 Z 방향(화면을 뚫고 나오는 방향) 순위를 가지고 있기 때문이다.



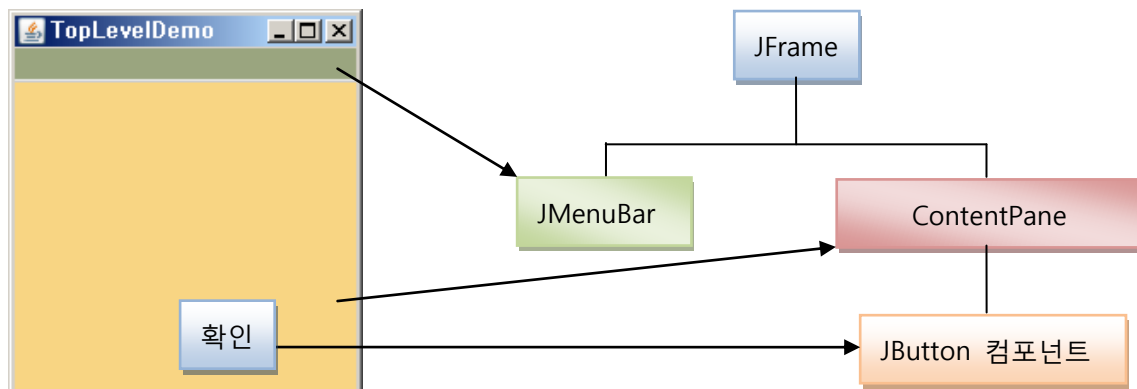


### 3) JMenuBar 와 ContentPane

JMenuBar 는 선택 사항으로 컨테이너의 setJMenuBar() 메소드를 통해 추가할 수 있다. ContentPane 은 일반적인 컴포넌트들이 배치되는 판이다. JMenuBar 와 ContentPane 에 대한 이해를 돕기 위해 책과 함께 제공되는 소스에서 TopLevelDemo.java 를 찾아 이클립스에서 실행하거나, 다음 URL 을 브라우저에서 요청하면 TopLevelDemo 프로그램이 실행될 것이다.

<http://docs.oracle.com/javase/tutorial/JWS/samples/uiswing/TopLevelDemoProject/TopLevelDemo.jnlp>

참고적으로 확인 버튼은 필자가 임의로 추가했다.



컴포넌트를 배치하거나 배치 관리자를 적용하는 작업은 모두 ContentPane 에서 해야 하기 때문에 JWindow, JFrame, JApplet, JDialog, JInternalFrame 컨테이너에서 컴포넌트를 배치할 때는 반드시 ContentPane 을 얻어서 ContentPane 의 add() 메소드로 배치해야 한다.

jWindow.add(컴포넌트)	→	jWindow.getContentPane().add(컴포넌트);
jFrame.add(컴포넌트);	→	jFrame.getContentPane().add(컴포넌트);
jApplet.add(컴포넌트);	→	jApplet.getContentPane().add(컴포넌트);
jDialog.add(컴포넌트);	→	jDialog.getContentPane().add(컴포넌트);
jInternalFrame.add(컴포넌트);	→	jInternalFrame.getContentPane().add(컴포넌트);

예를 들어 JFrame 에서 JMenuBar 를 추가하고, ContentPane 에 JButton 컴포넌트를 배치하는 코드를 간략히 요약하면 다음과 같다.

```
JFrame jFrame = new JFrame();

//MenuBar 추가
jFrame.setJMenuBar(new JMenuBar());

//컴포넌트 추가
```

```
jFrame.getContentPane().add(new JButton("확인"));
```

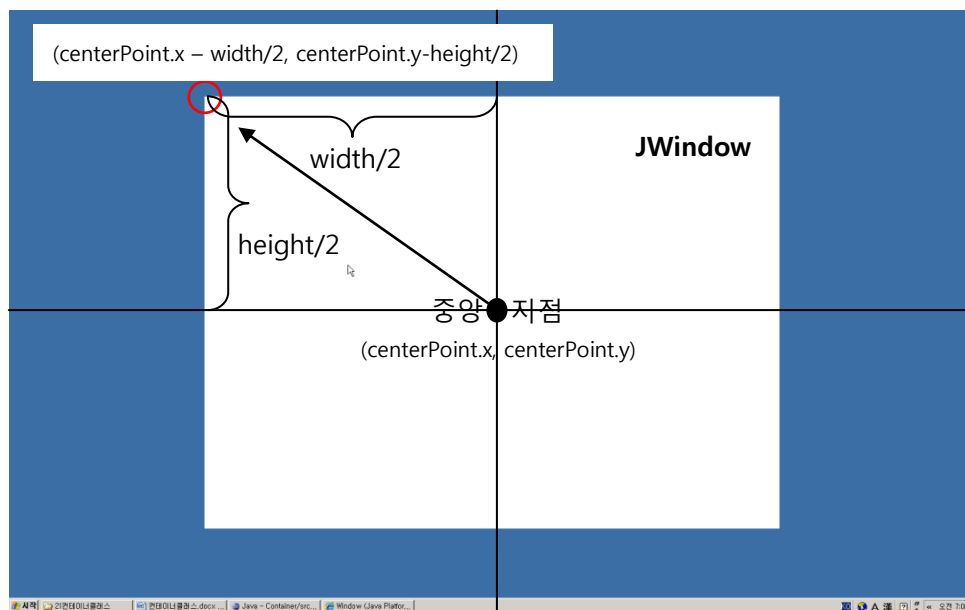
참고로 AWT 의 컨테이너인 Window, Frame, Applet, Dialog 에 컴포넌트를 배치할 때는 이들 컨테이너의 add() 메소드를 직접 사용한다. AWT 의 컨테이너들은 JRootPane 객체를 사용하지 않기 때문이다.

### 3.3 JWindow

JWindow 는 윈도우 경계선, 제목 표시줄, 크기 조절 버튼, 종료 버튼, 메뉴가 모두 없는 단순한 윈도우를 만드는 컨테이너로 컴포넌트만 배치할 수 있는 평면 공간만을 갖는다. 게임 애플리케이션처럼 제목 표시줄이 없는 윈도우를 만들 때 주로 이용된다. 새로운 JWindow 를 만들기 위해서는 다음과 같이 JWindow 를 상속해서 만든다.

```
public class JWindowExample extends JWindow {  
    public JWindowExample() {  
        this.setSize(450, 300);  
    }  
}
```

윈도우는 반드시 폭(width)과 높이(height)가 있어야 하기 때문에 생성자에서 setSize() 메소드로 폭과 높이를 주면 된다. 윈도우를 화면 중앙에 띄우기 위해서는 화면의 중앙 지점을 얻어서, 윈도우의 좌측 상단 모서리의 좌표를 계산해야 한다.



다음은 JWindow 를 화면 중앙에 띄우기 위해 필요한 코드들이다.

```
GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();
```

```
Point centerPoint = ge.getCenterPoint();
int leftTopX = centerPoint.x - getWidth()/2;
int leftTopY = centerPoint.y - getHeight()/2;
this.setLocation(leftTopX, leftTopY);
```

java.awt.GraphicsEnvironment 는 그래픽 환경에 대한 정보를 가지고 있는 객체이다. 이 객체는 정적(static) 메소드인 getLocalGraphicsEnvironment()를 호출해서 얻을 수 있다. GraphicsEnvironment 의 getContentPoint() 메소드는 화면 중앙 지점의 X 좌표와 Y 좌표를 가지고 있는 Point 객체를 리턴한다. 이렇게 얻은 화면 중앙 좌표와 윈도우 폭, 높이로 JWindow 의 좌측 상단 모서리 좌표를 계산할 수 있다. 그런 다음 JWindow 의 setLocation() 메소드로 좌측 상단 모서리 좌표를 설정해주면 된다. 최종적으로 JWindow 를 화면에 띄우려면 setVisible(true) 메소드를 호출하면 된다.

```
JWindowExample jWindow = new JWindowExample();
jWindow.setVisible(true);
```

반대로 setVisible(false)를 호출하면 JWindow 가 화면에서 사라지는데, 이것은 JWindow 가 화면에서 완전히 제거되는 것이 아니라, 숨겨질 뿐이다. 다시 setVisible(true)를 호출하면 언제든지 나타난다. 만약 JWindow 를 화면에서 완전히 제거하고 싶다면 dispose() 메소드를 호출하면 된다.

```
jWindow.dispose();
```

다음 예제는 JWindow 를 이용해서 게임 시작 화면을 이미지로 보여준다.

#### 【JWindowExample.java】 시작 로고창

```
1 public class JWindowExample extends JWindow {
2     public JWindowExample() {
3         //JWindow의 크기
4         this.setSize(600, 350);
5
6         //JWindow를 화면 중앙으로 띄우기
7         GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();
8         Point centerPoint = ge.getCenterPoint();
9         int leftTopX = centerPoint.x - this.getWidth()/2;
10        int leftTopY = centerPoint.y - this.getHeight()/2;
11        this.setLocation(leftTopX, leftTopY);
12
13        //JWindow에 이미지가 포함된 JLabel 추가
14        JLabel label = new JLabel();
15        label.setIcon(new ImageIcon(getClass().getResource("game.png")));
```

```

16     getContentPane().add(label, BorderLayout.CENTER);
17
18     //마우스 클릭 이벤트 처리
19     this.addMouseListener(new MouseAdapter() {
20         @Override
21         public void mouseClicked(MouseEvent e) {
22             System.exit(0);
23         }
24     });
25 }
26
27 public static void main(String[] args) {
28     SwingUtilities.invokeLater(new Runnable() {
29         public void run() {
30             JWindowExample jWindow = new JWindowExample();
31             jWindow.setVisible(true);
32         }
33     });
34 }
35 }

```

JWindow 에 이미지를 넣기 위해 14~15 라인에서 JLabel 컴포넌트를 활용하였다. JLabel 은 글자 및 이미지를 포함할 수 있는 컴포넌트인데, setIcon() 메소드로 ImageIcon 객체를 매개값으로 주면 이미지를 나타낼 수 있다. ImageIcon 생성자는 이미지 파일의 URL 객체를 매개값으로 받는데, JWindowExample.class 와 동일한 폴더에 있는 "game.png" 파일의 URL 객체를 얻기 위해 다음 코드를 사용하였다.

#### 【실행결과】



```
getClass().getResource("game.png")
```

16 라인은 JLabel 을 JWindow 의 중앙에 배치시킨다. 컴포넌트 배치는 4 절에서 설명한다. 19~24 라인은 마우스를 JWindow 에 클릭했을 때 애플리케이션이 종료되도록 이벤트를 처리한 것이다. 이벤트 처리는 5 절에서 설명한다.

### 3.4 JFrame

javax.swing.JFrame 은 JWindow 와는 달리 윈도우 경계선, 제목 표시줄, 크기 조절 버튼, 종료 버튼, 메뉴가 있는 윈도우를 만드는 컨테이너 클래스이다. 새로운 사용자 프레임을 만들기 위해서는 다음과 같이 JFrame 을 상속해서 만든다.

```
public class JFrameExample extends JFrame {
    public JFrameExample() {
        //제목
        this.setIconImage(new ImageIcon(getClass().getResource("icon.png")).getImage());
        this.setTitle("창제목");
        //JFrame 크기
        this.setSize(600, 500);
        //종료 버튼의 기본 기능
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

JFrame 의 제목 표시줄에는 아이콘, 타이틀, 크기 조절용 버튼, 종료 버튼으로 구성된다. 아이콘은 setIconImage() 메소드로 설정하면 되는데, ImageIcon 객체의 getImage() 메소드로 Image 객체를 얻어 매개값으로 설정하면 된다. 창 제목은 setTitle() 메소드로 설정할 수 있다. 종료 버튼의 기본 기능은 JFrame 을 단순히 숨기기만 하고 프로세스를 종료하지 않는다. 프로세스를 완전히 종료하려면



setDefaultCloseOperation() 메소드로 종료 버튼의 기본 기능을 변경해야 한다. setDefaultCloseOperation() 메소드에 지정할 수 있는 종료 버튼의 기능별 매개값은 다음 네가지 종류가 있다.

기능별 상수	설명
WindowConstants.DO_NOTHING_ON_CLOSE	아무 것도 하지 않음
WindowConstants.HIDE_ON_CLOSE	화면에서 JFrame 숨김 (기본)
WindowConstants.DISPOSE_ON_CLOSE	화면에서 JFrame 완전히 제거, 다른 JFrame 이 없다면 윈도우 프로세스를 실행 종료.
JFrame.EXIT_ON_CLOSE	윈도우 프로세스를 실행 종료

창크기는 setSize()로 폭과 높이를 주면 되고, JFrame 을 화면의 중앙 부분에 위치시키는 방법은 JWindow 에서 설명한 것과 동일한데, 화면 중앙 좌표와 윈도우 폭, 높이로 JFrame 의 좌측 상단 모서리 좌표를 구한 다음, setLocation() 메소드로 설정해주면 된다. 최종적으로 JFrame 를 화면에 띄우려면 setVisible(true) 메소드를 호출하면 된다.

```
JFrameExample jFrame = new JFrameExample ();
jFrame.setVisible(true);
```

반대로 setVisible(false)를 호출하면 JFrame 이 화면에서 사라지는데, 이것은 JFrame 이 화면에서 완전히 제거되는 것이 아니라, 숨겨질 뿐이다. 다시 setVisible(true)를 호출하면 언제든지 나타난다. 만약 JFrame 를 화면에서 완전히 제거하고 싶다면 dispose() 메소드를 호출하면 된다.

```
jFrame.dispose();
```

다음 예제는 JFrame 의 제목을 설정하고, 종료 버튼을 클릭했을 때 프로세스가 종료되도록 하였다.

#### 【JFrameExample.java】 제목 표시줄이 있는 윈도우

```
1 public class JFrameExample extends JFrame {
2     public JFrameExample() {
3         //JWindow의 크기
4         this.setSize(600, 500);
5
6         //제목 표시줄 내용
7         this.setIconImage(new ImageIcon(getClass().getResource("icon.png")).getImage());
8         this.setTitle("메인창");
9
10        //종료 버튼의 기본 기능
11        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
12
13        //JWindow를 화면 중앙으로 띄우기
14        GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();
15        Point centerPoint = ge.getCenterPoint();
16        int leftTopX = centerPoint.x - this.getWidth()/2;
17        int leftTopY = centerPoint.y - this.getHeight()/2;
18        this.setLocation(leftTopX, leftTopY);
19    }
20
21    public static void main(String[] args) {
22        SwingUtilities.invokeLater(new Runnable() {
23            public void run() {
```

```

24         JFrameExample jFrame = new JFrameExample();
25         jFrame.setVisible(true);
26     }
27     });
28 }
29 }

```

### 3.5 JTabbedPane

JTabbedPane 은 탭(tab) 별로 다른 내용을 보여주기 위해 사용되는 컨테이너이다. JTabbedPane 은 독립적인 윈도우 모양을 갖고 있지 않기 때문에 JWindow, JFrame, JDialog 등과 같은 최상위 레벨 컨테이너에 배치된다. 옆그림은 JTabbedPane 의 모습을 보여준다. 탭의 위치는 상단, 왼쪽, 오른쪽, 하단에 위치 시킬 수 있다. JTabbedPane 을 생성하려면 다음과 같이 기본 생성자를 호출하면 된다.



```
JTabbedPane jTabbedPane = new JTabbedPane();
```

탭의 위치를 설정하려면 setTabPlacement() 메소드로 탭의 위치 상수를 다음과 같이 지정하면 된다.

```

jTabbedPane.setTabPlacement(
    JTabbedPane.TOP | JTabbedPane.BOTTOM | JTabbedPane.LEFT | JTabbedPane.RIGHT
);

```

JTabbedPane 에 탭을 추가하려면 addTab() 메소드를 이용한다. addTab() 메소드는 탭의 이름과 탭안에 배치될 컴포넌트를 매개값으로 받는데, 컴포넌트는 주로 JPanel 을 객체를 사용한다.

```

jTabbedPane.addTab("TapName1", jPanel1);
jTabbedPane.addTab("TapName2", jPanel2);

```

각 탭에 포함될 컴포넌트는 각 탭의 JPanel 에 배치하면 된다. 이렇게 생성된 JTabbedPane 은 다른 컴포넌트와 마찬가지로 최상위 레벨 컨테이너에 배치된다. 다음은 JFrame 의 중앙에 배치하는 코드이다.

```
jFrame.getContentPane().add("Center", jTabbedPane);
```

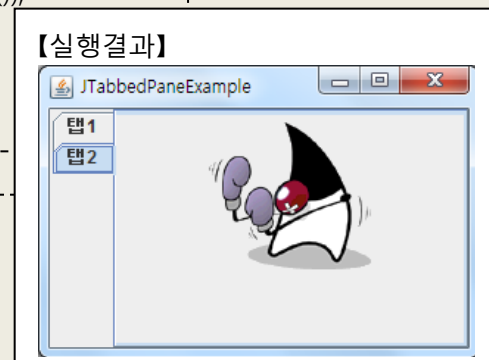
다음 예제는 두 개의 탭을 추가했는데 각 탭을 클릭하면 두 개의 JPanel 이 교체해가며 이미지를 보여준다.

#### 【JTabbedPaneExample.java】 탭을 이용해서 이미지 보기

```

1 public class JTabbedPaneExample extends JFrame {
2     private JTabbedPane jTabbedPane;
3     private JPanel tab1Panel;
4     private JPanel tab2Panel;
5
6     public JTabbedPaneExample() {
7         this.setTitle("JTabbedPaneExample");
8         this.setSize(300, 200);
9         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10        this.getContentPane().add(getJTabbedPane(), BorderLayout.CENTER);
11    }
12
13    private JTabbedPane getJTabbedPane() {
14        if (jTabbedPane == null) {
15            jTabbedPane = new JTabbedPane();
16            jTabbedPane.setTabPlacement(JTabbedPane.LEFT);
17            jTabbedPane.addTab("탭1", getTab1Panel());
18            jTabbedPane.addTab("탭2", getTab2Panel());
19        }
20        return jTabbedPane;
21    }
22
23    private JPanel getTab1Panel() {
24        if(tab1Panel == null) {
25            tab1Panel = new JPanel();
26            JLabel jLabel = new JLabel();
27            jLabel.setIcon(new ImageIcon(getClass().getResource("duke1.gif")));
28            tab1Panel.add(jLabel);
29        }
30        return tab1Panel;
31    }
32
33    private JPanel getTab2Panel() {
34        if(tab2Panel == null) {
35            tab2Panel = new JPanel();

```





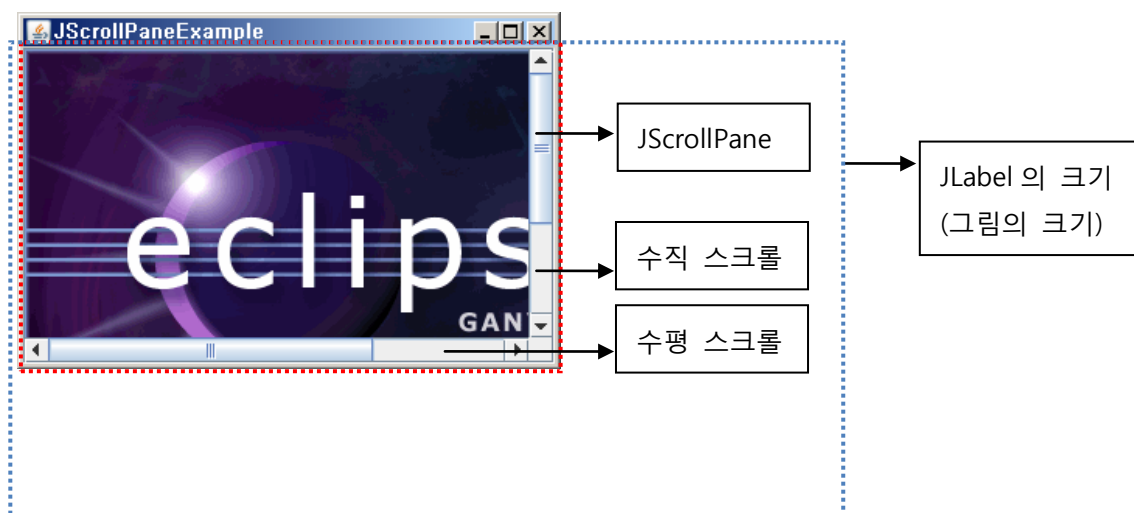
```

36     JLabel jLabel = new JLabel();
37     jLabel.setIcon(new ImageIcon(getClass().getResource("duke2.gif")));
38     tab2Panel.add(jLabel);
39 }
40 return tab2Panel;
41 }
42
43 public static void main(String[] args) {
44     SwingUtilities.invokeLater(new Runnable() {
45         public void run() {
46             JTabbedPaneExample jFrame = new JTabbedPaneExample();
47             jFrame.setVisible(true);
48         }
49     });
50 }
51 }

```

### 3.6 JScrollPane

JScrollPane 은 포함된 컴포넌트의 크기가 JScrollPane 자신보다 큰 경우 수평 또는 수직 스크롤바를 이용해서 볼 수 있게 해준다. JScrollPane 은 다른 컨테이너와는 달리 단 하나의 컴포넌트만을 포함시킬 수 있다.



위 그림을 보면 JScrollPane 은 JFrame 의 중앙에 위치하고 있다. JScrollPane 안에는 그림을 포함하고 있는 JLabel 배치되어 있는데, 그림의 크기가 JScrollPane 크기를 초과하기 때문에 수직 및 수평 스크롤이 생긴다. 스크롤이 필요한 컴포넌트에는 큰 그림을 포함하고 있는 JLabel,

JTextArea, JList, JTable, JTree 등이 있다. 이들 컴포넌트에 스크롤을 적용시키려면 다음과 같이 JScrollPane 생성자에 이들 컴포넌트를 매개값으로 주면 된다.

```
JScrollPane scrollJList = new JScrollPane(jLabel);
JScrollPane scrollJTextArea = new JScrollPane(jTextArea);
JScrollPane scrollJList = new JScrollPane(jList);
JScrollPane scrollJTable = new JScrollPane(jTable);
```

이렇게 생성된 JScrollPane 은 컴포넌트가 배치될 수 있는 곳이라면 어디든지 배치가 가능하다. 다음 예제는 JLabel 에 큰 이미지를 넣고, JScrollPane 에 JLabel 을 추가시켰다. 그리고 나서 JFrame 중앙에 JScrollPane 을 배치시켰는데, JFrame 의 사이즈가 이미지보다 작기 때문에 스크롤이 자동 생성된다.

#### 【JScrollPaneExample.java】 큰 그림을 스크롤로 이동해서 보기

```
1 public class JScrollPaneExample extends JFrame {
2     private JScrollPane scrollImage;
3     private JLabel lblImage;
4
5     public JScrollPaneExample() {
6         this.setTitle("JScrollPaneExample");
7         this.setSize(350, 230);
8         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9         this.getContentPane().add(getScrollImage(), BorderLayout.CENTER);
10    }
11
12    private JScrollPane getScrollImage() {
13        if (scrollImage == null) {
14            scrollImage = new JScrollPane(getLblImage());
15        }
16        return scrollImage;
17    }
18
19    public JLabel getLblImage() {
20        if (lblImage == null) {
21            lblImage = new JLabel();
22            lblImage.setIcon(new ImageIcon(getClass().getResource("snow.jpg")));
23        }
24        return lblImage;
25    }
26 }
```



```

27 public static void main(String[] args) {
28     SwingUtilities.invokeLater(new Runnable() {
29         public void run() {
30             JScrollPaneExample jFrame = new JScrollPaneExample();
31             jFrame.setVisible(true);
32         }
33     });
34 }
35 }

```

## 5 절. 컴포넌트 배치

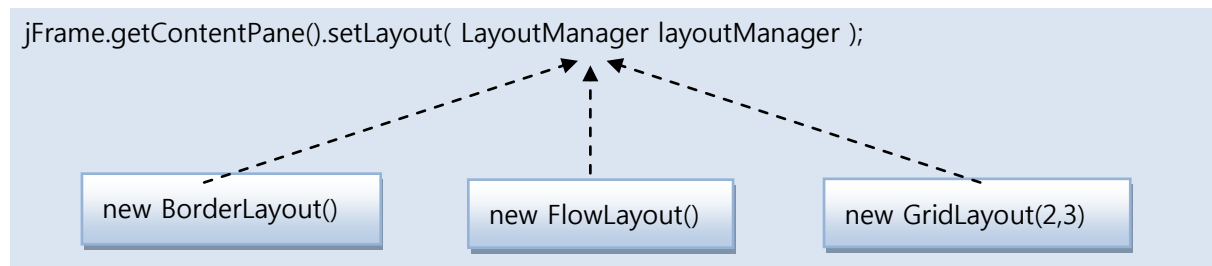
컨테이너에는 UI 컴포넌트들이 배치된다. 대표적인 컴포넌트에는 버튼, 체크 박스, 라디오 버튼, 콤포, 리스트등이 있다. 컨테이너들은 기본적으로 배치 관리자로 컴포넌트를 배치한다. 배치 관리자는 좌표값으로 컴포넌트를 배치하지 않고, 컨테이너를 몇 개의 구획으로 나누고, 하나의 구획에 하나의 컴포넌트를 배치해 준다. 배치 관리자로 배치하게 되면 컨테이너의 크기가 사용자에게 의해 변경되더라도 컴포넌트의 크기가 비율적으로 늘거나 줄게되어 배치 모양이 그대로 유지된다는 장점이 있다. 크기가 고정된 컨테이너일 경우, 세밀한 배치를 위해서 좌표값으로 컴포넌트를 배치할 수도 있다. 컨테이너의 좌측 상단 모서리를 (0,0) 으로 보고 x 축과 y 축 좌표로 컴포넌트의 위치를 정해서 배치한다.

### 4.1 배치 관리자(LayoutManager)

컨테이너가 컴포넌트를 배치할 때에는 설정된 배치 관리자(Layout Manager)가 무엇이냐에 따라 달라진다. JWindow, JFrame, JDialog 은 기본적으로 BorderLayout 배치 관리자를 사용하고 JPanel 은 FlowLayout 을 사용한다. 기본 배치 관리자는 변경이 가능한데, 자바는 java.awt 패키지에서 다음과 같은 배치 관리자를 제공하고 있다.

배치 관리자	설명
BorderLayout	동·서·남·북·중앙으로 컴포넌트를 배치
CardLayout	여러장의 카드에 컴포넌트를 각각 배치
FlowLayout	왼쪽에서 오른쪽으로 컴포넌트를 배치
GridLayout	바둑판과 같은 격자에 컴포넌트를 배치
GridBagLayout	바둑판과 같은 격자에 컴포넌트를 배치하지만 격자간 병합이 가능

모든 배치 관리자는 공통된 인터페이스인 `java.awt.LayoutManager` 를 구현하고 하고 있다. 컨테이너의 기본 배치 관리자 대신 다른 것을 사용하고 싶다면 컨테이너의 `setLayout()` 메소드로 배치 관리자를 변경할 수 있다. `setLayout()`의 매개변수 데이터 타입은 `LayoutManager` 인터페이스이므로 모든 배치 관리자의 인스턴스가 올 수 있다. 예를 들어 다음은 `JFrame` 의 배치 관리자를 변경하는 방법을 보여준다.

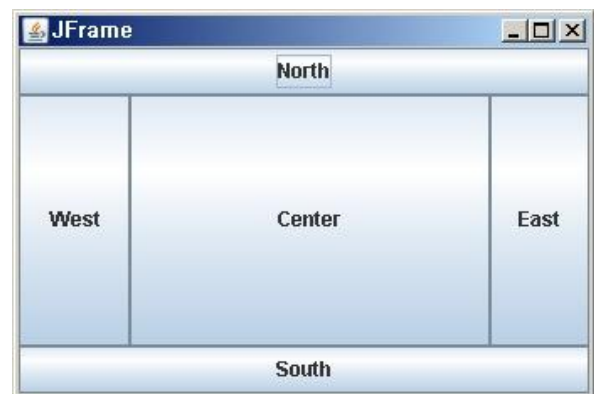


컨테이너의 배치 관리자 설정은 컴포넌트를 배치하기 전에 변경하는 것이 좋다. 컨테이너에서 사용하는 배치 관리자를 얻고 싶다면 컨테이너의 `getLayout()` 메소드를 호출하면 된다. `getLayout()`의 리턴타입은 `LayoutManager` 인터페이스이므로 다음과 같이 타입 변환을 해야 한다.

```
BorderLayout borderLayout = (BorderLayout) JFrame.getContentPane().getLayout();
```

## 4.2 BorderLayout

`BorderLayout` 배치 관리자는 컨테이너를 중앙·동·서·남·북으로 구획 짓고, 각 구획에 하나의 컴포넌트 또는 컨테이너를 배치한다. 일반적으로 각 구획에는 `JPanel` 컨테이너가 배치되어, 복잡한 형태의 UI 를 만들어 낸다. `BorderLayout` 을 기본적으로 사용하는 컨테이너는 `JWindow`, `JFrame`, `JDialog` 등이 있다. 옆 그림은 `JFrame` 의 각 구획에 `JButton` 컴포넌트를 배치한 것이다. `BorderLayout` 이 적용된 컨테이너에 컴포넌트를 배치할 때에는 다음과 같은 `add()` 메소드를 사용해야 한다.



```

JFrame.getContentPane().add(컴포넌트, BorderLayout.CENTER);
JFrame.getContentPane().add(컴포넌트, BorderLayout.EAST);
JFrame.getContentPane().add(컴포넌트, BorderLayout.WEST);
JFrame.getContentPane().add(컴포넌트, BorderLayout.SOUTH);
JFrame.getContentPane().add(컴포넌트, BorderLayout.NORTH);

```

첫 번째 매개값에는 배치할 컴포넌트 객체가 오고, 두 번째 매개값에는 어떤 구획에 배치할 것인지 지정하는 BorderLayout 의 상수가 온다. 만약 동·서·남·북 중에서 컴포넌트가 배치되지 않은 구획이 있다면 중앙에 배치된 컴포넌트가 해당 구획까지 확장된다. 다음 예제는 중앙, 북쪽, 남쪽에만 컴포넌트를 배치하고, 동쪽과 서쪽은 배치하지 않았다. 그래서 중앙에 배치된 컴포넌트가 동쪽과 서쪽으로 확장되었다.

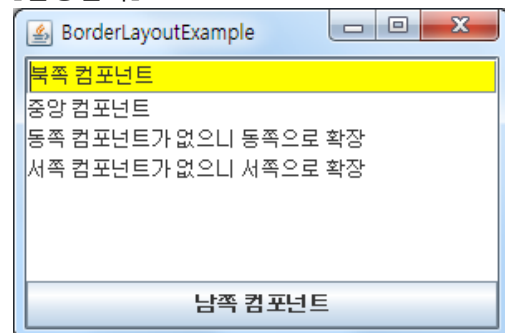
#### 【BorderLayoutExample.java】 BorderLayout 으로 컴포넌트 배치

```

1 public class BorderLayoutExample extends JFrame {
2     private JTextField txtNorth;
3     private JTextArea txtCenter;
4     private JButton btnSouth;
5
6     public BorderLayoutExample() {
7         this.setTitle("BorderLayoutExample");
8         this.setSize(300, 200);
9         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10
11         this.getContentPane().add(getTxtNorth(), BorderLayout.NORTH);
12         this.getContentPane().add(getTxtCenter(), BorderLayout.CENTER);
13         this.getContentPane().add(getBtnSouth(), BorderLayout.SOUTH);
14     }
15
16     private JTextField getTxtNorth() {
17         if(txtNorth == null) {
18             txtNorth = new JTextField();
19             txtNorth.setText("북쪽 컴포넌트");
20             txtNorth.setBackground(Color.YELLOW);
21         }
22         return txtNorth;
23     }
24
25     private JTextArea getTxtCenter() {
26         if(txtCenter == null) {
27             txtCenter = new JTextArea();
28             txtCenter.append("중앙 컴포넌트\n");
29             txtCenter.append("동쪽 컴포넌트가 없으니 동쪽으로 확장\n");
30             txtCenter.append("서쪽 컴포넌트가 없으니 서쪽으로 확장\n");
31         }
32         return txtCenter;
33     }

```

#### 【실행결과】



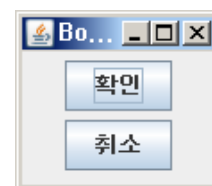
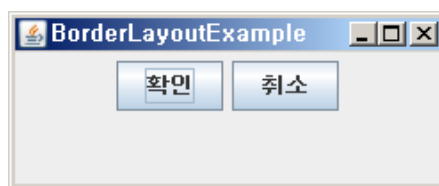
```

34
35 private JButton getBtnSouth() {
36     if(btnSouth == null) {
37         btnSouth = new JButton();
38         btnSouth.setText("남쪽 컴포넌트");
39     }
40     return btnSouth;
41 }
42
43 public static void main(String[] args) {
44     SwingUtilities.invokeLater(new Runnable() {
45         public void run() {
46             BorderLayoutExample jFrame = new BorderLayoutExample();
47             jFrame.setVisible(true);
48         }
49     });
50 }
51 }

```

## 4.3 FlowLayout

FlowLayout 배치 관리자는 이미 배치된 컴포넌트의 오른쪽 옆에 새로운 컴포넌트를 배치한다. 오른쪽에 배치할 공간이 부족하면 아래쪽에 배치하기 때문에 사용자에 의해 컨테이너의 폭(width)이 변경되면 컴포넌트의 배치 위치가 변경될 수 있다.



FlowLayout 이 적용된 컨테이너가 컴포넌트를 배치할 때는 컴포넌트만 매개 변수로 갖는 add() 메소드를 사용한다. 예를 들어 JFrame 이 FlowLayout 을 사용하여 컴포넌트를 배치한다면, 다음과 같이 add() 메소드로 컴포넌트를 배치해야 한다.

```

jFrame.getContentPane().setLayout(new FlowLayout());
jFrame.getContentPane().add(컴포넌트);

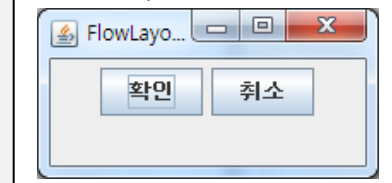
```

FlowLayout 을 기본적으로 사용하는 컨테이너에는 JPanel 이 있다. JPanel 은 다른 컨테이너에 컴포넌트를 배치하는데 도움을 주는 보조 컨테이너 역할을 한다. 다음 예제는 JFrame 의 배치 관리자로 FlowLayout 을 적용하고 버튼 두 개를 배치하였다.

#### 【FlowLayoutExample.java】 FlowLayout 으로 컴포넌트 배치

```
1 public class FlowLayoutExample extends JFrame {
2     private JButton btnOk;
3     private JButton btnCancel;
4
5     public FlowLayoutExample() {
6         this.setTitle("FlowLayoutExample");
7         this.setSize(200, 100);
8         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9
10        this.setLayout(new FlowLayout());
11        this.getContentPane().add(getBtnOk());
12        this.getContentPane().add(getBtnCancel());
13    }
14
15    private JButton getBtnOk() {
16        if(btnOk == null) {
17            btnOk = new JButton();
18            btnOk.setText("확인");
19        }
20        return btnOk;
21    }
22
23    private JButton getBtnCancel() {
24        if(btnCancel == null) {
25            btnCancel = new JButton();
26            btnCancel.setText("취소");
27        }
28        return btnCancel;
29    }
30
31    public static void main(String[] args) {
32        SwingUtilities.invokeLater(new Runnable() {
33            public void run() {
34                FlowLayoutExample jFrame = new FlowLayoutExample();
35                jFrame.setVisible(true);
```

#### 【실행결과】



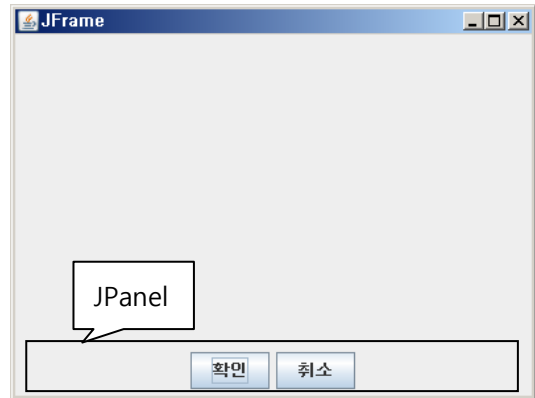
```

36         }
37     });
38 }
    }

```

## 4.4 JPanel

JPanel 은 JWindow, JFrame, JDialog 처럼 하나의 윈도우 창을 만드는 최상위 레벨 컨테이너가 아니라, 이들 컨테이너 속에서 컴포넌트의 배치를 위해 사용되는 투명한 보조 컨테이너이다. JPanel 은 기본적으로 FlowLayout 을 사용한다. 만약 JFrame 남쪽에 두 개의 JButton 을 배치하려면 어떻게 해야할까? 이 경우에는 JPanel 을 JFrame 남쪽에 배치하고, JPanel 내부에 JButton 컴포넌트 두 개를 배치하면 된다. JPanel 은 컴포넌트가 배치 될 수 있는 어떤 곳이라도 배치가 가능하다. 심지어는 JPanel 에 또다른 JPanel 을 배치하는 것도 가능하다. 다음은 JFrame 남쪽에 JPanel 을 배치하는 코드이다.



```

JPanel jPanel = new JPanel();
jFrame.getContentPane().add(jPanel, BorderLayout.SOUTH);

```

JPanel 은 기본적으로 FlowLayout 을 사용하지만, 다음과 같이 setLayout() 메소드로 배치 관리자를 변경할 수 있다.

```

jPanel.setLayout(new BorderLayout());

```

JPanel 은 JWindow, JFrame, JDialog 처럼 JRootPane 객체를 이용해서 컴포넌트를 관리하지 않는다. 따라서 ContentPane 이 없으므로 JPanel 의 add() 메소드로 컴포넌트를 배치하면 된다.

```

FlowLayout 일 경우:    jPanel.add(컴포넌트);
BorderLayout 일 경우:  jPanel.add(컴포넌트, BorderLayout.CENTER);

```

JPanel 을 사용하지 않고 JFrame, JDialog 에서 복잡한 형태로 컴포넌트를 배치할 수 없기 때문에 거의 필수적으로 사용된다. 다음 예제는 JFrame 의 남쪽에 JPanel 을 배치하고 확인 및 취소 버튼을 JPanel 에 배치했다.

### 【JPanelExample.java】 JPanel 을 남쪽에 배치한 JFrame

```

1 public class JPanelExample extends JFrame {
2     private JPanel panelSouth;

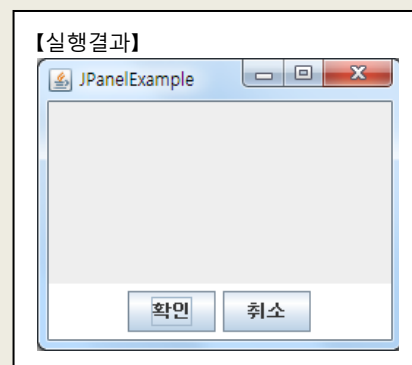
```



```

3 private JButton btnOk;
4 private JButton btnCancel;
5
6 public JPanelExample() {
7     this.setTitle("JPanelExample");
8     this.setSize(250, 200);
9     this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10    this.getContentPane().add(getPanelSouth(), BorderLayout.SOUTH);
11 }
12
13 public JPanel getPanelSouth() {
14     if (panelSouth == null) {
15         panelSouth = new JPanel();
16         panelSouth.setBackground(Color.WHITE);
17         panelSouth.add(getBtnOk());
18         panelSouth.add(getBtnCancel());
19     }
20     return panelSouth;
21 }
22
23 public JButton getBtnOk() {
24     if(btnOk == null) {
25         btnOk = new JButton();
26         btnOk.setText("확인");
27     }
28     return btnOk;
29 }
30
31 public JButton getBtnCancel() {
32     if(btnCancel == null) {
33         btnCancel = new JButton();
34         btnCancel.setText("취소");
35     }
36     return btnCancel;
37 }
38
39 public static void main(String[] args) {
40     SwingUtilities.invokeLater(new Runnable() {
41         public void run() {
42             JPanelExample jFrame = new JPanelExample();

```



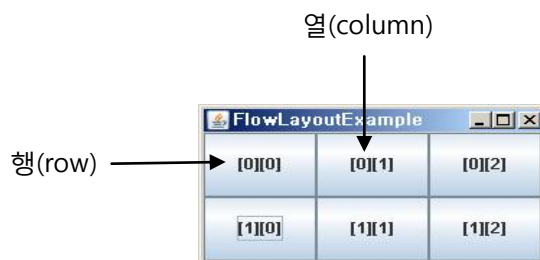
```

43         JFrame.setVisible(true);
44     }
45 });
46 }
47 }

```

## 4.5 GridLayout

GridLayout 배치 관리자는 컨테이너를 행(row)과 열(column)로 구성된 테이블 모양으로 구획 짓고, 각 구획에 하나의 컴포넌트를 배치한다.



행과 열의 수는 GridLayout 객체를 생성할 때 생성자의 매개값으로 주거나, 객체 생성 후 `setRows()`, `setColumns()` 메소드로 지정할 수도 있다. 다음은 생성자에서 행과 열의 수를 지정하여 GridLayout 을 생성한 뒤, JFrame 의 배치 관리자로 설정한다.

```
jFrame.getContentPane().setLayout(new GridLayout(행수, 열수));
```

GridLayout 을 배치 관리자로 사용하는 컨테이너가 컴포넌트를 배치할 때는 FlowLayout 과 마찬가지로 컴포넌트만 매개 변수로 갖는 `add()` 메소드를 한다.

```
jFrame.getContentPane().add(컴포넌트);
```

컴포넌트 배치 순서는 첫 번째 행의 첫 번째 열부터 배치되고, 행의 마지막 열까지 배치가 끝나면 다음 행의 첫 번째 열부터 다시 차례대로 배치된다.

### 【GridLayoutExample.java】 GridLayout 으로 컴포넌트 배치

```

1 public class GridLayoutExample extends JFrame {
2     private JButton[][] btn;
3
4     public GridLayoutExample() {
5         setTitle("GridLayoutExample");
6         setSize(300, 100);
7         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

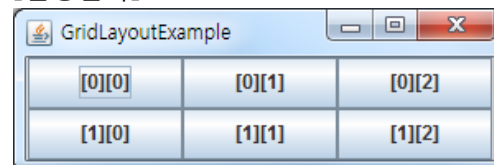
```

```

8
9      setLayout(new GridLayout(2,3));
10     for(int r = 0; r<2; r++) {
11         for(int c=0; c<3; c++) {
12             getContentPane().add(getBtn()[r][c]);
13         }
14     }
15 }
16
17 public JButton[][] getBtn() {
18     if(btn == null) {
19         btn = new JButton[2][3];
20         for(int r = 0; r<2; r++) {
21             for(int c=0; c<3; c++) {
22                 btn[r][c] = new JButton();
23                 btn[r][c].setText "[" + r + "]" + " + c + " ");
24             }
25         }
26     }
27     return btn;
28 }
29
30 public static void main(String[] args) {
31     SwingUtilities.invokeLater(new Runnable() {
32         public void run() {
33             GridLayoutExample jFrame = new GridLayoutExample();
34             jFrame.setVisible(true);
35         }
36     });
37 }
38 }

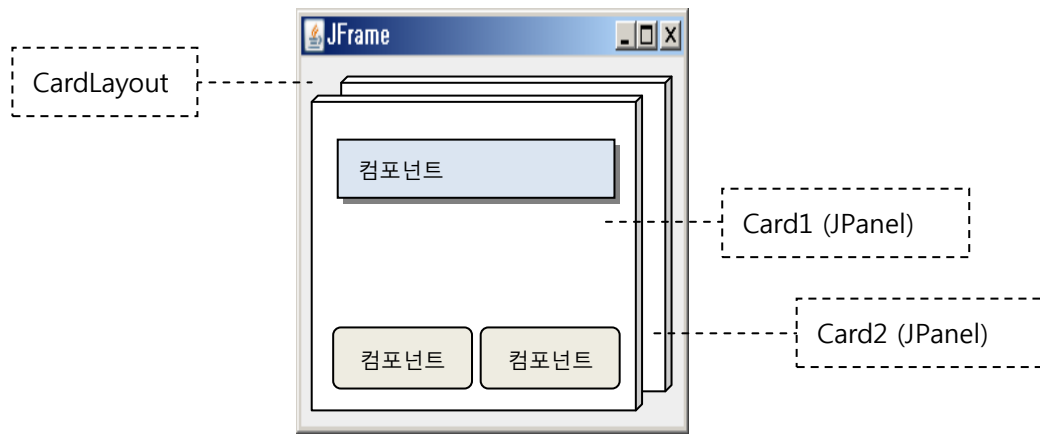
```

【실행결과】



## 4.6 CardLayout

CardLayout 배치 관리자는 이름에서도 알 수 있듯이, 여러 장의 카드를 포개 놓고 한 번에 하나의 카드를 보여주는 역할을 한다. 이 때 카드는 하나의 JPanel 로 구성된다.



CardLayout 이 적용된 컨테이너에 카드 하나를 추가할 때에는 카드의 이름과 JPanel 을 추가할 수 있는 add() 메소드를 사용한다.

```
jFrame.getContentPane().add("Card1", jPanel1);
jFrame.getContentPane().add("Card2", jPanel2);
```

여러개의 카드가 추가되더라도 제일 먼저 추가된 카드만 보이게 된다. 다른 카드는 아래에 겹쳐 있어 볼 수 없는데, 이들 카드를 나타나도록 하기 위해서 CardLayout 의 first(), last(), next(), show() 메소드를 호출하면 된다.

CardLayout 메소드	설명
first(Container container)	첫번째 배치한 카드를 보이게 한다.
last(Container container)	마지막에 배치한 카드를 보이게 한다.
next(Container container)	현재 카드 다음에 배치한 카드를 보이게 한다.
show(Container container, String name)	지정된 이름의 카드를 보이게 한다.

메소드의 첫 번째 매개값인 Container 는 CardLayout 을 사용하는 컨테이너이고, show() 메소드의 두번째 매개값은 배치할 때 주어진 카드 이름이다. 다음 예제는 3 개의 색깔 카드를 JFrame 에 추가하고 2 초 간격으로 카드를 변경해서 보여준다.

### 【CardLayoutExample.java】 3 개의 색깔 카드를 갖는 JFrame

```
1 public class CardLayoutExample extends JFrame {
2     private JPanel redCard, greenCard, blueCard;
```

```

3
4 public CardLayoutExample() {
5     this.setTitle("CardLayoutExample");
6     this.setSize(250, 400);
7     this.setResizable(false);
8     this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9
10    this.getContentPane().setLayout(new CardLayout());
11    this.getContentPane().add("RedCard", getRedCard());
12    this.getContentPane().add("GreenCard", getGreenCard());
13    this.getContentPane().add("BlueCard", getBlueCard());
14 }
15
16 public JPanel getRedCard() {
17     if(redCard == null) {
18         redCard = new JPanel();
19         redCard.setBackground(Color.RED);
20     }
21     return redCard;
22 }
23
24 public JPanel getGreenCard() {
25     if(greenCard == null) {
26         greenCard = new JPanel();
27         greenCard.setBackground(Color.GREEN);
28     }
29     return greenCard;
30 }
31
32 public JPanel getBlueCard() {
33     if(blueCard == null) {
34         blueCard = new JPanel();
35         blueCard.setBackground(Color.BLUE);
36     }
37     return blueCard;
38 }
39
40 public static void main(String[] args) {
41     SwingUtilities.invokeLater(new Runnable() {
42         public void run() {

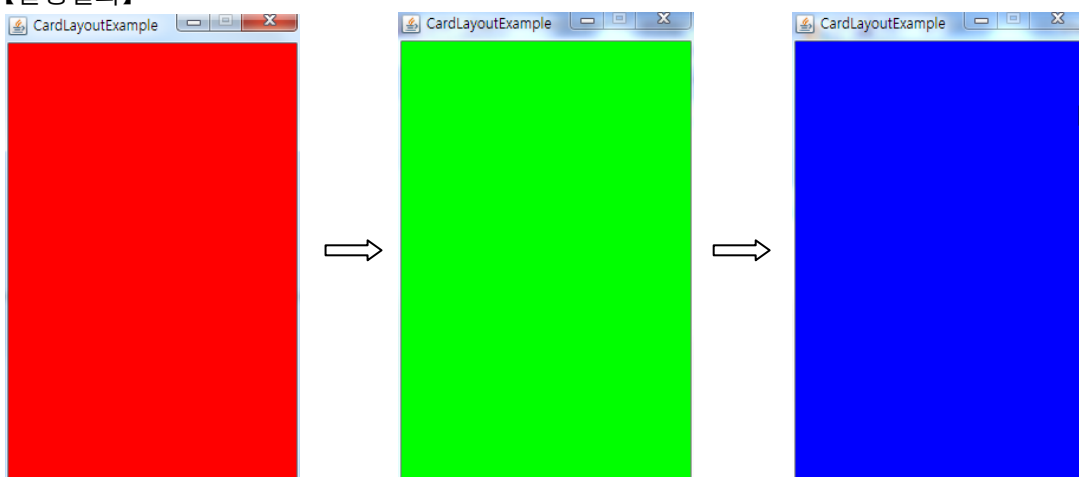
```

```

43     final CardLayoutExample JFrame = new CardLayoutExample();
44     JFrame.setVisible(true);
45     Thread thread = new Thread() {
46         @Override
47         public void run() {
48             for(int i=0; i<2; i++) {
49                 try { Thread.sleep(2000); } catch(InterruptedException e) {}
50                 SwingUtilities.invokeLater(new Runnable() {
51                     @Override
52                     public void run() {
53                         CardLayout cardLayout =
54                             (CardLayout) JFrame.getContentPane().getLayout();
55                         cardLayout.next(JFrame.getContentPane());
56                     }
57                 });
58             }
59         }
60     };
61     thread.start();
62 }
63 };
64 }
65 }

```

#### 【실행결과】



## 4.7 NullLayout

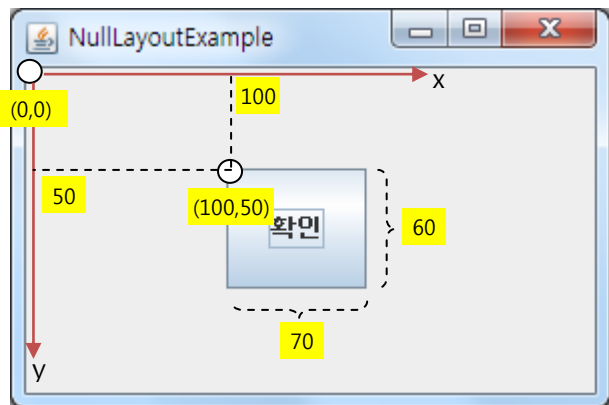
NullLayout 은 컨테이너의 `setLayout()` 메소드에 배치 관리자 대신 매개값으로 `null` 로 설정한 것을 말한다. 이것은 어떠한 배치 관리자도 사용하지 않고, 좌표값으로 컴포넌트를 배치함을 뜻한다.

```
jFrame.getContentPane().setLayout(null);
```

컨테이너가 컴포넌트를 배치할 때 좌표 값을 주는 것이 아니라, 컴포넌트가 컨테이너의 어떤 위치에 배치될 것인지 `setBounds()` 메소드로 좌표값을 설정해야 한다.

```
컴포넌트.setBounds(int x, int y, int width, int height);
```

x, y 좌표값은 픽셀 단위를 사용한다. 컨테이너의 좌측 상단이 0, 0 이고 우측이 x 축, 아래쪽이 y 축이다. 좌표값으로 컴포넌트를 배치할 경우 대개 컨테이너는 크기가 늘어나지 않는 고정 크기를 갖는다. 따라서 x 의 최대 값은 컨테이너의 폭(width)가 되고, y 의 최대 값은 컨테이너의 높이(height)에 해당 된다. `setBounds()` 메소드의 세번째 매개 변수인 width 는 컴포넌트의 폭을 말하고, 네번째 매개 변수인 height 는 컴포넌트의 높이를 말한다. 다음 예제는 JButton 을 옆 그림과 같이 배치한다.



### 【NullLayoutExample.java】좌표값으로 컴포넌트 배치

```
1 public class NullLayoutExample extends JFrame {
2     private JButton btnOk;
3
4     public NullLayoutExample() {
5         this.setTitle("NullLayoutExample");
6         this.setSize(300, 200);
7         this.setResizable(false);
8         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9
10        this.getContentPane().setLayout(null);
11        this.getContentPane().add(getBtnOk());
12    }
13
14    public JButton getBtnOk() {
15        if(btnOk == null) {
```

```

16     btnOk = new JButton();
17     btnOk.setText("확인");
18     btnOk.setBounds(100, 50, 70, 60);
19 }
20 return btnOk;
21 }
22
23 public static void main(String[] args) {
24     SwingUtilities.invokeLater(new Runnable() {
25         public void run() {
26             NullLayoutExample jFrame = new NullLayoutExample();
27             jFrame.setVisible(true);
28         }
29     });
30 }
31 }

```

## 4.8 pack() 메소드

JWindow, JFrame, JDialog 와 같이 java.awt.Window 를 상속받는 최상위 레벨 컨테이너는 pack() 이라는 메소드를 사용해서 내부의 컴포넌트의 크기에 맞게 컨테이너의 크기를 자동으로 조절한다. 컴포넌트들은 PreferredSize 라는 속성이 있는데, 이것은 컴포넌트의 기본 배치 크기를 말한다. 컨테이너의 pack() 메소드가 호출되면 내부 컴포넌트의 getPreferredSize()를 호출해서 컴포넌트의 기본 배치 크기를 알아낸 뒤, 컨테이너의 크기를 계산한다. 컨테이너의 setSize() 메소드는 직접 컨테이너의 폭과 높이를 설정하지만, pack() 메소드는 내부 컴포넌트의 크기에 따라 컨테이너의 크기가 결정된다. 따라서 pack() 메소드를 호출하는 시점은 컨테이너에 컴포넌트들이 모두 배치가 끝난 시점이어야 한다. 다음 예제는 JFrame 에 두개의 JButton 을 추가하고 pack() 메소드를 호출했다. JFrame 의 크기는 두 버튼의 크기에 최대한 맞추게 된다.

### **【PackExample.java】 pack() 메소드로 컨테이너 크기 결정**

```

1 public class PackExample extends JFrame {
2     private JButton btnOk;
3     private JButton btnCancel;
4
5     public PackExample() {
6         this.setTitle("FlowLayoutExample");
7         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
8         this.setLayout(new FlowLayout());
9         this.getContentPane().add(getBtnOk());

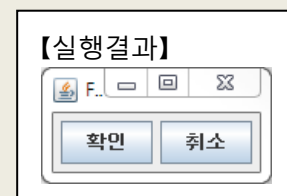
```



```

10     this.getContentPane().add(getBtnCancel());
11     this.pack();
12 }
13
14 private JButton getBtnOk() {
15     if(btnOk == null) {
16         btnOk = new JButton();
17         btnOk.setText("확인");
18     }
19     return btnOk;
20 }
21
22 private JButton getBtnCancel() {
23     if(btnCancel == null) {
24         btnCancel = new JButton();
25         btnCancel.setText("취소");
26     }
27     return btnCancel;
28 }
29
30 public static void main(String[] args) {
31     SwingUtilities.invokeLater(new Runnable() {
32         public void run() {
33             PackExample jFrame = new PackExample();
34             jFrame.setVisible(true);
35         }
36     });
37 }
38 }

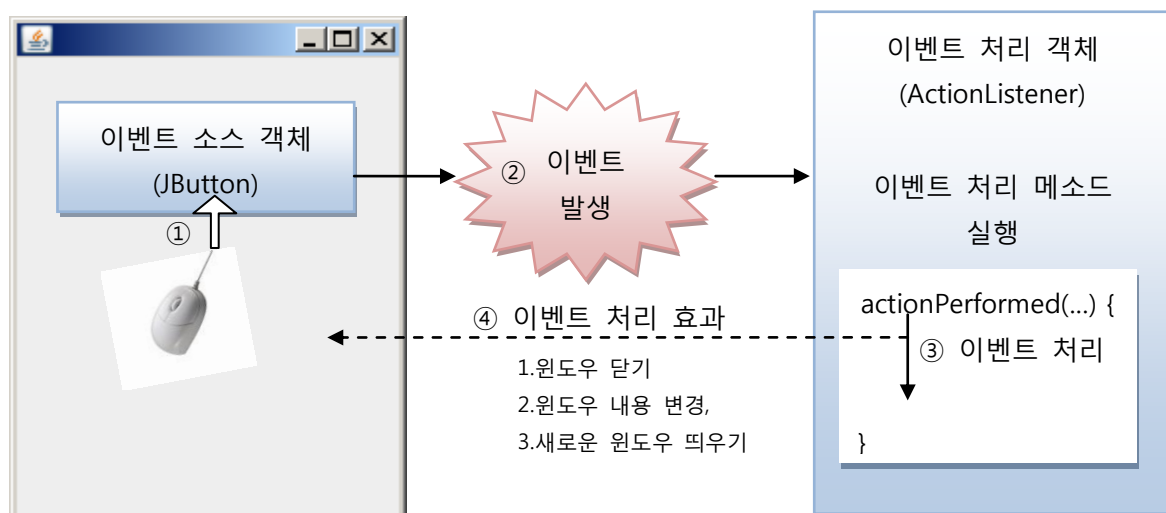
```



## 6 절. 이벤트 처리

### 5.1 이벤트 처리 방법

UI 프로그램은 사용자와 상호 작용을 하면서 코드를 실행한다. 사용자가 UI 의 컴포넌트를 사용하는 순간 이벤트(event)가 발생하고 프로그램은 이벤트를 처리하기 위해 코드를 실행한다. 자바는 이벤트 소스 객체(컨테이너, 컴포넌트) 와 이벤트 처리 객체(리스너:Listener)를 분리하는 위임형(Delegation) 방식을 사용한다. 위임형 방식이란 이벤트 소스에서 이벤트가 발생하면, 이벤트 소스가 이벤트를 직접 처리하지 않고, 이벤트 소스에 등록된 리스너에게 이벤트를 처리를 위임하는 방식이다. 예를 들어 사용자가 JButton (이벤트 소스 객체)를 클릭하면 액션 이벤트(ActionEvent)가 발생하고, JButton 에 등록된 ActionListener 객체(이벤트 처리 객체)가 액션 이벤트를 처리한다.



이벤트 처리 객체인 리스너(Listener)는 컴포넌트에서 이벤트가 발생하면, 이벤트 처리 메소드를 실행시킨다. 이벤트 처리 메소드는 주로 현재 윈도우를 닫거나, 내용을 변경, 새로운 윈도우 또는 다이얼로그를 띄우는 내용으로 구성된다. 컴포넌트는 하나의 이벤트만 발생하는 것이 아니라 동시에 여러 개의 이벤트가 발생하기도 한다. 예를 들어 JButton 을 마우스로 클릭하면 액션 이벤트(ActionEvent)도 발생하지만, 마우스 이벤트(MouseEvent)도 발생한다. 액션 이벤트는 마우스로 클릭하거나, 엔터 키를 눌러 사용하는 컴포넌트에서 주로 발생하는 이벤트이고, 마우스 이벤트는 UI 컴포넌트에서 거의 대부분 발생하는 이벤트이다.

컴포넌트에서 발생하는 모든 이벤트를 처리하기 위해서는 이벤트별로 이벤트 처리 객체인 리스너가 필요하다. 예를 들어 JButton 에서 발생하는 액션 이벤트와 마우스 이벤트를 동시에 처리하기 위해서는 액션 리스너 객체와 마우스 리스너 객체가 모두 필요하다. 하지만, 동시에 발생하는 이벤트가 많다고 하더라도 모두 처리할 필요가 없다. 처리하고 싶은 관심 이벤트에

대해서만 리스너 객체를 만들면 된다. 다음 표는 컨테이너 및 컴포넌트에서 발생할 수 있는 대표적인 이벤트와 이벤트 처리 리스너를 정리한 표이다. 대표적인 이벤트란 프로그램에서 주로 처리되는 이벤트를 말한다.

이벤트 소스	발생 이벤트	발생 원인	리스너
JFrame	WindowEvent	 중 하나를 클릭했을 때	WindowListener
JDialog	WindowEvent	 중 하나를 클릭했을 때	WindowListener
JTextField	ActionEvent	엔터키를 눌렀을 때	ActionListener
JButton	ActionEvent	클릭했을 때	ActionListener
JRadioButton	ActionEvent	클릭했을 때	ActionListener
JCheckBox	ActionEvent	클릭했을 때	ActionListener
JMenuItem	ActionEvent	선택했을 때	ActionListener
JComboBox	ActionEvent	다른 항목을 선택했을 때	ActionListener
JList	ListSelectionEvent	다른 항목을 선택했을 때	ListSelectionListener

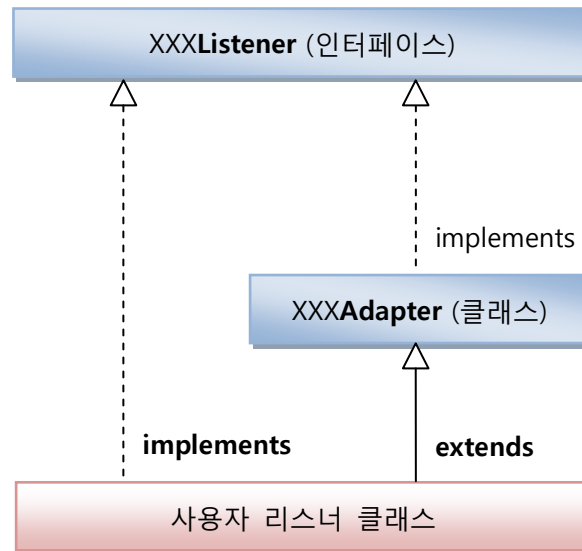
이벤트 소스에 리스너가 등록되어 있지 않으면 이벤트가 발생하더라도 아무런 작업을 하지 않는다. 이벤트를 처리하고 싶다면 반드시 이벤트 소스에 리스너를 등록해야 한다. 이벤트 소스에 리스너를 등록할 때에는 addXXXListener() 메소드를 이용하는데, 여기서 XXX 는 이벤트명이다. 예를 들어 Window 이벤트, Action 이벤트, ListSelection 이벤트를 등록하기 위해 다음 메소드를 사용할 수 있다.

```
jFrame.addWindowListener(WindowListener listener);
jButton.addActionListener(ActionListener listener);
jList.addListSelectionListener(ListSelectionListener listener);
```

자바 API 도큐먼트를 보면 컴포넌트에서 어떤 이벤트들이 발생하는지 쉽게 알 수 있다. 해당 컴포넌트의 메소드 목록을 자세히 살펴보면 자신이 가지고 있거나, 아니면 상위 클래스가 가지고 있는 addXXXListener() 메소드를 많이 볼 수 있다. 이것은 해당 컴포넌트가 XXXEvent 가 발생할 수 있으니 XXXListener 를 등록할 수 있다는 말이다. 예를 들어 대부분의 Swing 컴포넌트들은 java.awt.Component 를 상속하는데, Component 클래스는 addKeyListener()와 addMouseListener() 메서드를 가지고 있다. 따라서 대부분의 Swing 컴포넌트들은 KeyEvent 와 MouseEvent 가 발생할 수 있고, 이들을 처리하기 위해 KeyListener 와 MouseListener 를 등록할 수 있다.

## 5.2 Listener 와 Adapter

컴포넌트에서 발생하는 이벤트를 처리하기 위해서는 리스너 구현 클래스를 우선적으로 작성해야 한다. 리스너 구현 객체를 만드는 방법은 리스너 인터페이스를 구현하는 방법과 어댑터 클래스를 상속하는 방법이 있다.



리스너 인터페이스를 구현하는 방법은 리스너 인터페이스에 정의되어 있는 이벤트 처리 메소드들을 모두 재정의하는 것이다. WindowListener 에는 windowClosing() 메소드를 포함하여 7 개의 메소드가 정의되어 있다. 이들 메소드들은 WindowEvent 가 발생했을 때 사용자의 행위에 따라 개별적으로 실행된다. 예를 들어 사용자가 윈도우 상단 우측 닫기(x) 버튼을 클릭하면 windowClosing() 메소드가 실행되고, 최소화(\_) 버튼을 클릭하면 windowIconified() 메소드가 실행된다. 닫기(x) 버튼에서 발생하는 WindowEvent 만 처리하고 싶어도 windowClosing() 및 나머지 6 개를 모두 재정의하는 방법이 리스너 인터페이스를 구현하는 방법이다.

```

class MyWindowListener implements WindowListener {
    public void windowActivated(WindowEvent e) {}
    public void windowClosed(WindowEvent e) {}
    public void windowClosing(WindowEvent e) {
        //닫기(x) 버튼을 클릭했을 때 처리 방법 코딩
    }
    public void windowDeactivated(WindowEvent e) {}
    public void windowDeiconified(WindowEvent e) {}
    public void windowIconified(WindowEvent e) {}
    public void windowOpened(WindowEvent e) {}
}
  
```

이 방법 보다는 어댑터 클래스를 상속하는 방법이 좀 더 효율적인데, 관심 있는 이벤트 처리 메소드만 재정의할 수 있기 때문이다. 예를 들어 WindowAdapter 클래스를 상속할 경우 windowClosing() 메소드만 재정의하면 된다. 나머지 6 개의 메소드는 WindowAdapter 에서 내용이 없는 채로 이미 구현되어 있기 때문이다.

```

class MyWindowListener extends WindowAdapter {
    public void windowClosing(WindowEvent e) {
        //닫기(x) 버튼을 클릭했을 때 처리 방법 코딩
    }
}
  
```

```
}
}
```

리스너 인터페이스에 대응되는 어댑터 클래스가 모두 존재하는 것은 아니다. 리스너 인터페이스에 2 개 이상의 이벤트 처리 메소드가 정의 되어 있을 경우에만 어댑터 클래스가 제공된다. `ActionEvent` 를 처리하는 `ActionListener` 일 경우 `actionPerformed()` 메소드 한 개만 정의되어 있기 때문에 `ActionAdapter` 는 제공되지 않는다. 다음 표는 리스너 인터페이스와 대응되는 어댑터 클래스를 보여준다.

리스너 인터페이스	어댑터
<code>java.awt.event.WindowListener</code>	<code>java.awt.event.WindowAdapter</code>
<code>java.awt.event.MouseListener</code>	<code>java.awt.event.MouseAdapter</code>
<code>java.awt.event.KeyListener</code>	<code>java.awt.event.KeyAdapter</code>
<code>java.awt.event.ActionListener</code>	없음
<code>javax.swing.event.ListSelectionListener</code>	없음

다음 예제는 `JFrame` 의 제목 표시줄의 닫기(x) 버튼과 하단에 있는 `btnClose` 버튼 중 하나를 클릭하면 프로그램이 종료되도록 하였다. 제목 표시줄의 닫기 버튼은 `WindowAdapter` 를, `btnClose` 는 `ActionListener` 를 이용해서 리스너 객체를 생성하였다.

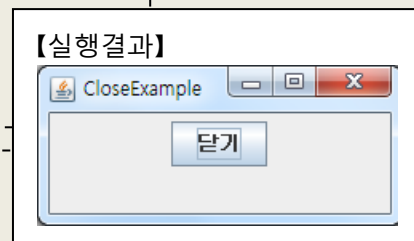
#### 【ClosableExample1.java】 윈도우 닫기 기능 구현

```
1 public class ClosableExample1 extends JFrame {
2     private JButton btnClose;
3
4     public ClosableExample1() {
5         this.setTitle("CloseExample");
6         this.setSize(300, 100);
7
8         this.setLayout(new FlowLayout());
9         this.getContentPane().add(getBtnClose());
10
11         this.addWindowListener(new MyWindowAdapter());
12     }
13
14     private JButton getBtnClose() {
15         if(btnClose == null) {
16             btnClose = new JButton();
17             btnClose.setText("닫기");
18             btnClose.addActionListener(new MyActionListener());
19         }
20     }
21 }
```

```

20     return btnClose;
21 }
22
23 public static void main(String[] args) {
24     SwingUtilities.invokeLater(new Runnable() {
25         public void run() {
26             ClosableExample1 jFrame = new ClosableExample1();
27             jFrame.setVisible(true);
28         }
29     });
30 }
31 }
32
33 class MyWindowAdapter extends WindowAdapter {
34     @Override
35     public void windowClosing(WindowEvent e) {
36         System.exit(0);
37     }
38 }
39
40 class MyActionListener implements ActionListener {
41     @Override
42     public void actionPerformed(ActionEvent e) {
43         System.exit(0);
44     }
45 }

```



### 5.3 익명 리스너 클래스

이벤트를 처리할 때 컨테이너의 필드와 메소드를 사용하는 경우가 많이 있기 때문에 리스너 클래스를 외부 클래스로 선언하게 되면 이들 필드와 메소드에 접근하는 것이 불편하다. 그래서 리스너 클래스는 일반적으로 익명 클래스로 작성한다. 익명 클래스는 매개값 위치 또는 필드 초기값 위치에 작성할 수 있다. 다음 예제는 이전 예제를 수정한 것인데 익명 클래스를 매개값 위치에 작성했다.

#### 【ClosableExample2.java】 윈도우 닫기 기능 구현

```

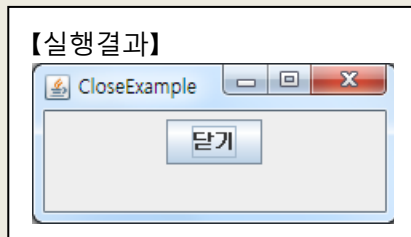
1 public class ClosableExample2 extends JFrame {
2     private JButton btnClose;

```

```

3
4 public ClosableExample() {
5     this.setTitle("CloseExample");
6     this.setSize(300, 100);
7
8     this.setLayout(new FlowLayout());
9     this.getContentPane().add(getBtnClose());
10
11     this.addWindowListener(new WindowAdapter() {
12         @Override
13         public void windowClosing(WindowEvent e) {
14             System.exit(0);
15         }
16     });
17 }
18
19 private JButton getBtnClose() {
20     if(btnClose == null) {
21         btnClose = new JButton();
22         btnClose.setText("닫기");
23         btnClose.addActionListener(new ActionListener() {
24             @Override
25             public void actionPerformed(ActionEvent e) {
26                 System.exit(0);
27             }
28         });
29     }
30     return btnClose;
31 }
32
33 public static void main(String[] args) {
34     SwingUtilities.invokeLater(new Runnable() {
35         public void run() {
36             ClosableExample2 jFrame = new ClosableExample2();
37             jFrame.setVisible(true);
38         }
39     });
40 }
41 }

```



익명 리스너 클래스를 매개값에 작성하면 해당 컴포넌트에서 발생하는 이벤트만 처리한다. 만약 복수 개의 컴포넌트에서 동일한 익명 리스너 클래스를 사용코자 한다면 필드 초기값 위치에 작성하는 것이 좋다. 예를 들어 두개의 btnOk, btnCancel 버튼이 있다고 가정해 보자. 각각의 버튼에서 자신의 ActionEvent 를 처리하기 위해 addActionListener() 메소드의 매개값으로 개별적인 ActionListener 익명 클래스를 생성하는 것 보다는 ActionListener 타입의 필드를 하나 선언하고 각각의 버튼에서 addActionListener() 메소드의 매개값으로 필드를 대입하는 것이 코드를 줄일 수 있다. 단, 필드로 선언된 리스너 구현 객체는 어떤 컴포넌트에서 이벤트가 발생되었는지 구분해서 코드를 처리해야 한다. 이벤트가 발생된 컴포넌트는 이벤트 객체의 getSource() 메소드를 이용하면 얻을 수 있다. 다음 예제는 두 개의 버튼에서 발생하는 ActionEvent 를 필드로 선언된 하나의 ActionListener 가 처리하는 방법을 보여준다.

#### [ActionListenerExample.java] 필드로 선언한 리스너 구현 객체

```

1 public class ActionListenerExample extends JFrame {
2     private JButton btnOk;
3     private JButton btnCancel;
4
5     public ActionListenerExample() {
6         this.setTitle("ActionListenerExample");
7         this.setSize(300, 100);
8         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9
10        this.setLayout(new FlowLayout());
11        this.getContentPane().add(getBtnOk());
12        this.getContentPane().add(getBtnCancel());
13    }
14
15    private ActionListener actionListener = new ActionListener() {
16        @Override
17        public void actionPerformed(ActionEvent e) {
18            if(e.getSource() == btnOk) {
19                System.out.println("확인 버튼을 클릭했습니다.");
20            } else if(e.getSource() == btnCancel) {
21                System.out.println("취소 버튼을 클릭했습니다.");
22            }
23        }
24    };
25
26    private JButton getBtnOk() {
27        if(btnOk == null) {
28            btnOk = new JButton();

```

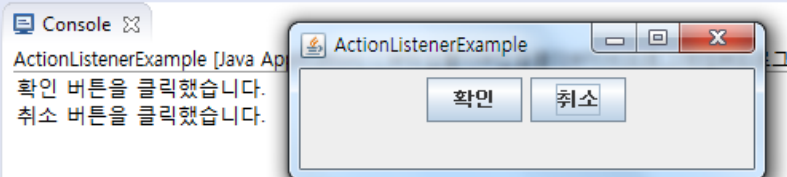


```

29     btnOk.setText("확인");
30     btnOk.addActionListener(actionListener);
31 }
32 return btnOk;
33 }
34
35 private JButton getBtnCancel() {
36     if(btnCancel == null) {
37         btnCancel = new JButton();
38         btnCancel.setText("취소");
39         btnCancel.addActionListener(actionListener);
40     }
41     return btnCancel;
42 }
43
44 public static void main(String[] args) {
45     SwingUtilities.invokeLater(new Runnable() {
46         public void run() {
47             ActionListenerExample jFrame = new ActionListenerExample();
48             jFrame.setVisible(true);
49         }
50     });
51 }
52 }

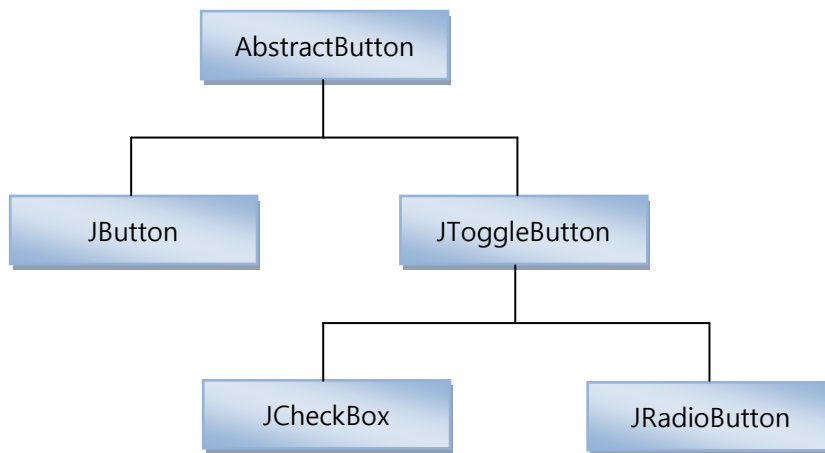
```

#### 【실행결과】



## 7 절. 버튼 컴포넌트

버튼 컴포넌트는 `AbstractButton` 을 상속받은 하위 클래스들을 말한다. 버튼 컴포넌트에는 `JButton`, `JToggleButton`, `JRadioButton`, `JCheckBox` 가 있는데, 모두 사용자가 마우스로 클릭하여 사용할 수 있도록 되어 있다. 다음은 버튼 컴포넌트의 상속 관계를 보여준다.



버튼 컴포넌트를 마우스로 클릭하면 모두 `ActionEvent` 가 발생한다. 그래서 `addActionListener()` 메소드로 `ActionListener` 객체를 등록하여 이벤트를 처리할 수 있다.

## 6.1 JButton

`JButton` 은 이미지와 텍스트로 구성된 일반적인 버튼을 만들 때 사용한다. `JButton` 의 `setText()` 메소드는 버튼의 텍스트를 설정하고, `setIcon()` 메소드는 버튼의 이미지를 설정한다.

```

JButton jButton = new JButton();
jButton.setText("새문서");
jButton.setIcon( new ImageIcon( getClass().getResource("new.gif") ) );
  
```

다음 예제는 텍스트, 이미지, 텍스트+이미지 버튼을 생성하고 이벤트를 처리했다.

### 【JButtonExample.java】 JButton

```

1 public class JButtonExample extends JFrame {
2     private JButton btn1, btn2, btn3;
3
4     public JButtonExample() {
5         this.setTitle("JButtonExample");
6         this.setSize(300, 100);
7         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
8         this.getContentPane().setLayout(new FlowLayout());
9         this.getContentPane().add(getBtn1());
10        this.getContentPane().add(getBtn2());
11        this.getContentPane().add(getBtn3());
  
```

```

12     }
13
14     //필드 Getter
15     public JButton getBtn1() {
16         if(btn1 == null) {
17             btn1 = new JButton();
18             btn1.setText("새문서");
19             btn1.addActionListener(new ActionListener() {
20                 public void actionPerformed(ActionEvent e) {
21                     JFileChooser jFileChooser = new JFileChooser();
22                     jFileChooser.showOpenDialog(JButtonExample.this);
23                 }
24             });
25         }
26         return btn1;
27     }
28
29     public JButton getBtn2() {
30         if(btn2 == null) {
31             btn2 = new JButton();
32             btn2.setIcon(new ImageIcon(getClass().getResource("new.gif")));
33             btn2.addActionListener(new ActionListener() {
34                 public void actionPerformed(ActionEvent e) {
35                     JFileChooser jFileChooser = new JFileChooser();
36                     jFileChooser.showOpenDialog(JButtonExample.this);
37                 }
38             });
39         }
40         return btn2;
41     }
42
43     public JButton getBtn3() {
44         if(btn3 == null) {
45             btn3 = new JButton();
46             btn3.setText("새문서");
47             btn3.setIcon(new ImageIcon(getClass().getResource("new.gif")));
48             btn3.addActionListener(new ActionListener() {
49                 public void actionPerformed(ActionEvent e) {
50                     JFileChooser jFileChooser = new JFileChooser();
51                     jFileChooser.showOpenDialog(JButtonExample.this);

```

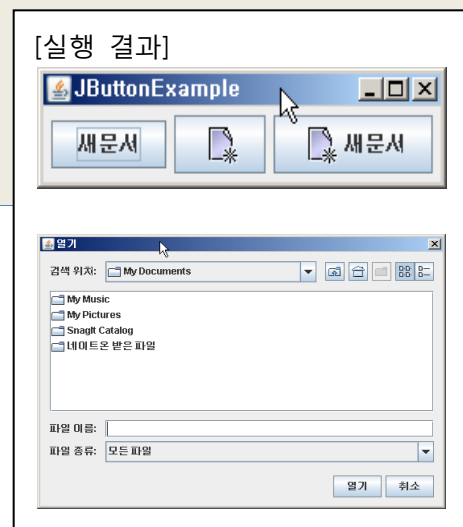
```

52     }
53     });
54 }
55     return btn3;
56 }
57
58 public static void main(String[] args) {
59     SwingUtilities.invokeLater(new Runnable() {
60         public void run() {
61             JButtonExample JFrame = new JButtonExample();
62             JFrame.setVisible(true);
63         }
64     });
65 }
66 }

```

버튼의 이벤트 처리는 파일 열기 대화상자를 보여 주도록 했다. 이 대화상자를 활용하는 방법은 13 절에서 설명한다.

## 6.2 JToggleButton



JToggleButton 은 선택된 상태와 그렇지 않은 두 가지 상태를 가지는 버튼이다. 생성 방법은 JButton 과 유사해서 텍스트와 이미지를 설정할 수 있다.

```

JToggleButton jButton = new JToggleButton();
jToggleButton.setText("확인");
jToggleButton.setIcon(new ImageIcon( getClass().getResource("ok.gif") ));

```

JToggleButton 은 선택된 상태를 가지는 버튼이기 때문에 ActionListener 보다는 다음과 같이 ItemListener 로 이벤트를 처리하는 것이 좋다. ItemEvent 의 getStateChange() 메소드는 JToggleButton 이 선택되었을 경우 ItemEvent.SELECTED 상수값을 리턴한다.

```

jToggleButton.addItemListener(new ItemListener() {
    @Override
    public void itemStateChanged(ItemEvent e) {
        if(e.getStateChange() == ItemEvent.SELECTED) {
            //선택된 상태
        } else {
            //해제된 상태
        }
    }
});

```

```

    }
}
});

```

JToggleButton 은 단독으로도 사용 가능하지만, JToggleButton 두개를 ButtonGroup 에 포함시키면 두 버튼을 배타적으로 선택할 수 있다. ButtonGroup 내부에서는 하나의 버튼만이 선택된 상태로 존재할 수 있기 때문이다.

```

ButtonGroup buttonGroup = new ButtonGroup();
buttonGroup.add( jToggleButton1 );
buttonGroup.add( jToggleButton2 );

```

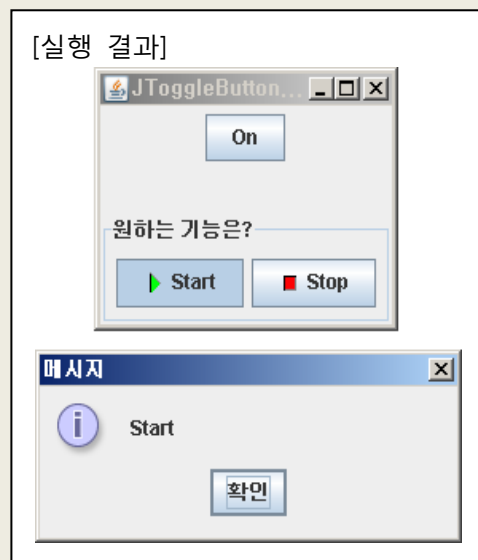
#### [JToggleButtonExample.java] JToggleButton 컴포넌트

```

1 public class JToggleButtonExample extends JFrame {
2     private JPanel pFirst;
3     private JPanel pSecond;
4     private JToggleButton tbOnOff;
5     private JToggleButton tbStart;
6     private JToggleButton tbStop;
7
8     public JToggleButtonExample() {
9         this.setTitle("JToggleButtonExample");
10        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
11        this.getContentPane().setLayout(new GridLayout(2, 1));
12        this.getContentPane().add(getPFirst());
13        this.getContentPane().add(getPSecond());
14        this.pack();
15    }
16
17    public JPanel getPFirst() {
18        if(pFirst == null) {
19            pFirst = new JPanel();
20            pFirst.add(getTbOnOff());
21        }
22        return pFirst;
23    }
24
25    public JPanel getPSecond() {
26        if(pSecond == null) {
27            pSecond = new JPanel();

```

[실행 결과]



```

28     pSecond.setBorder(new TitledBorder("원하는 기능은?"));
29     pSecond.add(getTbStart());
30     pSecond.add(getTbStop());
31     ButtonGroup buttonGroup = new ButtonGroup();
32     buttonGroup.add(getTbStart());
33     buttonGroup.add(getTbStop());
34 }
35 return pSecond;
36 }
37
38 public JToggleButton getTbOnOff() {
39     if(tbOnOff == null) {
40         tbOnOff = new JToggleButton();
41         tbOnOff.setText("On");
42         tbOnOff.addItemListener(new ItemListener() {
43             @Override
44             public void itemStateChanged(ItemEvent e) {
45                 if(e.getStateChange() == ItemEvent.SELECTED) {
46                     getTbOnOff().setText("Off");
47                 } else {
48                     getTbOnOff().setText("On");
49                 }
50             }
51         });
52     }
53     return tbOnOff;
54 }
55
56 public JToggleButton getTbStart() {
57     if(tbStart == null) {
58         tbStart = new JToggleButton();
59         tbStart.setText("Start");
60         tbStart.setIcon(new ImageIcon(getClass().getResource("start.gif")));
61         tbStart.addActionListener(new ActionListener() {
62             public void actionPerformed(ActionEvent e) {
63                 JOptionPane.showMessageDialog(JToggleButtonExample.this, "Start");
64             }
65         });
66     }
67     return tbStart;

```

```

68     }
69
70     public JToggleButton getTbStop() {
71         if(tbStop == null) {
72             tbStop = new JToggleButton();
73             tbStop.setText("Stop");
74             tbStop.setIcon(new ImageIcon(getClass().getResource("stop.gif")));
75             tbStop.addActionListener(new ActionListener() {
76                 public void actionPerformed(ActionEvent e) {
77                     JOptionPane.showMessageDialog(JToggleButtonExample.this, "Stop");
78                 }
79             });
80         }
81         return tbStop;
82     }
83
84     public static void main(String[] args) {
85         SwingUtilities.invokeLater(new Runnable() {
86             public void run() {
87                 JToggleButtonExample jFrame = new JToggleButtonExample();
88                 jFrame.setVisible(true);
89             }
90         });
91     }
92 }

```

### 6.3 JRadioButton

JRadioButton 은 JToggleButton 을 하위 클래스로서, 둥근 모양의 선택과 텍스트를 함께 보여주는 버튼이다. JRadioButton 들은 동일한 ButtonGroup 에 포함되어 한 번에 하나의 JRadioButton 만 선택된 상태를 가진다.

```

JRadioButton jRadioButton1 = new JRadioButton();
jRadioButton1.setText("남자");

JRadioButton jRadioButton2 = new JRadioButton();
jRadioButton2.setText("여자");

```

```
ButtonGroup buttonGroup = new ButtonGroup();
buttonGroup.add(jRadioButton1);
buttonGroup.add(jRadioButton2);
```

JRadioButton 은 마우스로 클릭했을 때 `ActionEvent` 가 발생하므로 `ActionListener` 로 이벤트를 처리할 수 있다.

#### **[JRadioButtonExample.java] JRadioButton**

```
1 public class JRadioButtonExample extends JFrame {
2     private JPanel radioPanel;
3     private JRadioButton rbBird;
4     private JRadioButton rbCat;
5     private JLabel lblPicture;
6
7     public JRadioButtonExample() {
8         setTitle("JRadioButtonExample");
9         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10        this.getContentPane().add(getRadioPanel(), BorderLayout.WEST);
11        this.getContentPane().add(getLblPicture(), BorderLayout.CENTER);
12        pack();
13    }
14
15    public JPanel getRadioPanel() {
16        if(radioPanel == null) {
17            radioPanel = new JPanel();
18
19            radioPanel.setLayout(new GridLayout(2, 1));
20            radioPanel.add(getRbBird());
21            radioPanel.add(getRbCat());
22
23            ButtonGroup group = new ButtonGroup();
24            group.add(getRbBird());
25            group.add(getRbCat());
26        }
27        return radioPanel;
28    }
29 }
```

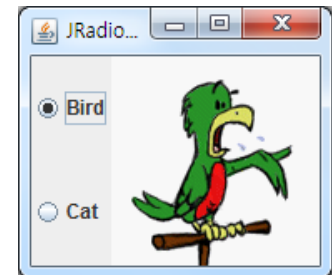


```

30 public JRadioButton getRbBird() {
31     if(rbBird == null) {
32         rbBird = new JRadioButton();
33         rbBird.setText("Bird");
34         rbBird.setSelected(true);
35         rbBird.addActionListener(new ActionListener() {
36             public void actionPerformed(ActionEvent e) {
37                 getLblPicture().setIcon(new ImageIcon(getClass().getResource("Bird.gif")));
38             }
39         });
40     }
41     return rbBird;
42 }
43
44 public JRadioButton getRbCat() {
45     if(rbCat == null) {
46         rbCat = new JRadioButton();
47         rbCat.setText("Cat");
48         rbCat.addActionListener(new ActionListener() {
49             public void actionPerformed(ActionEvent e) {
50                 getLblPicture().setIcon(new ImageIcon(getClass().getResource("Cat.gif")));
51             }
52         });
53     }
54     return rbCat;
55 }
56
57 public JLabel getLblPicture() {
58     if(lblPicture == null) {
59         lblPicture = new JLabel();
60         lblPicture.setIcon(new ImageIcon(getClass().getResource("Bird.gif")));
61     }
62     return lblPicture;
63 }
64
65 public static void main(String[] args) {
66     SwingUtilities.invokeLater(new Runnable() {
67         public void run() {
68             JRadioButtonExample jFrame = new JRadioButtonExample();
69             jFrame.setVisible(true);

```

[실행 결과]



```

70     }
71     });
72 }
73 }

```

## 6.4 JCheckBox

JCheckBox 는 사각형의 체크박스와 텍스트가 함께 보여주는 컴포넌트이다. 이 컴포넌트도 JToggleButton 을 상속하며 체크와 언체크의 두 가지 상태를 갖는다.

```

JCheckBox jCheckBox1 = new JCheckBox();
jCheckBox1.setText("축구");

JCheckBox jCheckBox2 = new JCheckBox();
jCheckBox2.setText("농구");

```

JCheckBox 도 마우스로 클릭했을 때 `ActionEvent` 가 발생하므로 `ActionListener` 로 이벤트를 처리할 수 있다. JCheckBox 가 선택되었는지 확인하는 방법은 `isSelected()` 메소드의 리턴값을 확인하면 된다. `true` 가 리턴되면 선택이 된 것이다.

### [JCheckBoxExample.java] JCheckBox

```

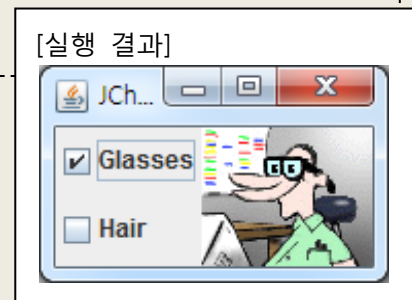
1 public class JCheckBoxExample extends JFrame {
2     private JPanel pWest;
3     private JCheckBox cbGlasses;
4     private JCheckBox cbHair;
5     private JLabel lblPicture;
6
7     public JCheckBoxExample() {
8         this.setTitle("JCheckBoxExample");
9         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10        this.getContentPane().add(getPWest(), BorderLayout.WEST);
11        this.getContentPane().add(getLblPicture(), BorderLayout.CENTER);
12        this.pack();
13    }
14
15    public JPanel getPWest() {
16        if(pWest == null) {
17            pWest = new JPanel(new GridLayout(2,1));
18            pWest.add(getCbGlasses());

```

```

19     pWest.add(getCbHair());
20 }
21 return pWest;
22 }
23
24 public JCheckBox getCbGlasses() {
25     if(cbGlasses == null) {
26         cbGlasses = new JCheckBox();
27         cbGlasses.setText("Glasses");
28         cbGlasses.addActionListener(actionListener);
29     }
30     return cbGlasses;
31 }
32
33 public JCheckBox getCbHair() {
34     if(cbHair == null) {
35         cbHair = new JCheckBox();
36         cbHair.setText("Hair");
37         cbHair.addActionListener(actionListener);
38     }
39     return cbHair;
40 }
41
42 public JLabel getLblPicture() {
43     if(lblPicture == null) {
44         lblPicture = new JLabel();
45         lblPicture.setIcon(new ImageIcon(getClass().getResource("geek.gif")));
46     }
47     return lblPicture;
48 }
49
50 private ActionListener actionListener = new ActionListener() {
51     @Override
52     public void actionPerformed(ActionEvent e) {
53         if(cbGlasses.isSelected() && cbHair.isSelected()) {
54             lblPicture.setIcon(new ImageIcon(getClass().getResource(
55                 "geek-glasses-hair.gif")));
56         } else if(cbGlasses.isSelected()) {
57             lblPicture.setIcon(new ImageIcon(getClass().getResource("geek-glasses.gif")));
58         } else if(cbHair.isSelected()) {

```



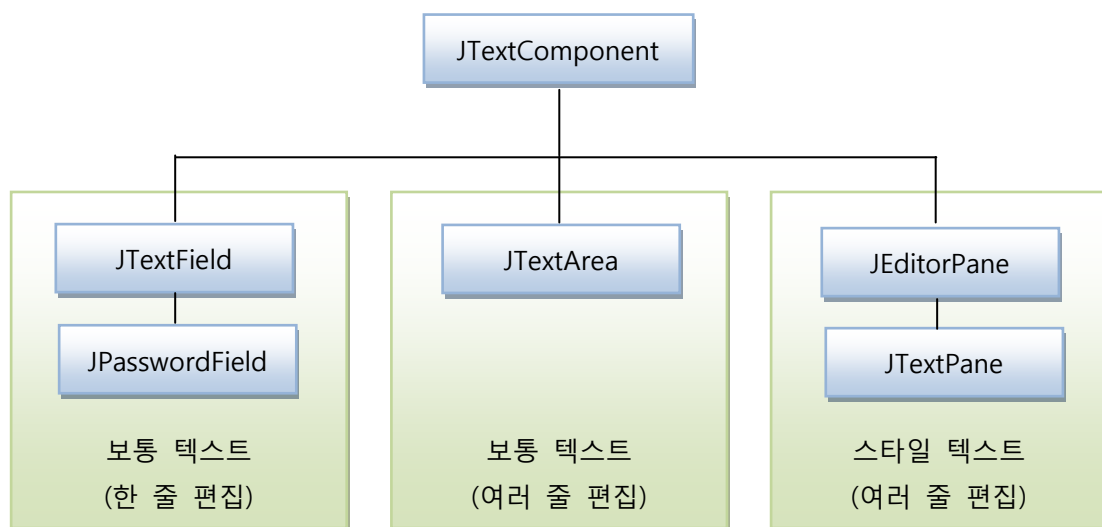
```

59     lblPicture.setIcon(new ImageIcon(getClass().getResource("geek-hair.gif")));
60 } else {
61     lblPicture.setIcon(new ImageIcon(getClass().getResource("geek.gif")));
62 }
63 }
64 };
65
66 public static void main(String[] args) {
67     SwingUtilities.invokeLater(new Runnable() {
68         public void run() {
69             JCheckBoxExample jFrame = new JCheckBoxExample();
70             jFrame.setVisible(true);
71         }
72     });
73 }
74 }

```

## 8 절. 텍스트 컴포넌트

텍스트 컴포넌트는 텍스트를 나타내거나, 편집할 수 있는 컴포넌트를 말한다. 텍스트 컴포넌트에는 JLabel, JTextField, JPasswordField, JTextArea, JEditorPane, JTextPane 등이 있다. 이 중에서 JLabel 만 텍스트를 편집할 수 없고, 나머지는 텍스트를 편집할 수 있다. 편집 가능한 텍스트 컴포넌트는 모두 JTextComponent 를 상속받아서, 각 컴포넌트의 특징에 맞게 설계되었다.



JTextField 와 JPasswordField 는 단일 라인의 텍스트를 편집할 수 있고, JTextArea, JEditorPane, JTextPane 은 멀티 라인 편집을 지원한다.

## 7.1 JLabel

JLabel 은 편집할 수 없는 한 줄의 간단한 텍스트와 정적인 이미지를 보여주는 컴포넌트이다. JLabel 에 텍스트와 이미지를 설정하는 방법은 다음과 같다.

```
JLabel jLabel = new JLabel();
jLabel.setText( "텍스트" );
jLabel.setIcon( new ImageIcon( getClass().getResource("이미지파일") ) );
```

텍스트와 이미지의 배치는 정렬(alignment)과 위치(position) 그리고 간격(gap)으로 조절할 수 있다. 정렬은 JLabel 전체 내용물의 위치를 의미하고, 위치는 이미지와 텍스트 사이의 상대적인 위치를 의미한다. 그리고 간격은 텍스트와 이미지의 간격이다.

```
setHorizontalAlignment(
    JLabel.LEFT | JLabel.CENTER | JLabel.RIGHT | JLabel.LEADING | JLabel.TRAILING);
setVerticalAlignment( JLabel.TOP | JLabel.CENTER | JLabel.BOTTOM );

setHorizontalTextPosition(
    JLabel.LEFT | JLabel.CENTER | JLabel.RIGHT | JLabel.LEADING | JLabel.TRAILING );
setVerticalTextPosition( JLabel.TOP | JLabel.CENTER | JLabel.BOTTOM );

setIconTextGap( iconTextGap );
```

JLabel 의 경계선은 기본적으로 없기 때문에 경계선의 모양을 주고 싶다면 setBorder() 메소드를 사용하면 된다. setBorder() 메소드는 JComponent 에 선언된 메소드이기 때문에 대부분의 Swing 컴포넌트에서 사용할 수 있다.

```
setBorder(Border border);
```

매개값은 Border 인터페이스 구현 객체인데, Border 구현 클래스는 javax.swing.border 패키지에 포함되어 있다. 예를 들어 조각칼로 판 모양의 경계를 사용하고 싶다면 다음과 같이 EtchedBorder 를 지정하면 된다.

```
setBorder(new EtchedBorder());
```

JLabel 은 마우스로 클릭할 수 없고, 키보드로 편집할 수도 없기 때문에, 특별한 이벤트가 발생하지 않는다.

**[JLabelExample.java] JLabel**

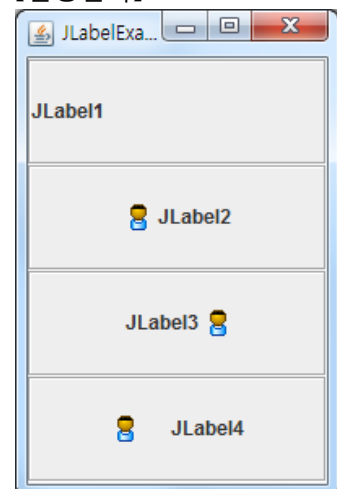
```
1 public class JLabelExample extends JFrame {
2     private JLabel jLabel1, jLabel2, jLabel3, jLabel4;
3
4     public JLabelExample() {
5         this.setTitle("JLabelExample");
6         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7         this.getContentPane().setLayout(new GridLayout(4,1));
8
9         this.getContentPane().add(getJLabel1());
10        this.getContentPane().add(getJLabel2());
11        this.getContentPane().add(getJLabel3());
12        this.getContentPane().add(getJLabel4());
13        this.setSize(200, 300);
14    }
15
16    public JLabel getJLabel1() {
17        if(jLabel1 == null) {
18            jLabel1 = new JLabel();
19            jLabel1.setText("JLabel1");
20            jLabel1.setHorizontalAlignment(JLabel.LEFT);
21            jLabel1.setBorder(new EtchedBorder());
22        }
23        return jLabel1;
24    }
25
26    public JLabel getJLabel2() {
27        if(jLabel2 == null) {
28            jLabel2 = new JLabel();
29            jLabel2.setText("JLabel2");
30            jLabel2.setIcon(new ImageIcon(getClass().getResource("user.gif")));
31            jLabel2.setHorizontalAlignment(JLabel.CENTER);
32            jLabel2.setBorder(new EtchedBorder());
33        }
34        return jLabel2;
35    }
36
37    public JLabel getJLabel3() {
38        if(jLabel3 == null) {
```

```

39     jLabel3 = new JLabel();
40     jLabel3.setText("JLabel3");
41     jLabel3.setIcon(new ImageIcon(getClass().getResource("user.gif")));
42     jLabel3.setHorizontalAlignment(JLabel.CENTER);
43     jLabel3.setHorizontalTextPosition(JLabel.LEFT);
44     jLabel3.setBorder(new EtchedBorder());
45 }
46 return jLabel3;
47 }
48
49 public JLabel getJLabel4() {
50     if(jLabel4 == null) {
51         jLabel4 = new JLabel();
52         jLabel4.setText("JLabel4");
53         jLabel4.setIcon(new ImageIcon(getClass().getResource("user.gif")));
54         jLabel4.setHorizontalAlignment(JLabel.CENTER);
55         jLabel4.setIconTextGap(20);
56         jLabel4.setBorder(new EtchedBorder());
57     }
58     return jLabel4;
59 }
60
61 public static void main(String[] args) {
62     SwingUtilities.invokeLater(new Runnable() {
63         public void run() {
64             JLabelExample jFrame = new JLabelExample();
65             jFrame.setVisible(true);
66         }
67     });
68 }
69 }

```

#### 【실행 결과】



## 7.2 JTextField 와 JPasswordField

JTextField 와 JPasswordField 는 단일 라인의 텍스트 입력란을 제공하는 컴포넌트이다. 차이점은 JPasswordField 는 사용자의 입력을 다른 사람이 볼 수 없도록 숨긴다.

```

JTextField jTextField = new JTextField();
JPasswordField jPasswordField = new JPasswordField();

```

사용자가 입력한 텍스트는 JTextField 일 경우 getText() 메소드로, JPasswordField 일 경우에는 getPassword() 메소드로 얻을 수 있다. getText()는 String 타입으로 리턴하지만, getPassword()는 char[] 배열로 리턴하므로 String 타입으로 변환할 필요가 있다.

```
String inputData = jTextField.getText();
String inputData = new String( jPasswordField.getPassword() );
```

JTextField 와 JPasswordField 는 두 가지의 주요 이벤트가 발생한다. 키보드로 문자를 입력할 때 마다 KeyEvent 가 발생하고, 엔터키(Enter Key)를 입력하면(ActionEvent)가 발생한다. 사용자가 입력하는 각 문자 마다 처리할 내용이 있다면 KeyEvent 를 처리하는 것이 좋고, 엔터키를 누르기 전까지 입력된 모든 문자들을 한꺼번에 처리하려면(ActionEvent)를 처리하는 것이 좋다. KeyEvent 를 처리하기 위해 KeyListener 객체를 등록하는 코드는 다음과 같다. KeyListener 의 keyPressed() 메소드는 키보드를 누를 때 마다 실행된다.

```
jTextField.addKeyListener( new KeyAdapter() {
    public void keyPressed(KeyEvent e) {
        char keyCar = e.getKeyChar(); //입력된 문자 얻기
        int keyCode = e.getKeyCode(); //입력된 키코드 얻기
        ...
    }
});
```

KeyEvent 의 getKeyChar() 메소드는 입력된 키문자를 리턴하고, getKeyCode() 메소드는 키코드를 리턴한다. 키코드는 KeyEvent 클래스의 상수로 선언되어 있기 때문에 키보드에서 어떤 키가 입력되었는지 확인하려면 상수와 비교하면 된다. 예를 들어 다음은 F1, F2, F3 키가 입력되었는지 확인한다.

```
if(e.getKeyCode() == KeyEvent.VK_F1) { ... }
if(e.getKeyCode() == KeyEvent.VK_F2) { ... }
if(e.getKeyCode() == KeyEvent.VK_F3) { ... }
```

입력한 키가 유효한 키코드 범위에 속하는 지 검사할 수도 있다. 예를 들어 다음은 알파벳을 입력했는지 확인한다.

```
if( (e.getKeyCode() >= KeyEvent.VK_A) && (e.getKeyCode() <= KeyEvent.VK_Z) ) { ... }
```

다음 예제는 아이디 입력 내용이 알파벳인지 검사하고, 패스워드를 입력하고 엔터키를 누르면 입력한 패스워드를 보여준다.



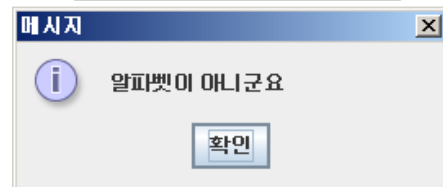
【JTextFieldJPasswordFieldExample.java】 JTextField 와 JPasswordField

```

1 public class JTextFieldJPasswordFieldExample extends JFrame {
2     private JTextField txtId;
3     private JPasswordField txtPassword;
4
5     public JTextFieldJPasswordFieldExample() {
6         this.setTitle("JTextField & JPasswordField");
7         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
8         this.getContentPane().setLayout(new GridLayout(2,2));
9         this.getContentPane().add(new JLabel("아이디", JLabel.CENTER));
10        this.getContentPane().add(getTxtId());
11        this.getContentPane().add(new JLabel("패스워드", JLabel.CENTER));
12        this.getContentPane().add(getTxtPassword());
13        this.setSize(200, 100);
14    }
15
16    public JTextField getTxtId() {
17        if(txtId == null) {
18            txtId = new JTextField();
19            txtId.addKeyListener(new KeyAdapter() {
20                public void keyPressed(KeyEvent e) {
21                    if(e.getKeyCode() >= KeyEvent.VK_A && e.getKeyCode() <= KeyEvent.VK_Z) {
22                        JOptionPane.showMessageDialog(JTextFieldJPasswordFieldExample.this,
23                            "알파벳 이군요");
24                    } else {
25                        JOptionPane.showMessageDialog(JTextFieldJPasswordFieldExample.this,
26                            "알파벳이 아니군요");
27                    }
28                }
29            });
30        }
31        return txtId;
32    }
33
34    public JPasswordField getTxtPassword() {
35        if(txtPassword == null) {
36            txtPassword = new JPasswordField();
37            txtPassword.addActionListener(new ActionListener() {
38                public void actionPerformed(ActionEvent e) {
39                    String password = new String(txtPassword.getPassword());

```

[실행 결과]



```

40         JOptionPane.showMessageDialog(JTextFieldJPasswordFieldExample.this,
41             "입력한 패스워드: " + password);
42     }
43     });
44 }
45 return txtPassword;
46 }
47
48 public static void main(String[] args) {
49     SwingUtilities.invokeLater(new Runnable() {
50         public void run() {
51             JTextFieldJPasswordFieldExample jFrame =
52                 new JTextFieldJPasswordFieldExample();
53             jFrame.setVisible(true);
54         }
55     });
56 }
57 }

```

### 7.3 JTextArea

JTextArea 는 멀티 라인의 텍스트를 편집할 수 있는 컴포넌트이다. JTextArea 는 자체적으로 스크롤을 제공하지 않으므로 JScrollPane 에 추가해서 사용된다.

```

JTextArea jTextArea = new JTextArea();
JScrollPane jScrollPane = new JScrollPane( jTextArea );

```

JTextArea 에서 키보드로 텍스트를 편집할 경우에는 스크롤이 따라 움직이지만, 프로그램에 의해서 편집이 될 경우에는 스크롤이 따라 움직이지 않는 경우가 있다. 이럴 때에는 다음 코드를 추가해 주면 스크롤이 자동적으로 내용에 맞게 움직이게 된다.

```

jTextArea.setCaretPosition( jTextArea.getText().length() );

```

다음 예제는 채팅창을 흉내내어 입력한 내용을 전송하면 JTextArea 에 출력하도록 했다.

#### 【JTextAreaExample.java】JTextArea 컴포넌트

```

1 public class JTextAreaExample extends JFrame {
2     private JTextArea txtDisplay;
3     private JPanel pSouth;

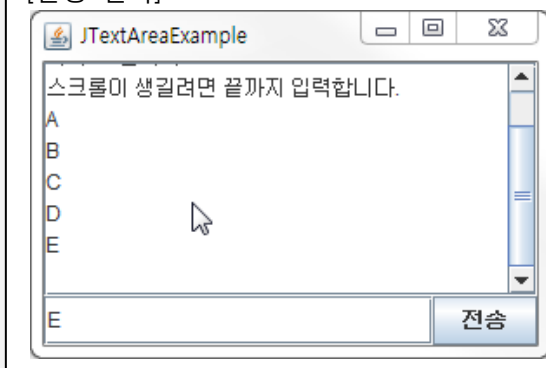
```

```

4 private JTextField txtInput;
5 private JButton btnSend;
6
7 public JTextAreaExample() {
8     this.setTitle("JTextAreaExample");
9     this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10    this.getContentPane().add(new JScrollPane(getTxtDisplay()), BorderLayout.CENTER);
11    this.getContentPane().add(getPSouth(), BorderLayout.SOUTH);
12    this.setSize(300, 200);
13 }
14
15 public JTextArea getTxtDisplay() {
16     if(txtDisplay == null) {
17         txtDisplay = new JTextArea();
18         txtDisplay.setEditable(false);
19     }
20     return txtDisplay;
21 }
22
23 public JPanel getPSouth() {
24     if(pSouth == null) {
25         pSouth = new JPanel();
26         pSouth.setLayout(new BorderLayout());
27         pSouth.add(getTxtInput(), BorderLayout.CENTER);
28         pSouth.add(getBtnSend(), BorderLayout.EAST);
29     }
30     return pSouth;
31 }
32
33 public JTextField getTxtInput() {
34     if(txtInput == null) {
35         txtInput = new JTextField();
36     }
37     return txtInput;
38 }
39
40 public JButton getBtnSend() {
41     if(btnSend == null) {
42         btnSend = new JButton();
43         btnSend.setText("전송");

```

[실행 결과]



```

44     btnSend.addActionListener(new ActionListener() {
45         public void actionPerformed(ActionEvent e) {
46             getTxtDisplay().append(getTxtInput().getText() + "\n");
47         }
48     });
49 }
50 return btnSend;
51 }
52
53 public static void main(String[] args) {
54     SwingUtilities.invokeLater(new Runnable() {
55         public void run() {
56             JTextAreaExample jFrame = new JTextAreaExample();
57             jFrame.setVisible(true);
58         }
59     });
60 }
61 }

```

JTextArea 내용 맨 뒤에 주어진 문자열을 추가

## 7.4 JEditorPane

JEditorPane 은 다양한 타입의 문서를 보여주거나 편집이 가능한 멀티 라인의 텍스트 컴포넌트이다. 기본적으로 단순 텍스트(text/plain), HTML(text/html), RTF(text/rtf) 타입의 문서를 지원하고 다른 타입의 문서를 지원하도록 구현할 수가 있다. 다음 코드는 JEditorPane 에 문서 경로를 지정하는 방법을 보여준다.

```

JEditorPane jEditorPane = new JEditorPane();
try {
    jEditorPane.setPage(getClass().getResource("문서파일명"));
} catch (Exception e) {}
jEditorPane.setEditable(false);

```

HTML 문서를 표시할 경우에는 setEditable(false)를 호출해서 편집할 수 없도록 하고, 사용자가 링크를 클릭할 경우 HyperlinkListener 를 다음과 같이 등록해서 HyperlinkEvent 이벤트를 처리할 수 있다.

```

jEditorPane.addHyperlinkListener(new HyperlinkListener() {
    public void hyperlinkUpdate(HyperlinkEvent e) {
        if(e.getEventType() == HyperlinkEvent.EventType.ACTIVATED) {
            try {

```

```

        jEditorPane.setPage(e.getURL());
    } catch (Exception e) {}
}
}
});

```

HyperlinkEvent 는 마우스를 링크 위로 가져갔을 경우와 링크 바깥으로 가져갔을 경우, 그리고 링크를 클릭했을 경우 모두 발생한다. 그래서 HyperlinkEvent 의 getEventType() 메소드는 어떤 이벤트가 발생했느냐를 구별해주는 HyperlinkEvent.EventType 형의 상수를 리턴한다. 마우스를 링크 위로 가져갔을 경우에는 ENTERED 를 리턴하고, 마우스를 링크 바깥으로 가져갔을 경우에는 EXITED 를 리턴한다. 그리고 링크를 클릭했을 경우에는 ACTIVATED 를 리턴한다. ENTERED 와 EXITED 일 경우에는 링크의 색깔을 바꾸도록 처리하면 좋고, ACTIVATED 는 실제로 링크된 문서를 불러와 보여주도록 처리하면 된다. ACTIVATED 가 리턴되었을 때 HyperlinkEvent 의 getURL()을 호출하면 링크된 문서의 URL 을 리턴한다. 예를 들어 <a href="source.html"> 일 경우에 source.html 의 URL 을 리턴한다. 리턴된 URL 을 JEditorPane 의 setPage() 메소드의 매개 변수로 주게되면 JEditorPane 은 링크된 문서를 표시한다.

아쉬운 점은 JEditorPane 은 HTML 3.2 태그만 지원한다. HTML 4.0 이나, XHTML, HTML5 는 지원하지 않는다. 현재 대부분의 사이트가 HTML 4.0 으로 작성된 페이지들이기 때문에 JEditorPane 에서 올바르게 표시되지 않는다. 또한 플러그인을 실행할 수 없기 때문에 플래시 화면도 볼 수 없다.

#### 【JEditorPaneExample.java】 HTML 뷰어

```

1  import java.awt.*;
2  import java.io.*;
3  import javax.swing.*;
4  import javax.swing.event.*;
5
6  public class JEditorPaneExample extends JFrame {
7      private JEditorPane jEditorPane;  ● --- HTML 뷰어로서 JEditorPane 필드 선언
8
9      public JEditorPaneExample() {
10         initialize();
11     }
12
13     private void initialize() {
14         this.setTitle("JEditorPaneExample");
15         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
16         this.getContentPane().add(new JScrollPane(getJEditorPane()), BorderLayout.CENTER);
17         this.setSize(400, 300);

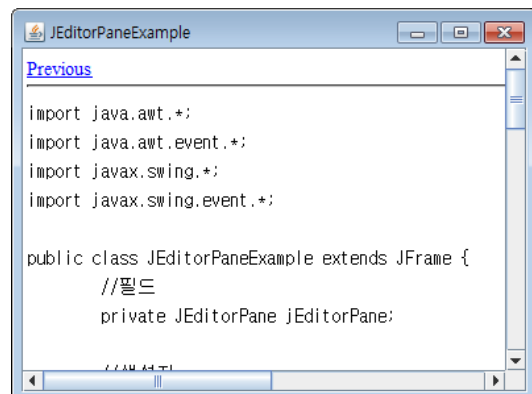
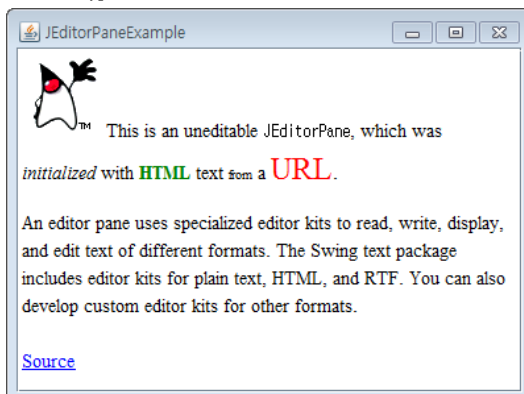
```

```

18     }
19
20     public JEditorPane getJEditorPane() {
21         if(jEditorPane == null) {
22             jEditorPane = new JEditorPane(); • --- JEditorPane 객체 생성
23             try {
24                 jEditorPane.setPage(getClass().getResource("/html/jeditorpane.html"));
25             } catch(Exception e) {}
26             jEditorPane.setEditable(false);
27             jEditorPane.addHyperlinkListener(new HyperlinkListener() {
28                 public void hyperlinkUpdate(HyperlinkEvent e) {
29                     if(e.getEventType() == HyperlinkEvent.EventType.ACTIVATED) {
30                         try {
31                             jEditorPane.setPage(e.getURL()); • --- 링크된 문서로 내용을 변경
32                         } catch(IOException e2) {}
33                     }
34                 }
35             });
36         }
37         return jEditorPane;
38     }
39
40     public static void main(String[] args) {
41         JEditorPaneExample example = new JEditorPaneExample();
42         example.setVisible(true);
43     }
44 }

```

#### [실행 결과]



## 9 절.리스트 컴포넌트

리스트 컴포넌트는 나열되어 있는 항목을 선택할 수 있는 컴포넌트를 말한다. 리스트 컴포넌트에는 JList, JComboBox 가 있다. JList 는 나열되어 있는 항목중에서 하나 이상의 항목을 선택할 수 있고, JComboBox 는 드롭다운리스트(drop down list)라고 하는데, 버튼을 클릭했을 때 항목이 나열되고 그 중 하나를 선택할 수 있다.

### 8.1 JList

JList 는 나열되어 있는 항목 중에서 하나 또는 여러 개를 선택할 수 있도록 하는 컴포넌트이다. JList 의 항목은 다음과 같이 Object[] 배열로 주거나,

```
String[] stringItems = { "one", "two", "tree", "four" };  
JList jList = new JList(stringItems);
```

다음과 같이 Vector 객체로 줄 수 있다.

```
Vector stringItems = new Vector();  
stringItems.add("one");  
stringItems.add("two");  
stringItems.add("tree");  
stringItems.add("four");  
JList jList = new JList(stringItems);
```

JList 의 기본적인 항목 컴포넌트는 JLabel 로 구성되어 있기 때문에 텍스트 이외에도 이미지 표현도 가능하다.

```
Vector imageItems = new Vector();  
imageItems.add( new ImageIcon( getClass().getResource("fruit1.gif") ));  
imageItems.add( new ImageIcon( getClass().getResource("fruit2.gif") ));  
JList jList = new JList(imageItems);
```

JList 는 자동으로 스크롤 바가 생성되지 않으므로 JScrollPane 에 추가해서 사용된다.

```
JScrollPane jScrollPane = new JScrollPane( jList );
```

사용자가 JList 의 항목을 선택하면 ListSelectionEvent 가 발생하기 때문에 ListSelectionListener 를 등록하면 이벤트를 처리할 수 있다.

```

jList.addListSelectionListener(new ListSelectionListener() {
    public void valueChanged(ListSelectionEvent e) {
        if(e.getValueIsAdjusting() == false) {
            //이벤트 처리 코드
        }
    }
});

```

JList 는 선택 항목이 변경되면 ListSelectionEvent 를 두 번 발생시킨다. 이것을 해결하기 위해서 ListSelectionEvent 에는 getValueIsAdjusting() 메소드를 제공하고 있다. 첫 번째 이벤트가 발생했을 때는 false 를 리턴하고 두 번째 이벤트가 연이어 발생하면 true 를 리턴한다. getValueIsAdjusting()가 false 를 반환할 때만 이벤트 처리 코드를 실행하도록 해야한다. JList 에서 선택된 항목을 알아내는 방법은 두가지가 있다. getSelectedIndex() 메소드는 선택된 항목의 인덱스를 리턴하고, getSelectedValue()는 선택된 항목의 값이 리턴된다. 어떤 정보가 이벤트 처리시 유리한지를 판단하고 사용하면 된다.

```

int selectedIndex = jList.getSelectedIndex();
//항목이 텍스트일 경우
String selectedItem = (String) jList.getSelectedValue();
//항목이 이미지일 경우
ImageIcon selectedItem = (ImageIcon) jList.getSelectedValue();

```

만약 Shift 또는 Ctrl 키를 이용해서 두 개 이상의 항목이 선택했다면 선택된 항목의 인덱스와 값을 배열로 리턴하는 다음 두 메소드를 사용할 수 있다.

```

int[] selectedIndices = jList.getSelectedIndices();
//항목이 텍스트일 경우
String[] selectedItems = (String[]) jList.getSelectedValues();
//항목이 이미지일 경우
ImageIcon[] selectedItem = (ImageIcon[]) jList.getSelectedValue();

```

#### **【JListExample.java】과일을 선택하는 JList**

```

1 public class JListExample extends JFrame {
2     private JPanel pWest;
3     private JList listString;
4     private JList listImage;
5     private JLabel jLabel;
6
7     public JListExample() {
8         this.setTitle("JListExample");

```



```

9      this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10     this.setSize(250, 200);
11     this.getContentPane().setBackground(Color.WHITE);
12     this.getContentPane().add(getPWest(), BorderLayout.WEST);
13     this.getContentPane().add(getJLabel(), BorderLayout.CENTER);
14 }
15
16 public JPanel getPWest() {
17     if(pWest == null) {
18         pWest = new JPanel();
19         pWest.setLayout(new GridLayout(2,1));
20         pWest.add(new JScrollPane(getListString()));
21         pWest.add(new JScrollPane(getListImage()));
22     }
23     return pWest;
24 }
25 public JList getListString() {
26     if(listString == null) {
27         String[] arrString = {
28             "Cantaloupe", "Grapefruit", "Grapes", "Kiwi", "Peach",
29             "pineapple", "strawberry", "tomato", "watermelon"
30         };
31         listString = new JList(arrString);
32         listString.addListSelectionListener(new ListSelectionListener() {
33             public void valueChanged(ListSelectionEvent e) {
34                 if(!e.getValueIsAdjusting()) {
35                     int selectedIndex = listString.getSelectedIndex();
36                     ImageIcon image = new ImageIcon(
37                         getClass().getResource("fruit" + (selectedIndex+1) + ".jpg"));
38                     getJLabel().setIcon(image);
39                 }
40             }
41         });
42     }
43     return listString;
44 }
45
46 public JList getListImage() {
47     if(listImage == null) {
48         Vector vImage = new Vector();

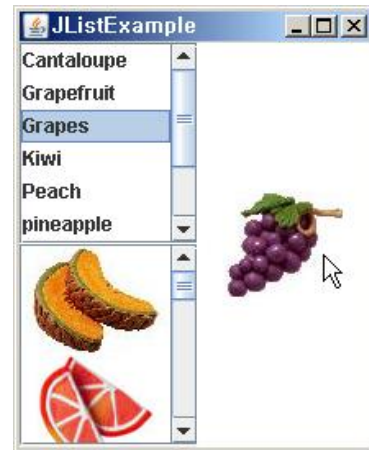
```

```

49         for(int i=1;i<10;i++) {
50             ImageIcon image = new ImageIcon(
51                 getClass().getResource("fruit" + i + ".jpg"));
52             vImage.addElement(image);
53         }
54         listImage = new JList(vImage);
55         listImage.addListSelectionListener(new ListSelectionListener() {
56             public void valueChanged(ListSelectionEvent e) {
57                 if(!e.getValueIsAdjusting()) {
58                     ImageIcon image = (ImageIcon) listImage.getSelectedValue();
59                     getJLabel().setIcon(image);
60                 }
61             }
62         });
63     }
64     return listImage;
65 }
66
67 public JLabel getJLabel() {
68     if(jLabel == null) {
69         jLabel = new JLabel();
70         jLabel.setHorizontalAlignment(JLabel.CENTER);
71     }
72     return jLabel;
73 }
74
75 public static void main(String[] args) {
76     SwingUtilities.invokeLater(new Runnable() {
77         public void run() {
78             JListExample jFrame = new JListExample();
79             jFrame.setVisible(true);
80         }
81     });
82 }
83

```

[실행 결과]



## 8.2 JComboBox

JComboBox 는 드롭다운리스트(drop down list)라고 하는데, 버튼을 클릭하면 항목이 나열되고 그 중 하나를 선택할 수 있다. JComboBox 의 항목은 다음과 같이 Object[] 배열로 주거나,

```
String[] stringItems = { "one", "two", "tree", "four" }  
JComboBox jComboBox = new JComboBox(stringItems);
```

다음과 같이 Vector 객체로 줄 수 있다.

```
Vector stringItems = new Vector();  
stringItems.add("one");  
stringItems.add("two");  
stringItems.add("tree");  
stringItems.add("four");  
JComboBox jComboBox = new JComboBox(stringItems);
```

JComboBox 의 항목은 문자열 이외에도 이미지도 가능하다. 이미지 항목은 ImageIcon 객체를 배열을 만들거나, Vector 항목으로 만들면 된다.

```
Vector imageItems = new Vector();  
imageItems.add( new ImageIcon( getClass().getResource("fruit1.gif") ) );  
imageItems.add( new ImageIcon( getClass().getResource("fruit2.gif") ) );  
JComboBox jComboBox = new JComboBox(imageItems);
```

JComboBox 에서도 선택된 항목을 알아내는 방법은 두가지가 있다. `getSelectedIndex()` 메소드는 선택된 항목의 인덱스를 리턴하고, `getSelectedItem()`은 선택된 항목의 값을 리턴한다. 어떤 정보가 이벤트 처리시 유리한지를 판단하고 사용하면 된다.

```
int selectedIndex = jComboBox.getSelectedIndex();  
//항목이 텍스트일 경우  
String selectedItem = (String) jComboBox.getSelectedItem();  
//항목이 이미지일 경우  
ImageIcon selectedItem = (ImageIcon) jComboBox.getSelectedItem();
```

### [JComboBoxExample.java] 과일을 선택하는 JComboBox

```
1 public class JComboBoxExample extends JFrame {
```

```

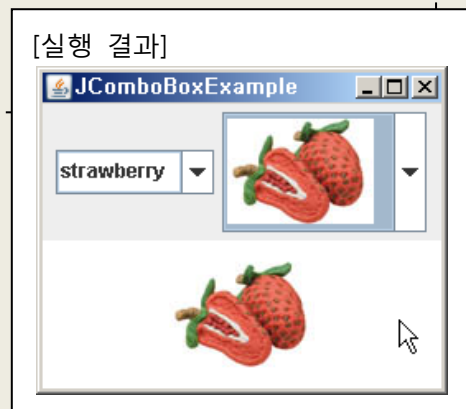
2 private JPanel pNorth;
3 private JComboBox comboString;
4 private JComboBox comboImage;
5 private JLabel jLabel;
6
7 public JComboBoxExample() {
8     this.setTitle("JComboBoxExample");
9     this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10    this.getContentPane().setBackground(Color.WHITE);
11    this.getContentPane().add(getPNorth(), BorderLayout.NORTH);
12    this.getContentPane().add(getJLabel(), BorderLayout.CENTER);
13    this.setSize(250, 200);
14 }
15
16 public JPanel getPNorth() {
17     if(pNorth == null) {
18         pNorth = new JPanel();
19         pNorth.add(getComboString());
20         pNorth.add(getComboImage());
21     }
22     return pNorth;
23 }
24
25 public JComboBox getComboString() {
26     if(comboString == null) {
27         String[] arrString = {
28             "Cantaloupe", "Grapefruit", "Grapes", "Kiwi", "Peach",
29             "pineapple", "strawberry", "tomato", "watermelon"
30         };
31         comboString = new JComboBox(arrString);
32         comboString.setBackground(Color.WHITE);
33         comboString.addActionListener(new ActionListener() {
34             public void actionPerformed(ActionEvent e) {
35                 int selectedIndex = comboString.getSelectedIndex();
36                 ImageIcon image = new ImageIcon(getClass().getResource(
37                     "fruit" + (selectedIndex+1) + ".jpg"));
38                 getJLabel().setIcon(image);
39             }
40         });
41     }

```

```

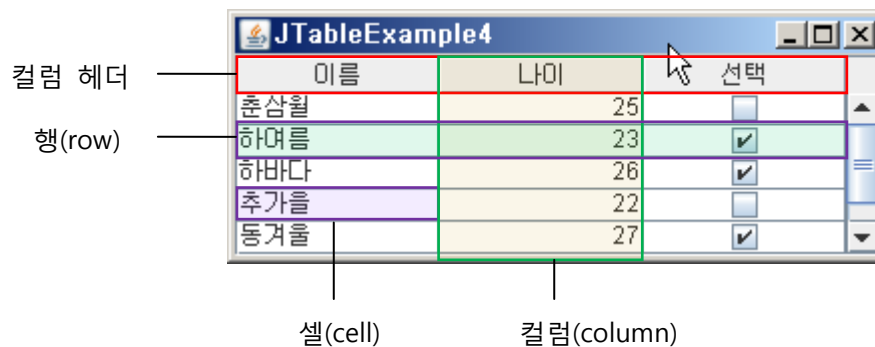
42     return comboString;
43 }
44
45 public JComboBox getComboImage() {
46     if(comboImage == null) {
47         Vector vImage = new Vector();
48         for(int i=1;i<10;i++) {
49             ImageIcon image = new ImageIcon(getClass().getResource("fruit" + i + ".jpg"));
50             vImage.add(image);
51         }
52         comboImage = new JComboBox(vImage);
53         comboImage.setBackground(Color.WHITE);
54         comboImage.addActionListener(new ActionListener() {
55             public void actionPerformed(ActionEvent e) {
56                 ImageIcon image = (ImageIcon) comboImage.getSelectedItem();
57                 getJLabel().setIcon(image);
58             }
59         });
60     }
61     return comboImage;
62 }
63
64 public JLabel getJLabel() {
65     if(jLabel == null) {
66         jLabel = new JLabel();
67         jLabel.setHorizontalAlignment(JLabel.CENTER);
68     }
69     return jLabel;
70 }
71
72 public static void main(String[] args) {
73     SwingUtilities.invokeLater(new Runnable() {
74         public void run() {
75             JComboBoxExample jFrame = new JComboBoxExample();
76             jFrame.setVisible(true);
77         }
78     });
79 }
80 }

```



## 10 절. 테이블 컴포넌트

JTable 은 테이블 형식의 데이터를 표시하고 편집할 수 있는 컴포넌트이다. JTable 은 다른 컴포넌트에 비해서 다소 복잡한 구조로 이루어져 있다.



테이블은 컬럼(column)과 행(row)으로 구성되어 있고, 컬럼과 행이 만나는 곳이 셀(cell)이다. 셀은 실제 데이터가 표시되는 곳이다. 하나의 컬럼을 구성하는 셀들은 동일한 데이터 타입을 가져야 한다. 위 테이블에서 이름 컬럼의 데이터는 모두 텍스트 이고, 나이 컬럼의 데이터는 모두 숫자로 구성되어 있는 것을 볼 수 있다.

### 9.1 테이블 생성

간단한 JTable 객체를 만들기 위해서는 먼저 컬럼 이름을 포함하고 있는 1 차원 String 배열과 셀의 데이터인 2 차원 Object 배열을 생성해야 한다. 그리고 이들을 JTable 생성자를 호출할 때 매개 변수로 넘겨준다.

```
String[] columnNames = {"이름", "나이"};
Object[][] rowData = {
    { "춘삼월", new Integer(25) },
    { "하여름", new Integer(23), },
    { "하바다", new Integer(26), },
};
JTable jTable = new JTable( rowData, columnNames );
```

2 차원 Object 배열을 생성할 때 주의할 점은 하나의 컬럼을 구성하는 셀의 데이터들은 모두 같은 타입이어야 한다. 위 코드를 보면 이름 컬럼의 데이터는 모두 String 객체이고 나이는 모두 Integer 객체로 주었다. 각 컬럼의 폭을 변경하고 싶다면 주어진 컬럼 이름에 해당하는 TableColumn 을 getColumn() 메소드로 얻고, TableColumn 의 setPreferredWidth() 메소드로 폭을 변경하면 된다.

```
TableColumn tableColumn = jTable.getColumn("컬럼 이름");
tableColumn.setPreferredWidth(폭길이);
```

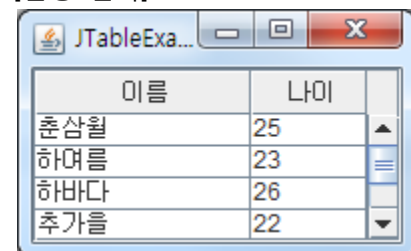
JTable 은 자체적으로 스크롤을 지원하지 않으므로 JScrollPane 에 추가해서 사용한다.

```
JScrollPane jScrollPane = new JScrollPane( jTable);
jFrame.getCententPane().add(jScrollPane, BorderLayout.CENTER);
```

#### 【JTableExample.java】 JTable 생성

```
1 public class JTableExample extends JFrame {
2     private JTable jTable;
3
4     public JTableExample() {
5         this.setTitle("JTableExample");
6         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7         this.getContentPane().add(new JScrollPane(getJTable()), BorderLayout.CENTER);
8         this.setSize(200, 125);
9     }
10
11    public JTable getJTable() {
12        if(jTable == null) {
13            String[] columnNames = { "이름", "나이" };
14            Object[][] rowData = {
15                {"춘삼월", new Integer(25) },
16                {"하여름", new Integer(23) },
17                {"하바다", new Integer(26) },
18                {"추가울", new Integer(22) },
19                {"동겨울", new Integer(27) },
20                {"동장군", new Integer(15) }
21            };
22            jTable = new JTable(rowData, columnNames);
23            jTable.getColumn("이름").setPreferredWidth(100);
24            jTable.getColumn("나이").setPreferredWidth(50);
25        }
26        return jTable;
27    }
28
29    public static void main(String[] args) {
30        SwingUtilities.invokeLater(new Runnable() {
```

[실행 결과]



이름	나이
춘삼월	25
하여름	23
하바다	26
추가울	22
동겨울	27
동장군	15

```

31         public void run() {
32             JTableExample jFrame = new JTableExample();
33             jFrame.setVisible(true);
34         }
35     });
36 }
37 }

```

## 9.2 테이블 모델

JTable 은 TableModel 객체를 사용해서 컬럼과 셀의 데이터를 관리한다. JTable 을 생성할 때 컬럼 이름인 1 차원 String 배열과 셀의 데이터인 2 차원 Object 배열을 생성자로 넘겨주면, TableModel 이 내부적으로 생성되고 이들 데이터를 관리하게 된다. JTable 은 내부적으로 생성된 TableModel 을 프로그램에서 사용할 수 있도록 getModel() 메소드를 제공하고 있다.

```
TableModel tableModel = jTable.getModel();
```

TableModel 에는 데이터에 대한 여러가지 정보를 제공하는 다음 메소드들이 있다.

메소드		용도
int	getColumnCount()	총 컬럼의 수 얻기
String	getColumnName(int columnIndex)	컬럼의 이름 얻기
int	getRowCount()	총 행의 수 얻기
Object	getValueAt(int rowIndex, int columnIndex)	셀의 데이터 얻기
void	setValueAt(Object aValue, int rowIndex, int columnIndex)	셀의 데이터 변경
void	setDataVector(Object[][] rows, Object[] columnNames) setDataVector(Vector rows, Vector columnNames);	전체 테이블 데이터를 주어진 매개값으로 대체

### 【JTableExample.java】 테이블 모델을 이용해서 정보 얻기

```

1 public class JTableExample extends JFrame {
2     private JTable jTable;
3     private JButton btnInfo;
4
5     public JTableExample() {
6         this.setTitle("JTableExample");
7         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
8         this.getContentPane().add(new JScrollPane(getJTable()), BorderLayout.CENTER);
9         this.getContentPane().add(getBtnInfo(), BorderLayout.SOUTH);
10        this.setSize(200, 200);

```

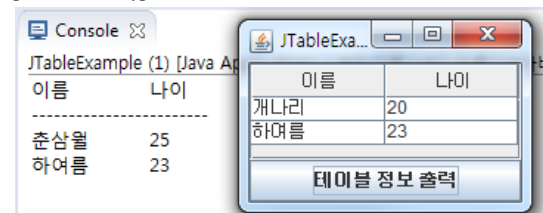


```

11     }
12
13     public JTable getJTable() {
14         if(jTable == null) {
15             String[] columnNames = { "이름", "나이" };
16             Object[][] rowData = {
17                 {"춘삼월", new Integer(25) },
18                 {"하여름", new Integer(23) }
19             };
20             jTable = new JTable(rowData, columnNames);
21         }
22         return jTable;
23     }
24
25     public JButton getBtnInfo() {
26         if(btnInfo == null) {
27             btnInfo = new JButton();
28             btnInfo.setText("테이블 정보 출력");
29             btnInfo.addActionListener(new ActionListener() {
30                 public void actionPerformed(ActionEvent e) {
31                     //테이블 모델 얻기
32                     TableModel tableModel = getJTable().getModel();
33                     //전체 컬럼수 얻기
34                     int columnCount = tableModel.getColumnCount();
35                     //전체 행수 얻기
36                     int rowCount = tableModel.getRowCount();
37                     //컬럼 이름 출력
38                     for(int i=0; i<columnCount; i++) {
39                         String columnName = tableModel.getColumnName(i);
40                         System.out.print(columnName + "WtWt");
41                     }
42                     System.out.println();
43                     System.out.println("-----");
44                     //행의 데이터 출력
45                     for(int i = 0; i<rowCount; i++) {
46                         String column0 = (String) tableModel.getValueAt(i, 0);
47                         Integer column1 = (Integer) tableModel.getValueAt(i, 1);
48                         System.out.println(column0 + "WtWt" + column1);
49                     }
50                     //1행 셀 데이터 변경

```

#### [실행 결과]



```

51         tableModel.setValueAt("개나리", 0, 0);
52         tableModel.setValueAt(new Integer(20), 0, 1);
53     }
54 });
55 }
56 return btnInfo;
57 }
58
59 public static void main(String[] args) {
60     SwingUtilities.invokeLater(new Runnable() {
61         public void run() {
62             JTableExample jFrame = new JTableExample();
63             jFrame.setVisible(true);
64         }
65     });
66 }
67 }

```

### 9.3 데이터 표현 방법 변경

컬럼 헤더나 셀의 텍스트 내용을 수평 정렬하거나, 셀의 내용을 이미지등의 여러가지 컴포넌트로 표현해야 할 경우에는 새로운 셀렌더러(TableCellRenderer)를 정의해야 한다. 새로운 셀렌더러는 표현하고자 모양의 컴포넌트를 상속하고 TableCellRenderer 인터페이스를 구현해서 만들어 진다.

```

public class MyTableCellRenderer extends 컴포넌트 implements TableCellRenderer {
    public Component getTableCellRendererComponent(...) {
        //컴포넌트를 초기화하고 반환하는 코드
        return this;
    }
};

```

TableCellRenderer 의 getTableCellRendererComponent() 메소드는 셀에 표현할 컴포넌트를 매개 변수에 따라 초기화하고 리턴해야 한다. getTableCellRendererComponent() 메소드의 매개 변수는 다음과 같다.

매개 변수	값 또는 참조
table	셀을 포함하고 있는 JTable
value	셀에 표시될 데이터

isSelected	셀을 포함하고 있는 행이 선택 되었는지 여부
hasFocus	셀에 포커스가 있는지 여부
row	셀을 포함하고 있는 행의 순번
column	셀을 포함하고 있는 컬럼의 순번

새로운 셀렌더러가 정의되었다면 TableColumn 의 setHeaderRenderer() 또는 setCellRenderer() 메소드로 헤더와 셀 렌더러를 변경하면 된다. 헤더와 셀을 다른 모양으로 변경하려면 다른 셀렌더러를 지정하면 된다.

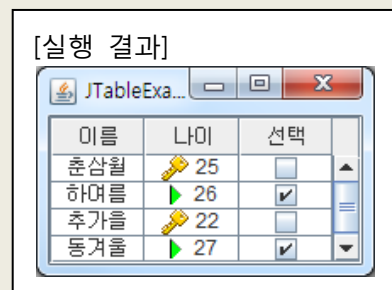
```
TableColumn tc = jTable.getColumn("컬럼 이름");
TableCellRenderer tc = new MyTableCellRenderer();
//헤더 모양 변경
tableColumn.setHeaderRenderer(tc);
//셀 모양 변경
tableColumn.setCellRenderer(tc);
```

다음 예제는 첫번째 셀은 글자만 있는 JLabel 로 표현했고, 두번째 셀을 아이콘과 글자가 포함된 JLabel 로 표현했다. 그리고 세번째 셀을 JCheckBox 로 표현했다.

#### [JTableExample7.java] 데이터 표시 방법을 변경한 테이블

```
1 public class JTableExample extends JFrame {
2     private JTable jTable;
3
4     public JTableExample() {
5         this.setTitle("JTableExample");
6         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7         this.getContentPane().add(new JScrollPane(getJTable()), BorderLayout.CENTER);
8         this.setSize(200, 125);
9     }
10
11     public JTable getJTable() {
12         if(jTable == null) {
13             String[] columnNames = {"이름", "나이", "선택"};
14             Object[][] rowData = {
15                 {"춘삼월", new Integer(25), new Boolean(false)},
16                 {"하여름", new Integer(26), new Boolean(true)},
17                 {"추가울", new Integer(22), new Boolean(false)},
18                 {"동겨울", new Integer(27), new Boolean(true)},
```

[실행 결과]



```

19     };
20     jTable = new.JTable(rowData, columnNames);
21     TableColumn tcName = jTable.getColumn("이름");
22     tcName.setCellRenderer(new NameTableCellRenderer());
23     TableColumn tcAge = jTable.getColumn("나이");
24     tcAge.setCellRenderer(new AgeTableCellRenderer());
25     TableColumn tcSelect = jTable.getColumn("선택");
26     tcSelect.setCellRenderer(new SelectTableCellRenderer());
27 }
28 return jTable;
29 }
30
31 public class NameTableCellRenderer extends JLabel implements TableCellRenderer {
32     public Component getTableCellRendererComponent(JTable table, Object value,
33         boolean isSelected, boolean hasFocus, int row, int column) {
34         setText(value.toString());
35         setFont(new Font(null, Font.PLAIN, 12));
36         setHorizontalAlignment(JLabel.CENTER);
37         setOpaque(true);
38         if(isSelected) { setBackground(Color.YELLOW); }
39         else { setBackground(Color.WHITE); }
40         return this;
41     }
42 }
43
44 public class AgeTableCellRenderer extends JLabel implements TableCellRenderer {
45     public Component getTableCellRendererComponent(JTable table, Object value,
46         boolean isSelected, boolean hasFocus, int row, int column) {
47         int age = ((Integer)value).intValue();
48         if(age <= 25) {
49             setIcon(new ImageIcon(getClass().getResource("key.gif")));
50         } else {
51             setIcon(new ImageIcon(getClass().getResource("start.gif")));
52         }
53         setText(value.toString());
54         setFont(new Font(null, Font.PLAIN, 12));
55         setHorizontalAlignment(JLabel.CENTER);
56         setOpaque(true);
57         if(isSelected) { setBackground(Color.YELLOW); }
58         else { setBackground(Color.WHITE); }

```

```

59     return this;
60 }
61 }
62
63 public class SelectTableCellRenderer extends JCheckBox implements TableCellRenderer {
64     public Component getTableCellRendererComponent(JTable table, Object value,
65         boolean isSelected, boolean hasFocus, int row, int column) {
66         Boolean boolWrapper = (Boolean) value;
67         setSelected(boolWrapper.booleanValue());
68         setHorizontalAlignment(CENTER);
69         if(isSelected) { setBackground(Color.YELLOW); }
70         else { setBackground(Color.WHITE); }
71         return this;
72     }
73 }
74
75 public static void main(String[] args) {
76     SwingUtilities.invokeLater(new Runnable() {
77         public void run() {
78             JTableExample jFrame = new JTableExample();
79             jFrame.setVisible(true);
80         }
81     });
82 }
83 }

```

37 라인과 56 라인의 `setOpaque(true)`는 테이블의 행을 선택했을 때 배경이 노란색으로 그려질 수 있도록 `JLabel` 을 불투명하게 만든다. 기본적으로는 `JLabel` 의 `Opaque` 는 `false` 이다. 즉 완전 투명이기 때문에 배경 색상을 지정해도 색상이 나오지 않는다.

## 9.4 선택 이벤트 처리

`JTable` 은 `MouseEvent` 가 발생하므로 `MouseListener` 를 등록해서 이벤트를 처리할 수 있다. `MouseListener(MouseAdapter)`의 `mouseClicked()` 메소드에서 `getSelectedColumn()`과 `getSelectedRow()` 메소드들을 이용해서 어떤 컬럼과 행이 클릭되었는지 알아낼 수 있다. 만약 선택된 셀이 없다면 `getSelectedColumn()`과 `getSelectedRow()` 메소드들은 `-1` 을 리턴한다. 따라서 `-1` 이 아닐 때 이벤트 처리 코드를 실행하면 된다.

```
jTable.addMouseListener(new MouseAdapter() {
```

```

public void mouseClicked(MouseEvent e) {
    int rowIndex = jTable.getSelectedRow();
    int columnIndex = jTable.getSelectedColumn();
    if(rowIndex != -1 || columnIndex != -1) {
        //이벤트 처리 코드
    }
}
});

```

다음 예제는 사용자가 행을 선택하면 행의 데이터를 읽고 이름과 나이를 JTextField에 표시한다.

#### 【JTableExample8.java】선택 이벤트를 처리하는 테이블

```

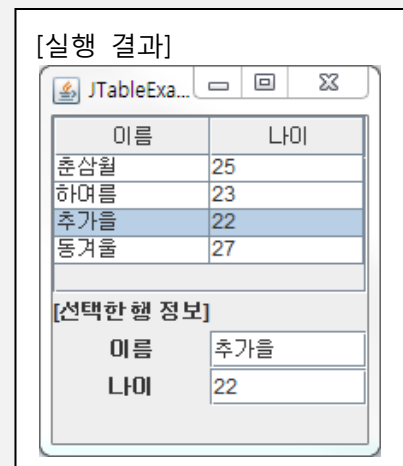
1 public class JTableExample extends JFrame {
2     private JTable jTable;
3     private JPanel pSouth;
4     private JTextField txtName;
5     private JTextField txtAge;
6     private Object[][] rowData;
7
8     public JTableExample() {
9         this.setTitle("JTableExample");
10        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
11        this.getContentPane().add(new JScrollPane(getJTable()), BorderLayout.CENTER);
12        this.getContentPane().add(getPSouth(), BorderLayout.SOUTH);
13        this.setSize(200, 230);
14    }
15
16    public JTable getJTable() {
17        if(jTable == null) {
18            String[] columnNames = new String[] {"이름", "나이" };
19            rowData = new Object[][] {
20                {"춘삼월", new Integer(25)},
21                {"하여름", new Integer(23)},
22                {"추가울", new Integer(22)},
23                {"동겨울", new Integer(27)}

```

```

24     };
25     jTable = new.JTable(rowData, columnNames);
26     jTable.addMouseListener(new MouseAdapter() {
27         public void mouseClicked(MouseEvent e) {
28             int rowIndex = jTable.getSelectedRow();
29             if(rowIndex != -1) {
30                 TableModel tableModel = jTable.getModel();
31                 String name = (String) tableModel.getValueAt(rowIndex, 0);
32                 Integer age = (Integer) tableModel.getValueAt(rowIndex, 1);
33
34                 getTxtName().setText(name);
35                 getTxtAge().setText(String.valueOf(age.intValue()));
36             }
37         }
38     });
39 }
40
41 return jTable;
42
43 }
44
45 public JPanel getPSouth() {
46     if(pSouth == null) {
47         pSouth = new JPanel();
48         pSouth.setLayout(new GridLayout(4,2));
49         pSouth.add(new JLabel("[선택한 행 정보]"));
50         pSouth.add(new JLabel(""));
51         pSouth.add(new JLabel("이름", JLabel.CENTER));
52         pSouth.add(getTxtName());
53         pSouth.add(new JLabel("나이", JLabel.CENTER));
54         pSouth.add(getTxtAge());
55     }
56     return pSouth;
57 }
58
59 public JTextField getTxtName() {
60     if(txtName == null) {
61         txtName = new JTextField();
62     }
63     return txtName;
64 }

```



```

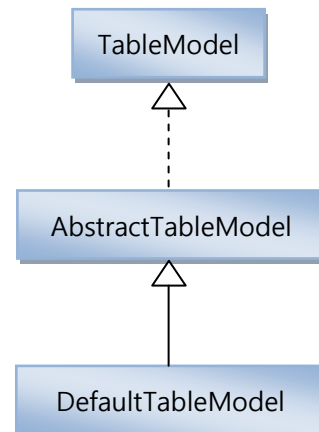
64 public JTextField getTxtAge() {
65     if(txtAge == null) {
66         txtAge = new JTextField();
67     }
68     return txtAge;
69 }
70
71 public static void main(String[] args) {
72     SwingUtilities.invokeLater(new Runnable() {
73         public void run() {
74             JTableExample jFrame = new JTableExample();
75             jFrame.setVisible(true);
76         }
77     });
78 }
79 }
80

```

## 9.5 행 추가, 수정, 삭제

TableModel 타입은 JTable 이 어떤 생성자로 만들어졌는지에 따라 다르다. JTable(Object[][] rowData, String[] columnNames) 생성자로 만들어졌을 경우 AbstractTableModel 을 상속해서 만들어 지고, 매개 변수가 없는 JTable() 생성자로 만들어졌다면 내부적으로 DefaultTableModel 을 상속해서 만들어 진다.

AbstractTableModel 과 DefaultTableModel 의 차이점은 DefaultTableModel 은 행을 추가, 삽입, 삭제할 수 있는 다음 메소드들이 추가적으로 제공된다는 점이다.



	메소드	용도
void	addRow(Object[] rowData ) addRow(Vector rowData)	맨 뒤에 행 추가
void	insertRow(int rowIndex, Object[] rowData )	주어진 인덱스에 행 삽입



	insertRow(int rowIndex, Vector rowData)	
void	removeRow(int rowIndex)	주어진 인덱스의 행 삭제

이들 메소드를 이용하려면 JTable 을 반드시 매개 변수가 없는 생성자로 만들고, getModel() 메소드의 리턴 객체를 DefaultTableModel 로 타입 변환해야 한다.

```
JTable jTable = new JTable();
DefaultTableModel tableModel = (DefaultTableModel) jTable.getModel();
//맨 뒤에 행 추가
tableModel.addRow(rowData);
//0 인덱스에 행 삽입
tableModel.insertRow(0, rowData);
//2 인덱스의 행 삭제
tableModel.removeRow(2);
```

DefaultTableModel 에는 행을 수정하는 메소드는 없는데, 행을 수정하려면 직접 행의 데이터를 얻어 변경해야 한다. 행을 추가 및 삽입할 때 데이터를 Object[] 배열로 주더라도 내부적으로 Vector 로 생성되어 관리된다. 각 행은 Vector 로 생성되고, 이 Vector 는 다시 전체 행을 관리하는 Vector 에 저장된다.

```
Vector(전체행 관리)
|-- Vector(0 인덱스 행 데이터)
|-- Vector(1 인덱스 행 데이터)
|-- ...
```

각 행의 데이터에 접근하기 위해서는 우선 DefaultTableModel 의 getDataVector() 메소드로 전체 행을 관리하는 Vector 를 얻어내야 한다.

```
Vector<Vector> rows = tableModel.getDataVector();
```

그리고 나서 원하는 인덱스의 행 Vector 를 다음과 같이 얻으면 된다.

```
Vector row = rows.elementAt(jTable.getSelectedRow());
```

행 Vector 의 내부 요소를 변경하기 위해서는 Vector 의 set() 메소드를 이용하면 되는데, 첫번째 매개값은 컬럼의 인덱스 번호이고, 두번째 매개값은 컬럼의 값이다.

```
row.set(0, changeValue);    //0 번 컬럼의 값을 changeValue 로 수정
row.set(1, changeVlaue);    //1 번 컬럼의 값을 changeVallue 로 수정
```

행의 데이터를 직접 Vector 를 이용해서 수정하였다면 수정된 내용을 JTable 에 반영하기 위해 DefaultTableModel 의 fireTableDataChanged() 메소드를 호출해야 한다.

```
tableModel.fireTableDataChanged();
```

#### 【JTableExample.java】 행 추가, 수정, 삭제

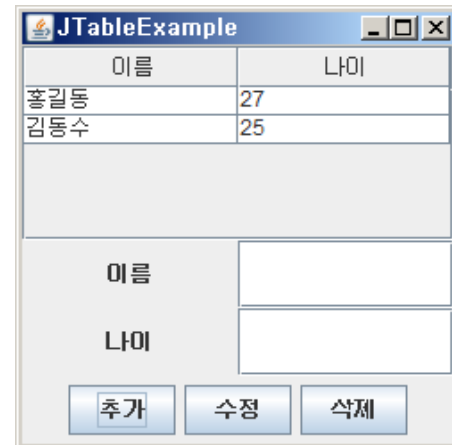
```
1 public class JTableExample extends JFrame {
2     private JTable jTable;
3     private JPanel pSouth;
4     private JTextField txtName, txtAge;
5     private JButton btnInsert, btnUpdate, btnDelete;
6
7     public JTableExample() {
8         this.setTitle("JTableExample");
9         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10        this.getContentPane().add(new JScrollPane(getJTable()), BorderLayout.CENTER);
11        this.getContentPane().add(getPSouth(), BorderLayout.SOUTH);
12        this.setSize(250, 250);
13    }
14
15    public JTable getJTable() {
16        if(jTable == null) {
17            jTable = new JTable();
18            final DefaultTableModel tableModel = (DefaultTableModel) jTable.getModel();
19            tableModel.addColumn("이름");
20            tableModel.addColumn("나이");
21
22            jTable.addMouseListener(new MouseAdapter() {
23                public void mouseClicked(MouseEvent e) {
24                    int rowIndex = jTable.getSelectedRow();
25                    if(rowIndex != -1) {
26                        String name = (String) tableModel.getValueAt(rowIndex, 0);
27                        String age = (String) tableModel.getValueAt(rowIndex, 1);
28                        txtName.setText(name);
29                        txtAge.setText(age.toString());
30                    }
31                }
32            });
33        }
34    }
```

```

35     return jTable;
36 }
37
38 public JPanel getPSouth() {
39     if(pSouth == null) {
40         pSouth = new JPanel();
41
42         pSouth.setLayout(new GridLayout(3,1));
43
44         JPanel pNameInput = new JPanel();
45         pNameInput.setLayout(new GridLayout(1,2));
46         pNameInput.add(new JLabel("이름", JLabel.CENTER));
47         pNameInput.add(getTxtName());
48         pSouth.add(pNameInput);
49
50         JPanel pAgeInput = new JPanel();
51         pAgeInput.setLayout(new GridLayout(1,2));
52         pAgeInput.add(new JLabel("나이", JLabel.CENTER));
53         pAgeInput.add(getTxtAge());
54         pSouth.add(pAgeInput);
55
56         JPanel pButton = new JPanel();
57         pButton.add(getBtnInsert());
58         pButton.add(getBtnUpdate());
59         pButton.add(getBtnDelete());
60         pSouth.add(pButton);
61     }
62     return pSouth;
63 }
64
65 public JTextField getTxtName() {
66     if(txtName == null) {
67         txtName = new JTextField();
68     }
69     return txtName;
70 }
71
72 public JTextField getTxtAge() {
73     if(txtAge == null) {
74         txtAge = new JTextField();

```

[실행 결과]



이름	나이
홍길동	27
김동수	25

이름:

나이:

```

75     }
76     return txtAge;
77 }
78
79 public JButton getBtnInsert() {
80     if(btnInsert == null) {
81         btnInsert = new JButton();
82         btnInsert.setText("추가");
83         btnInsert.addActionListener(new ActionListener() {
84             public void actionPerformed(ActionEvent e) {
85                 Object[] rowData = {
86                     txtName.getText(),
87                     txtAge.getText()
88                 };
89                 DefaultTableModel tableModel = (DefaultTableModel) getJTable().getModel();
90                 tableModel.addRow(rowData);
91                 txtName.setText("");
92                 txtAge.setText("");
93             }
94         });
95     }
96     return btnInsert;
97 }
98
99 public JButton getBtnUpdate() {
100     if(btnUpdate == null) {
101         btnUpdate = new JButton();
102         btnUpdate.setText("수정");
103         btnUpdate.addActionListener(new ActionListener() {
104             public void actionPerformed(ActionEvent e) {
105                 DefaultTableModel tableModel = (DefaultTableModel) getJTable().getModel();
106                 Vector<Vector> rows = tableModel.getDataVector();
107                 Vector row = rows.elementAt(jTable.getSelectedRow());
108                 row.set(0, txtName.getText());
109                 row.set(1, txtAge.getText());
110                 tableModel.fireTableDataChanged();
111                 txtName.setText("");
112                 txtAge.setText("");
113             }
114         });

```

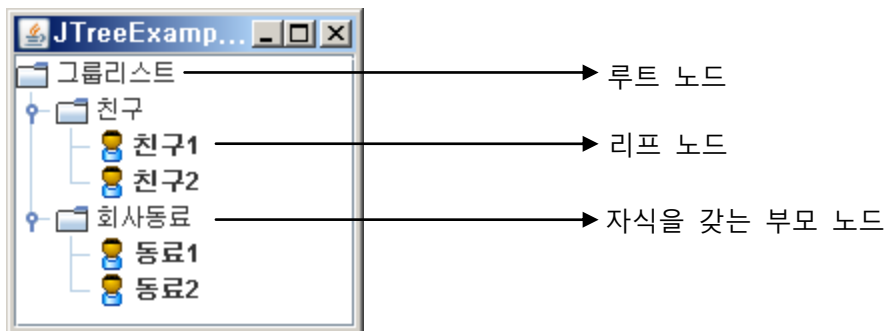
```

115     }
116     return btnUpdate;
117 }
118
119 public JButton getBtnDelete() {
120     if(btnDelete == null) {
121         btnDelete = new JButton();
122         btnDelete.setText("삭제");
123         btnDelete.addActionListener(new ActionListener() {
124             public void actionPerformed(ActionEvent e) {
125                 int rowIndex = getJTable().getSelectedRow();
126                 if(rowIndex != -1) {
127                     DefaultTableModel tableModel =
128                         (DefaultTableModel) getJTable().getModel();
129                     tableModel.removeRow(rowIndex);
130                     txtName.setText("");
131                     txtAge.setText("");
132                 }
133             }
134         });
135     }
136     return btnDelete;
137 }
138
139 public static void main(String[] args) {
140     SwingUtilities.invokeLater(new Runnable() {
141         public void run() {
142             JTableExample jFrame = new JTableExample();
143             jFrame.setVisible(true);
144         }
145     });
146 }
147

```

## 11 절.트리 컴포넌트

트리 컴포넌트는 계층적인 데이터를 표시하는 최적의 컴포넌트이다. 자바에서는 트리 컴포넌트로 JTree 를 제공하고 있다. JTree 는 하나의 루트 노드(root node) 아래에 여러 개의 자식 노드(child node)를 가진다. 자식 노드는 또 다시 자식 노드를 가질 수 있고, 자식 노드를 가지지 않는 노드를 리프(leaf) 노드라고 부른다. 그리고 동일한 부모 노드(parent node)를 갖는 노드들을 묶어서 형제 노드(sibling node)라고 부른다.



### 10.1 트리 생성

트리를 생성하려면 JTree 의 인스턴스를 생성하면 된다. 생성자의 매개값으로 루트 노드를 대입해야 하는데, 루트 노드는 DefaultMutableTreeNode 로 생성하면 된다. DefaultMutableTreeNode 는 루트 노드 뿐만 아니라 부모 노드, 리프 노드를 생성하는데 사용된다. 다음은 각 노드 인스턴스가 DefaultMutableTreeNode 임을 보여주고 JTree 를 생성할 때 루트 노드인 DefaultMutableTreeNode 를 생성자 매개값으로 제공해야됨을 보여준다.

```
DefaultMutableTreeNode
|-- DefaultMutableTreeNode
|-- DefaultMutableTreeNode
|-- DefaultMutableTreeNode
|-- DefaultMutableTreeNode

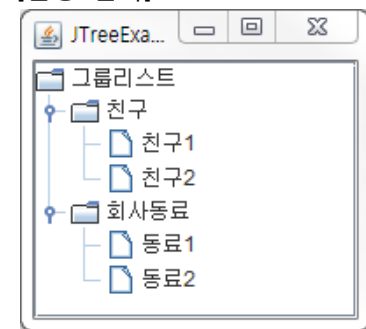
JTree jTree = new JTree( );
```

노드의 표현 방법은 DefaultTreeCellRenderer 가 결정한다. DefaultTreeCellRenderer 는 JLabel 의 하위 클래스로서 노드를 아이콘과 텍스트로 조합해서 보여준다. 기본적으로 자식을 갖는 루트 노드나 부모 노드는 폴더 모양의 아이콘으로 표현하고, 리프 노드는 문서 모양의 아이콘으로 표현한다.

### [JTreeExample1.java] 기본 트리

```
1 public class JTreeExample extends JFrame {
2     private JTree jTree;
3
4     public JTreeExample() {
5         this.setTitle("JTreeExample");
6         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7         this.getContentPane().add(new JScrollPane(getJTree()), BorderLayout.CENTER);
8         this.setSize(200, 150);
9     }
10
11    public JTree getJTree() {
12        if(jTree == null) {
13            DefaultMutableTreeNode root = new DefaultMutableTreeNode("그룹리스트");
14
15            DefaultMutableTreeNode node1 = new DefaultMutableTreeNode("친구");
16            node1.add(new DefaultMutableTreeNode("친구1"));
17            node1.add(new DefaultMutableTreeNode("친구2"));
18            root.add(node1);
19
20            DefaultMutableTreeNode node2 = new DefaultMutableTreeNode("회사동료");
21            node2.add(new DefaultMutableTreeNode("동료1"));
22            node2.add(new DefaultMutableTreeNode("동료2"));
23            root.add(node2);
24
25            jTree = new JTree(root);
26        }
27        return jTree;
28    }
29
30    public static void main(String[] args) {
31        SwingUtilities.invokeLater(new Runnable() {
32            public void run() {
33                JTreeExample jFrame = new JTreeExample();
34                jFrame.setVisible(true);
35            }
36        });
37    }
38 }
```

[실행 결과]



부모 노드(DefaultMutableTreeNode)는 자식 노드(DefaultMutableTreeNode)를 add() 메소드로 추가한다. 그룹 리스트 노드는 친구와 회사동료 노드를 add() 메소드로 추가했고, 친구 노드는 친구 1 과 친구 2 노드를 add() 메소드로 추가했다.

## 10.2 노드 표현 방법 변경

노드의 아이콘과 리프 노드의 표현 방법을 프로그램의 성격에 맞게 다르게 변형하고 싶다면 새로운 TreeCellRenderer 를 만들어 기본 렌더러인 DefaultTreeCellRenderer 를 대체하면 된다. 새로운 TreeCellRenderer 를 만드는 방법은 TreeCellRenderer 의 구현 클래스를 정의하고 getTreeCellRendererComponent() 메소드를 재정의하면 된다.

```
public class MyTreeCellRenderer implements TreeCellRenderer {
    public Component getTreeCellRendererComponent(...) {
        //컴포넌트를 초기화하고 리턴하는 코드
    }
}
```

getTreeCellRendererComponent() 메소드는 주어진 매개값을 이용해서 노드에 표현할 컴포넌트를 초기화하고 리턴하는 역할을 한다. getTreeCellRendererComponent() 메소드의 매개변수는 총 6 개로 다음과 같은 값들을 가지고 있다.

매개변수	값 또는 참조
tree	노드를 포함하고 있는 JTree 참조
value	노드인 DefaultMutableTreeNode 객체
sel	노드가 선택되었는지 여부
expanded	부모 노드가 펼쳐졌는지 여부
leaf	리프 노드인지 여부
hasFocus	포커스를 가지고 있는지 여부

새로운 TreeCellRenderer 렌더러가 정의되었다면 JTree 의 setCellRenderer() 메소드로 기본 렌더러를 변경하면 된다.

```
jTree.setCellRenderer(new MyTreeCellRenderer());
```



다음 예제는 부모 노드의 아이콘을 변경하고 리프 노드의 내용으로 아이콘+글자+아이콘으로 표현한다.

#### 【JTreeExample2.java】 노드의 다양한 표현

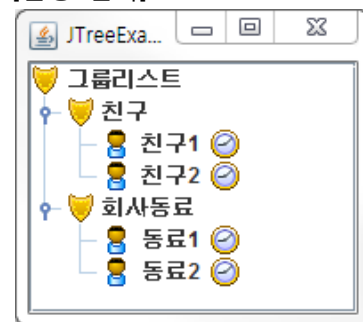
```
1 public class JTreeExample extends JFrame {
2     private JTree jTree;
3
4     public JTreeExample() {
5         this.setTitle("JTreeExample");
6         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7         this.getContentPane().add(new JScrollPane(getJTree()), BorderLayout.CENTER);
8         this.setSize(200, 150);
9     }
10
11    public JTree getJTree() {
12        if(jTree == null) {
13            DefaultMutableTreeNode root = new DefaultMutableTreeNode("그룹리스트");
14
15            DefaultMutableTreeNode node1 = new DefaultMutableTreeNode("친구");
16            node1.add(new DefaultMutableTreeNode("친구1"));
17            node1.add(new DefaultMutableTreeNode("친구2"));
18            root.add(node1);
19
20            DefaultMutableTreeNode node2 = new DefaultMutableTreeNode("회사동료");
21            node2.add(new DefaultMutableTreeNode("동료1"));
22            node2.add(new DefaultMutableTreeNode("동료2"));
23            root.add(node2);
24
25            jTree = new JTree(root);
26            jTree.setCellRenderer(new MyTreeCellRenderer());
27        }
28        return jTree;
29    }
30
31    public class MyTreeCellRenderer implements TreeCellRenderer {
32        private JPanel jPanel;
33        private JLabel lblWest, lblCenter, lblEast;
34        public Component getTreeCellRendererComponent(JTree tree, Object value,
35            boolean sel, boolean expanded, boolean leaf, int row, boolean hasFocus) {
```

```

36     if(leaf) {
37         JPanel jPanel = new JPanel();                리프 노드의 컴포넌트를 JPanel 로 하고
38         jPanel.setBackground(Color.WHITE);           서쪽, 중앙, 동쪽에 각각 JLabel 을 추가한다.
39         jPanel.setLayout(new BorderLayout());          서쪽 JLabel 은 이미지,
40                                                         중앙 JLabel 은 글자,
41                                                         동쪽 JLabel 은 이미지를 넣는다.
42         lblWest = new JLabel(new ImageIcon(getClass().getResource("logon.gif")));
43         lblCenter = new JLabel(" " + value.toString() + " ");
44         lblEast = new JLabel(new ImageIcon(getClass().getResource("time.gif")));
45         jPanel.add(lblWest, BorderLayout.WEST);
46         jPanel.add(lblCenter, BorderLayout.CENTER);    리프 노드가 선택되면
47                                                         바탕색을 노란색으로
48                                                         설정한다.
49         if(sel) { jPanel.setBackground(Color.ORANGE); }
50     } else {
51         JLabel jLabel = new JLabel();
52         jLabel.setIcon(new ImageIcon(getClass().getResource("parentnode.gif")));
53         jLabel.setText(value.toString());
54         return jLabel;                                부모 노드의 컴포넌트를 JLabel 로 하고
55                                                         아이콘과 글자를 넣는다.
56     }
57 }
58
59 public static void main(String[] args) {
60     SwingUtilities.invokeLater(new Runnable() {
61         public void run() {
62             JTreeExample JFrame = new JTreeExample();
63             JFrame.setVisible(true);
64         }
65     });
66 }
67 }

```

[실행 결과]



## 10.1 선택 이벤트 처리

JTree 를 클릭했을 때 무언가를 하고 싶다면 JTree 에서 발생하는 TreeSelectionEvent 를 처리하면 된다. TreeSelectionListener 의 valueChanged() 메소드는 선택이 변경되었을 때에 호출되는데, 같은 노드를 2 회 이상 클릭했을 때에는 호출되지 않는다. 다음은 TreeSelectionListener 의 구현 클래스를 작성하는 방법을 보여준다.

```
public class MyTreeSelectionListener implements TreeSelectionListener {
    public void valueChanged(TreeSelectionEvent e) {
        //선택된 노드의 전체 경로 얻기
        TreePath treePath = e.getPath();
        //선택된 노드의 DefaultMutableTreeNode 얻기
        DefaultMutableTreeNode node =
            (DefaultMutableTreeNode) treePath.getLastPathComponent();
        //선택된 노드의 문자열 얻기
        String userObject = (String) node.getUserObject();
        //처리 코드
        //~
    }
}
```

TreeSelectionEvent 의 getPath() 메소드는 선택된 노드의 전체 경로 즉 루트 노드에서부터 선택된 노드까지의 정보를 가지고 있는 TreePath 를 리턴한다. 선택된 노드는 전체 경로의 마지막 노드이므로 TreePath 의 getLastPathComponent() 메소드로 얻을 수 있다. 이 때 리턴 타입이 Object 이므로 노드의 실제 타입인 DefaultMutableTreeNode 로 타입 변환을 해야한다. 노드의 문자열은 DefaultMutableTreeNode 의 getUserObject() 메소드로 얻을 수 있다. 만약 프로그램이 노드 선택 변경보다는 클릭과 더블 클릭에 의해서 무엇을 처리해야한다면 다음과 같이 MouseEvent 를 처리하는 것이 좋다.

```
public class MyMouseListener extends MouseAdapter {
    public void mousePressed(MouseEvent e) {
        //JTree 얻기
        JTree jTree = (JTree) e.getSource();
        //선택된 노드의 전체 경로 얻기
        TreePath treePath = jTree.getPathForLocation(e.getX(), e.getY());
        //선택된 노드가 있을 경우
        if(selRow != -1) {
            if(e.getClickCount() == 1) {
                //클릭했을 경우 실행할 코드
            } else if(e.getClickCount() == 2) {
            }
        }
    }
}
```

```

        //더블 클릭했을 경우 실행할 코드
    }
}
}
}

```

JTree 에는 클릭된 좌표를 가지고 선택된 노드의 전체 경로를 리턴하는 `getPathForLocation()` 메소드를 가지고 있다. 이 메소드를 호출하면 선택된 노드의 전체 경로 즉 루트 노드에서부터 선택된 노드까지의 정보를 가지고 있는 `TreePath` 를 얻을 수 있다. 마우스를 클릭했느냐 더블 클릭했느냐는 `MouseEvent` 의 `getClickCount()`의 리턴값으로 구분할 수 있으므로 if 문을 사용해서 실행할 코드를 구분한다.

#### 【JTreeExample.java】선택된 노드의 처리

```

1  public class JTreeExample extends JFrame {
2      private JTree jTree;
3
4      public JTreeExample() {
5          this.setTitle("JTreeExample");
6          this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7          this.getContentPane().add(new JScrollPane(getJTree()), BorderLayout.CENTER);
8          this.setSize(200, 150);
9      }
10
11     public JTree getJTree() {
12         if(jTree == null) {
13             DefaultMutableTreeNode root = new DefaultMutableTreeNode("그룹리스트");
14
15             DefaultMutableTreeNode node1 = new DefaultMutableTreeNode("친구");
16             node1.add(new DefaultMutableTreeNode("친구1"));
17             node1.add(new DefaultMutableTreeNode("친구2"));
18             root.add(node1);
19
20             jTree = new JTree(root);
21             jTree.setCellRenderer(new MyTreeCellRenderer());
22             jTree.addTreeSelectionListener(treeSelectionListener);
23             jTree.addMouseListener(mouseAdapter);
24
25         }
26         return jTree;
27     }

```

```

28
29 private TreeSelectionListener treeSelectionListener = new TreeSelectionListener() {
30     @Override
31     public void valueChanged(TreeSelectionEvent e) {
32         TreePath treePath = e.getPath();
33         DefaultMutableTreeNode treeNode =
34             (DefaultMutableTreeNode) treePath.getLastPathComponent();
35         String nodeText = (String) treeNode.getUserObject();
36         JOptionPane.showMessageDialog(JTreeExample.this, "노드 변경: " + nodeText);
37     }
38 };
39
40 private MouseAdapter mouseAdapter = new MouseAdapter() {
41     public void mouseClicked(java.awt.event.MouseEvent e) {
42         if(e.getClickCount() == 2) {                                더블 클릭 했을 때 실행
43             TreePath treePath = jTree.getPathForLocation(e.getX(), e.getY());
44             DefaultMutableTreeNode treeNode =
45                 (DefaultMutableTreeNode) treePath.getLastPathComponent();
46             String nodeText = (String) treeNode.getUserObject();
47             JOptionPane.showMessageDialog(JTreeExample.this, "더블 클릭: " + nodeText);
48         }
49     };
50 };
51
52 public class MyTreeCellRenderer implements TreeCellRenderer {
53     private JPanel jPanel;
54     private JLabel lblWest, lblCenter, lblEast;
55     public Component getTreeCellRendererComponent(JTree tree, Object value,
56         boolean sel, boolean expanded, boolean leaf, int row, boolean hasFocus) {
57         if(leaf) {
58             jPanel = new JPanel();
59             jPanel.setBackground(Color.WHITE);
60             jPanel.setLayout(new BorderLayout());
61
62             lblWest = new JLabel(new ImageIcon(getClass().getResource("logon.gif")));
63             lblCenter = new JLabel(" " + value.toString() + " ");
64             lblEast = new JLabel(new ImageIcon(getClass().getResource("time.gif")));
65             jPanel.add(lblWest, BorderLayout.WEST);
66             jPanel.add(lblCenter, BorderLayout.CENTER);
67             jPanel.add(lblEast, BorderLayout.EAST);

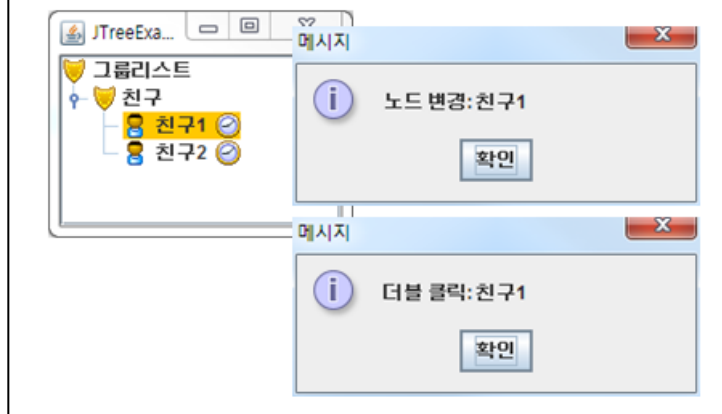
```

```

68
69         if(sel) { jPanel.setBackground(Color.ORANGE); }
70         return jPanel;
71     } else {
72         JLabel jLabel = new JLabel();
73         jLabel.setIcon(new ImageIcon(getClass().getResource("parentnode.gif")));
74         jLabel.setText(value.toString());
75         return jLabel;
76     }
77 }
78 }
79
80 public static void main(String[] args) {
81     SwingUtilities.invokeLater(new Runnable() {
82         public void run() {
83             JTreeExample JFrame = new JTreeExample();
84             JFrame.setVisible(true);
85         }
86     });
87 }
88 }

```

[실행 결과]

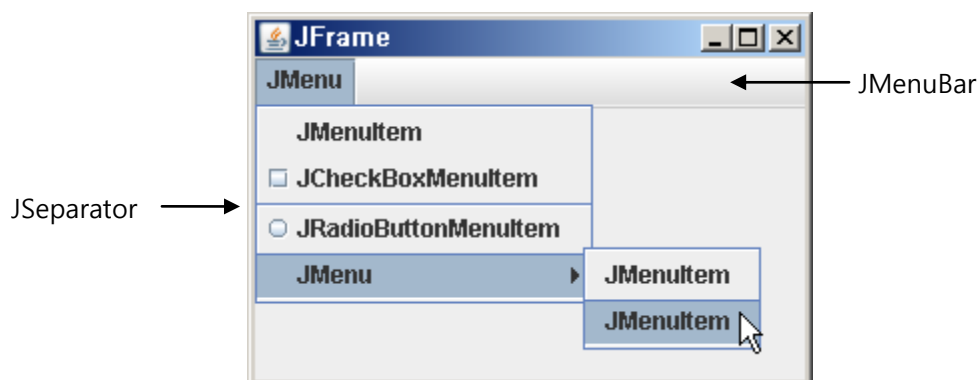


## 12 절. 메뉴 컴포넌트

GUI 프로그램에서 메뉴는 빠질 수 없는 구성 요소이다. 자바는 메뉴 생성을 위해서 다음과 같은 컴포넌트를 javax.swing 패키지에서 제공하고 있다.

컴포넌트	설명
JMenuBar	메뉴바 컴포넌트
JMenu	주 메뉴 및 자식 메뉴 아이템을 갖는 서브 메뉴 컴포넌트
JPopupMenu	팝업 메뉴 컴포넌트
JMenuItem	메뉴 아이템 컴포넌트
JCheckBoxMenuItem	JCheckBox 로 선택할 수 있는 메뉴 아이템 컴포넌트
JRadioButtonMenuItem	JRadioButton 으로 선택할 수 있는 메뉴 아이템 컴포넌트
JSeparator	메뉴를 수직 또는 수평으로 분리시키는 컴포넌트

각 컴포넌트의 사용 위치를 그림으로 표시하면 다음과 같다.



### 11.1 메뉴 생성

메뉴를 생성하기 위해서는 제일 먼저 JMenuBar 를 생성하고 윈도우 컨테이너 상단에 배치해야 한다. JMenuBar 를 배치할 수 있는 윈도우 컨테이너는 JFrame, JDialog, JApplet, JInternalFrame 등이 있다. 이들 컨테이너는 JMenuBar 를 상단에 배치하기 위해 setMenuBar() 메소드를 제공하고 있다.

```
JMenuBar jMenuBar = new JMenuBar();
jFrame.setJMenuBar(jMenuBar);
```

JMenuBar 는 윈도우 상단에 메뉴를 수평으로 배치하는 역할만 하므로, 만약 메뉴가 한 개라도 포함되어 있지 않으면 윈도우 상단에 보이지 않는다. JMenu 로 생성된 메뉴를 JMenuBar 에 다음과 같이 추가해야만 윈도우 상단에 나타난다.

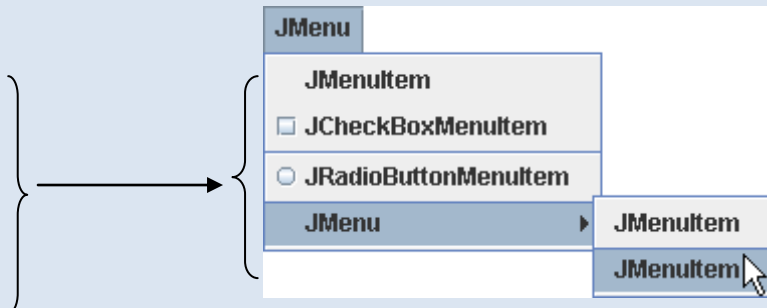
```
JMenu jMenuItem = new JMenu("메뉴명");
jMenuBar.add(jMenuItem);
```

JMenuBar 에 추가된 JMenu 는 마우스로 클릭했을 때 보여줄 메뉴 아이템과 서브 메뉴를 가져야 한다. 메뉴 아이템은 JMenuItem, JCheckBoxMenuItem, JRadioButtonMenuItem 들이고, 서브 메뉴는 JMenu 를 말한다. 이들 메뉴 아이템들은 JMenu 의 add() 메소드에 의해 추가된다.

```
//메뉴 아이템 생성
JMenuItem menuItem1 = new JMenuItem("JMenuItem");
JMenuItem menuItem2 = new JCheckBoxMenuItem("JCheckBoxMenuItem");
JMenuItem menuItem3 = new JRadioButtonMenuItem("JRadioButtonMenuItem");
```

```
//서브 메뉴 생성
JMenu subMenu = new JMenu("JMenu");
JMenuItem subMenuItem1 = new JMenuItem("JMenuItem");
JMenuItem subMenuItem2 = new JMenuItem("JMenuItem");
jMenuSub.add(subMenuItem1);
jMenuSub.add(subMenuItem2);
```

```
//메뉴 아이템과 서브 메뉴 추가
jMenu.add(menuItem1);
jMenu.add(menuItem2);
jMenu.add(new JSeparator());
jMenu.add(menuItem3);
jMenu.add(subMenu);
```



메뉴 앞에 아이콘을 넣고 싶다면 JMenuItem, JCheckBoxMenuItem, JRadioButtonMenuItem 생성자에 ImageIcon 객체를 다음과 같이 추가하면 된다.

```
JMenuItem mi = new JMenuItem("메뉴명", new ImageIcon( getClass().getResource("파일명")));
```

#### 【JMenuExample1.java】메뉴 생성

```
1 public class JMenuExample extends JFrame {
2     private JMenuBar jMenuBar;
3     private JMenu menuFile, menuNew, menuHelp;
4     private JMenuItem menuItemNewFile, menuItemNewFolder;
5     private JMenuItem menuItemOpen, menuItemSave, menuItemExit;
6
7     public JMenuExample() {
```

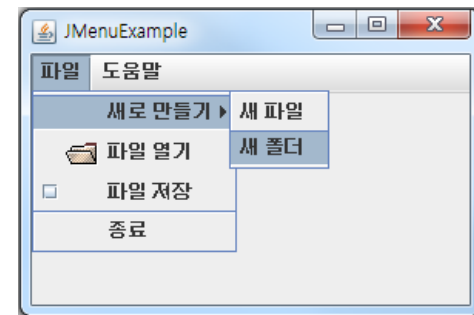


```

8      this.setTitle("JMenuExample");
9      this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
10     this.setJMenuBar(getJMenuBar());
11     this.setSize(300, 200);
12 }
13
14 public JMenuBar getJMenuBar() {
15     if(jMenuBar == null) {
16         jMenuBar = new JMenuBar();
17         jMenuBar.add(getMenuFile());
18         jMenuBar.add(getMenuHelp());
19     }
20     return jMenuBar;
21 }
22
23 public JMenu getMenuFile() {
24     if(menuFile == null) {
25         menuFile = new JMenu("파일");
26         menuFile.add(getMenuNew());
27         menuFile.add(getMenuItemOpen());
28         menuFile.add(getMenuItemSave());
29         menuFile.add(new JSeparator());
30         menuFile.add(getMenuItemExit());
31     }
32     return menuFile;
33 }
34
35 public JMenu getMenuHelp() {
36     if(menuHelp == null) {
37         menuHelp = new JMenu("도움말");
38     }
39     return menuHelp;
40 }
41
42 public JMenu getMenuNew() {
43     if(menuNew == null) {
44         menuNew = new JMenu("새로 만들기");
45         menuNew.add(getMenuItemNewFile());
46         menuNew.add(getMenuItemNewFolder());
47     }

```

[실행 결과]



```

48     return menuNew;
49 }
50
51 public JMenuItem getMenuNewItemNewFile() {
52     if(menuItemNewFile == null) {
53         menuItemNewFile = new JMenuItem("새 파일");
54     }
55     return menuItemNewFile;
56 }
57
58 public JMenuItem getMenuNewItemNewFolder() {
59     if(menuItemNewFolder == null) {
60         menuItemNewFolder = new JMenuItem("새 폴더");
61     }
62     return menuItemNewFolder;
63 }
64
65 public JMenuItem getMenuNewItemOpen() {
66     if(menuItemOpen == null) {
67         menuItemOpen = new JMenuItem("파일 열기",
68                                     new ImageIcon(getClass().getResource("open.gif")));
69     }
70     return menuItemOpen;
71 }
72
73 public JMenuItem getMenuNewItemSave() {
74     if(menuItemSave == null) {
75         menuItemSave = new JCheckBoxMenuItem("파일 저장");
76     }
77     return menuItemSave;
78 }
79
80 public JMenuItem getMenuNewItemExit() {
81     if(menuItemExit == null) {
82         menuItemExit = new JMenuItem("종료");
83     }
84     return menuItemExit;
85 }
86
87 public static void main(String[] args) {

```

```

88 SwingUtilities.invokeLater(new Runnable() {
89     public void run() {
90         JMenuExample jFrame = new JMenuExample();
91         jFrame.setVisible(true);
92     }
93 });
94 }
95 }

```

## 11.2 메뉴 이벤트 처리

사용자가 마우스로 메뉴 아이템을 선택하면 `ActionEvent` 가 발생하게 된다. 따라서 메뉴 아이템에 `ActionListener` 를 등록하여 이벤트를 처리할 수 있다. 메뉴 아이템 마다 별도의 `ActionListener` 를 추가하지 않고, 하나의 `ActionListener` 로 여러 메뉴 아이템의 `ActionEvent` 를 처리하는 것이 좋다. `ActionEvent` 의 `getSource()`는 마우스로 선택한 메뉴 아이템의 객체를 얻을 수 있고, `getActionCommand()`는 메뉴 아이템의 문자열을 얻을 수 있다.

```

public class MenuActionListener implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        //선택된 메뉴 아이템으로 구분
        JMenuItem selected = e.getSource();
        if(selected == menuItem1) { //처리 코드 }
        else if(selected == menuItem2) { //처리 코드 }

        //선택된 메뉴 아이템의 문자열로 구분
        String ac = e.getActionCommand();
        if(ac.equals("메뉴 문자열 1")) { //처리 코드 }
        else if(ac.equals("메뉴 문자열 2")) { //처리 코드 }
    }
}

```

`JCheckBoxMenuItem` 과 `JRadioButtonMenuItem` 의 경우 현재 선택되어 있는지, 아닌지에 따라 처리 코드 내용이 달라 질 수 있기 때문에 선택 여부를 알 필요가 있고, 선택 및 해제가 코드로 가능해야 한다. 두 컴포넌트 모두 `AbstractButton` 를 상속하기 때문에 `isSelected()` 와 `setSelected()` 메소드를 사용할 수 있다. `isSelected()` 메소드는 선택 여부를 알아 낼 때 사용할 수 있고, `setSelected()` 메소드는 선택 및 해제 설정할 때 사용할 수 있다. `JCheckBoxMenuItem` 는 추가적으로 `getState()`와 `setState()`를 제공하는데, 이들 메소드를 사용해도 좋다.

```

jCheckBoxMenuItem.addActionListener( new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        if(jCheckBoxMenuItem.isSelected()) {
            //선택 상태일 때 실행할 코드
        } else {
            //해제 상태일 때 실행할 코드
        }
    }
}
}

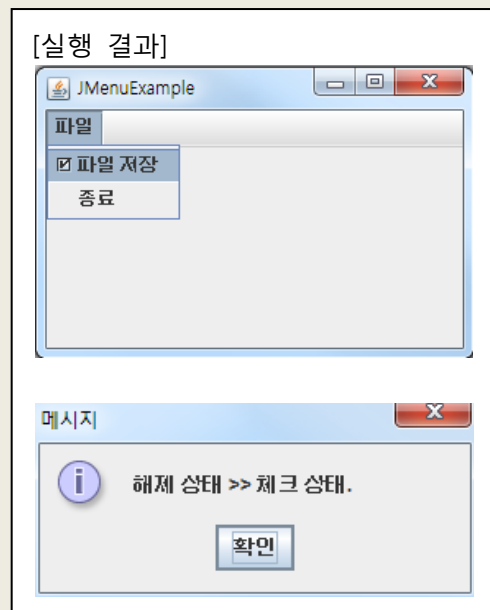
```

#### 【JMenuExample.java】 메뉴의 이벤트 처리

```

1  public class JMenuExample extends JFrame {
2      private JMenuBar jMenuBar;
3      private JMenu menuFile;
4      private JMenuItem menuItemSave, menuItemExit;
5
6      public JMenuExample() {
7          this.setTitle("JMenuExample");
8          this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9          this.setJMenuBar(getJMenuBar());
10         this.setSize(300, 200);
11     }
12
13     public JMenuBar getJMenuBar() {
14         if(jMenuBar == null) {
15             jMenuBar = new JMenuBar();
16             jMenuBar.add(getMenuFile());
17         }
18         return jMenuBar;
19     }
20
21     public JMenu getMenuFile() {
22         if(menuFile == null) {
23             menuFile = new JMenu("파일");
24             menuFile.add(getMenuItemSave());
25             menuFile.add(getMenuItemExit());
26         }
27         return menuFile;
28     }
29

```



```

30 public JMenuItem getMenuItemsSave() {
31     if(menuItemsSave == null) {
32         menuItemSave = new JCheckBoxMenuItem("파일 저장");
33         menuItemSave.addActionListener(actionListener);
34     }
35     return menuItemSave;
36 }
37
38 public JMenuItem getMenuItemsExit() {
39     if(menuItemsExit == null) {
40         menuItemExit = new JMenuItem("종료");
41         menuItemExit.addActionListener(actionListener);
42     }
43     return menuItemExit;
44 }
45
46 private ActionListener actionListener = new ActionListener() {
47     @Override
48     public void actionPerformed(ActionEvent e) {
49         if(e.getSource() == menuItemSave) {
50             if(menuItemSave.isSelected()) {                파일 저장 메뉴 아이템을 클릭했을 때
51                 JOptionPane.showMessageDialog(
52                     JMenuExample.this, "해제 상태 >> 체크 상태.");
53             } else {
54                 JOptionPane.showMessageDialog(
55                     JMenuExample.this, "체크 상태 >> 해제 상태.");
56             }
57         } else if(e.getSource() == menuItemExit) {
58             System.exit(0);                                종료 메뉴 아이템을 클릭했을 때
59         }
60     }
61 };
62
63 public static void main(String[] args) {
64     SwingUtilities.invokeLater(new Runnable() {
65         public void run() {
66             JMenuExample jFrame = new JMenuExample();
67             jFrame.setVisible(true);
68         }
69     });

```

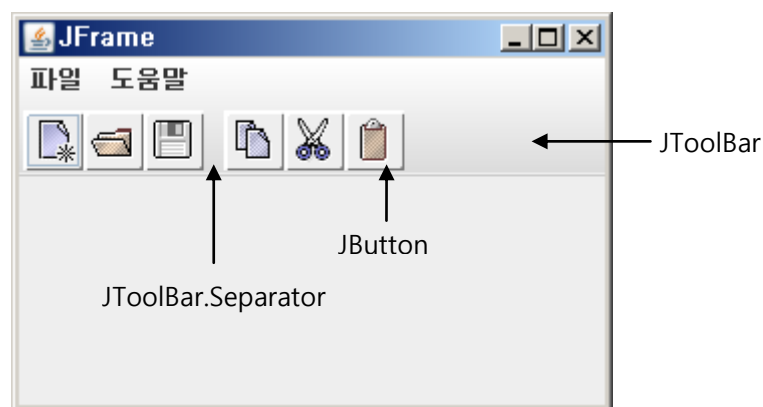
70	}
71	}

## 13 절. 툴바 컴포넌트

툴바는 메뉴바 아래에 위치하고 주로 버튼들이 배치되는 컨테이너다. 메뉴보다는 빠르게 원하는 기능을 마우스로 선택할 수 있기 때문에 사용자들은 메뉴 보다는 툴바를 더 선호한다. 그래서 대부분의 GUI 프로그램은 툴바를 제공한다. Swing 은 툴바 생성을 위해서 다음과 같은 컴포넌트를 제공하고 있다.

컴포넌트	설명
JToolBar	툴바 생성을 위한 컴포넌트
JToolBar.Separator	내부 컴포넌트를 시각적으로 그룹짓는 공백을 제공하는 컴포넌트

그림으로 각 컴포넌트의 사용 위치를 표시하면 다음과 같다.



JToolBar 를 상단에 배치하는 특별한 메소드는 없고, JToolBar 객체를 생성한 뒤, BorderLayout 을 갖는 컨테이너의 북쪽에 위치하도록 배치하면 된다.

```
JToolBar jToolBar = new JToolBar();
jFrame.getContentPane().add(jToolBar, BorderLayout.NORTH);
```

JToolBar 내부에 컴포넌트가 추가되지 않으면 JToolBar 가 북쪽에 위치하더라도 보이지 않는다. 최소한 한 개의 컴포넌트가 추가되어야 JToolBar 가 보이게 된다. JToolBar 의 내부 컴포넌트는 일반적으로 JButton 으로 구성되나 다른 컴포넌트도 추가 가능하다. 다음은 JButton 을 JToolBar 에 추가하는 코드이다.

```

JButton jButton = new JButton();
jButton.setIcon(new ImageIcon(getClass().getResource("image.gif"))); //아이콘
jButton.setBorder(new SoftBevelBorder(SoftBevelBorder.RAISED)); //돌출된 모양
jButton.setToolTipText("새로만들기"); //마우스를 올려 놓았을 때 나오는 풍선 도움말
jToolBar.add(jButton);

```

실행 상태에서 JToolBar 의 요철 부분을 마우스로 드래그해서 컨테이너의 동·서·남·북에 붙일 수 있다. 또한 JToolBar 를 중앙에 끌어서 놓으면 툴바 윈도우가 만들어진다. 만약 JToolBar 의 끌기 기능을 없애려면 setFloatable(false)를 호출해서 floatable 속성을 끄면 된다.

#### 【JToolBarExample.java】 툴바 만들기

```

1 public class JToolBarExample extends JFrame {
2     private JMenuBar jMenuBar;
3     private JToolBar jToolBar;
4     private JButton btnNew, btnSave, btnCopy, btnPaste;
5
6     public JToolBarExample() {
7         this.setTitle("JFrame");
8         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9         this.setJMenuBar(getJMenuBar());
10        this.getContentPane().add(getJToolBar(), BorderLayout.NORTH);
11        this.setSize(300, 200);
12    }
13
14    public JMenuBar getJMenuBar() {
15        if(jMenuBar == null) {
16            jMenuBar = new JMenuBar();
17            jMenuBar.add(new JMenu("파일"));
18            jMenuBar.add(new JMenu("도움말"));
19        }
20        return jMenuBar;

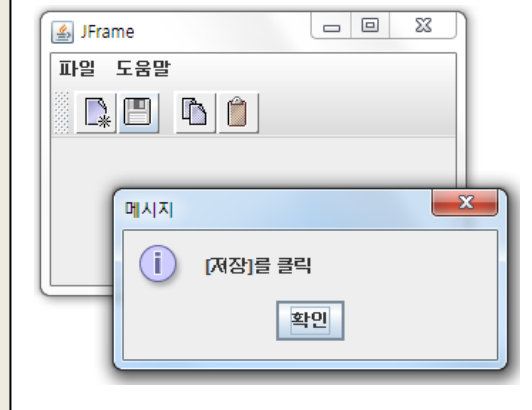
```

```

21 }
22
23 public JToolBar getJToolBar() {
24     if(jToolBar == null) {
25         jToolBar = new JToolBar();
26         jToolBar.add(getBtnNew());
27         jToolBar.add(getBtnSave());
28         jToolBar.addSeparator();
29         jToolBar.add(getBtnCopy());
30         jToolBar.add(getBtnPaste());
31     }
32     return jToolBar;
33 }
34
35 public JButton getBtnNew() {
36     if(btnNew == null) {
37         btnNew = new JButton();
38         btnNew.setIcon(new ImageIcon(getClass().getResource("new.gif")));
39         btnNew.setBorder(new SoftBevelBorder(SoftBevelBorder.RAISED));
40         btnNew.setToolTipText("새로만들기");
41         btnNew.addActionListener(actionListener);
42     }
43     return btnNew;
44 }
45
46 public JButton getBtnSave() {
47     if(btnSave == null) {
48         btnSave = new JButton();
49         btnSave.setIcon(new ImageIcon(getClass().getResource("save.gif")));
50         btnSave.setBorder(new SoftBevelBorder(SoftBevelBorder.RAISED));
51         btnSave.setToolTipText("저장");
52         btnSave.addActionListener(actionListener);
53     }
54     return btnSave;
55 }
56
57 public JButton getBtnCopy() {
58     if(btnCopy == null) {
59         btnCopy = new JButton();
60         btnCopy.setIcon(new ImageIcon(getClass().getResource("copy.gif")));

```

[실행 결과]





```

61     btnCopy.setBorder(new SoftBevelBorder(SoftBevelBorder.RAISED));
62     btnCopy.setToolTipText("복사");
63     btnCopy.addActionListener(actionListener);
64 }
65 return btnCopy;
66 }
67
68 public JButton getBtnPaste() {
69     if(btnPaste == null) {
70         btnPaste = new JButton();
71         btnPaste.setIcon(new ImageIcon(getClass().getResource("paste.gif")));
72         btnPaste.setBorder(new SoftBevelBorder(SoftBevelBorder.RAISED));
73         btnPaste.setToolTipText("붙여넣기");
74         btnPaste.addActionListener(actionListener);
75     }
76     return btnPaste;
77 }
78
79 private ActionListener actionListener = new ActionListener() {
80     @Override
81     public void actionPerformed(ActionEvent e) {
82         if(e.getSource() == btnNew) {
83             JOptionPane.showMessageDialog(JToolBarExample.this, "[새로만들기]를 클릭");
84         } else if(e.getSource() == btnSave) {
85             JOptionPane.showMessageDialog(JToolBarExample.this, "[저장]를 클릭");
86         } else if(e.getSource() == btnCopy) {
87             JOptionPane.showMessageDialog(JToolBarExample.this, "[복사]를 클릭");
88         } else if(e.getSource() == btnPaste) {
89             JOptionPane.showMessageDialog(JToolBarExample.this, "[붙여넣기]를 클릭");
90         }
91     }
92 };
93
94 public static void main(String[] args) {
95     SwingUtilities.invokeLater(new Runnable() {
96         public void run() {
97             JToolBarExample jFrame = new JToolBarExample();
98             jFrame.setVisible(true);
99         }
100     });

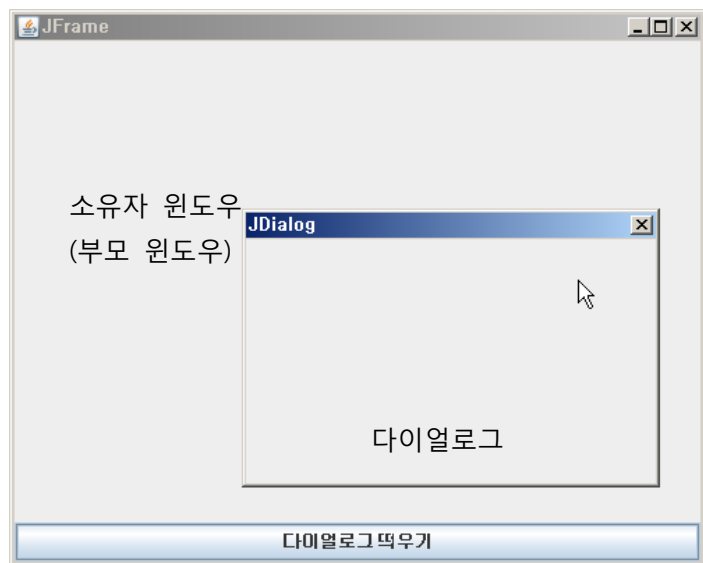
```

101	}
102	}

## 14 절. 다이얼로그

다이얼로그(Dialog)는 주 윈도우에서 사용자의 선택 또는 입력을 위해서 내부적으로 띄우는 서브 윈도우라고 볼 수 있다. 다이얼로그는 자체적으로 실행될 수 없고, 주 윈도우가 실행되고 나서 필요에 의해서 띄운다. 이 때 다이얼로그를 띄우는 주 윈도우를 다이얼로그의 부모(parent) 윈도우 또는 다이얼로그의 소유자(owner) 윈도우라고 한다. 다이얼로그는 모달(modal)과 모달리스(modalless) 두가지 종류가 있다. 모달 다이얼로그는 다이얼로그를 닫기 전까지 부모 윈도우를 사용할 수 없는 다이얼로그를 말한다. 모달리스 다이얼로그는 부모 윈도우를 계속 사용할 수 있는 다이얼로그를 말한다. 다이얼로그의 기본은 모달이다.

자바는 다이얼로그를 생성하기 위해서 JDialog 를 제공한다. 사용자 정의 다이얼로그는 이 JDialog 를 상속해서 만들어 진다. 자바는 프로그램에서 자주 사용되는 다이얼로그를 쉽게 생성할 수 있도록 여러가지 클래스를 제공하고 있는데, 파일 선택을 위한 JFileChooser, 색상 선택을 위한 JColorChooser, 메시지·확인·간단한 입력을 위한 JOptionPane 이 있다. 이들 다이얼로그를 띄울 때는 소유자 윈도우가 반드시 필요하다. JDialog 의 소유자 윈도우는 JWindow, JFrame, JApplet 등이 될 수 있고, JDialog 에서 또다른 JDialog 를 생성할 수 있기 때문에 JDialog 도 소유자 윈도우가 될 수 있다.



### 13.1 사용자 정의 다이얼로그

사용자 다이얼로그를 만들기 위해서는 다음과 같이 JDialog 를 상속해야 한다. JDialog 는 소유자 윈도우가 반드시 있어야 하므로 생성자는 소유자 윈도우를 매개변수로 가져야 한다. 소유자 윈도우가 될 수 있는 것은 JWindow, JFrame, JApplet, JDialog 이다. 생성자에서 작성해야할

첫번째 코드는 매개변수로 받은 소유자 윈도우를 상위 JDialog 생성자에게 넘겨주기 위해 super(owner)를 호출하는 것이다.

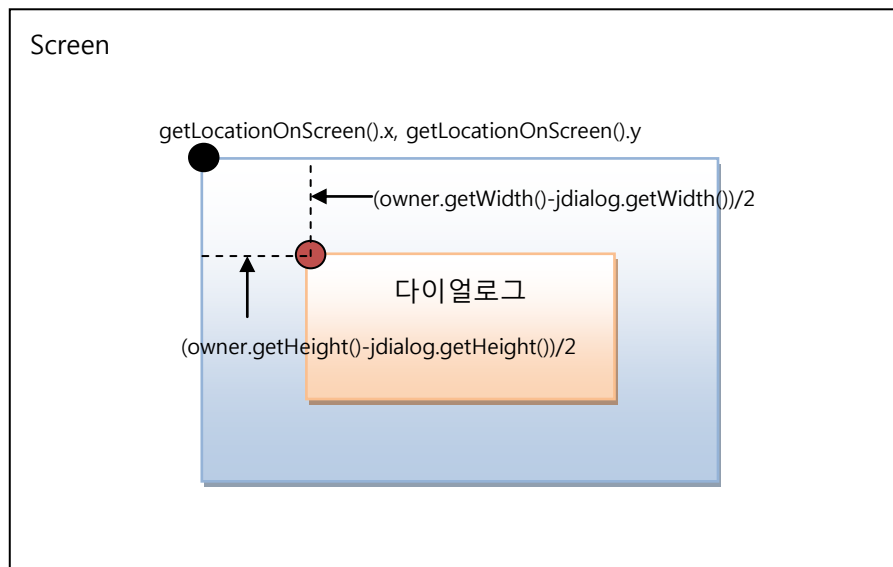
```
public class MyDialog extends JDialog {
    public MyDialog(JFrame owner) {
        super(owner);
        setTitle("Title");
        setSize(width, height);
        setModal(true);
        setResizable(false);
        setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);
        setLocation(
            owner.getLocationOnScreen().x + (owner.getWidth()-getWidth())/2,
            owner.getLocationOnScreen().y + (owner.getHeight()-getHeight())/2
        );
    }
}
```

setTitle()과 setSize()는 JFrame 과 마찬가지로 제목과 폭과 높이를 설정한다. setModal() 은 모달 또는 모달리스로 설정하고, setResizable()은 사용자에게 의해서 크기가 변경될 수 있는지 여부를 설정한다. JDialog 는 JFrame 과 마찬가지로 종료 버튼을 제공하는데 종료 버튼의 기본 기능은 단순히 JDialog 를 화면에서 숨기는 역할만 한다. 만약 JDialog 를 완전히 제거하고 싶다면 setDefaultCloseOperation(JDialog.DISPOSE\_ON\_CLOSE) 를 호출하면 된다. 종료 버튼에 적용할 수 있는 기능은 다음 세가지이다.

기능별 상수	설명
WindowConstants.DO_NOTHING_ON_CLOSE	아무 것도 하지 않음
WindowConstants.HIDE_ON_CLOSE	화면에서 JDialog 숨김(기본)
WindowConstants.DISPOSE_ON_CLOSE	화면에서 JDialog 완전히 제거

다이얼로그는 소유자 윈도우 중앙에서 띄워지는 것이 보기 좋기 때문에 소유자 윈도우 좌상단 좌표를 이용해서 다음과 같이 다이얼로그 좌상단 좌표를 구할 수 있다.

```
int left = owner.getLocationOnScreen().x + (owner.getWidth()-dialog.getWidth())/2;
int top = owner.getLocationOnScreen().y + (owner.getHeight()-dialog.getHeight())/2;
```



JDialog 을 화면에 띄우려면 다음과 같이 JDialog 객체 생성을 하고, setVisible(true) 메소드를 호출하면 된다. 주의할 점은 생성자 호출시 소유자 윈도우를 반드시 제공해야 한다.

```
MyDialog jDialog = new MyDialog(jFrame);
jDialog.setVisible(true);
```

JDialog 는 기본적으로 BorderLayout 배치 관리자로 설정되어 있고 언제든지 배치 관리자를 변경할 수 있다. 내부 컴포넌트는 JFrame 과 동일하게 getContentPane()을 얻어 배치해야 한다.

#### 【JDialogExample.java】사용자 다이얼로그

```
1 public class JDialogExample extends JDialog {
2     private JButton btnClose;
3
4     public JDialogExample( JFrame owner ) {
5         super(owner);
6         this.setTitle("JDialogExample");
7         this.setSize(300, 200);
8         this.setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
9         this.setResizable(false);
10        this.setModal(true);
11
12        this.setLocation(
13            owner.getLocationOnScreen().x + (owner.getWidth()-this.getWidth())/2,
14            owner.getLocationOnScreen().y + (owner.getHeight()-this.getHeight())/2
15        );
16    }
```

```

17     this.getContentPane().add(getBtnClose(), BorderLayout.SOUTH);
18 }
19
20 public JButton getBtnClose() {
21     if(btnClose == null) {
22         btnClose = new JButton();
23         btnClose.setText("닫기");
24         btnClose.addActionListener(new ActionListener() {
25             @Override
26             public void actionPerformed(ActionEvent e) {
27                 JDialogExample.this.dispose();
28             }
29         });
30     }
31     return btnClose;
32 }
33 }

```

#### 【JFrameExample.java】소유자 윈도우

```

1 public class JFrameExample extends JFrame {
2     private JButton btnOpenDialog;
3
4     public JFrameExample() {
5         this.setTitle("JFrame");
6         this.setSize(500, 400);
7         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
8         this.getContentPane().add(getBtnOpenDialog(), BorderLayout.SOUTH);
9     }
10
11     private JButton getBtnOpenDialog() {
12         if (btnOpenDialog == null) {
13             btnOpenDialog = new JButton();
14             btnOpenDialog.setText("다이얼로그 띄우기");
15             btnOpenDialog.addActionListener(new java.awt.event.ActionListener() {
16                 public void actionPerformed(java.awt.event.ActionEvent e) {
17                     JDialogExample jDialog = new JDialogExample(JFrameExample.this);
18                     jDialog.setVisible(true);
19                 }
20             });
21         }

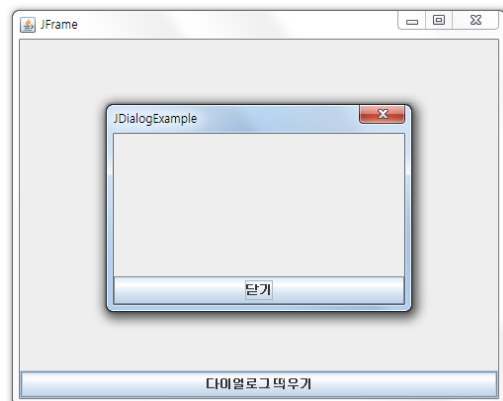
```

```

22     return btnOpenDialog;
23 }
24
25 public static void main(String[] args) {
26     SwingUtilities.invokeLater(new Runnable() {
27         public void run() {
28             JFrameExample jFrame = new JFrameExample();
29             jFrame.setVisible(true);
30         }
31     });
32 }
33 }

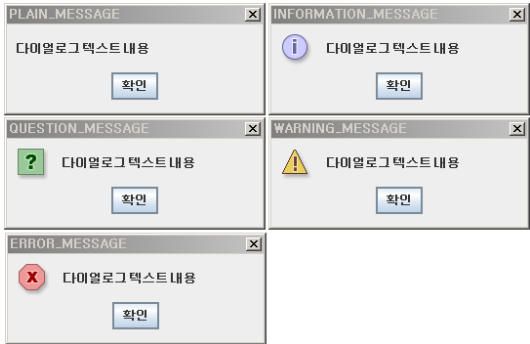
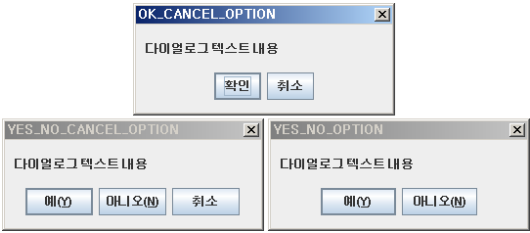
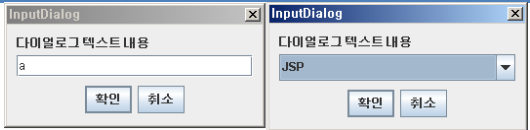
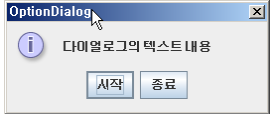
```

#### 【실행결과】



## 13.2 표준화된 다이얼로그

자바는 자주 사용되는 표준화된 다이얼로그(MessageDialog, ConfirmDialog, InputDialog, OptionDialog)를 쉽게 생성하기 위해 JOptionPane 클래스를 제공한다. JOptionPane 는 자체가 다이얼로그가 아니고, showXXXDialog() 메소드를 호출하면 XXX 다이얼로그가 생성된다.

제공되는 메소드	제공되는 컴포넌트	다이얼로그 모양
showMessageDialog( ... )	[확인] 버튼	
showConfirmDialog( ... )	[확인, 취소] 버튼 [예, 아니오, 취소] 버튼 [예, 아니오] 버튼	
showInputDialog( ... )	하나의 입력 컴포넌트 [확인, 취소] 버튼	
showOptionDialog( ... )	사용자 정의 버튼	

showXXXDialog() 메소드는 다음과 같은 매개 변수를 가지는데, 이들 매개 변수로 다이얼로그의 모양을 결정 짓는다.

매개 변수	설명	값의 종류
Component parentComponent	다이얼로그의 소유자 윈도우, 다이얼로그는 소유자 윈도우의 중앙에 위치, 하지만 null 을 주면 스크린 중앙에 다이얼로그가 위치	JWindow, JFrame, JApplet, JDialog
Object message	다이얼로그의 텍스트 내용	String
String title	다이얼로그의 제목 내용	String
int optionType	버튼의 집합을 결정	OK_CANCEL_OPTION

	JOptionPane 의 상수 이용	YES_NO_OPTION YES_NO_CANCEL_OPTION
int messageType	보여줄 아이콘을 결정 JOptionPane 의 상수 이용	PLAIN_MESSAGE INFORMATION_MESSAGE QUESTION_MESSAGE WARNING_MESSAGE ERROR_MESSAGE
Icon icon	보여줄 아이콘을 직접 제공	ImageIcon
Object[] selectionValues	InputDialog 에서 사용됨. 사용할수 있는 선택 항목으로 12 미만이면 JComboBox 가 사용되고, 12 개 이상이면 JList 가 사용된다. 만약 null 이면 JTextField 가 표시된다.	String[] Icon[]
Object initialSelectionValue	InputDialog 에서 초기 선택값	String, Icon
Object[] options	OptionDialog 의 버튼들에 표시되는 문자열 또는 이미지	String[] Icon[]
Object initialValue	초기 포커스를 갖는 버튼의 문자열 또는 이미지	String Icon

showMessageDialog() 메소드를 제외하고 나머지 세 개의 showXXXDialog() 메소드는 모두 리턴값이 있다. 리턴값은 어떤 버튼이 눌러졌는지, 입력된 값이 무엇인지에 대한 정보를 담고 있다. ConfirmDialog 는 어떤 버튼이 눌러졌는지가 중요하기 때문에 showConfirmDialog() 메소드는 눌러진 버튼에 해당하는 int 타입의 JOptionPane 상수값을 리턴한다.

JOptionPane 상수	설명
JOptionPane.OK_OPTION	확인 버튼을 눌렀을 때
JOptionPane.CANCEL_OPTION	취소 버튼을 눌렀을 때
JOptionPane.YES_OPTION	예 버튼을 눌렀을 때
JOptionPane.NO_OPTION	아니오 버튼을 눌렀을 때
JOptionPane.CLOSED_OPTION	우측 상단의 x 버튼을 눌렀을 때

InputDialog 는 입력된 값이 중요하기 때문에 showInputDialog() 메소드는 입력 컴포넌트가 JTextField 이면 String 을 JList 나 JComboBox 면 선택된 Object 을 리턴한다. OptionDialog 역시 어떤 버튼이 눌러졌는지가 중요하기 때문에 showOptionDialog() 의 매개변수로 주어진 옵션(options) 배열에서 버튼 문자열에 해당하는 인덱스를 리턴한다.

#### 【JOptionPaneExampleOwner.java】표준화 다이얼로그

```
1 public class JOptionPaneExample extends JFrame {
```



```

2 private JButton btnMessage, btnConfirm;
3 private JButton btnInput, btnOption;
4
5 public JOptionPaneExample() {
6     this.setTitle("JOptionPaneExample");
7     this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
8     this.getContentPane().setLayout(new GridLayout(4,1));
9     this.getContentPane().add(getBtnMessage());
10    this.getContentPane().add(getBtnConfirm());
11    this.getContentPane().add(getBtnInput());
12    this.getContentPane().add(getBtnOption());
13    this.setSize(150, 150);
14 }
15
16 public JButton getBtnMessage() {
17     if(btnMessage == null) {
18         btnMessage = new JButton();
19         btnMessage.setText("MessageDialog");
20         btnMessage.addActionListener(new ActionListener() {
21             public void actionPerformed(ActionEvent e) {
22                 JOptionPane.showMessageDialog(
23                     JOptionPaneExample.this,
24                     "다이얼로그 텍스트 내용",
25                     "INFORMATION_MESSAGE",
26                     JOptionPane.INFORMATION_MESSAGE);
27             }
28         });
29     }
30     return btnMessage;
31 }
32
33 public JButton getBtnConfirm() {
34     if(btnConfirm == null) {
35         btnConfirm = new JButton();
36         btnConfirm.setText("ConfirmDialog");
37         btnConfirm.addActionListener(new ActionListener() {
38             public void actionPerformed(ActionEvent e) {
39                 int option = JOptionPane.showConfirmDialog(
40                     JOptionPaneExample.this,
41                     "다이얼로그 텍스트 내용",

```

**【실행결과】**



```

42         "OK_CANCEL_OPTION",
43         JOptionPane.OK_CANCEL_OPTION,
44         JOptionPane.PLAIN_MESSAGE,
45         null);
46     if(option == JOptionPane.OK_OPTION) {
47         System.out.println("확인 버튼을 눌렀군요");
48     } else if(option == JOptionPane.CANCEL_OPTION) {
49         System.out.println("취소 버튼을 눌렀군요");
50     } else if(option == JOptionPane.CLOSED_OPTION) {
51         System.out.println("닫기 버튼을 눌렀군요");
52     }
53 }
54 });
55 }
56 return btnConfirm;
57 }
58
59 public JButton getBtnInput() {
60     if(btnInput == null) {
61         btnInput = new JButton();
62         btnInput.setText("InputDialog");
63         btnInput.addActionListener(new ActionListener() {
64             public void actionPerformed(ActionEvent e) {
65                 String input = null;
66                 input = JOptionPane.showInputDialog(
67                     JOptionPaneExample.this,
68                     "다이얼로그 텍스트 내용",
69                     "InputDialog",
70                     JOptionPane.INFORMATION_MESSAGE);
71                 System.out.println("입력된 텍스트: " + input);
72
73                 input = (String) JOptionPane.showInputDialog(
74                     JOptionPaneExample.this,
75                     "다이얼로그 텍스트 내용",
76                     "InputDialog",
77                     JOptionPane.PLAIN_MESSAGE,
78                     null,
79                     new String[] {"Java", "JDBC", "JSP", "Spring"},
80                     "JDBC");
81                 System.out.println("선택된 항목: " + input);

```

```

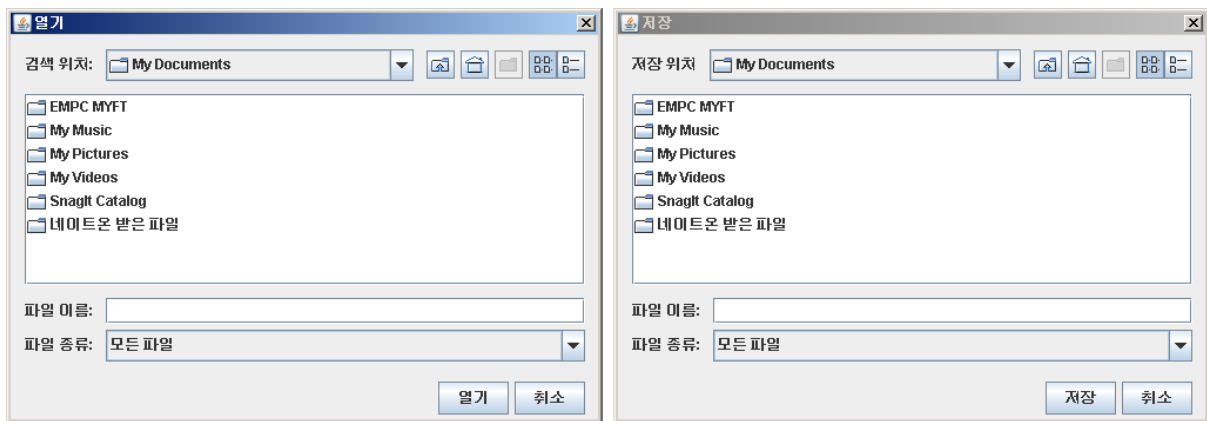
82     }
83     });
84 }
85 return btnInput;
86 }
87
88 public JButton getBtnOption() {
89     if(btnOption == null) {
90         btnOption = new JButton();
91         btnOption.setText("OptionDialog");
92         btnOption.addActionListener(new ActionListener() {
93             public void actionPerformed(ActionEvent e) {
94                 int option = JOptionPane.showOptionDialog(
95                     JOptionPaneExample.this,
96                     "다이얼로그의 텍스트 내용",
97                     "OptionDialog",
98                     JOptionPane.YES_NO_OPTION,
99                     JOptionPane.INFORMATION_MESSAGE,
100                    null,
101                    new String[] {"시작", "종료"},
102                    "시작");
103                 if(option == 0) {
104                     System.out.println("시작 버튼을 눌렀군요");
105                 } else if(option == 1) {
106                     System.out.println("종료 버튼을 눌렀군요");
107                 }
108             }
109         });
110     }
111     return btnOption;
112 }
113
114 public static void main(String[] args) {
115     SwingUtilities.invokeLater(new Runnable() {
116         public void run() {
117             JOptionPaneExample jFrame = new JOptionPaneExample();
118             jFrame.setVisible(true);
119         }
120     });
121 }

```

### 13.3 파일 다이얼로그

UI 프로그램에서 파일 다이얼로그는 사용자가 파일을 선택할 때 매우 편리함을 제공한다. 파일 다이얼로그는 용도에 따라 열기와 저장용으로 구분된다. 자바는 파일 다이얼로그를 위해 JFileChooser 를 제공하고 있다. JFileChooser 객체를 생성한 후, showOpenDialog()와 showSaveDialog() 메소드를 이용해서 열기와 저장용 파일 다이얼로그를 띄울 수 있다. showOpenDialog()와 showSaveDialog() 메소드의 매개 값은 소유자 윈도우이다.

```
JFileChooser jFileChooser = new JFileChooser();
jFileChooser.showOpenDialog(jFrame); 또는 jFileChooser.showSaveDialog(jFrame);
```



기본 검색 위치는 로그인한 사용자의 홈 디렉토리가 된다. 만약 기본 검색 위치를 변경하고 싶다면 setCurrentDirectory() 메소드를 이용하면 된다.

```
JFileChooser jFileChooser = new JFileChooser();
jFileChooser.setCurrentDirectory(new File("C:/Temp"));
jFileChooser.showOpenDialog(jFrame);
```

파일 종류는 JComboBox 로 선택할 수 있는데, 기본적으로 '모든 파일' 항목만 있다. 파일의 종류를 추가하기 위해서는 FileNameExtensionFilter 객체를 생성하고, addChoosableFileFilter() 메소드로 추가해 주면 된다.

```
JFileChooser jFileChooser = new JFileChooser();
jFileChooser.addChoosableFileFilter(new FileNameExtensionFilter("텍스트 파일(*.txt)", "txt"));
jFileChooser.showOpenDialog(jFrame);
```

showOpenDialog()와 showSaveDialog() 메소드는 모두 int 값을 리턴한다. 리턴값은 파일 다이얼로그에서 어떤 버튼이 눌려졌는지 구분할 용도로 사용되는데, 다음 상수값 중 하나를 리턴한다.

옵션 상수	설명
APPROVE_OPTION	[열기], [저장] 버튼을 클릭했을 때
CANCEL_OPTION	[취소], [x] 버튼을 클릭했을 때

```
JFileChooser jFileChooser = new JFileChooser();
int option = jFileChooser.showOpenDialog(jFrame);
if(option == JFileChooser.APPROVE_OPTION) {
    //[열기] 버튼을 눌렀을 경우 실행할 코드
} else {
    //[취소], [닫기] 버튼을 눌렀을 경우 실행할 코드
}
```

showOpenDialog()와 showSaveDialog() 메소드가 JFileChooser.APPROVE\_OPTION 을 리턴했다면 사용자가 파일을 선택하고, [열기] 또는 [저장] 버튼을 클릭한 경우이다. 사용자가 선택한 파일은 JFileChooser 의 getSelectedFile() 메소드로 알 수 있는데, 리턴 객체인 java.io.File 에는 사용자가 선택한 파일의 경로 정보가 담겨 있다.

```
File file = jFileChooser.getSelectedFile();
System.out.println("선택한 파일 절대경로: " + file.getAbsolutePath());
System.out.println("선택한 파일 이름: " + file.getName());
```

#### 【JFileChooserExample.java】 파일 다이얼로그

```
1 public class JFileChooserExample extends JFrame {
2     private JButton btnFileOpen, btnFileSave;
3
4     public JFileChooserExample() {
5         this.setTitle("JFileChooserExample");
6         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7         this.getContentPane().setLayout(new GridLayout(2,1));
8         this.getContentPane().add(getBtnFileOpen());
9         this.getContentPane().add(getBtnFileSave());
10        this.setSize(150, 100);
11    }
12
13    public JButton getBtnFileOpen() {
14        if(btnFileOpen == null) {
```

```

15     btnFileOpen = new JButton();
16     btnFileOpen.setText("File Open");
17     btnFileOpen.addActionListener(new ActionListener() {
18         public void actionPerformed(ActionEvent e) {
19             JFileChooser jFileChooser = new JFileChooser();
20             jFileChooser.addChoosableFileFilter(new FileNameExtensionFilter(
21                 "그림파일(*.jpg, *.gif, *.bmp)", "jpg", "gif", "bmp"));
22             jFileChooser.addChoosableFileFilter(
23                 new FileNameExtensionFilter("텍스트 파일(*.txt)", "txt"));
24             int option = jFileChooser.showOpenDialog(JFileChooserExample.this);
25             if(option == JFileChooser.APPROVE_OPTION) {
26                 File file = jFileChooser.getSelectedFile();
27                 System.out.println("열어야할 파일 절대경로: " + file.getAbsolutePath());
28                 System.out.println("열어야할 파일 이름: " + file.getName());
29             } else if(option == JFileChooser.CANCEL_OPTION) {
30                 System.out.println("취소 또는 닫기를 눌렀군요");
31             }
32         }
33     });
34 }
35 return btnFileOpen;
36 }
37
38 public JButton getBtnFileSave() {
39     if(btnFileSave == null) {
40         btnFileSave = new JButton();
41         btnFileSave.setText("File Save");
42         btnFileSave.addActionListener(new ActionListener() {
43             public void actionPerformed(ActionEvent e) {
44                 JFileChooser jFileChooser = new JFileChooser();
45                 jFileChooser.addChoosableFileFilter(new FileNameExtensionFilter(
46                     "그림파일(*.jpg, *.gif, *.bmp)", "jpg", "gif", "bmp"));
47                 jFileChooser.addChoosableFileFilter(
48                     new FileNameExtensionFilter("텍스트 파일(*.txt)", "txt"));
49                 int option = jFileChooser.showSaveDialog(JFileChooserExample.this);
50                 if(option == JFileChooser.APPROVE_OPTION) {
51                     File file = jFileChooser.getSelectedFile();
52                     System.out.println("저장할 파일: " + file.getAbsolutePath());
53                 } else if(option == JFileChooser.CANCEL_OPTION) {
54                     System.out.println("취소 또는 닫기를 눌렀군요");

```

```

55     }
56     }
57     });
58 }
59 return btnFileSave;
60 }
61
62 public static void main(String[] args) {
63     SwingUtilities.invokeLater(new Runnable() {
64         public void run() {
65             JFileChooserExample jFrame = new JFileChooserExample();
66             jFrame.setVisible(true);
67         }
68     });
69 }
70 }

```

【실행결과】



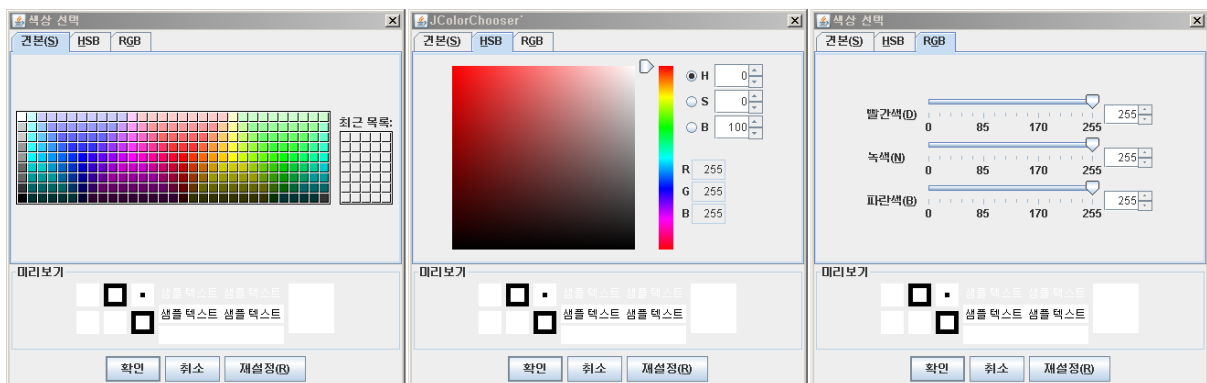
## 13.4 색상 다이얼로그

색상 다이얼로그는 색상을 사용자가 선택할 수 있는 편리함을 제공해 준다. 자바는 색상 다이얼로그를 위해 JColorChooser 를 제공하는데, JColorChooser 객체를 생성한 후, showDialog() 메소드를 이용해서 색상 다이얼로그를 띄울 수 있다.

```

JColorChooser jColorChooser = new JColorChooser();
Color color = jColorChooser.showDialog(jFrame, "색상 선택", null);

```



showDialog() 메소드의 첫번째 매개값은 소유자 윈도우이고, 두번째 매개값은 색상 다이얼로그의 제목을 주면 된다. 세번째 매개값은 미리보기 색상에 해당하는 Color 객체를 주면 되는데, 만약 null 을 주면 미리보기 색상은 하얀색이 된다. JColorChooser 는 사용자가 색상을 선택할 수 있도록 여러가지 탭을 제공하고 있다. 색상을 결정하고 나서 확인 버튼을 클릭하면 사용자가

선택한 색상은 Color 객체로 생성되고, showDialog() 메소드의 리턴값으로 나온다. 다음 예제는 색상 다이얼로그에서 선택한 색상을 버튼의 배경색으로 설정한다.

#### 【JColorChooserExample.java】 색상 다이얼로그

```
1 public class JColorChooserExample extends JFrame {
2     private JButton btnColor;
3
4     public JColorChooserExample() {
5         this.setTitle("JColorChooserExample");
6         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
7         this.getContentPane().setLayout(new GridLayout(1,1));
8         this.getContentPane().add(getBtnColor());
9         this.setSize(150, 60);
10    }
11
12    public JButton getBtnColor() {
13        if(btnColor == null) {
14            btnColor = new JButton();
15            btnColor.setText("JColorChooser");
16            btnColor.addActionListener(new ActionListener() {
17                public void actionPerformed(ActionEvent e) {
18                    JColorChooser jColorChooser = new JColorChooser();
19                    Color color = jColorChooser.showDialog(
20                        JColorChooserExample.this, "색상 선택", Color.BLUE);
21                    btnColor.setBackground(color);
22                }
23            });
24        }
25        return btnColor;
26    }
27
28    public static void main(String[] args) {
29        SwingUtilities.invokeLater(new Runnable() {
30            public void run() {
31                JColorChooserExample jFrame = new JColorChooserExample();
32                jFrame.setVisible(true);
33            }
34        });
35    }
36 }
```

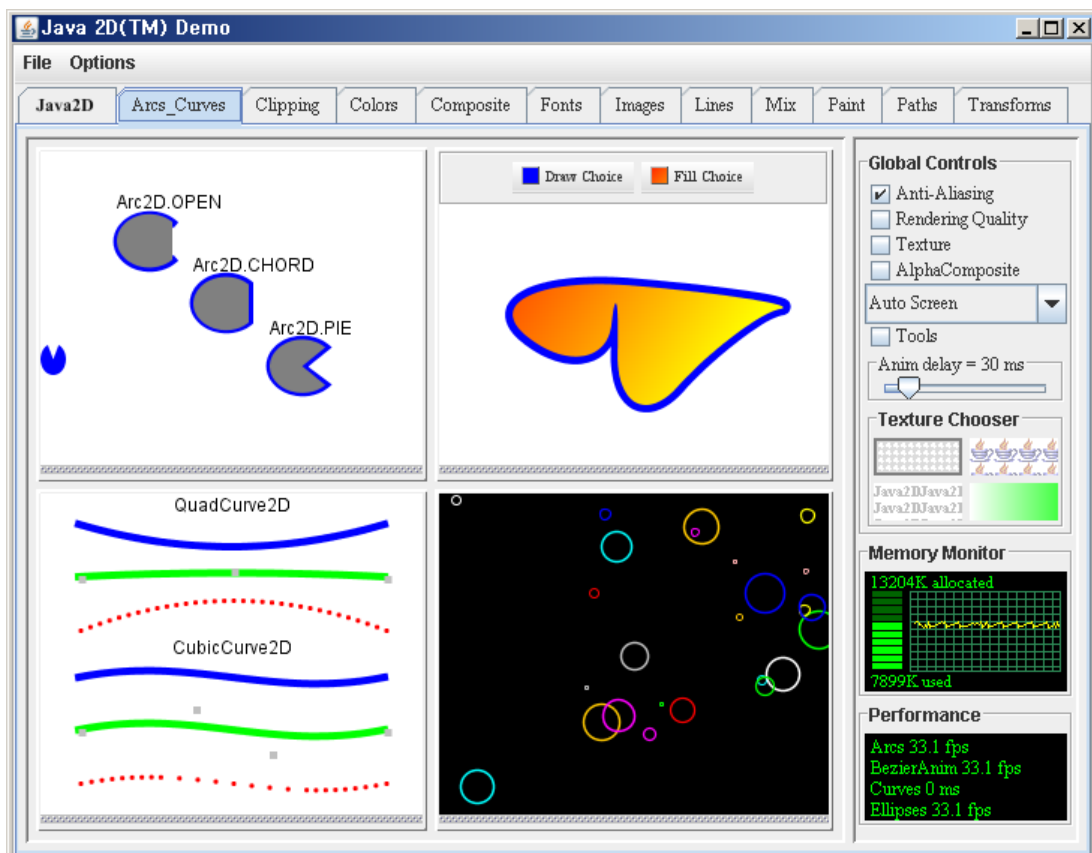
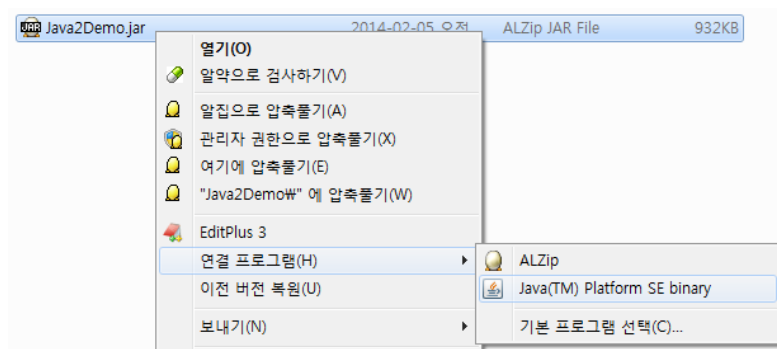
#### 【실행결과】





## 15 절. Java 2D

UI 프로그램의 구성 요소는 모든 것이 드로잉(drawing)된 것들이다. 여러분이 보는 윈도우 창이나 버튼, 이미지등 모든 것이 프로그램에 의해 화면에 드로잉된다. 자바는 드로잉 처리에 필요한 API 를 Java2D 라는 이름으로 제공하고 있다. Java2D 는 기본적으로 JDK 에 포함되어 있기 때문에 별도로 설치할 필요가 없다. Java2D 로 만들 수 있는 다양한 그래픽을 경험해 보고 싶다면 책소스에서 Java2Demo.jar 파일을 탐색기로 찾아 선택한 다음, Java(TM) Platform SE binary 를 연결 프로그램으로 선택하면 된다.

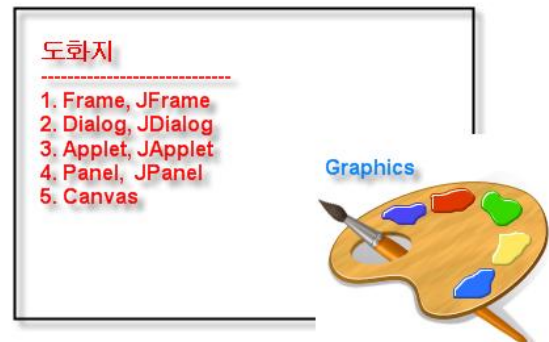


## 14.1 도화지 클래스와 붓 클래스

손으로 그림을 그리기 위해서는 도화지와 붓이 필요하다. 마찬가지로 자바에서 그림을 그릴 때 그런 역할을 하는 클래스들이 있다. 도화지는 일반적으로 Canvas 클래스를 주로 사용하지만 JFrame, JDialog, JPanel, JApplet 도 도화지가 될 수 있다. 붓은 Graphics 클래스이다. Graphics 는 도화지의 paint() 메소드의 매개값으로 제공된다. paint() 메소드는 주어진 Graphics 를 가지고 해당 도화지에 그림을 그린다.

도화지가 페인팅될 준비가 되면 그래픽 스레드는 도화지의 paint() 메소드를 호출해서 Graphics 로 도화지 위에 페인팅을 한다. 그리고 다음 두가지 경우에 paint() 메소드를 다시 호출하여 Graphics 로 도화지 위에 재 페인팅을 한다.

- 도화지가 아이콘화하였다가 다시 확대했을 때
- 도화지의 크기가 변경되었을 때
- 도화지가 스크린 밖으로 가려졌다가 다시 나타났을 때



아래 예제를 실행해 보면 윈도우 창이 다른 윈도우 창에 가려졌다가 나타나거나, 아이콘화되었다가 확대했을 때에 Canvas 의 paint() 메소드가 다시 호출되어 글자를 다시 그린다. 확인 방법은 콘솔창에서 "paint() 메소드 실행"이 계속 출력되는 것을 보면 된다.

### 【CanvasPaintExample.java】 Paint() 메소드가 자동적으로 호출되는 시점 알기

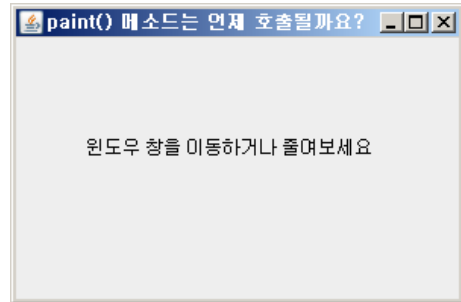
```
1 public class CanvasPaintExample extends JFrame {
2     public CanvasPaintExample() {
3         setTitle("paint() 메소드는 언제 호출될까요?");
4         getContentPane().add(new MyCanvas(), BorderLayout.CENTER);
5         setSize(300, 200);
6     }
7
8     public class MyCanvas extends Canvas {
9         public void paint(Graphics g) {
10             g.drawString("윈도우 창을 줄이거나 늘려보세요", 50, 80);
11             System.out.println("paint() 메소드 실행");
12         }
13     }
14
15     public static void main(String[] args) {
16         SwingUtilities.invokeLater(new Runnable() {
17             public void run() {
```

```

18 CanvasPaintExample JFrame = new CanvasPaintExample();
19 JFrame.setVisible(true);
20 }
21 };
22 }
23 }

```

【실행결과】



다음 예제는 JFrame 을 도화지로 하고 이전 예제와 동일한 내용을 드로잉한다. 차이점은 이전 내용을 지우는 코드가 추가되었다. 그 이유는 조금 후 설명된다.

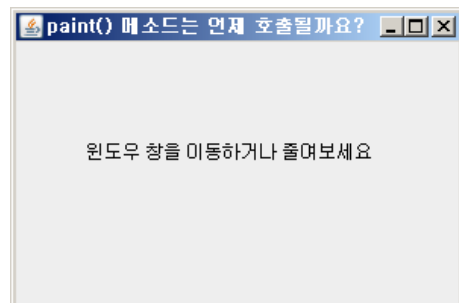
【JFramePaintExample.java】 Paint() 메소드가 자동적으로 호출되는 시점 알기

```

1 public class JFramePaintExample extends JFrame {
2     public JFramePaintExample() {
3         setTitle("paint() 메소드는 언제 호출될까요?");
4         setSize(300, 200);
5     }
6
7     @Override
8     public void paint(Graphics g) {
9         //이전 내용을 지우는 코드
10        g.clearRect(0, 0, getWidth(), getHeight());
11        //새로운 내용을 그리는 코드
12        g.setColor(Color.BLACK);
13        g.drawString("윈도우 창을 줄이거나 늘려보세요", 50, 80);
14        System.out.println("paint() 메소드 실행");
15    }
16
17    public static void main(String[] args) {
18        SwingUtilities.invokeLater(new Runnable() {
19            public void run() {
20                JFramePaintExample JFrame = new JFramePaintExample();
21                JFrame.setVisible(true);
22            }
23        });
24    }
25 }

```

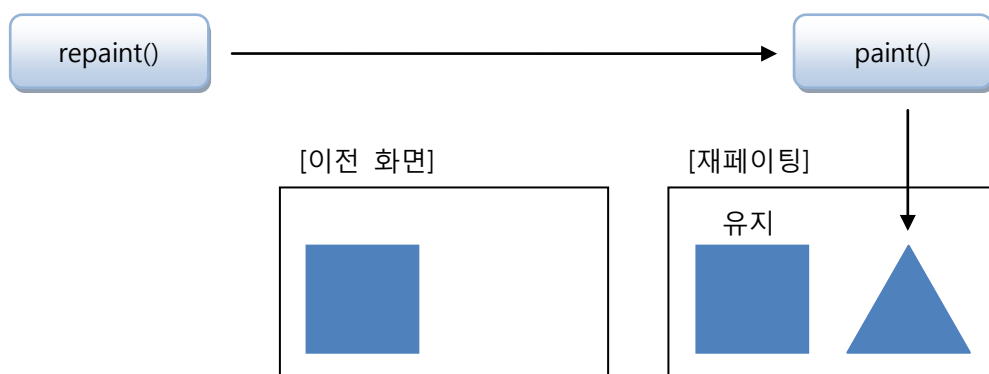
【실행결과】



## 14.2 수동으로 재페인팅

도화지에 페인팅하는 작업은 이벤트 디스패칭 스레드가 한다. 이벤트 디스패칭 스레드는 도화지의 내용이 갱신될 필요가 있을 때 도화지의 `paint()` 메소드를 다시 호출해서 재페인팅을 한다. 그렇다면 개발자가 필요에 따라 수동으로 재 페인팅하고 싶을 경우는 어떻게 해야할까? 개발자는 재페인팅을 위해서 도화지의 `paint()` 메소드를 직접 호출할 수는 없다. 개발자가 할 수 있는 유일한 방법은 도화지의 `repaint()` 메소드를 호출하는 방법뿐이다. `repaint()` 호출 이후의 내부 처리 과정은 Swing 과 AWT 가 조금씩 다르다.

Swing 도화지(`JFrame`, `JDialog`, `JApplet`, `JPanel` 등)에서 개발자가 `repaint()` 메소드를 호출하면 이벤트 디스패칭 스레드는 `paint()` 메소드를 호출하여 재페인팅 작업을 하게된다. 이때 이전 내용은 지우지 않고 새로 페인팅할 부분만 작업한다.



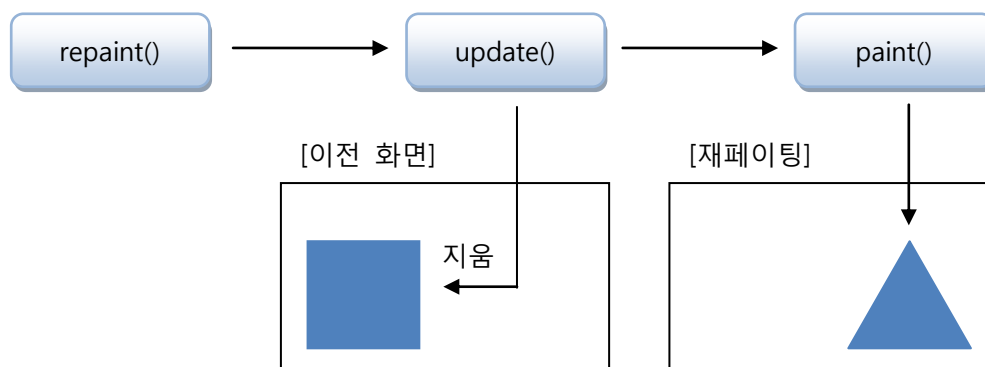
그렇기 때문에 이전 내용을 모두 지우고, 새로 페인팅을 하고 싶다면 `paint()` 메소드 내에서 이전 내용을 먼저 지우고, 새로운 내용을 그려야 한다.

```

public void paint(Graphics g) {
    //이전 내용을 지우는 코드
    g.clearRect(0, 0, getWidth(), getHeight());
    //새로운내용을 그리는 코드
    //~
}
  
```

AWT 도화지(`Frame`, `Dialog`, `Applet`, `Panel`, `Canvas`)에서 개발자가 `repaint()` 메소드를 호출하면 이벤트 디스패칭 스레드는 `update()` 메소드를 호출한다. 그리고 이 `update()` 메소드에서 `paint()` 메소드가 호출되어 재페인팅 작업이 일어난다. `update()` 메소드의 역할은 새로운 도화지를

준비하는 것이다. update() 메소드의 기본 동작은 paint() 메소드로 재페인팅하기 전에 이전 내용을 백그라운드 색깔로 덧칠해 모두 지운다.



AWT 도화지에서 `repaint()` 메소드를 호출할 때 이전 내용을 유지하면서 새로운 내용을 추가하려면 `update()` 메소드를 오버라이딩해서 이전 내용을 지우는 코드를 작성하지 않고 그냥 `paint(g)` 메소드만 호출하면 된다.

```

public void update(Graphics g) {
    paint(g);
}

public void paint(Graphics g) {
    //새로운내용을 그리는 코드
}
    
```

다음은 마우스를 누른 상태에서 드래그하면 마우스가 움직인 경로를 따라서 \*가 그려지는 예제이다. 이 예제의 핵심은 Canvas 가 AWT 도화지 이므로 `update()` 메소드를 오버라이딩하지 않았을 경우에는 이전 마우스의 자취는 남지 않는다는 것이다.

#### 【RepaintExample.java】수동으로 재 페인팅

```

1 public class RepaintExample extends JFrame {
2     public RepaintExample() {
3         setTitle("수동으로 재 페인팅");
4         getContentPane().add(new MyCanvas(), BorderLayout.CENTER);
5         setSize(500, 400);
6     }
7
8     public class MyCanvas extends Canvas implements MouseMotionListener {
9         private int x;
10        private int y;
11        public MyCanvas() {
    
```

```

12      addMouseMotionListener(this);
13  }
14  public void mouseDragged(MouseEvent e) {
15      x = e.getX();
16      y = e.getY();
17      repaint();
18  }
19  public void mouseMoved(MouseEvent e) {
20  }
21  public void update(Graphics g) {
22      paint(g);
23  }
24  public void paint(Graphics g) {
25      g.drawString("*", x, y);
26  }
27  }

28
29  public static void main(String[] args) {
30      SwingUtilities.invokeLater(new Runnable() {
31          public void run() {
32              RepaintExample JFrame = new RepaintExample();
33              JFrame.setVisible(true);
34          }
35      });
36  }
37  }

```



## 14.3 색상과 폰트 설정

Graphics 의 drawString()으로 그려지는 글자의 기본 색상은 검정색이고, 사이즈는 12, 폰트의 종류는 Dialog 이다. 이번 절에서는 글자의 속성인 색상, 사이즈, 폰트를 변경하는 방법에 대해서 알아보기로 하자.

### 1) 색상 설정

우리가 붓에 물감을 묻혀 그림을 그리듯이 Graphics 에 색상을 설정할 수 있다. Graphics 의 setColor() 메소드에 매개값으로 Color 객체를 넘겨주면, 해당 색상으로 Graphics 가 준비된다. 이렇게 준비된 Graphics 로 문자, 선, 도형을 그리면 해당 색상으로 모든 것이 그려진다. Color

클래스는 기본적으로 13 가지의 색깔을 표현하는 Color 객체 상수를 가지고 있다. 이들 상수를 이용하면 13 가지의 색깔 객체를 쉽게 얻을 수 있다. 자바 3 이전까지는 소문자 Color 상수만 있었는데, 상수는 대문자 관례를 따르기 위해 자바 4 이후부터는 대소문자 상수 모두를 가지고 있다.

Color.BLACK (black)	Color.WHITE(white)	Color.RED(red)
Color.GREEN(green)	Color.BLUE(blue)	Color.GRAY(gray)
Color.DARK_GRAY(darkGray)	Color.LIGHT_GRAY(lightGray)	Color.CYAN(cyan)
Color.MAGENTA(magenta)	Color.ORANGE(orange)	Color.PINK(pink)
Color.YELLOW(yellow)		

붓에 빨간색을 묻히고 싶다면 다음과 같이 코드를 작성하면 된다.

```
g.setColor(Color.RED);
```

13 가지의 상수로 정해진 색상 이외의 다른 색상을 얻고 싶다면 R(Red), G(Green), B(Blue) 값을 생성자 매개값으로 차례대로 지정해서 Color 객체를 만들면 된다. RGB 값은 각각 0~255 사이의 값을 주면 된다. 다음은 주어진 RGB 값으로 Color 객체를 얻고, 이것을 Graphics 에 설정한다.

```
Color color = new Color(100, 200, 50);
g.setColor(color);
```

## 2) 폰트 설정

운영체제에서 사용하는 폰트의 종류는 여러가지가 있다. Graphics 의 drawString() 메소드로 문자를 그릴 때 특정 폰트를 지정할 수 있다. Graphics 의 setFont() 메소드를 이용하면 되는데, 매개값으로 Font 객체를 넘겨주면 이후로 그려지는 문자는 해당 폰트로 그려진다. Font 객체를 만들 때에는 폰트의 이름, 스타일, 크기등의 정보를 생성자의 매개값으로 주면된다.

```
Font font = new Font(String name, int style, int size);
```

폰트 이름은 굴림체, 돋움체, 바탕체, Arial 등을 말한다. 폰트는 운영체제마다 차이가 있기 때문에 운영체제에서 제공되는 폰트의 이름을 알아내기 위해서 다음 코드를 사용할 수 있다.

```
GraphicsEnvironment ge = GraphicsEnvironment.getLocalGraphicsEnvironment();
String[] fontNames = ge.getAvailableFontFamilyNames();
for(String fontName : fontNames) {
    System.out.println(fontName);
}
```

폰트 스타일은 보통, 굵음, 기울임을 말한다. 폰트 스타일은 Font 클래스의 상수를 사용한다. 스타일 상수는 다음과 같이 세가지가 있다.

Font.PLAIN (보통 글자)	Font.BOLD (굵은 글자)	Font.ITALIC(기울인 글자)
--------------------	-------------------	---------------------

이들 상수는 | 연산자를 사용해서 두가지 스타일을 모두 적용할 수 있다. 다음은 돋움체이고, 굵고 기울어지게 15 사이즈로 Font 객체를 만들고 Graphics 에 설정한다.

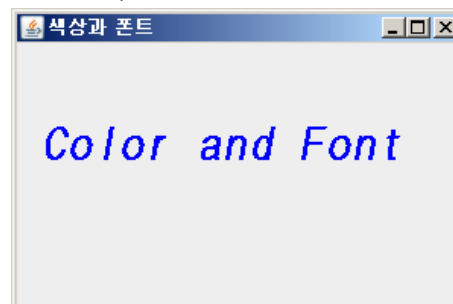
```
Font font = new Font("돋움체", Font.BOLD | Font.ITALIC, 15);
g.setFont(font);
```

#### 【ColorFontExampleExample.java】 색상과 폰트 설정

```

1 public class ColorFontExample extends JFrame {
2     public ColorFontExample() {
3         setTitle("색상과 폰트");
4         getContentPane().add(new MyCanvas(), BorderLayout.CENTER);
5         setSize(300, 200);
6     }
7
8     public class MyCanvas extends Canvas {
9         public void paint(Graphics g) {
10             g.setColor(Color.BLUE);
11             g.setFont(new Font("돋움체", Font.BOLD | Font.ITALIC, 30));
12             g.drawString("Color and Font", 20, 100);
13         }
14     }
15
16     public static void main(String[] args) {
17         SwingUtilities.invokeLater(new Runnable() {
18             public void run() {
19                 ColorFontExample jFrame = new ColorFontExample();
20                 jFrame.setVisible(true);
21             }
22         });
23     }
24 }
```

#### 【실행결과】



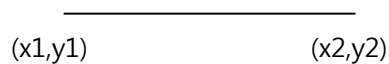


## 14.4 기본 도형 그리기

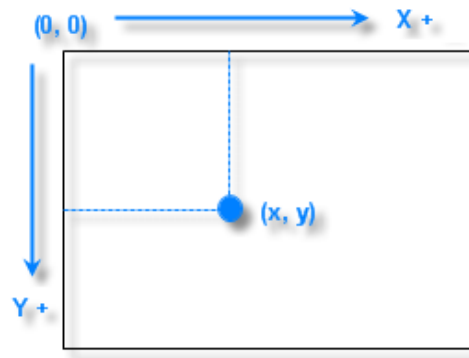
기본 도형에는 선, 원, 타원, 사각형, 호등이 있다. Graphics 에는 이들 기본 도형을 그리기 위한 메소드를 다음과 같이 제공하고 있다.

- 선을 그리는 메소드

- drawLine(int x1, int y1, int x2, int y2)

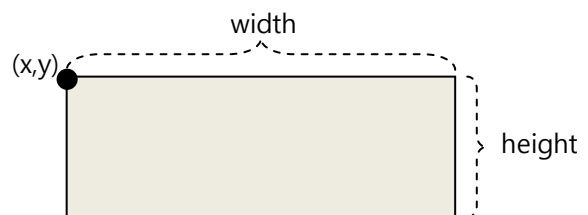


그래픽 좌표 시스템



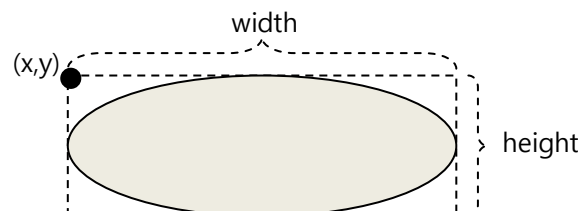
- 사각형을 그리는 메소드

- drawRect(int x, int y, int width, int height)
- fillRect(int x, int y, int width, int height)



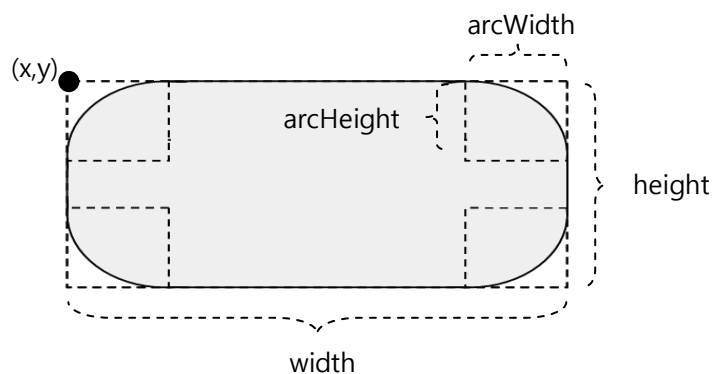
- 원 또는 타원을 그리는 메소드

- drawOval(int x, int y, int width, int height)
- fillOval(int x, int y, int width, int height)

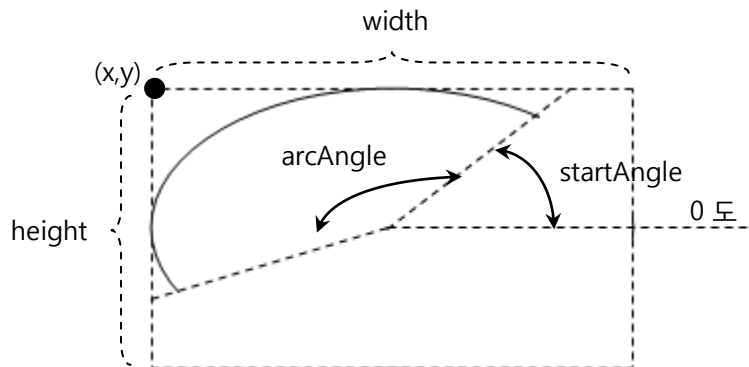


- 모서리가 둥근 사각형을 그리는 메소드

- drawRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)
- fillRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)



- 입체 사각형을 그리는 메소드
  - draw3DRect(int x, int y, int width, int height, boolean raised)
  - fill3DRect(int x, int y, int width, int height, boolean raised)
- 호를 그리는 메소드
  - drawArc(int x, int y, int width, int height, int startAngle, int arcAngle)
  - fillArc(int x, int y, int width, int height, int startAngle, int arcAngle)



- 다각형을 그리는 메소드
  - drawPolygon(int [] xPoints, int[] yPoints, int nPoints)
  - fillPolygon(int [] xPoints, int[] yPoints, int nPoints)
  - drawPolyline(int [] xPoints, int[] yPoints, int nPoints)
- 영역을 지우는 메소드
  - clearRect(int x, int y, int width, int height)

다음은 Graphics 가 제공하는 다양한 메소드를 이용해서 Canvas 에 그림을 그리는 예제이다. drawXXX()와 fillXXX()의 차이점은 선만 그리느냐 아니면 내부를 채우느냐이다.

#### 【ShapeExample.java】도형 그리기

```

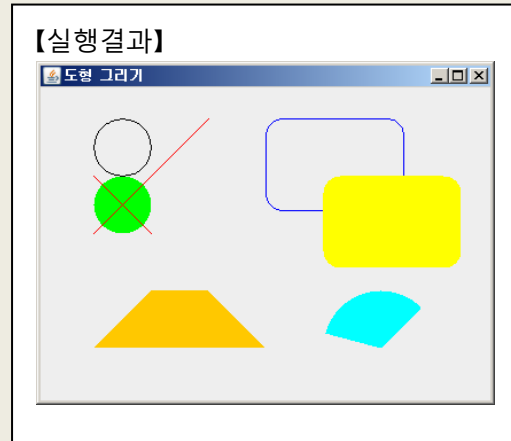
1 class ShapeExample extends JFrame {
2     public ShapeExample() {
3         setTitle("도형 그리기");
4         getContentPane().add(new MyCanvas(), BorderLayout.CENTER);
5         setSize(400,300);
6     }
7
8     public class MyCanvas extends Canvas {
9         public void paint(Graphics g) {

```

```

10    g.drawOval(50,50, 50,50);
11
12    g.setColor(Color.GREEN);
13    g.fillOval(50,100, 50,50);
14
15    g.setColor(Color.RED);
16    g.drawLine(50,100, 100,150);
17    g.drawLine(150,50, 50,150);
18
19    g.setColor(Color.BLUE);
20    g.drawRoundRect(200, 50, 120, 80, 30, 30);
21    g.setColor(Color.YELLOW);
22    g.fillRoundRect(250, 100, 120, 80, 30, 30);
23
24    g.setColor(Color.ORANGE);
25    g.fillPolygon(new int[]{50, 100, 150, 200}, new int[]{250, 200, 200, 250}, 4);
26
27    g.setColor(Color.cyan);
28    g.fillArc(250, 200, 100, 100, 45, 120);
29 }
30 }
31
32 public static void main(String[] args) {
33     SwingUtilities.invokeLater(new Runnable() {
34         public void run() {
35             ShapeExample jFrame = new ShapeExample();
36             jFrame.setVisible(true);
37         }
38     });
39 }
40 }

```



## 14.5 안티 알리아싱(Anti Aliasing)

비트맵 그래픽의 방식은 작은 사각형(픽셀)을 최소 단위로 하기 때문에 이 픽셀들이 모여 만들어진 원, 곡선, 사선은 경계 부분에서 거친 계단 현상(알리어싱:aliasing)이 나타난다. 이 문제를 해결하기 위해 안티알리어싱(anti-aliasing) 기능이 필요하다. 안티알리어싱은 배경색과 이미지 색상의 중간 색상을 단계적으로 채워주므로써 경계선을 부드럽게 만들어 주는 기능이다.

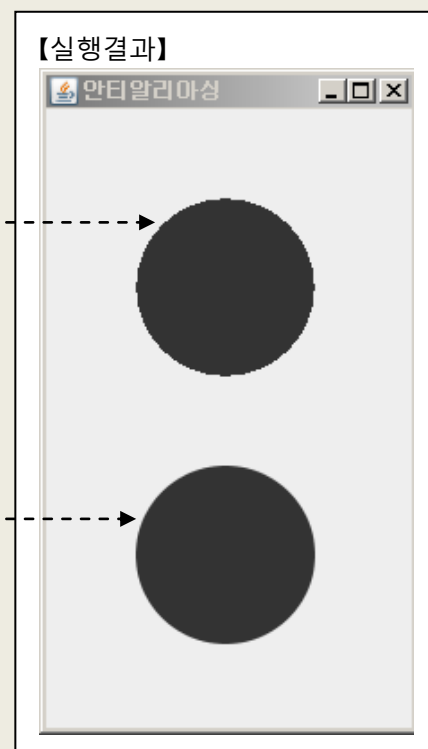
자바는 안티 알리어싱을 위해 Graphics2D 의 setRenderingHint() 메소드를 제공하고 있다. paint() 메소드의 매개변수 Graphics 는 사실 Graphics2D 객체를 참조하고 있기 때문에 다음과 같이 타입 변환을 통해 Graphics2D 객체를 얻고 setRenderingHint() 메소드를 호출할 수 있다.

```
public void paint(Graphics g) {
    Graphics2D g2 = (Graphics2D) g;
    g2.setRenderingHint(
        RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_ON);
    ...
}
```

Java 2D 는 여러가지 렌더링 알고리즘을 키와 값으로 구성된 맵(Map) 객체로 관리하고 있다. 안티 알리아싱 기능을 활성화할지 여부의 값은 RenderingHints.KEY\_ANTIALIASING 키로 관리하고 있기 때문에 이 키로 맵에 저장된 값을 RenderingHints.VALUE\_ANTIALIAS\_ON 으로 변경하면 된다. 다음 예제는 안티 알리아싱을 적용하지 않은 원과 안티 알리아싱을 적용한 원의 차이점을 보여준다.

#### 【AntiAliasingExample.java】 안티알리아싱 적용

```
1 public class AntiAliasingExample extends JFrame {
2     public AntiAliasingExample() {
3         setTitle("안티알리아싱");
4         getContentPane().add(new MyCanvas(), BorderLayout.CENTER);
5         setSize(200,350);
6     }
7
8     public class MyCanvas extends Canvas {
9         public void paint(Graphics g) {
10             g.fillOval(50, 50, 100, 100);
11
12             Graphics2D g2 = (Graphics2D) g;
13             g2.setRenderingHint(
14                 RenderingHints.KEY_ANTIALIASING,
15                 RenderingHints.VALUE_ANTIALIAS_ON);
16
17             g.fillOval(50, 200, 100, 100);
18         }
19     }
20
21     public static void main(String[] args) {
22         SwingUtilities.invokeLater(new Runnable() {
```



```

23         public void run() {
24             AntiAliasingExample jFrame = new AntiAliasingExample();
25             jFrame.setVisible(true);
26         }
27     });
28 }
    }

```

## 14.6 이미지 그리기

파일에 저장된 이미지를 화면에 보여주기 위해서는 이미지를 로딩하는 과정과 이를 페인팅하는 과정을 거쳐야 한다. 이미지 로딩이란 파일 시스템의 파일을 메모리로 읽어 들이는 것을 말한다. 이렇게 메모리로 읽어들인 이미지 데이터는 Image 객체에 할당된다. Image 객체를 얻는 방법은 두가지가 있는데, 첫번째 방법은 Toolkit 의 getImage() 메소드로 얻을 수 있다. getImage() 의 매개값에는 이미지 파일명 또는 이미지 파일 위치를 URL 객체로 넘겨주면 된다.

```

Toolkit toolkit = Toolkit.getDefaultToolkit();
//파일 시스템의 이미지 파일일 경우
Image image1 = toolkit.getImage(String filename);
//파일 시스템 또는 웹 서버의 이미지 파일
Image image2 = toolkit.getImage(URL url);

```

두번째 방법은 ImageIcon 객체를 생성해서 getImage() 메소드로 얻을 수 있다. ImageIcon 객체를 생성할 때 생성자에 이미지 파일명 또는 이미지 파일 위치를 URL 객체를 넘겨주면된다.

```

//파일 시스템의 이미지 파일일 경우
ImageIcon imageIcon1 = new ImageIcon(String filename);
Image image1 = imageIcon1.getImage();
//파일 시스템 또는 웹 서버의 이미지 파일일 경우
ImageIcon imageIcon2 = new ImageIcon(URL url);
Image image2 = imageIcon2.getImage();

```

URL 을 생성하는 방법은 다음과 같다.

```

//파일 시스템의 이미지 파일일 경우
URL url = new URL("file:///C:/wimages/logo.gif");
//클래스 파일과 동일한 디렉토리에 있는 이미지 파일일 경우
URL url = getClass().getResource("logo.gif");
//웹 서버의 이미지 파일일 경우

```

```
URL url = new URL("http://www.naver.com/logo.gif");
```

이미지를 화면에 보여주기 위해서는 Graphics 의 drawImage() 메소드로 페인팅을 해야 한다.

```
drawImage(Image img, int x, int y, ImageObserver observer)
```

페인팅할 Image 를 첫번째 매개값으로, 페인팅을 시작할 좌상단 좌표인 x 와 y 값을 두번째와 세번째 매개값으로 지정하면 된다. 네번째 매개값인 ImageObserver 는 이미지 로딩과 동시에 화면에 그려야하는 경우에 사용된다. Toolkit 의 getImage() 메소드가 실행되면 새로운 스레드가 시작하면서 이미지를 로딩한다. 이미지 로딩이 아직 끝나지 않은 상태에서 Graphics 의 drawImage()를 호출할 경우, 이미지 로딩 스레드는 주기적으로 ImageObserver 의 imageUpdate() 메소드를 호출하여 ImageObserver 에게 이미지 로딩 상태를 알려주게 된다. 그러면 ImageObserver 는 repaint() 메소드를 호출해서 그 때까지 로딩된 이미지를 다시 페인팅한다. 도화지로 사용되는 모든 컴포넌트는 ImageObserver 인터페이스를 구현하고 있기 때문에 ImageObserver 가 될 수 있다. 그래서 네번째 매개값은 Graphics 로 이미지를 그리는 도화지가 되므로 보통 this 를 대입하면 된다.

#### 【ImageExample.java】 이미지 그리기

```
1 public class ImageExample extends JFrame {
2     public ImageExample() {
3         setTitle("이미지 그리기");
4         getContentPane().add(new MyCanvas(), BorderLayout.CENTER);
5         add(new MyCanvas(), BorderLayout.CENTER);
6         setSize(500,350);
7     }
8
9     public class MyCanvas extends Canvas {
10        private Image imgSun, imgMoon;;
11        public MyCanvas() {
12            setBackground(Color.WHITE);
13            Toolkit toolkit = Toolkit.getDefaultToolkit();
14            imgSun = toolkit.getImage(getClass().getResource("sun.gif"));
15            imgMoon = new ImageIcon(getClass().getResource("moon.gif")).getImage();
16        }
17        public void paint(Graphics g) {
18            g.drawImage(imgSun, 10, 10, this);
19            g.drawImage(imgMoon, 300, 20, this);
20        }
21    }
22 }
```



```

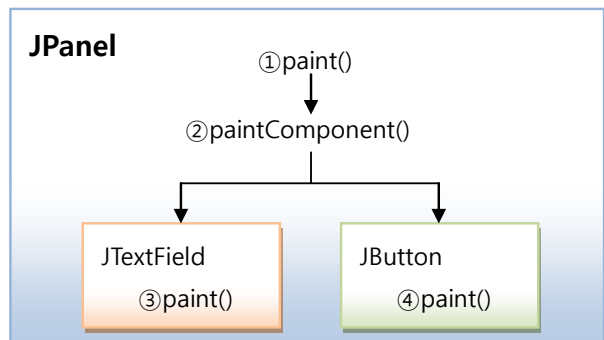
23 public static void main(String[] args) {
24     SwingUtilities.invokeLater(new Runnable() {
25         public void run() {
26             ImageExample JFrame = new ImageExample();
27             JFrame.setVisible(true);
28         }
29     });
30 }
31 }

```

## 14.7 배경 이미지 넣기

이번에는 JPanel 에 배경 이미지를 넣는 것을 알아보자. JPanel 은 paint() 메소드를 실행할 때 자신의 paintComponent() 메소드 먼저 호출하고, 자식 컴포넌트의 paint() 메소드를 나중에 호출하도록 되어 있다.

배경 그림은 모든 자식 컴포넌트보다 먼저 드로잉되어 자식 컴포넌트 밑에 깔려야하므로 paintComponent()에서 배경 그림이 드로잉되어야 한다.



### 【BackgroundImageExample.java】 배경 이미지 넣기

```

1 public class BackgroundImageExample extends JFrame {
2     private JTextField txtId;
3     private JButton btnLogin;
4
5     public BackgroundImageExample() {
6         this.setTitle("배경 그림 넣기");
7         this.getContentPane().add(new MyPanel(), BorderLayout.CENTER);
8         this.setSize(200, 270);
9         this.setResizable(false);
10        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
11    }
12
13    public JTextField getTxtId() {
14        if(txtId == null) {

```

```

15     txtId = new JTextField();
16     txtId.setBounds(50, 50, 100, 30);
17 }
18 return txtId;
19 }
20 public JButton getBtnLogin() {
21     if(btnLogin == null) {
22         btnLogin = new JButton("버튼");
23         btnLogin.setBounds(50, 100, 100, 30);
24     }
25     return btnLogin;
26 }
27
28 public class MyPanel extends JPanel {
29     public MyPanel() {
30         setLayout(null);
31         add(getTxtId());
32         add(getBtnLogin());
33     }
34     public void paintComponent(Graphics g) {
35         ImageIcon icon = new ImageIcon(this.getClass().getResource("bg.jpg"));
36         g.drawImage(icon.getImage(), 0, 0, this);
37     }
38 }
39
40 public static void main(String[] args) {
41     SwingUtilities.invokeLater(new Runnable() {
42         public void run() {
43             BackgroundImageExample jFrame = new BackgroundImageExample();
44             jFrame.setVisible(true);
45         }
46     });
47 }
48 }

```



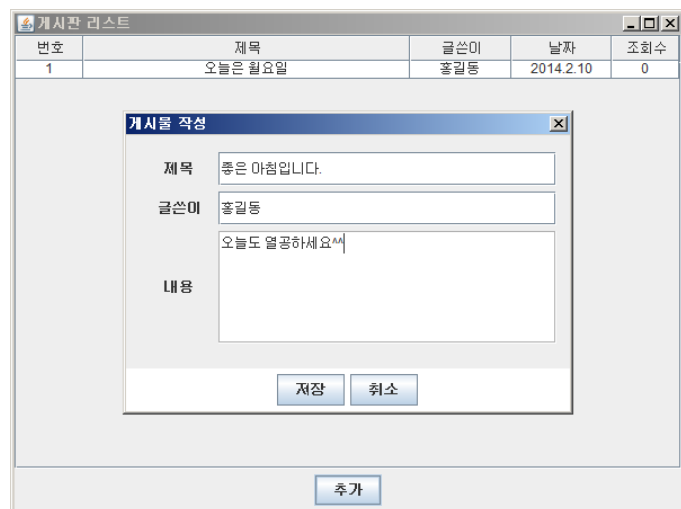


## 【확인문제】

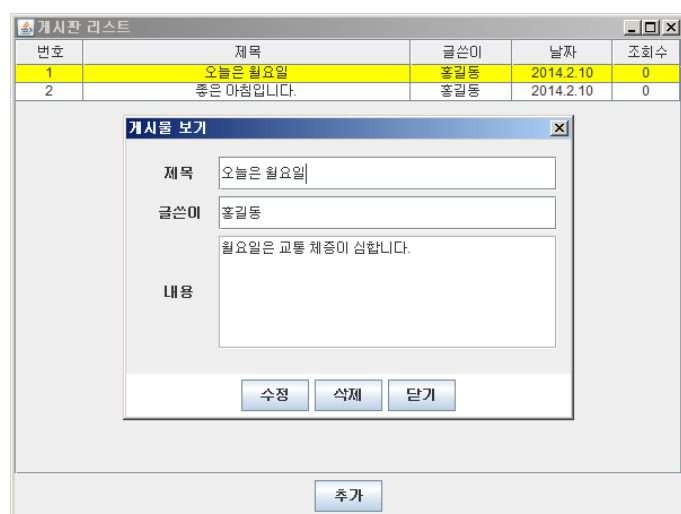
1. 게시판 애플리케이션의 첫 메인 창을 다음과 같이 JFrame 을 사용해서 구성하려고 합니다. 하단의 추가 버튼을 클릭하면 JTable 에 새로운 행이 삽입될 수 있도록 BoardApp.java 를 작성해 보세요.



2. 1 번에서 만든 BoardApp 을 수정하는데, [추가] 버튼을 클릭하면 다음과 같이 게시물 입력 다이얼로그를 띄우도록 합니다. 그리고 입력 다이얼로그에서 제목, 글쓴이, 내용을 입력하고 저장 버튼을 클릭하면 JTable 에 행이 추가되도록 작성해 보세요.

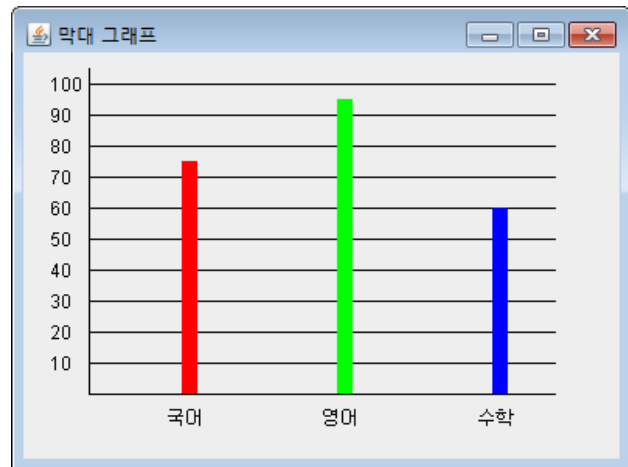


3. 2 번에 이어서 JTable 행을 클릭하면 다음과 같이 게시물 뷰 다이얼로그를 띄우려고 합니다. 내용을 수정하고 [수정] 버튼을 클릭하면 해당 행의 데이터가 수정되게 하고, [삭제] 버튼을 클릭하면 해당 행이 삭제가 되도록 작성해 보세요.



4. 국어, 영어, 수학 점수가 다음 표와 같습니다. JFrame 을 이용해서 막대 그래프로 보여주는 애플리케이션을 작성해보세요.

과목	점수
국어	75
영어	95
수학	60



5. 100 명의 개발자에게 자신이 제일 좋아하는 언어를 조사하였더니 다음과 같은 결과가 나왔습니다. JFrame 을 이용해서 파이 그래프로 그리는 프로그램을 작성해보세요.

언어	좋아하는 사람
VC	20
Java	40
C#	25
VB	15

