Java의 정석

제 13 장

AWT와 애플릿

2010. 1. 14

남궁성 강의

castello@naver.com

1. AWT

(Abstract Window Toolkit)

1.1 AWT(Abstract Window Toolkit)란?

▶ AWT

- GUI프로그래밍(윈도우 프로그래밍)을 위한 도구
- GUI프로그래밍에 필요한 다양한 컴포넌트를 제공한다.
- Java로 구현하지 않고, OS의 컴포넌트를 그대로 사용한다.

▶ Swing

- AWT를 확장한 GUI프로그래밍 도구
- AWT보다 더 많은 종류의 컴포넌트를 제공한다.
- OS의 컴포넌트를 사용하지 않고, 순수한 Java로 구현하였다.

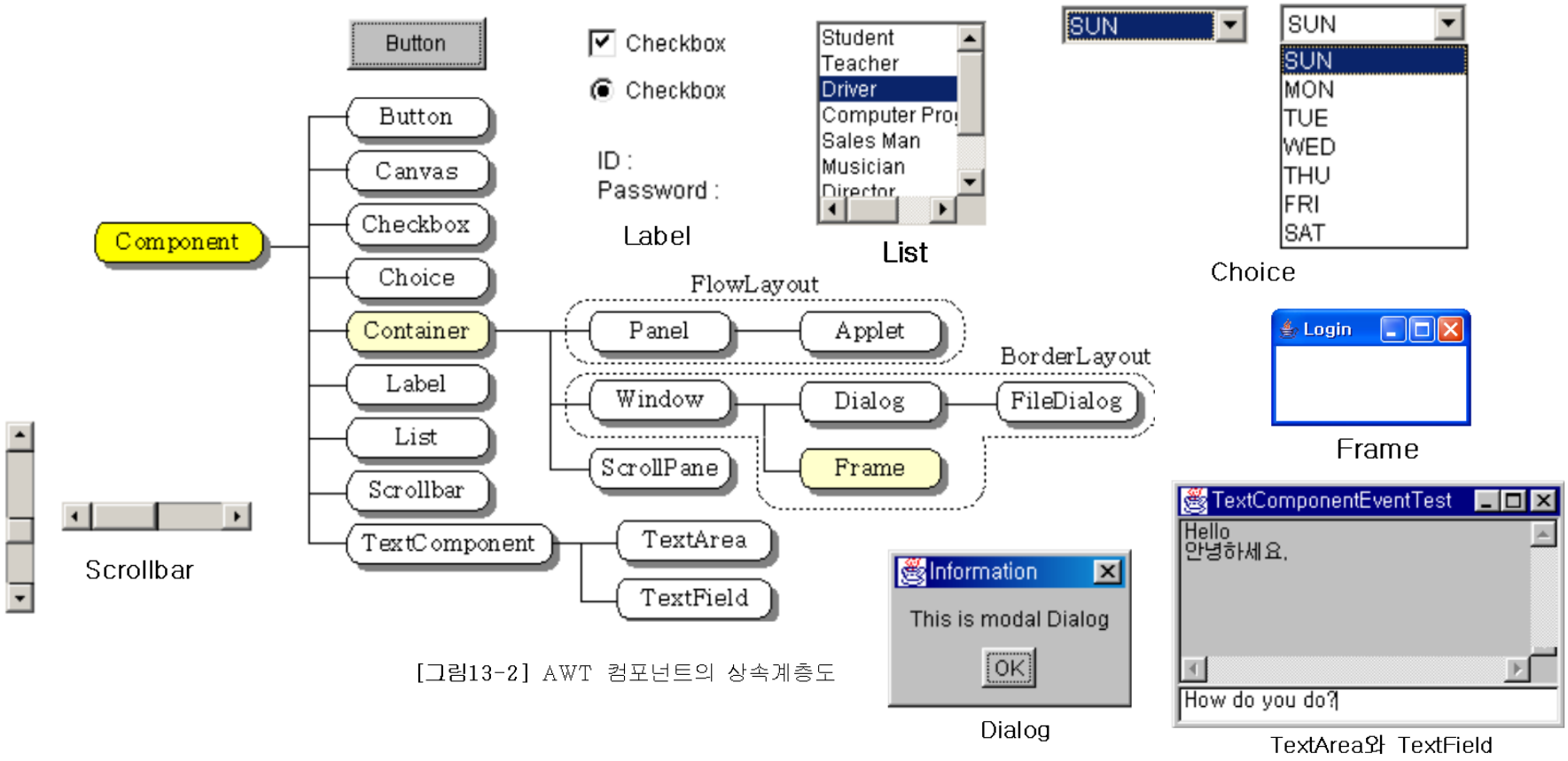
1.2 AWT의 구성(1/3)

- AWT관련 패키지는 모두 'java.awt'로 시작한다.
- 'java.awt'패키지와 'java.awt.event'패키지가 AWT의 핵심이다.

패키지명	설명
java.awt	AWT를 이용한 GUI어플리케이션을 작성하는데 필요한 기본적인 클래스와 컴포넌트를 제공한다.
java.awt.datatransfer	여러 어플리케이션 사이의, 또는 단일 어플리케이션 내에서의 데이터 전송을 구현하는데 필요한 클래스와 인터페이스를 제공한다.
java.awt.dnd	GUI의 장점 중의 하나인 끌어놓기(Drag and Drop)기능을 구현하는데 필요한 클래스들을 제공한다.
java.awt.event	GUI어플리케이션에서 발생하는 이벤트를 처리하는데 필요한 클래스와 인터페이스를 제공한다.
java.awt.font	폰트와 관련된 클래스와 인터페이스를 제공한다.
java.awt.image	이미지를 생성하거나 변경하는데 사용되는 클래스를 제공한다.
java.awt.print	출력에 관련된 클래스와 인터페이스를 제공한다.

1.2 AWT의 구성(2/3)

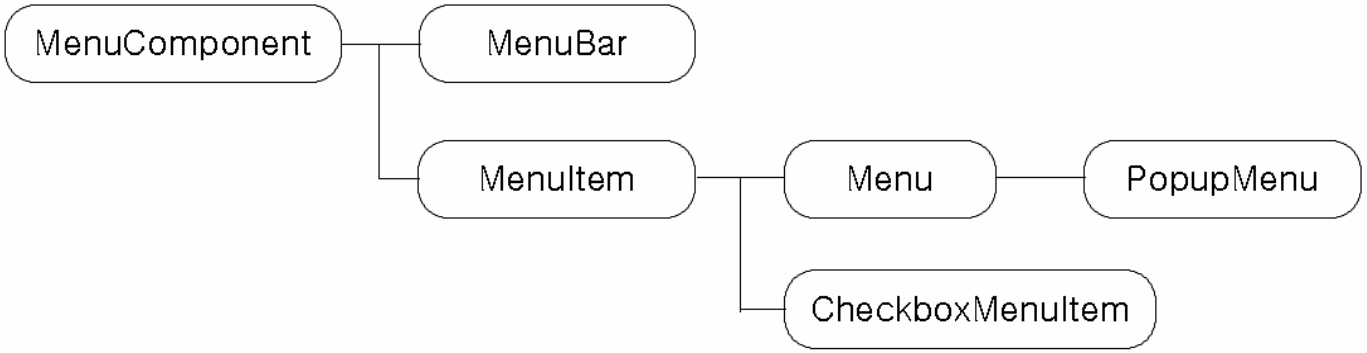
- 모든 AWT컴포넌트의 최고 조상은 java.awt.Component클래스이다. (메뉴관련 컴포넌트 제외)
- Container는 다른 컴포넌트를 담을 수 있는 컴포넌트이다.



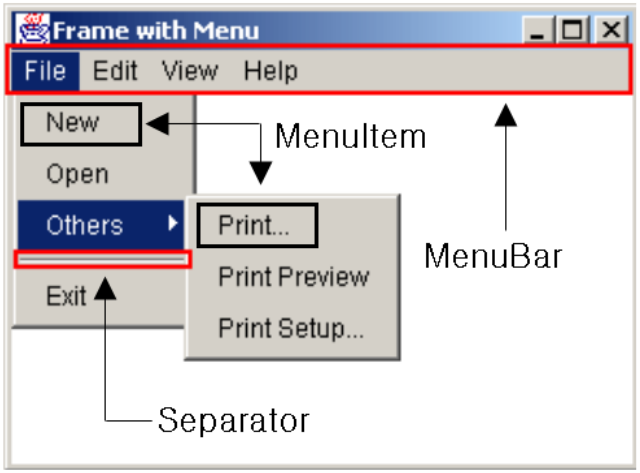
[그림13-2] AWT 컴포넌트의 상속계층도

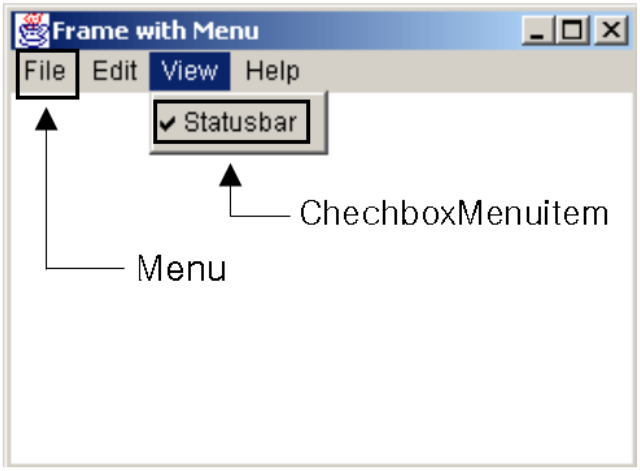
1.2 AWT의 구성(3/3)

- 메뉴관련 컴포넌트의 최고 조상은 java.awt.MenuComponent클래스이다.



[그림13-4] AWT 메뉴컴포넌트의 상속계층도





1.3 컴포넌트(Component)

- 모든 AWT컴포넌트(메뉴관련 컴포넌트 제외)의 최고 조상
- 컴포넌트라면 반드시 있어야 하는 공통적인 메서드들이 정의되어 있다.

메서드	설 명
Color getBackground()	컴포넌트의 배경색을 얻는다.
void setBackground(Color c)	컴포넌트의 배경색을 지정된 색으로 변경한다.
Cursor getCursor()	컴포넌트에 지정된 커서를 얻는다.
void setCursor(Cursor c)	컴포넌트의 커서(마우스포인터)를 지정한다.
Font getFont()	컴포넌트에 지정되어 있는 Font를 얻는다.
void setFont(Font f)	컴포넌트의 Font를 지정한다.
Color getForeground()	컴포넌트의 전경색을 얻는다.
void setForeground(Color c)	컴포넌트의 전경색을 지정한다.
int getHeigtht()	컴포넌트의 높이(Height)를 얻는다.
int getWidth()	컴포넌트의 폭(Width)를 얻는다.
void setBounds(int x, int y, int width, int height)	컴포넌트의 위치(x, y)와 크기(width, height)를 지정한다.
Rectangle getBounds()	컴포넌트의 위치와 크기(Rectangle객체)를 얻는다.
Point getLocation()	컴포넌트의 위치를 얻는다.
void setLocation(int x, int y)	컴포넌트의 위치를 지정한다.
Dimension getSize()	컴포넌트의 크기를 얻는다.
void setSize(int width, int height)	컴포넌트의 크기를 지정한다.
boolean hasFocus()	컴포넌트가 현재 focus를 갖고 있는지 알려준다.
void requestFocus()	컴포넌트가 focus를 갖도록 한다.
void paint(Graphics g)	컴포넌트를 화면에 그린다.
void repaint()	컴포넌트를 화면에 다시 그린다.
void setEnabled(boolean b)	컴포넌트를 사용가능(true)/불가능(false) 하게 한다.
Container getParent()	컴포넌트가 포함되어져 있는 컨테이너(parent)를 얻는다.
void setVisible(boolean b)	컴포넌트가 화면에 보이게(true)/보이지 않게(false) 한다.

1.4 컨테이너(Container)

- 다른 컴포넌트를 포함할 수 있는 컴포넌트. Container클래스와 그 자손들

1. 독립적인 컨테이너 – 독립적으로 사용될 수 있으며, 다른 컴포넌트나 종속적인 컨테이너를 포함할 수 있다.

컨테이너	설 명
Frame	가장 일반적인 컨테이너로 윈도우와 모양이 같다. titlebar와 크기조절버튼, 닫기버튼을 가지고 있다. 그리고 메뉴를 추가할 수 있다.
Window	Frame의 조상이며, 경계선, titlebar, 크기조절버튼, 닫기 버튼이 없으며, 메뉴도 추가할 수 없다. 단지 컴포넌트를 담을 수 있는 평면공간만을 갖는다.
Dialog	Frame처럼, titlebar와 닫기버튼을 갖고 있지만, 메뉴는 가질 수 없으며 기본적으로 크기를 변경할 수 없다. 주로 프로그램사용자에게 메시지를 보여 주거나, 응답을 받는데 사용한다.

2. 종속적인 컨테이너 – 독립적으로 사용될 수 없으며, 다른 컨테이너에 포함되어야 한다. 다른 컴포넌트나 종속적인 컨테이너를 포함할 수 있다.

컨테이너	설 명
Panel	평면공간으로 Frame과 같이 여러 컴포넌트를 담을 수 있으나 단독적으로 사용될 수는 없다.
ScrollPane	Panel과 같은 평면공간이지만, Panel과는 달리 단 하나의 컴포넌트만 포함할 수 있으며 자신보다 큰 컴포넌트가 포함되면 스크롤바가 자동적으로 나타난다.

1.4 컨테이너(Container) – 주요 메서드

- add()를 사용해서 컴포넌트를 컨테이너에 담는다.
- 컨테이너에 담긴 컴포넌트는 컨테이너의 전경색, 배경색, 폰트 등의 설정을 그대로 따르게 된다.(나중에 변경가능)

메서드	설 명
Component[] getComponents()	컨테이너에 포함되어 있는 모든 컴포넌트를 얻는다.
Component getComponent(int n)	컨테이너에 n번째로 추가된 컴포넌트를 얻는다.
Component getComponentAt(int x, int y)	컨테이너의 지정된 위치(x, y)에 있는 컴포넌트를 얻는다.
Component add(Component comp)	컨테이너에 컴포넌트를 추가한다.
void remove(Component comp)	컨테이너에서 지정된 컴포넌트를 제거한다.
Insets getInsets()	컨테이너의 경계의 크기를 알 수 있는 Inset객체를 얻는다.
LayoutManager getLayout()	컨테이너에 설정되어 있는 LayoutManager를 얻는다.
void setLayout(LayoutManager mgr)	컨테이너에 LayoutManager를 설정한다.

2. AWT의 주요 컴포넌트

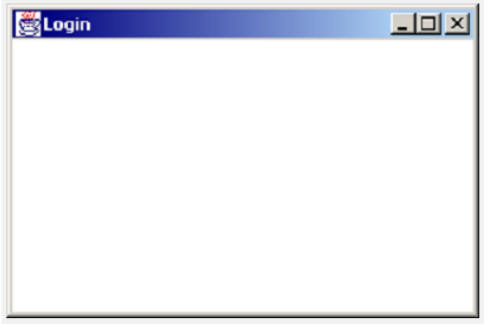
2.1 Frame

- titlebar와 최대화, 최소화, 닫기 버튼을 가진 윈도우(컨테이너)

메서드 또는 생성자	설 명
Frame(String title)	Frame을 생성한다. title - Frame의 titlebar에 보여 질 text
String getTitle()	titlebar에 있는 text를 얻는다.
void setTitle(String title)	titlebar의 text를 변경한다.
void setState(int state)	Frame의 상태를 변경할 수 있으며, state에는 아래의 두 가지 값중 하나를 사용할 수 있다. Frame.ICONIFIED - Frame을 최소화 상태가 되게 한다. Frame.NORMAL - Frame을 정상적인 상태(최소화 이전상태)가 되게 한다.
int getState()	Frame의 현재 상태를 알 수 있다.
void setResizable(boolean resizable)	Frame의 크기를 변경가능 또는 불가능하게 한다. (resizable의 값이 false로 하면 사용자가 Frame의 크기를 변경할 수 없다.)

```
import java.awt.*;

class FrameTest {
    public static void main(String args[]) {
        Frame f = new Frame("Login");     // Frame객체를 생성한다.
        f.setSize(300, 200);              // Frame의 크기를 설정한다.
        f.setVisible(true);               // 생성한 Frame을 화면에 보이도록
    }
}
```

[실행결과]



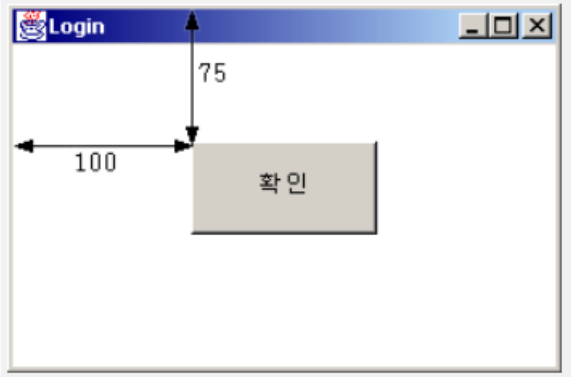
2.2 Button

- 사용자가 클릭했을 때 어떤 작업이 수행되도록 할 때 사용하는 컴포넌트

메서드 또는 생성자	설 명
Button(String label)	지정된 label을 가진 Button을 생성한다. label - Button위에 나타날 text
String getLabel()	Button에 나타나있는 text를 얻는다.
void setLabel(String label)	Button에 나타나있는 text를 변경한다.

```
import java.awt.*;

class ButtonTest2 {
    public static void main(String args[]) {
        Frame f = new Frame("Login");
        f.setSize(300, 200);
        f.setLayout(null);          // 레이아웃 매니저의 설정을 해제

        Button b = new Button("확 인");
        b.setSize(100, 50);         // Button의 크기를 설정한다.
        b.setLocation(100, 75);     // Frame내에서의 Button의 위치를 설정한다.

        f.add(b);
        f.setVisible(true);
    }
}
```

[실행결과]



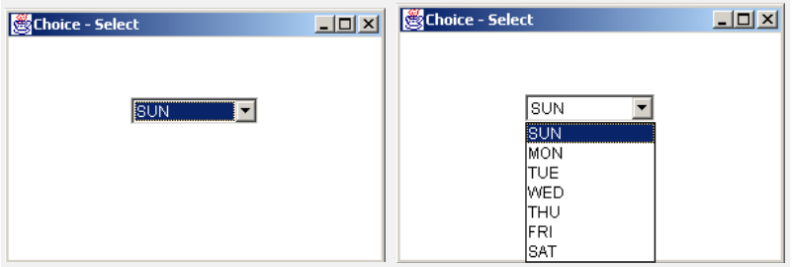
2.3 Choice

- 여러 item 중에서 하나를 선택할 수 있게 해주는 컴포넌트

메서드 또는 생성자	설 명
void add(String item)	Choice에 item을 추가한다.
void remove(String item)	Choice에서 item을 제거한다.
void remove(int index)	지정된 순서에 있는 item을 제거한다. index는 0부터 시작
void removeAll()	Choice의 모든 item을 제거한다.
void insert(String item, int index)	지정된 순서에 item을 추가한다. index는 0부터 시작
String getItem(int index)	지정된 순서의 item을 얻는다. index는 0부터 시작
int getItemCount()	현재 Choice에 추가되어 있는 item이 몇 개인지 알려준다.
int getSelectedIndex()	현재 선택되어져 있는 item의 index값을 얻는다.
String getSelectedItem()	현재 선택되어져 있는 item을 얻는다.

```
public static void main(String args[]) {
      Frame f = new Frame("Choice - Select");
      f.setSize(300, 200);
      f.setLayout(null);

      Choice day = new Choice();  // Choice객체를 생성한 다음
      day.add("SUN");             // Choice의 목록에 나타날 값들을 추가한다.
      day.add("MON");       day.add("TUE");
      day.add("WED");       day.add("THU");
      day.add("FRI");       day.add("SAT");

      day.setSize(100, 50);
      day.setLocation(100, 70);

      f.add(day);
      f.setVisible(true);
}
```

[실행결과]



2.4 List - 메서드

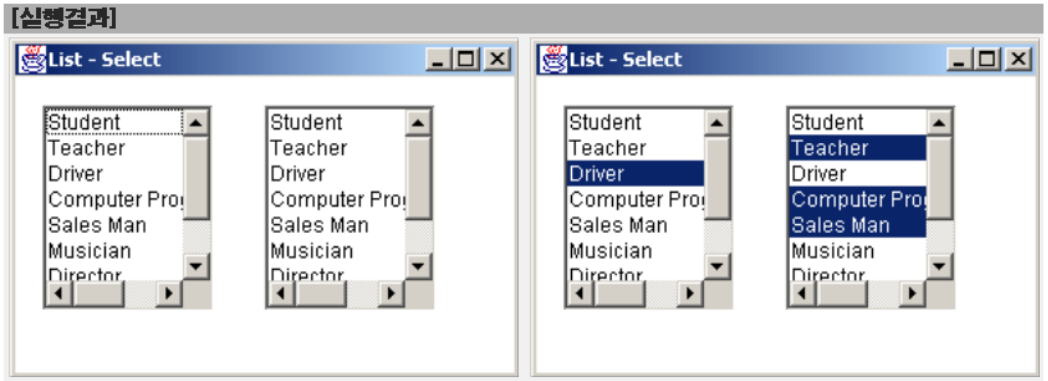
- 여러 item 중에서 하나를 선택할 수 있게 해주는 컴포넌트

메서드 또는 생성자	설 명
List(int rows, boolean multipleMode)	rows - 몇 줄짜리 크기의 List를 보여줄 것인지 지정한다. multipleMode - List목록 다중선택이 가능하도록 할 것인가를 지정할 수 있다. true로 하면 여러 개의 item을 선택할 수 있다.
List(int rows)	List에 보여줄 item 수만 지정한다. multipleMode의 값은 false로 지정되어 하나의 목록만 선택가능하다.
List()	rows의 값은 기본값인 4로 지정되고, multipleMode의 값은 false가 되어 하나의 item만 선택가능하다.
void add(String item)	item을 List에 추가한다.
void add(String item, int index)	지정된 위치(index)에 item을 추가한다.
void replaceItem(String newValue, int index)	지정된 위치(index)의 item을 새로운 item(newValue)로 바꾼다.
void remove(String item)	List에서 해당 item을 제거한다.
void remove(int index)	index - 지정된 위치에 있는 item을 제거한다.
void removeAll()	List의 모든 item을 제거한다.
int getRows()	List에 scroll없이 볼 수 있는 item의 수를 얻는다.
String getItem(int index)	index - 지정된 위치에 있는 item을 얻는다.
String[] getItems()	List에 있는 모든 item을 얻는다.
int getItemCount()	List에 있는 item이 모두 몇 개인지 알려준다.(getRows()와 비교해볼 것)
void select(int index)	지정된 위치에 있는 item을 선택한다.
void deselect(int index)	지정된 위치에 있는 item을 선택해제한다.
int getSelectedIndex()	현재 선택되어 있는 item의 index값을 얻는다.
int[] getSelectedIndexes()	현재 선택되어 있는 item들의 index값을 얻는다.(List의 multipleMode가 true인 경우)
String getSelectedItem()	현재 선택되어 있는 item을 얻는다.
String[] getSelectedItems()	현재 선택되어 있는 item들을 얻는다.(multipleMode가 true인 경우)
boolean isIndexSelected(int index)	지정된 위치의 item이 선택되어있는지 알려준다.
void setMultipleMode(boolean b)	List를 multipleMode로 할 것인지 결정한다.b - true면, List를 multipleMode로 설정한다.

2.4 List - 예제

```
public static void main(String args[]) {
      Frame f = new Frame("List - Select");
      f.setSize(300, 200);
      f.setLayout(null);

      List selectOne = new List(6);     // 6개 목록을 보여줄 수 있는 크기의 List를 만든다.
      selectOne.setLocation(20,40);
      selectOne.setSize(100, 120);
      selectOne.add("Student");
      selectOne.add("Teacher");
      selectOne.add("Driver");
      selectOne.add("Computer Programmer");
      selectOne.add("Sales Man");
      selectOne.add("Musician");
      selectOne.add("Director");

      // 생성자의 두번째 인자값을 true로 설정해서 List의 목록에서 여러 개를 선택할 수 있게 했다.
      List selectMany = new List(6, true);
      selectMany.setLocation(150, 40);
      selectMany.setSize(100,120);
      selectMany.add("Student");
      selectMany.add("Teacher");
      // ... 내용 중간생략...

      f.add(selectOne);
      f.add(selectMany);
      f.setVisible(true);
}
```

[실행결과]



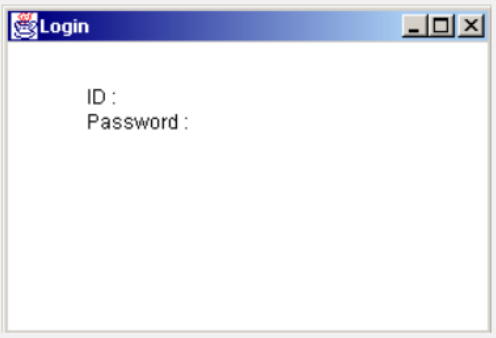
2.5 Label

- 화면에 텍스트를 표시하는데 사용되는 컴포넌트

메서드 또는 생성자	설 명
Label(String text, int alignment)	text - 화면에 나타낼 글(text)을 String으로 넣는다. alignment -text의 정렬방식을 지정한다. Label.LEFT, Label.CENTER, Label.RIGHT 중 하나를 사용.
Label(String text)	text - 화면에 나타낼 text를 지정한다. alignment의 기본값인 Label.LEFT로 설정된다.
String getText()	Label의 text를 얻어온다.
void setText(String text)	Label의 text를 주어진 값으로 변경한다.
void setAlignment(int alignment)	Label의 text 정렬을 지정한다. Label.LEFT, Label.CENTER, Label.RIGHT 중 하나를 사용.

```
public static void main(String args[]) {
      Frame f = new Frame("Login");
      f.setSize(300, 200);
      f.setLayout(null);

      Label id = new Label("ID :");      // Label을 생성하고 크기와 위치를 지정한다
      id.setBounds(50, 50, 30, 10);      // 50, 50위치에 크기가 가로 30, 세로 10

      Label pwd = new Label("Password :");
      pwd.setBounds(50, 65, 100, 10);

      f.add(id);                         // 생성한 Label을 Frame에 포함시킨다.
      f.add(pwd);
      f.setVisible(true);
}
```

[실행결과]



2.6 Checkbox - 메서드

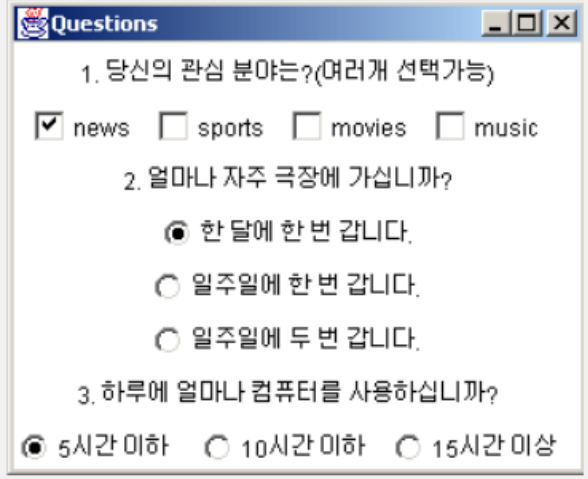
- '선택/비선택'을 표현하는데 사용되는 컴포넌트.
- CheckboxGroup을 사용하면 '4지선다'와 같이 여러 값 중의 하나를 선택하게 할 수 있다.

메서드 또는 생성자	설 명
Checkbox(String text, boolean state)	text - Checkbox와 함께 보여질 text를 지정한다. state - true이면 Checkbox가 선택된 상태로 생성되고, false이면 Checkbox가 선택해제된 상태로 생성된다.
Checkbox(String text)	Checkbox와 함께 보여질 text를 지정한다. Checkbox는 선택해제된 상태로 생성된다.
Checkbox()	text없이 Checkbox만 나타나고, Checkbox는 선택해제된 상태로 생성된다.
Checkbox(String text, CheckboxGroup group, boolean state)	group - CheckboxGroup객체의 참조. CheckboxGroup을 이용해서 radio button으로 만든다.
String getLabel()	Checkbox의 label을 얻는다.
void setLabel(String label)	Checkbox의 label을 주어진 값으로 변경한다.
boolean getState()	Checkbox의 상태를 얻는다. 결과값이 true면 체크되어있는 상태이다.
void setState(boolean state)	Checkbox의 상태를 설정한다. state값을 true로 하면 Checkbox가 체크되어 있는 상태가 되도록 한다.

2.6 Checkbox - 예제

```
public static void main(String args[]) {
      Frame f = new Frame("Questions");
      f.setSize(305, 250);
      // Frame의 LayoutManager를 FlowLayout으로 설정한다.
      f.setLayout(new FlowLayout());

      Label q1 = new Label("1. 당신의 관심 분야는?(여러개 선택가능)");
      Checkbox news = new Checkbox("news", true); // 선택된 상태로 생성
      Checkbox sports = new Checkbox("sports");
      Checkbox movies = new Checkbox("movies");
      Checkbox music = new Checkbox("music");

      f.add(q1); f.add(news); f.add(sports); f.add(movies); f.add(music);

      Label q2 = new Label("2. 얼마나 자주 극장에 가십니까?");
      CheckboxGroup group1 = new CheckboxGroup();
      Checkbox movie1 = new Checkbox("한 달에 한 번 갑니다.", group1, true);
      Checkbox movie2 = new Checkbox("일주일에 한 번 갑니다.", group1, false);
      Checkbox movie3 = new Checkbox("일주일에 두 번 갑니다.", group1, false);

      f.add(q2); f.add(movie1); f.add(movie2); f.add(movie3);

      Label q3 = new Label("3. 하루에 얼마나 컴퓨터를 사용하십니까?");
      CheckboxGroup group2 = new CheckboxGroup();
      Checkbox com1 = new Checkbox("5시간 이하 ", group2, true);
      Checkbox com2 = new Checkbox("10시간 이하", group2, false);
      Checkbox com3 = new Checkbox("15시간 이상", group2, false);

      f.add(q3); f.add(com1); f.add(com2); f.add(com3);
      f.setVisible(true);
}
```

[실행결과]



2.7 TextField - 메서드

- 사용자로부터 데이터를 자유롭게 입력받을 수 있는 컴포넌트
- 한 줄만 입력할 수 있어서 비교적 길지 않은 값의 입력에 사용된다.

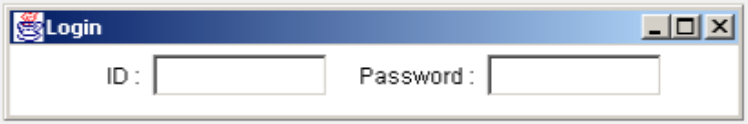
메서드 또는 생성자	설 명
TextField(String text, int col)	text - TextField에 보여질 text를 지정한다. col - 입력 받을 글자의 수를 적는다. col의 값에 따라서 TextField의 크기가 결정된다.
TextField(int col)	col - 입력 받을 글자의 수를 적는다.
TextField(String text)	text - TextField에 보여질 text를 지정한다.
void setEchoChar(char c)	지정된 문자를 EchoChar로 한다. (비밀번호 입력에 주로 사용됨)
int getColumns()	TextField의 칼럼 수를 얻는다.
void setText(String t)*	지정된 문자열을 TextField의 text로 한다.
String getText()*	TextField의 text를 얻는다.
void select(int selectionStart, int selectionEnd)*	selectionStart부터 selectionEnd까지의 text를 선택(하이라이트)한다.
void selectAll()*	TextField의 모든 text를 선택된 상태가 되도록 한다.
String getSelectedText()*	TextField의 text중 선택되어진 부분을 얻는다.
void setEditable(boolean b)*	TextField의 text를 편집가능(true)/불가능(false) 하도록 한다.

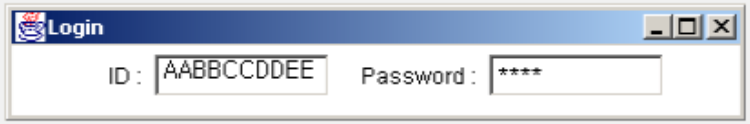
[참고] '*' 표시가 되어있는 것은 TextField의 조상인 TextComponent로부터 상속받은 것이다.

2.7 TextField - 예제

```
import java.awt.*;

class TextFieldTest {
    public static void main(String args[]) {
        Frame f = new Frame("Login");
        f.setSize(400, 65);
        f.setLayout(new FlowLayout()); // LayoutManager를 FlowLayout으로 한다.

        Label lid = new Label("ID :", Label.RIGHT); // 정렬을 오른쪽으로.
        Label lpwd = new Label("Password :", Label.RIGHT);

        TextField id = new TextField(10);//약 10개의 글자를 입력할 수 있는 TextField 생성
        TextField pwd = new TextField(10);
        pwd.setEchoChar('*');      // 입력한 값 대신 '*'가 보이도록 한다.

        f.add(lid);      // 생성한 컴포넌트들을 Frame에 포함시킨다.
        f.add(id);
        f.add(lpwd);
        f.add(pwd);
        f.setVisible(true);
    }
}
```

[실행결과]





2.8 TextArea - 메서드

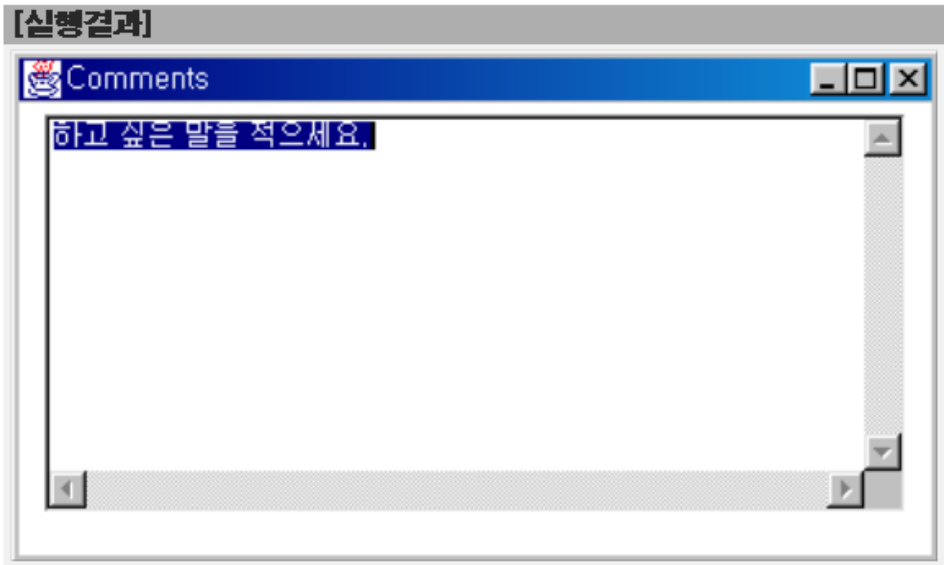
- 여러 줄의 텍스트를 입력하거나 보여줄 수 있는 편집가능한 컴포넌트

메서드 또는 생성자	설 명
TextArea(String text, int row, int col, int scrollbar)	text - TextArea에 보여질 text를 지정한다. row - TextArea의 줄(row) 수를 지정한다. col - TextArea의 열(column) 수를 적는다. scrollbar - TextArea에 사용할 scrollbar의 종류와 사용여부 지정. (아래의 값들 중 하나) TextArea.SCROLLBARS_BOTH,TextArea.SCROLLBARS_NONE TextArea.SCROLLBARS_HORIZONTAL_ONLY, TextArea.SCROLLBARS_VERTICAL_ONLY
TextArea(int row, int col)	row - TextArea의 줄(row) 수를 지정한다. col - TextArea의 열(column) 수를 적는다. 아무런 text도 없는 빈 TextArea를 생성한다. scrollbar는 수평(HORIZONTAL), 수직(VERTICAL) 모두 갖는다.
int getRows()	TextArea의 행(row)의 개수를 얻는다.
int getColumns()	TextArea의 열(column)의 개수를 얻는다.
void setRows(int rows)	지정된 값으로 TextArea의 행(row)의 개수를 설정한다.
void setColumns(int columns)	지정된 값으로 TextArea의 열(column)의 개수를 설정한다.
void append(String str)	TextArea에 있는 text의 맨 마지막에 문자열을 덧붙인다.
void insert(String str, int pos)	TextArea에 있는 text의 지정된 위치에 문자열을 넣는다.
void replaceRange(String str, int start, int end)	TextArea에 있는 text의 start부터 end범위에 있는 문자열을 str에 지정된 값으로 변경한다.
void setText(String t)*	지정된 문자열을 TextArea의 text로 한다.
String getText()*	TextArea의 text를 얻는다.
void select(int selectionStart, int selectionEnd)*	selectionStart부터 selectionEnd까지의 text가 선택되게 한다.
void selectAll()*	TextArea의 모든 text를 선택되게 한다.
String getSelectedText()*	TextArea의 text중 선택된 부분을 얻는다.
void setEditable(boolean b)*	TextArea의 text를 편집가능(true)/불가능(false) 하도록 한다.

2.8 TextArea - 예제

```
import java.awt.*;

class TextAreaTest {
    public static void main(String args[]) {
        Frame f = new Frame("Comments");
        f.setSize(400, 220);
        f.setLayout(new FlowLayout());

        TextArea comments = new TextArea("하고 싶은 말을 적으세요.", 10, 50);

        f.add(comments);
        comments.selectAll();  // TextArea의 text 전체가 선택 되도록 한다.
        f.setVisible(true);
    }
}
```

[실행결과]



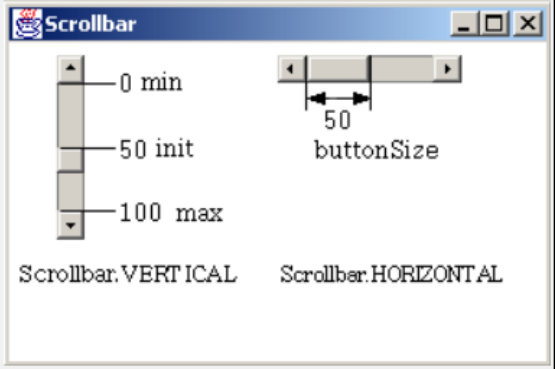
2.9 Scrollbar

- 사용자가 정해진 범위에서 값을 조절할 수 있게 해주는 컴포넌트

메서드 또는 생성자	설 명
Scrollbar() Scrollbar(int orientation) Scrollbar(int orientation, int value, int visible, int min, int max)	orientation - Scrollbar의 종류를 지정해준다. Scrollbar.VERTICAL, Scrollbar.HORIZONTAL 중의 하나 value - Scrollbar의 초기값 visible - Scroll버튼(bubble)의 크기 min - Scrollbar가 가질 수 있는 최소값 max - Scrollbar가 가질 수 있는 최대값
int getValue()	현재 설정된 Scrollbar의 값을 얻어온다.
void setValue(int newValue)	Scrollbar의 값을 지정된 값(newValue)으로 설정한다.

```
public static void main(String args[]) {
      Frame f = new Frame("Scrollbar");
      f.setSize(300, 200);
      f.setLayout(null);

      Scrollbar hor = new Scrollbar(Scrollbar.HORIZONTAL, 0, 50, 0, 100);
      hor.setSize(100, 15);
      hor.setLocation(60, 30);
      Scrollbar ver = new Scrollbar(Scrollbar.VERTICAL, 50, 20, 0, 100);
      ver.setSize(15, 100);
      ver.setLocation(30, 30);

      f.add(hor);
      f.add(ver);
      f.setVisible(true);
}
```

[실행결과]

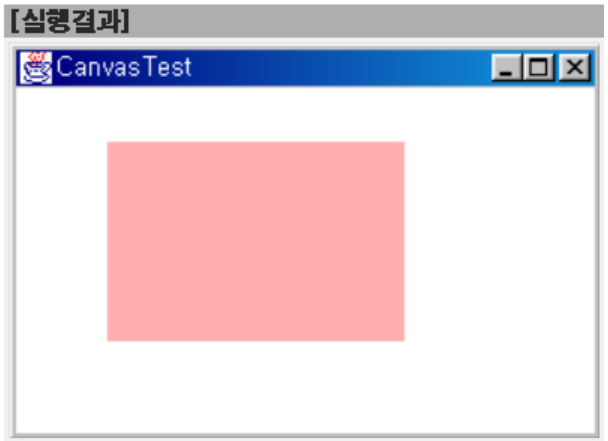


2.10 Canvas

- 주로 그림을 그리거나 이미지를 위한 공간으로 사용되는 컴포넌트

```
import java.awt.*;

class CanvasTest {
    public static void main(String args[]) {
        Frame f = new Frame("CanvasTest");
        f.setSize(300, 200);
        f.setLayout(null);          // Frame의 Layout Manager설정을 해제한다.

        Canvas c = new Canvas();
        c.setBackground(Color.pink);    // Canvas의 배경을 분홍색(pink)으로 한다.
        c.setBounds(50, 50, 150, 100);

        f.add(c);          // Canvas을 Frame에 포함시킨다.
        f.setVisible(true);
    }
}
```

[실행결과]

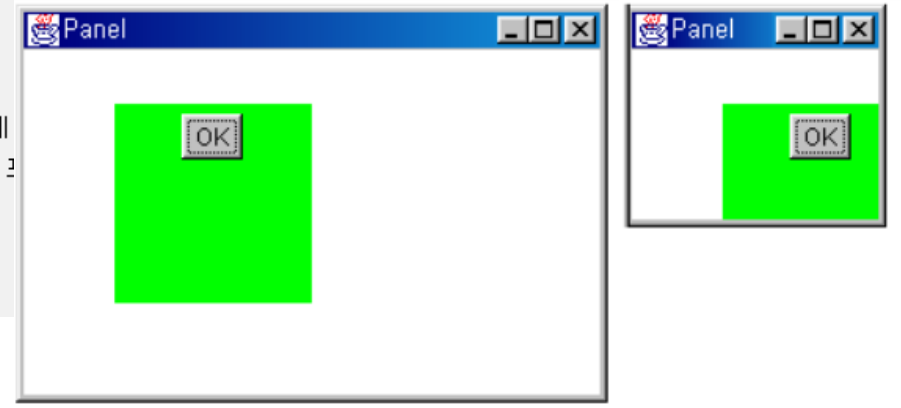


2.11 Panel

- 빈 평면 공간만 가지고 있는 종속적인 컨테이너.
- Panel안에 Panel을 넣을 수 있어서 컴포넌트의 다양한 배치에 유용하다.

```
import java.awt.*;

class PanelTest {
    public static void main(String args[]) {
        Frame f = new Frame("Panel");
        f.setSize(300, 200);
        f.setLayout(null);          // Frame의 Layout Manager를 사용하지 않도록 한다.

        Panel p = new Panel();
        p.setBackground(Color.green);   // Panel의 배경을 녹색으로 한다.
        p.setSize(100, 100);
        p.setLocation(50, 50);

        Button ok = new Button("OK");

        p.add(ok);      // Button을 Panel에
        f.add(p);       // Panel을 Frame에
        f.setVisible(true);
    }
}
```

[실행결과]



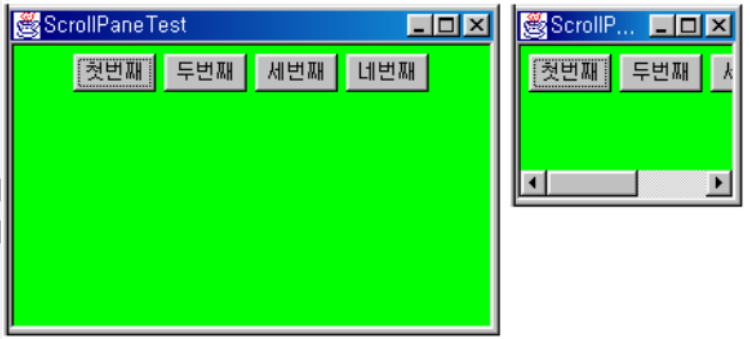
2.12 ScrollPane

- 단 하나의 컴포넌트만 포함할 수 있는 종속적인 컨테이너.
- 제한된 공간에서 큰 컴포넌트를 화면에 보여줄 때 사용한다.

메서드 또는 생성자	설 명
ScrollPane(int scrollbarDisplayPolicy)	srollbarDisplayPolicy - 아래 값 중의 하나를 지정한다. SCROLLBARS_ALWAYS - 스크롤바가 항상 보이게 한다. SCROLLBARS_AS_NEEDED - 필요할 때만 보이게 한다. SCROLLBARS_NEVER - 항상 보이지 않도록 한다.
ScrollPane()	ScrollPane의 객체를 생성한다.

```
public static void main(String args[]) {
      Frame f = new Frame("ScrollPaneTest");
      f.setSize(300, 200);

      ScrollPane sp = new ScrollPane();
      Panel p = new Panel();
      p.setBackground(Color.green);   // Panel의 배경을 green으로 한다.
      p.add(new Button("첫번째"));      // Button을
      p.add(new Button("두번째"));
      p.add(new Button("세번째"));
      p.add(new Button("네번째"));

      sp.add(p);       // Panel을 ScrollPane에 포함시
      f.add(sp);       // ScrollPane을 Frame에 포함시
      f.setVisible(true);
}
```

[실행결과]



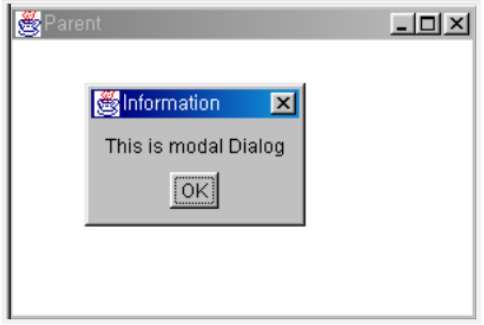
2.13 Dialog

- Frame과 같은 독립적인 컨테이너. titlebar와 닫기버튼을 가지고 있다.
- 주로 화면에 메시지창을 보여주거나 사용자로부터 입력을 받을 때 사용

메서드 또는 생성자	설 명
Dialog(Frame parent, String title, boolean modal)	parent - 어떤 Frame에 속한 것인지 지정. title - Dialog의 titlebar에 나타날 text지정 modal - Dialog를 modal(필수응답)로 할 것 인지를 결정.
Dialog(Frame parent, String title)	modal을 지정하지 않으면, false가 된다.
void show()	Dialog가 화면에 나타나도록 한다.
void hide()	Dialog가 화면에 보이지 않도록 한다.(메모리에는 남아있다.)
void dispose()	Dialog를 닫는다.(화면에 안 보이게 한 후, 메모리에서 제거)
String getTitle()	Dialog의 titlebar에 나타난 text를 얻는다.
void setModal(boolean b)	Dialog를 modal(true) 또는 modaless(false)로 한다.
void setResizable(boolean resizable)	사용자에 의해서 Dialog의 크기가 변경가능/불가능 하도록 한다.(기본적으로 Dialog는 크기변경이 불가)

```
public static void main(String args[]) {
      Frame f = new Frame("Parent");
      f.setSize(300, 200);
      // parent Frame을 f로 하고, modal을 true로 해서 필수응답 Dialog로 함.
      Dialog info = new Dialog(f, "Information", true);
      info.setSize(140, 90);
      info.setLocation(50, 50); // parent Frame이 아닌, 화면이 위치의 기준이 된다.
      info.setLayout(new FlowLayout());

      Label msg = new Label("This is modal Dialog", Label.CENTER);
      Button ok = new Button("OK");
      info.add(msg); info.add(ok);

      f.setVisible(true);
      info.setVisible(true);     // Dialog를 화면에 보이게 한다.
}
```

[실행결과]

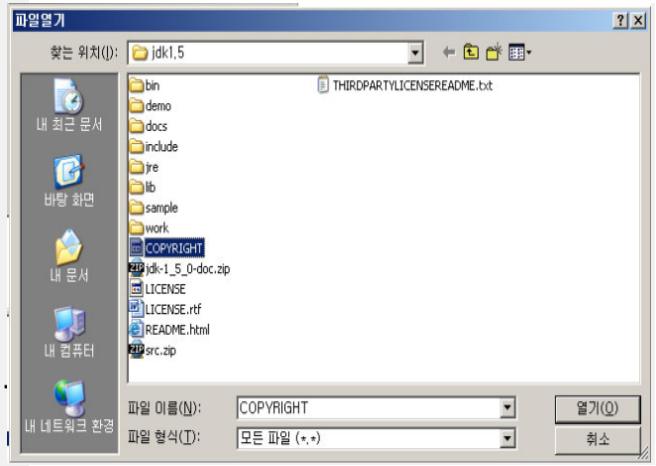


2.14 FileDialog

- 파일을 열거나 저장할 때 사용되는 Dialog

메서드 또는 생성자	설 명
FileDialog(Frame parent, String title, int mode)	parent - 어떤 Frame에 속한 것인지 지정. title - FileDialog의 titlebar에 나타날 text지정 mode - FileDialog.LOAD, FileDialog.SAVE 중 하나 선택
FileDialog(Frame parent, String title)	mode를 생략하면, 디폴드로 FileDialog.LOAD가 사용된다.
String getFile()	FileDialog에 의해 선택된 파일의 이름을 얻는다.
String getDirectory()	FileDialog에 의해 선택된 파일의 경로(path)를 얻는다.
void setFile(String file)	FileDialog에 지정된 파일을 설정한다.
void setDirectory(String dir)	FileDialog에 지정된 디렉토리를 설정한다.

```
public static void main(String args[]) {
      Frame f = new Frame("Parent");
      f.setSize(300, 200);

      FileDialog fileOpen = new FileDialog(f, "파일열기", FileDialog.LOAD);

      f.setVisible(true);
      fileOpen.setDirectory("c:\\jdk1.5");
      fileOpen.setVisible(true);

      //파일을 선택한 다음, FileDialog의 열기버튼을 누르면,
      // getFile()과 getDirectory()를 이용해서 파일이름과 위치한 디렉토리를 얻을 수 있다.
      System.out.println(fileOpen.getDirectory() + fileOpen.getFile());
}
```



2.15 Font

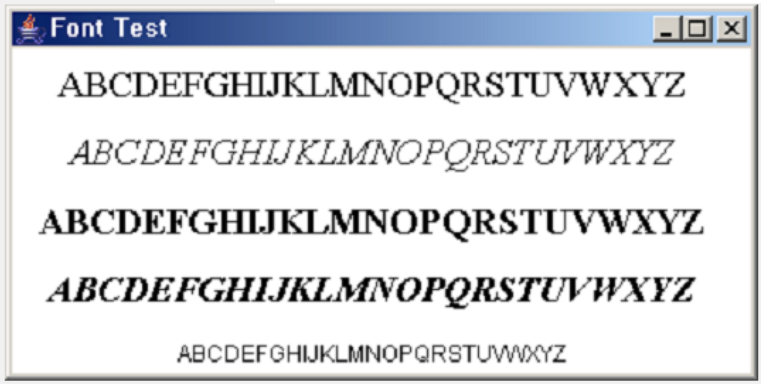
- Component클래스의 setFont(Font f)를 사용하면, 폰트를 변경할 수 있다.

메서드 또는 생성자	설 명
Font(String name, int style, int size)	name - 사용할 폰트의 이름. (예: "Serif") style - 폰트의 스타일을 지정한다. 보통체(Font.PLAIN), 굵은체(Font.BOLD), 기울임체(Font.ITALIC), 굵은 기울임체(Font.BOLD +Font.ITALIC) 이들 중 하나를 선택. size - 폰트의 크기

```
Frame f = new Frame("Font Test");
String abc = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

// A 부터 Z를 내용으로 갖는 Label들을 생성한다.
Label abc1 = new Label(abc);          Label abc2 = new Label(abc);
Label abc3 = new Label(abc);          Label abc4 = new Label(abc);
Label abc5 = new Label(abc);

// Serif체이며, 크기가 20인 Font
Font f1 = new Font("Serif", Font.PLAIN, 20);      // 보통체
Font f2 = new Font("Serif", Font.ITALIC, 20);     // 기울임체
Font f3 = new Font("Serif", Font.BOLD, 20);       // 굵은체
Font f4 = new Font("Serif", Font.BOLD+Font.ITALIC, 20); // 굵은 기울임체

// Label에 새로운 Font를 적용한다.
abc1.setFont(f1);          abc2.setFont(f2);
abc3.setFont(f3);          abc4.setFont(f4);

f.setLayout(new FlowLayout());
f.add(abc1);          f.add(abc2);
f.add(abc3);          f.add(abc4);
f.add(abc5);

f.setSize(400, 200);
f.setVisible(true);
```

```
public static final int BOLD    = 1;
```

- Serif
- SansSerif
- Dialog
- DialogInput
- Monospaced



2.16 Color

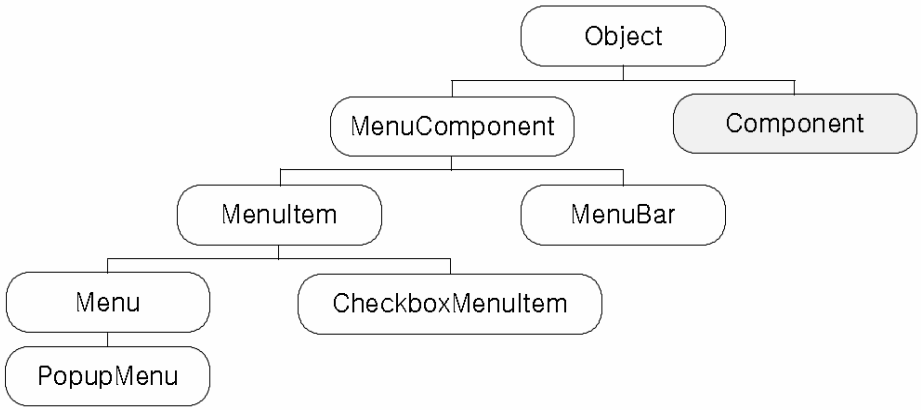
- 색의 표현에 사용되는 클래스. RGB값이나 미리 정의된 색 사용가능.

메서드 또는 생성자	설 명
Color(int r, int g, int b)	r - red, g - green, b - blue r, g, b 모두 0~255사이의 정수값을 갖는다.
Color(float r, float g, float b)	r - red, g - green, b - blue r, g, b 모두 0.0~1.0사이의 실수값을 갖는다.
Color(int r, int g, int b, int a)	a - alpha값(불투명도)으로 0~255사이의 정수값
Color(float r, float g, float b, float a)	a - alpha값(불투명도)으로 0.0~1.0사이의 실수값

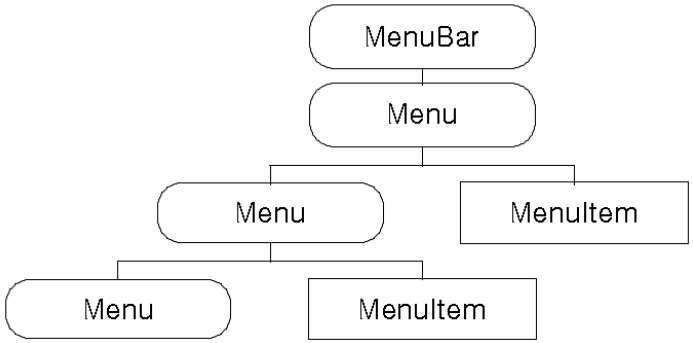
```
Frame f = new Frame("Color Test");
f.setLayout(new GridLayout(14, 2));
Panel p1 = new Panel();     p1.setBackground(Color.black);
Panel p2 = new Panel();     p2.setBackground(Color.blue);
Panel p3 = new Panel();     p3.setBackground(Color.cyan);
Panel p4 = new Panel();     p4.setBackground(Color.darkGray);
Panel p5 = new Panel();     p5.setBackground(Color.gray);
Panel p6 = new Panel();     p6.setBackground(Color.green);
Panel p7 = new Panel();     p7.setBackground(Color.lightGray);
Panel p8 = new Panel();     p8.setBackground(Color.magenta);
Panel p9 = new Panel();     p9.setBackground(Color.orange);
Panel p10 = new Panel();    p10.setBackground(Color.pink);
Panel p11 = new Panel();    p11.setBackground(Color.red);
Panel p12 = new Panel();    p12.setBackground(Color.white);
Panel p13 = new Panel();    p13.setBackground(Color.yellow);
Panel p14 = new Panel();    p14.setBackground(new Color(50,100,100));
f.add(new Label("Color.black"));          f.add(p1);
f.add(new Label("Color.blue"));           f.add(p2);
// ... 내용 생략 ...
```

3. 메뉴 만들기

3.1 메뉴를 구성하는 컴포넌트

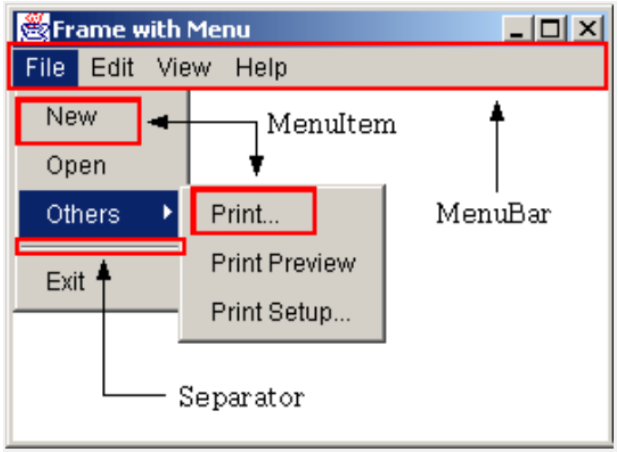


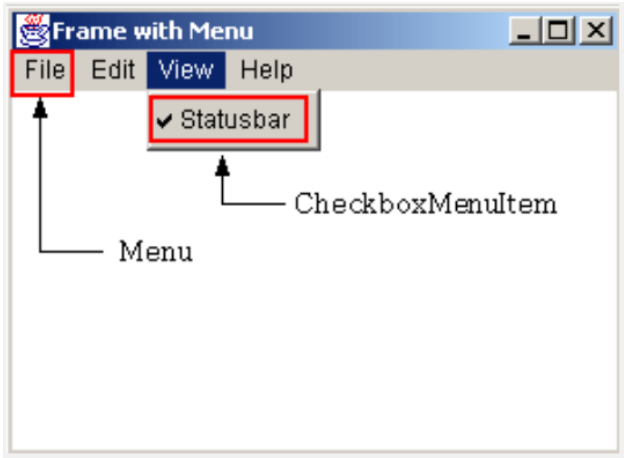
[그림13-7] 메뉴관련 컴포넌트의 상속계층도



[그림13-8] 메뉴 컴포넌트간의 포함관계

- MenuBar에는 Menu만 포함될 수 있다.(MenuItem은 포함시킬 수 없다.)
- Menu에는 Menu와 MenuItem이 포함될 수 있다.
- 계층형 menu를 만들고자 할 때는 Menu에 Menu를 포함시키면 된다.

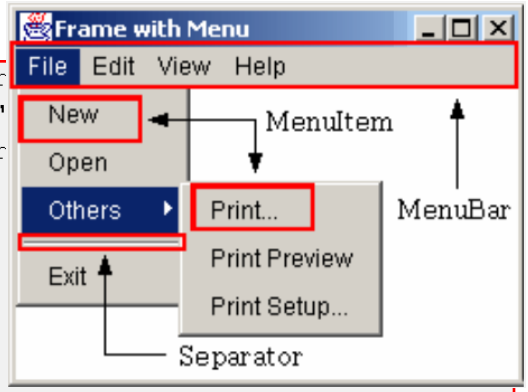


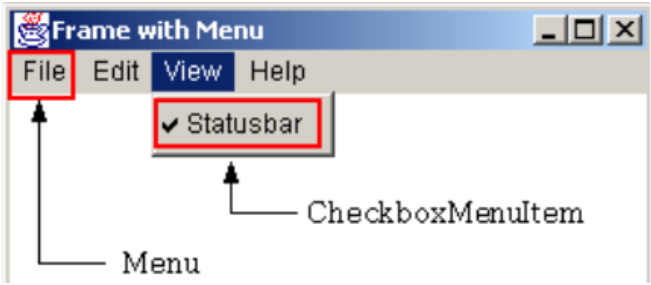


3.1 메뉴를 구성하는 컴포넌트 - 예제

메서드 또는 생성자	설 명
setHelpMenu(Menu menu)	인자로 넘겨받은 menu를 Help menu로 지정한다.
addSeparator()	Menu에 분리선(Separator)을 추가한다.
CheckboxMenuItem(String name, boolean status)	name - CheckboxMenuItem에 보여질 menu이름. status - 값이 true면, 체크된 상태로 생성된다.
CheckboxMenuItem(String name)	name - CheckboxMenuItem에 보여질 menu이름. status값을 지정하지 않으면 기본적으로 false값을 갖는다.

```
MenuBar mb = new MenuBar();
Menu mFile = new Menu("File");

MenuItem miNew = new MenuItem("New");
MenuItem miOpen = new MenuItem("Open");
Menu mOthers = new Menu("Others");
MenuItem miExit = new MenuItem("Exit");

mFile.add(miNew); //Menu에 MenuItem을 추가
mFile.add(miOpen);
mFile.add(mOthers); //Menu에 Menu를 추가
mFile.addSeparator(); //메뉴 분리선을 넣는다.
mFile.add(miExit);
```



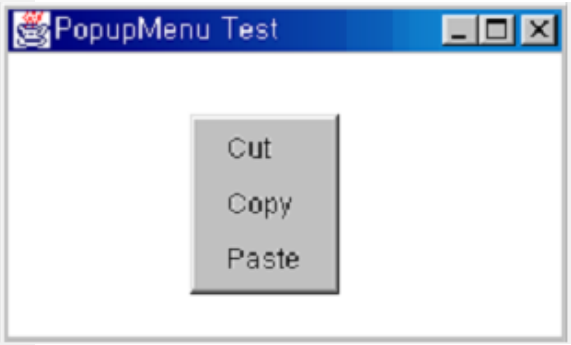



```
MenuItem miPrint = new MenuItem("Pr
MenuItem miPreview = new MenuItem("
MenuItem miSetup = new MenuItem("Pr
mOthers.add(miPrint);
mOthers.add(miPreview);
mOthers.add(miSetup);

Menu mEdit = new Menu("Edit");
Menu mView = new Menu("View");
Menu mHelp = new Menu("Help");
CheckboxMenuItem miStatusbar = new CheckboxMenuItem("Statusbar");
mView.add(miStatusbar);

mb.add(mFile);              // MenuBar에 Menu를 추가한다.
mb.add(mEdit);
mb.add(mView);
mb.setHelpMenu(mHelp);      // mHelp를 HelpMenu로 지정한다.

f.setMenuBar(mb);           // Frame에 MenuBar를 포함시킨다.
f.setVisible(true);
```

3.2 PopupMenu

- 윈도우(Frame) 내에서 오른쪽 마우스버튼을 누르면 나타나는 메뉴

```
public static void main(String args[]) {
      final Frame f = new Frame("PopupMenu Test");
      f.setSize(300, 200);

      final PopupMenu pMenu = new PopupMenu("Edit");
      MenuItem miCut = new MenuItem("Cut");
      MenuItem miCopy = new MenuItem("Copy");
      MenuItem miPaste = new MenuItem("Paste");
      pMenu.add(miCut); // PopupMenu에 MenuItem들을 추가한다.
      pMenu.add(miCopy);
      pMenu.add(miPaste);

      f.add(pMenu);   // PopupMenu를 Frame에 추가한다.
      f.addMouseListener( new MouseAdapter() { // 익명클래스
            public void mousePressed(MouseEvent me) {
                  // 오른쪽 마우스버튼을 누르면 PopupMenu를 화면에 보여준다.
                  if(me.getModifiers() == me.BUTTON3_MASK)
                        pMenu.show(f, me.getX(), me.getY());
            }
      });
      f.setVisible(true);
}
```

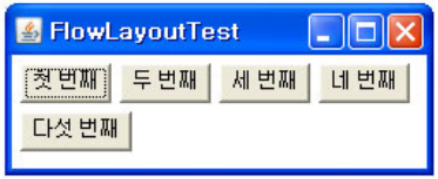


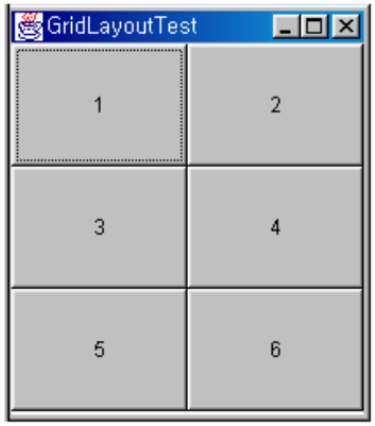
4. 레이아웃 매니저
(Layout Manager)

4.1 레이아웃 매니저를 이용한 컴포넌트 배치

- 레이아웃 매니저는 컨테이너에 포함된 컴포넌트의 배치를 자동관리한다.
- 레이아웃 매니저를 사용하면 컨테이너의 크기가 변경되거나 새로운 컴포넌트가 추가될 때, 컴포넌트를 재배치하는 코드를 작성할 필요가 없다.
- AWT에서는 아래와 같이 5개의 레이아웃 매니저를 제공한다.

BorderLayout, FlowLayout, GridLayout, CardLayout, GridbagLayout





▶ 컨테이너별 기본 레이아웃 매니저

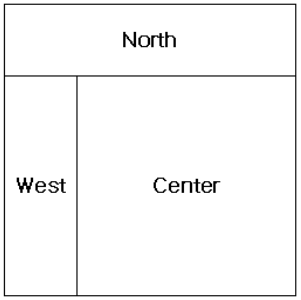
FlowLayout - Panel, Applet

BorderLayout - Window, Dialog, Frame

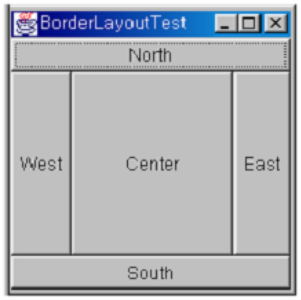
4.2 BorderLayout

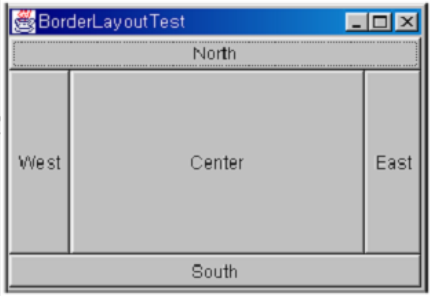
- 모두 5개의 영역으로 나누고, 각 영역에 하나의 컴포넌트만 넣을 수 있다.
- 한 영역에 하나 이상의 컴포넌트를 넣으려면 Panel을 사용하면 된다.

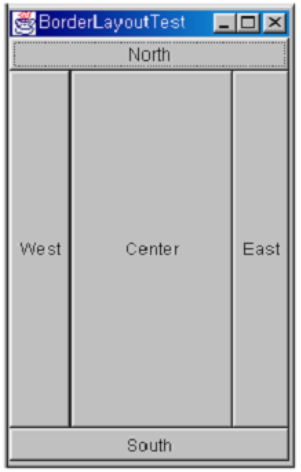
메서드 또는 생성자	설 명
BorderLayout(int hgap, int vgap)	각 영역 사이에 간격이 있는 BorderLayout을 생성한다. hgap - 각 영역의 사이에 간격을 준다.(좌우) vgap - 각 영역의 사이에 간격을 준다.(위아래)
BorderLayout()	영역 사이에 간격이 없는 BorderLayout을 생성한다.
add(String name, Component c) 또는 add(Component c, String name)	c - 추가하려는 컴포넌트 name- "North", "South", "East", "West", "Center" 중의 하나 또는 BorderLayout.NORTH, BorderLayout.SOUTH, BorderLayout.EAST, BorderLayout.WEST, BorderLayout.CENTER 중의 하나




```
Frame f = new Frame("BorderLayoutTest");
f.setSize(200, 200);
// Frame은 기본적으로 BorderLayout로 설정되어 있으므로 따로 설정하지 않아도 됨
f.setLayout(new BorderLayout());
Button north = new Button("North");
Button south = new Button("South");
Button east = new Button("East");
Button west = new Button("West");
Button center = new Button("Center");

// Frame의 5개의 각 영역에 Button을 하나씩 추가한다.
f.add(north, "North"); // f.add("North",north);와 같이 쓸 수도
f.add(south, "South"); // South의 대소문자 정확히
f.add(east, "East");   // East대신, BorderLayout.EAST 사용가능
f.add(west, "West");
f.add(center, "Center");
```





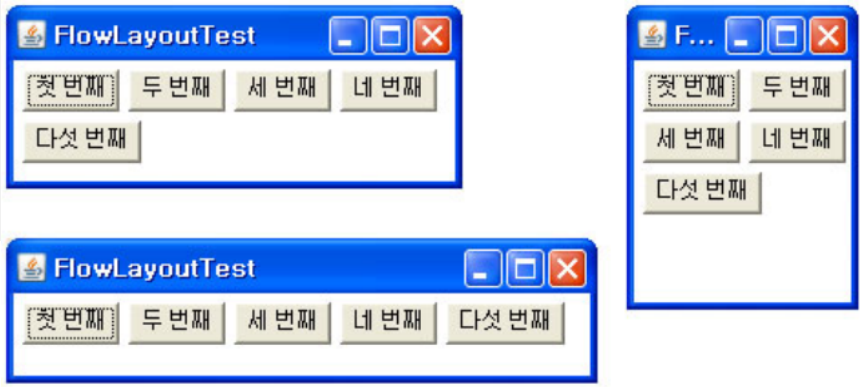


4.3 FlowLayout

- 컴포넌트를 워드프로세서와 같은 방식, 즉 왼쪽에서 오른쪽으로 배치한다.
- 3가지 정렬방식(왼쪽, 가운데, 오른쪽)이 가능하다.

메서드 또는 생성자	설 명
FlowLayout(int align, int hgap, int vgap)	align - 컴포넌트들의 정렬 방법을 지정한다. FlowLayout.LEFT(왼쪽), FlowLayout.CENTER(가운데),FlowLayout.RIGHT(오른쪽) 중 하나 hgap - 각 컴포넌트간의 사이에 간격을 준다.(좌우) vgap - 각 컴포넌트간의 사이에 간격을 준다.(위아래)
FlowLayout(int align)	align - 컴포넌트들의 정렬 방법을 지정한다. hgap과 vgap이 5픽셀인 FlowLayout을 생성한다.
FlowLayout()	가운데 정렬이면서 hgap과 vgap이 5픽셀인 FlowLayout을 생성한다.

```
Frame f = new Frame("FlowLayoutTest");
f.setSize(250, 100);
f.setLayout(new FlowLayout(FlowLayout.LEFT));
f.add(new Button("첫 번째"));
f.add(new Button("두 번째"));
f.add(new Button("세 번째"));
f.add(new Button("네 번째"));
f.add(new Button("다섯 번째"));
f.setVisible(true);
```



4.4 GridLayout

- 컴포넌트를 워드프로세서와 같은 방식, 즉 왼쪽에서 오른쪽으로 배치한다.
- 3가지 정렬방식(왼쪽, 가운데, 오른쪽)이 가능하다.

메서드 또는 생성자	설 명
GridLayout(int row, int col, int hgap, int vgap)	영역들 간의 사이에 간격이 있는 GridLayout을 생성한다. row - 컨테이너를 몇 개의 행(row)으로 나눌 것인지 적는다. col - 컨테이너를 몇 개의 열(column)로 나눌 것인지 적는다. hgap - 각 영역간의 사이에 간격을 준다.(좌우) vgap - 각 영역간의 사이에 간격을 준다.(위아래)
GridLayout(int row, int col)	영역들 간의 사이에 간격이 없는 GridLayout을 생성한다. row - 컨테이너를 몇 개의 행(row)으로 나눌 것인지 적는다. col - 컨테이너를 몇 개의 열(column)로 나눌 것인지 적는다.

```
Frame f = new Frame("GridLayoutTest");
f.setSize(150, 150);
f.setLayout(new GridLayout(3, 2)); // 3행 2열의 테이블을 만든다.
f.add(new Button("1"));  // 추가되는 순서대로 Button에 번호를 붙였다.
f.add(new Button("2"));
f.add(new Button("3"));
f.add(new Button("4"));
f.add(new Button("5"));
f.add(new Button("6"));
```

4.5 CardLayout

- 여러 컨테이너를 슬라이드처럼 바꿔가며 보여줄 수 있다.
- 앨범이나 퀴즈 또는 설치 프로그램에 주로 사용된다.

메서드 또는 생성자	설 명
CardLayout(int hgap, int vgap)	hgap - 컨테이너와 CardLayout 사이에 간격을 준다.(수평) vgap - 컨테이너와 CardLayout 사이에 간격을 준다.(수직)
CardLayout()	컨테이너와 간격이 없는 CardLayout을 생성한다.
add(Container parent, String name)	주어진 이름(name)으로 지정된 컨테이너(parent)에 추가한다.
show(Container parent,String name)	주어진 이름(name)의 컴포넌트를 컨테이너(parent)에 보여준다. name - 나타낼 컨테이너의 이름(추가할 때 사용한 이름)
first(Container parent)	parent - 지정된 컨테이너에 첫 번째로 추가된 컴포넌트를 보여준다.
last(Container parent)	parent - 지정된 컨테이너에 마지막으로 추가된 컴포넌트를 보여준다.
previous(Container parent)	지정된 컨테이너에 현재 보이는 컴포넌트보다 이전에 추가된 것을 보여준다. (현재 보이는 것이 첫 번째 것이면, 제일 마지막 것이 나타나게 된다.)
next(Container parent)	지정된 컨테이너에 현재 보이는 컴포넌트보다 다음에 추가된 것을 보여준다. (현재 보이는 것이 마지막 것이면, 제일 첫 번째 것이 나타나게 된다.)

4.5 CardLayout - 예제

```
public static void main(String args[]) {
      final Frame f = new Frame("CardLayoutTest");
      final CardLayout card = new CardLayout(10, 10);
      f.setLayout(card);

      Panel card1= new Panel();
      card1.setBackground(Color.lightGray);
      card1.add(new Label("Card 1"));
      Panel card2= new Panel();
      card2.add(new Label("Card 2"));
      card2.setBackground(Color.orange);
      Panel card3= new Panel();
      card3.add(new Label("Card 3"));
      card3.setBackground(Color.cyan);

      f.add(card1, "1");          // Frame에 card1을 "1"이라고 이름 붙여 추가한다.
      f.add(card2, "2");
      f.add(card3, "3");

      card1.addMouseListener(new Handler());
      card2.addMouseListener(new Handler());
      card3.addMouseListener(new Handler());

      f.setSize(200, 200);
      f.setLocation(200, 200);
      f.setVisible(true);
      card.show(f,"1");     // Frame에 추가된 Component중 이름이 "1"인 것을 보여준다.
}
```

```
class Handler extends MouseAdapter {
      public void mouseClicked(MouseEvent e) {
            // 마우스 오른쪽 버튼을 눌렀을 때
            if(e.getModifiers() == e.BUTTON3_MASK)      {
                  card.previous(f);  // 이전 Panel을 보여준다.
            } else {  // 마우스 왼쪽 버튼을 눌렀을 때
                  card.next(f);     //  다음 Panel을 보여준다.
            }
      }
} // class Handler
```

5. 이벤트 처리

(Event handling)

5.1 이벤트(Event)란?

- 사용자 또는 프로그램에 의해 발생할 수 있는 하나의 사건.

종류	설명
이벤트 소스 (Event Source, 이벤트 발생지)	이벤트가 발생한 컴포넌트. 사용자가 Button을 눌렀을 때 이벤트가 발생하고, Button은 이 이벤트의 이벤트 소스가 된다.
이벤트 핸들러 (Event Handler, 이벤트 처리기)	이벤트가 발생했을 때 실행될 코드를 구현해 놓은 클래스
이벤트 리스너 (Event Listener, 이벤트 감지기)	이벤트를 감지하고 처리한다. 이벤트 핸들러를 이벤트 리스너로 이벤트 소스에 연결해야 이벤트가 발생했을 때 이벤트가 처리된다.

```
class EventTest extends Frame {
    Button b = new Button("확 인");

    EventTest(String title) {
        super(title);
        setLayout(null);
        b.setBounds(100,75,100,50);
        // 이벤트 핸들러를 버튼의 이벤트 리스너를 등록한다.
        b.addActionListener(new EventHandler());

        add(b); // 버튼을 Frame에 추가한다.
        setSize(300, 200);
        setVisible(true);
    }

    public static void main(String args[]) {
        EventTest mainWin = new EventTest("Event Test");
    } // main
}
```

```
class EventHandler implements ActionListener { // 이벤트 핸들러
    public void actionPerformed(ActionEvent ae) {
        System.out.println("버튼이 눌러졌습니다.");
    }
}
```

5.2 이벤트 처리(Event handling)

- 이벤트가 발생했을 때, 어떤 작업이 수행되도록 코드를 작성하는 것

▶ 이벤트 처리 방법

1. 표13-25에 있는 메서드 중에서 필요한 것을 찾는다.

```
windowClosing(WindowEvent we)
```

2. 선택한 메서드가 속해있는 인터페이스를 구현하는 클래스를 작성한다.(표13-24 참고)

```
class EventHandler implements WindowListener {
    public void windowClosing(WindowEvent e) { /* 코드 작성 */  }
     ...
}
```

3. 위에서 구현한 클래스의 인스턴스를 생성해서 이벤트 소스에 Listener로 등록한다.

```
f.addWindowListener(new EventHandler());
```

이벤트	인터페이스	메서드
ActionEvent	ActionListener	actionPeformed(ActonEvent ae)
…	…	…
WindowEvent	WindowListener	windowClosing(WindowEve
		windowOpened(WindowEve
		windowIconified(WindowEv
		windowDeiconified(Window
		windowClosed(WindowEver
		windowActivated(WindowEv
		windowDeactivated(WindowEvent we)
…	…	…

메서드	호출시기
actionPeformed(ActonEvent ae)	Button을 클릭했을 때, Menu를 클릭했을 때, TextField에서 Enter키를 눌렀을 때, List의 item하나를 선택하여 더블클릭했을 때
…	…
windowClosing(WindowEvent we)	윈도우가 닫힐 때(닫기 버튼을 눌렀을 때)
…	…

5.2 이벤트 처리(Event handling) - 예제

```
class FrameTest3 {
    public static void main(String args[]) {
        Frame f = new Frame("Login");
        f.setSize(300, 200);

        // EventHandler클래스의 객체를 생성해서 Frame의 WindowListener로 등록한다.
        f.addWindowListener(new EventHandler());
        f.setVisible(true);
    }
}
```


```
class EventHandler implements WindowListener {
    public void windowOpened(WindowEvent e) {}
    public void windowClosing(WindowEvent e) { // Frame의 닫기 버튼을 눌렀을 때 호출된다.
        e.getWindow().setVisible(false);       // Frame을 화면에서 보이지 않도록 하고
        e.getWindow().dispose();               // 메모리에서 제거한다.
        System.exit(0);                        // 프로그램을 종료한다.
    }
    public void windowClosed(WindowEvent e) {} // 아무내용도 없는 메서드 구현
    public void windowIconified(WindowEvent e) {}
    public void windowDeiconified(WindowEvent e) {}
    public void windowActivated(WindowEvent e) {}
    public void windowDeactivated(WindowEvent e) {}
}
```

사용자가 Frame의 닫기 버튼을 누르면,
1. WindowEvent가 발생하고(WindowEvent의 인스턴스가 생성됨),
2. Frame에 WindowListener로 등록되어 있는 이벤트 핸들러의 windowClosing메서드를 호출한다. 이 메서드 내에서는 이벤트 발생시 생성된 WindowEvent인스턴스의 참조를 사용할 수 있어서 WindowsEvent인스턴스의 메서드들을 사용할 수 있다.

5.3 ActionEvent

- 컴포넌트에 정의된 특정 동작이 수행되었을 때 발생하는 고수준 이벤트
- Button을 누르는 방법은 두 가지(마우스 클릭, spacebar누르기)가 있다.
- Button을 누르면, MouseEvent나 KeyEvent가 발생하지만…
  ActionEvent도 발생한다.
- MouseEvent와 KeyEvent에 각각 별도의 이벤트처리를 하는 것보다
  ActionEvent에만 이벤트처리를 하는 것이 낫다.(코드의 중복제거)

▶ ActionEvent가 발생하는 경우

- Button이 눌러졌을 때
- Menu를 클릭했을 때
- TextField에서 Enter키를 눌렀을 때
- List의 item하나를 선택하여 더블클릭했을 때

5.4 Adapter클래스

- Listener인터페이스를 아무런 내용없이 구현해 놓은 클래스

```
interface KeyListener {
    public void keyPressed(KeyEvent e);
    public void keyReleased(KeyEvent e);
    public void keyTyped(KeyEvent e);
}
class KeyAdapter implements KeyListener {
    public void keyPressed(KeyEvent e) {}
    public void keyReleased(KeyEvent e) {}
    public void keyTyped(KeyEvent e) {}
}
```

Adapter클래스	이벤트 리스너(interface)
ComponentAdapter	ComponentListener
ContainerAdapter	ContainerListener
FocusAdapter	FocusListener
KeyAdapter	KeyListener
MouseAdapter	MouseListener
MouseMotionAdapter	MouseMotionListener
WindowAdapter	WindowListener

```
class KeyHandler implements KeyListener
{
   public void keyPressed(KeyEvent e) {
      System.out.println(e.getKeyChar());
   }
   public void keyReleased(KeyEvent e) {}
   public void keyTyped(KeyEvent e) {}
}
```

→

```
class KeyHandler extends KeyAdapter {
   public void keyPressed(KeyEvent e) {
      System.out.println(e.getKeyChar());
   }
}
```

6. AWT의 그래픽

6.1 paint()와 Graphics

- Component클래스의 paint()는 컴포넌트에 그림을 그리기 위한 것이다.

```
public void paint(Graphics g) {
    ...
}
```

- 컴포넌트에 그림을 그리려면, paint()를 알맞게 오버라이딩하면 된다.
- 모든 컴포넌트에는 Graphics객체가 있으며, getGraphics()로 얻을 수 있다.

```
Panel p = new Panel();
Graphics g = p.getGraphics();
g.drawLine(0,0, 10, 10); //Panel에 두 점 (0,0)과 (10,10)을 잇는 선을 그린다.
```

- Graphics클래스는 그림을 그리는데 필요한 다양한 메서드를 제공한다.
  문자 출력, font와 color지정, 다양한 선과 도형, 이미지 출력 등...

6.2 AWT쓰레드와 repaint()

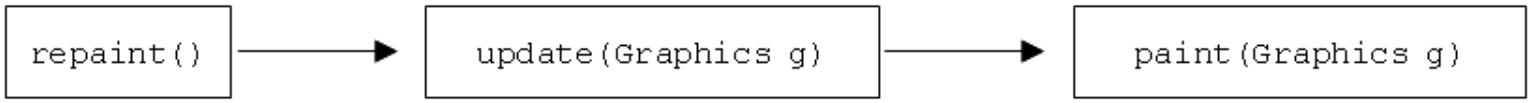
- AWT쓰레드는 다음과 같은 경우, paint()를 자동호출해서 화면을 갱신한다.

● 처음 화면에 나타날 때
● 다른 화면에 가려져 있던 부분이 다시 화면에 나타날 때
● 아이콘화 되어 있다가 원래 크기로 화면에 나타날 때

- 화면이 강제적으로 다시 그려지게 하려면 repaint()를 호출하면 된다.
- 화면갱신 요청을 받으면, AWT쓰레드는 update()를 호출하고, update()는 화면을 지운 후에 paint()를 호출한다.

```
public void update(Graphics g) {
      g.clearRect(0,0, width, height);
      paint(g);
}
```

repaint() - AWT쓰레드에게 화면을 갱신할 것을 요청
AWT쓰레드는 0.1초마다 확인해서 요청이 있으면 update()를 호출한다.
update(Graphics g) - 화면을 지우고 paint(Graphics g)를 호출한다.



6.2 AWT쓰레드와 repaint() - 예제

```
class GraphicsEx5 extends Frame implements MouseMotionListener {
    int x = 0;
    int y = 0;

    Image img = null;
    Graphics gImg = null;

    public static void main(String[] arg
        new GraphicsEx5("GraphicsEx5")
    }

    public GraphicsEx5(String title) {
        super(title);
        addMouseMotionListener(this);
        // ... 중간 생략 ...
        img = createImage(500, 500);
        gImg = img.getGraphics();
        gImg.drawString("왼쪽버튼을 누른 채로 마우스를 움직여보세요." ,10, 50);
    }

    public void paint(Graphics g) {
        if(img==null) return;
        g.drawImage(img,0,0,this); // 가상화면에 그려진 그림을 Frame에 복사
    }

    public void mouseMoved(MouseEvent me) {
        x = me.getX();
        y = me.getY();
    }
```

```
    public void mouseDragged(MouseEvent me) {
        if (me.getModifiersEx()!=MouseEvent.BUTTON1_DOWN_MASK )
            return;
        gImg.drawLine(x,y, me.getX(),me.getY());
        x = me.getX();
        y = me.getY();
        repaint();
    }
} // class
```

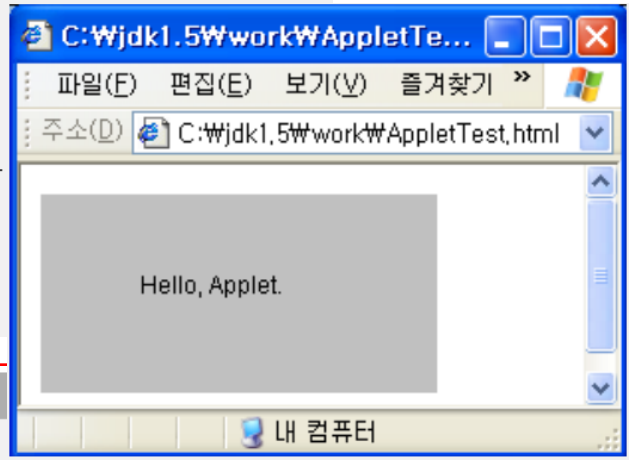
7. 애플릿(Applet)

7.1 애플릿(Applet)이란?

- 웹브라우저를 통해 실행될 수 있는 '작은 어플리케이션(Applet)'

1. 애플릿 관련정보가 포함된 HTML문서를 작성해야한다.
2. java.exe가 아닌 웹브라우저를 통해 실행된다.
3. public static void main(String args[])이 필요없다.
4. 애플릿은 java.applet.Applet을 상속하는 public클래스이어야 한다.

```
import java.awt.*;

public class HelloApplet extends java.applet.Applet
{
    public void paint(Graphics g) {
        setBackground(Color.lightGray); // 애플릿의 바
        g.drawString("Hello, Applet.",50,50);
    }
}
```



AppletTest.html

```
<html>
    <applet code="HelloApplet.class" width=200 height=100>
    </applet>
</html>
```

7.2 Applet의 생명주기(Life cycle)

- 애플릿이 담긴 HTML페이지가 브라우저에 로딩되면서 애플릿은 시작된다.

1. 웹브라우저가 애플릿이 포함된 HTML문서를 읽는다.
2. 웹브라우저가 애플릿(자바클래스)을 다운로드한다.
3. 웹브라우저가 애플릿의 인스턴스를 생성한다.
4. 애플릿이 초기화된다. - init()이 호출된다.
5. 애플릿이 실행된다. - start()가 호출된다.

애플릿의 메서드	설 명
init()	애플릿이 생성될 때 호출된다. 객체생성이나 이미지 로딩 등, 애플릿의 초기화 작업에 사용된다. 생성자 다음에 호출된다.
start()	애플릿의 실행이 시작 또는 재시작될 때 호출된다. 웹브라우저가 아이콘화되었다가 화면에 다시 나타날 때 그리고 init()이 호출된 직 후에는 반드시 호출된다.
stop()	애플릿의 실행을 중지시킨다. 웹브라우저가 아이콘화되던가, 다른 페이지로 이동할 때, 그리고 destroy()가 호출되기 직전에 반드시 호출된다.
destroy()	웹브라우저가 닫히면서 애플릿이 소멸되기 직전에 stop() 다음으로 호출된다. 애플릿이 사용하던 자원을 반환하는 용도로 사용된다.

[표13-29] 애플릿의 생명주기와 관련된 메서드

7.3 Applet의 보안 제약(Security restriction)

- 애플릿은 외부로부터 다운받는 프로그램이므로 잠재적인 위험요소가 있다.
- 사용자의 컴퓨터를 보호하기 위해 다음과 같은 '보안 제약'을 정해놓았다.

1. 사용자의 컴퓨터에 있는 실행파일(*.exe, *.com, *.bat)을 실행하는 것

```
Runtime rt = Runtime.getRuntime();
rt.exec("C:\\WINDOWS\\system32\\calc.exe"); // 계산기를 실행시킨다.
```

2. 사용자 컴퓨터의 파일을 읽거나 쓰기

```
FileWriter fw = new FileWriter("test.txt"); // test.txt에 abc를 출력한다.
fw.write("abc");
```

3. 사용자 컴퓨터의 정보를 읽기

```
String userName = System.getProperty("user.name"); // 사용자계정
String dir = System.getProperty("java.home"); // JDK가 설치된 위치
```

4. 애플릿을 제공한 서버가 아닌 다른 컴퓨터에서 소켓 열기

5. 네이티브 메서드(native method) 호출하기

7.4 Applet과 HTML태그

태그 옵션	설 명
ARCHIVE	애플릿에 사용되는 클래스파일이나 이미지파일이 담긴 jar파일을 지정하는데 사용된다. jar 파일이 하나 이상일 때는 쉼표(,)를 구분자로 사용한다. 예: ARCHIVE="a.jar, b.jar"
CODE	애플릿 클래스의 클래스파일을 지정하는데 사용한다. 패키지명.클래스명.class형태로 지정한다.
WIDTH, HEIGHT	애플릿이 웹브라우저에 보여질 영역의 폭(WIDTH)과 넓이(HEIGHT)를 지정하는데 사용된다.
CODEBASE	애플릿을 다운로드할 경로(URL)를 지정하는 데 사용된다. 이 옵션이 없으면 애플릿이 담긴 HTML의 경로가 codebase가 된다.
ALT	<APPLET>태그를 이해하지 못하는 웹브라우저에서 애플릿 대신 보여줄 텍스트를 지정하는데 사용한다.
NAME	애플릿의 이름을 지정하는 데 사용한다. 하나의 HTML페이지에 둘 이상의 애플릿이 있을 때 서로 구별되도록 할 수 있다.
ALIGN	애플릿의 정렬위치(alignment)를 지정하는데 사용된다. HTML의 태그에 사용되는 align옵션과 동일한 값이 사용될 수 있다. 예를 들면 left, right, top, bottom 등의 값이 사용된다.
VSPACE, HSPACE	애플릿의 외부 여백을 지정하는 데 사용된다.
<PARAM NAME=attribute VALUE=value>	외부에서 애플릿에 값을 전달할 때 사용한다. 자바 어플리케이션을 실행할 때 커맨드 라인에서 값을 입력하는 것과 유사하다. 애플릿 내부에서는 getParameter(String name)을 사용해서 값을 읽을 수 있다.

```
<APPLET
      [ARCHIVE=archiveList] CODE=appletFile.class
      WIDTH=pixels HEIGHT=pixels
      [CODEBASE=codebaseURL] [ALT=alternateText]
      [NAME=appletInstanceName] [ALIGN=alignment]
      [VSPACE=pixels] [HSPACE=pixels]
>
      [<PARAM NAME=appletAttribute1 VALUE=value>]
      [<PARAM NAME=appletAttribute2 VALUE=value>]
      ...
</APPLET>
```

감사합니다.

더 많은 동영상강좌를 아래의 사이트에서 구하실 수 있습니다.

http://www.javachobo.com

관련문의 : 남궁성 castello@naver.com