AWT

- 자바에서의 GUI 프로그래밍 기법을 익히고 제작해 본다
- AWT의 기본 개념 및 구조를 알아본다.
- 자바에서 제공하는 컴포넌트 및 배치 관리자에 대해 알아본다.

Section 01 GUI 프로그래밍

GUI 프로그래밍이란

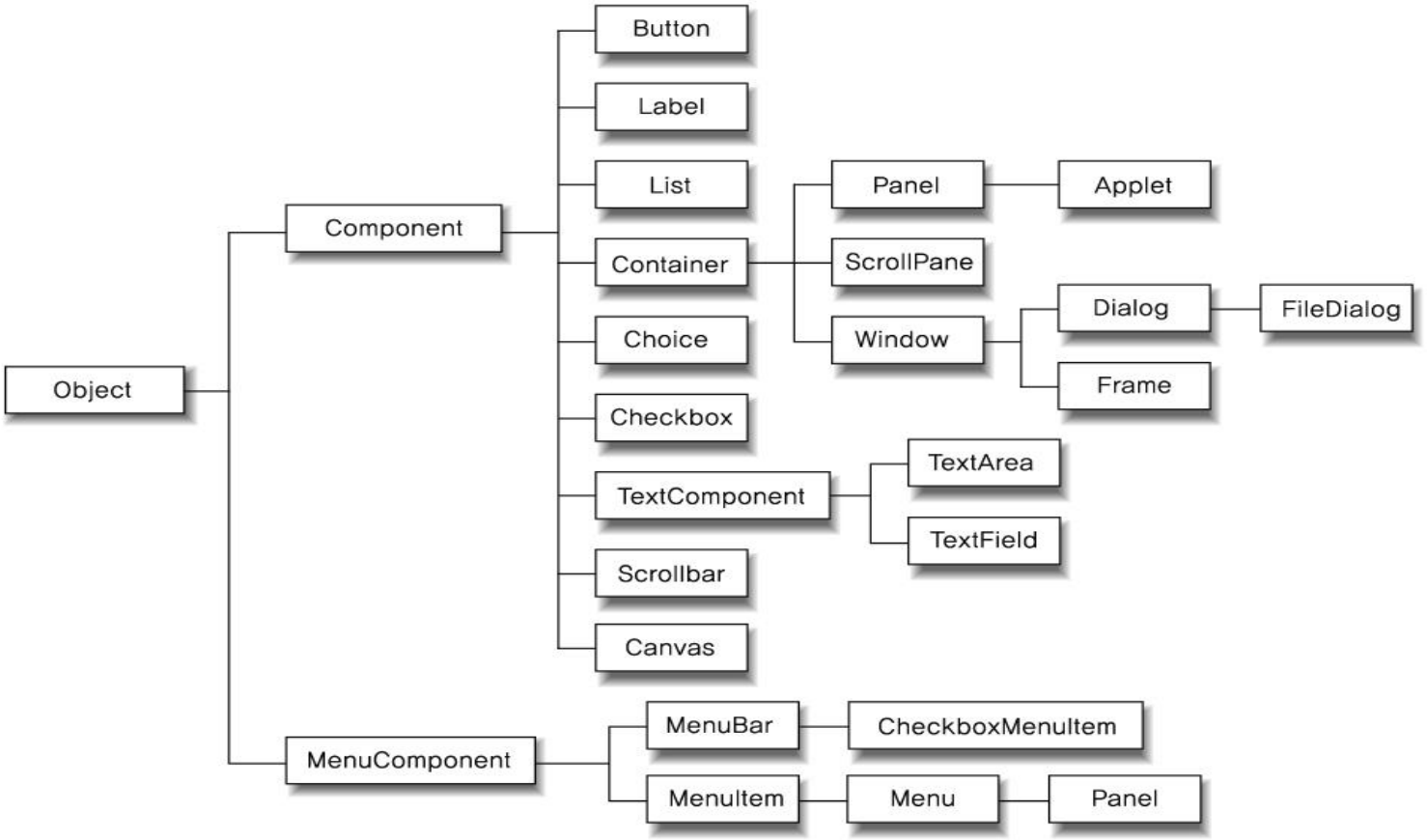
GUI는 과거에 사용하였던 DOS(CUI방식)와 같은 방식의 텍스트 기반 운영체제가 아닌 그래픽을 이용하여 사용자와 프로그램 간의 상호작용을 할 수 있도록 해주는 인터페이스를 의미한다. 자바에서 이러한 그래픽 프로그래밍을 지원하기 위해 나온 것이 바로 AWT인 것이다.

AWT의 기본 개념

- AWT(Abstract Window Toolkit)는 GUI 프로그래밍을 제작하기 위해 자바에서 제공하는 라이브러리를 모아놓은 것이다.
- AWT는 모든 GUI 프로그램에 사용되는 컴포넌트 및 툴킷을 제공하고 있으며 향후에는 JFC와 같은 스윙(Swing) 및 Java2D의 모태가 되는 개념이다.
- AWT는 운영체제에 구해받지 않고 쓸 수 있도록 운영체제의 것을 그대로 사용하지 않고 공통적이고 기본적인 컴포넌트들을 추상화시켜 제공한다.
- 실행되는 운영체제에 따라 다르게 보이거나 동작 방식에 차이가 있을 수 있다.
  이러한 단점을 극복하기 위해 개발된 것이 JFC(Java Foundation Classes)이다.

Section 01 GUI 프로그래밍

java.awt 패키지



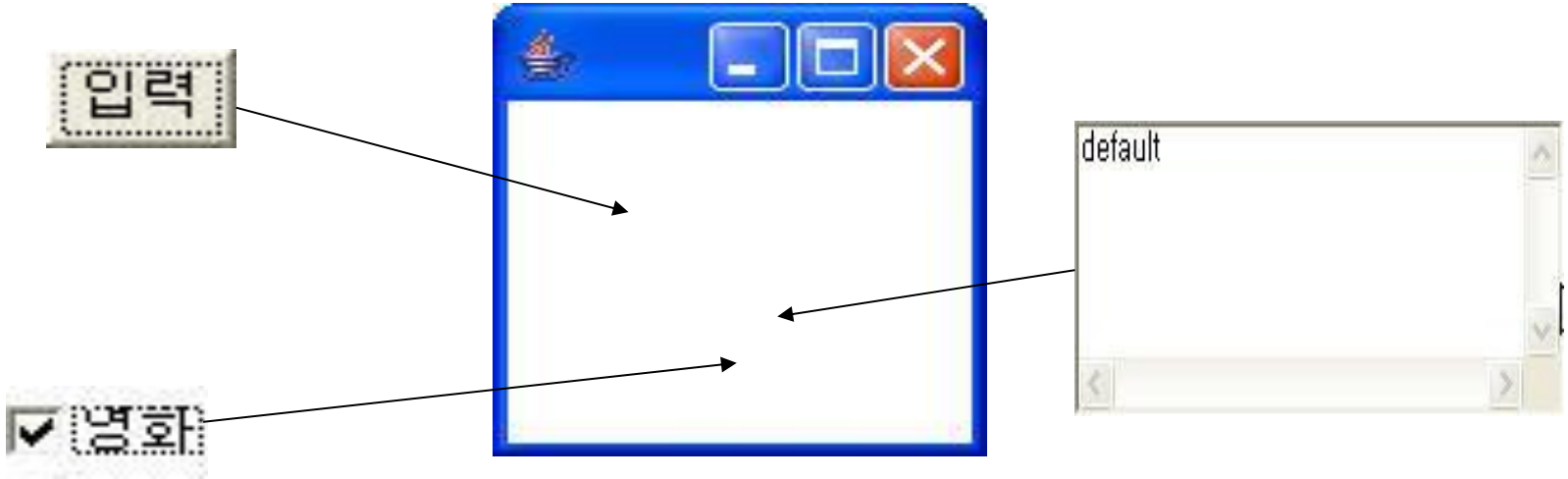
[그림 9-1] java.awt 패키지 구조도

Section 02 Container

Container

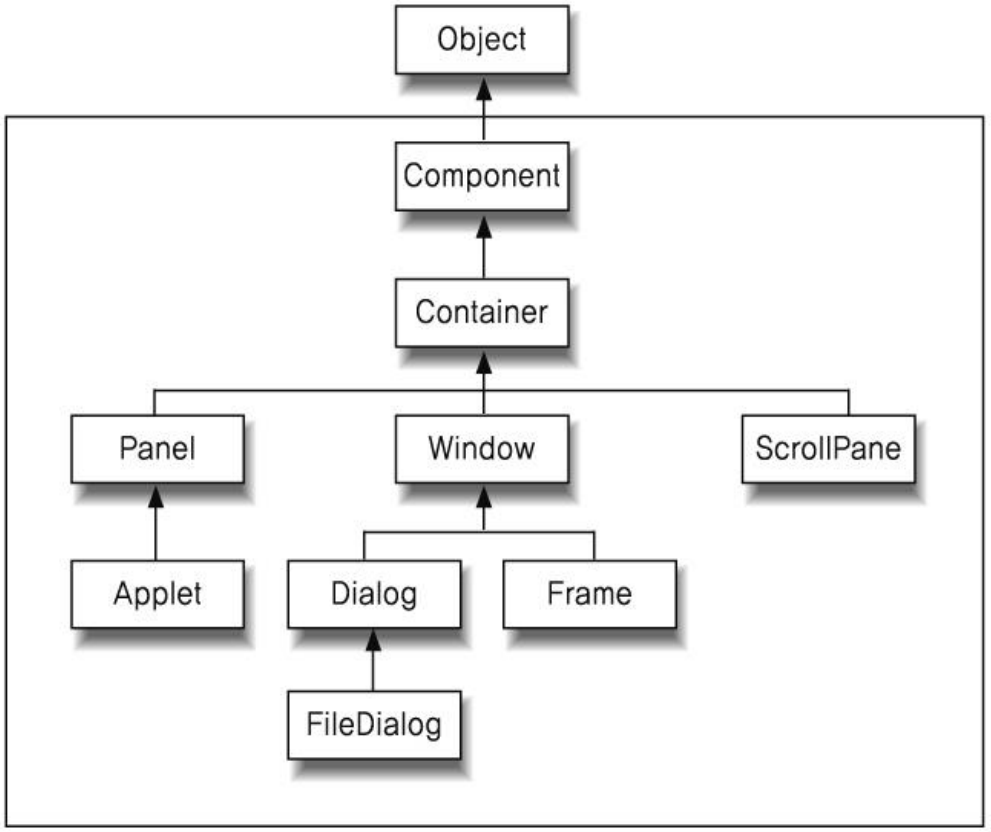
- 자신의 영역에 컴포넌트를 포함시키고 관리하는 역할을 하며 컨테이너가 다른 컨테이너를 포함할 수도 있다.
- 컴포넌트도 또한 컨테이너에 부착시키지 않으면 독자적으로 화면에 출력될 수가 없고 반드시 컨테이너에 부착을 시켜야만 화면에 출력이 될 수 있다.
- 컨테이너의 종류에는 Frame, Window, Panel, Applet, Dialog, FileDialog, ScrollPane이 있다.
- 컨테이너에 컴포넌트를 부착시키기 위해 add()메서드를 사용한다.

컨테이너와 컴포넌트 관계



Section 02 Container

컨테이너 종류 및 상속관계



[그림 9-3] 컨테이너 종류와 상속 관계

Section 02 Container

Frame

- Window 클래스의 하위 클래스로 일반적인 응용프로그램에서 윈도우를 생성하기 위해 사용되는 클래스이다.
- Frame 클래스의 상위 클래스인Window 클래스는 타이틀, 메뉴 등이 지원되지 않기 때문에 일반적으로 사용하지 않고 Frame 클래스를 사용한다.
- Frame 클래스는 기본적으로 경계선(Border), 타이틀, 메뉴, 시스템상자(최소화, 최대화, 종료 버튼) 등의 기능을 제공한다.
- Frame은 다른 윈도우에 속해 있지 않은 윈도우로 최상위 레벨 윈도우라 한다. setSize(),
- setBounds()메서드 등을 이용해서 Window의 크기를 설정한 후 setVisible(), show()메서드를 통해서 화면에 출력시킬 수 있다.

Frame 클래스의 생성자

[표 9-1] Frame 클래스의 주요 생성자

생성자	설명
Frame( )	가장 일반적인 생성자로 타이틀이 빈 상태로 생성한다.
Frame(GraphicsConfiguration gc)	화면 장치의 GraphicsConfiguration을 이용하여 프레임을 생성한다.
Frame(String title)	Title(윈도우의 타이틀 바에 나타낼 문자열)을 지정하여 프레임을 생성한다.
Frame(String title, GraphicsConfiguration gc)	Title(윈도우의 타이틀 바에 나타낼 문자열)과 GraphicsConfiguration을 이용하여 프레임을 생성한다.

Frame 클래스의 주요메서드

[표 9-2] Frame 클래스의 주요 메서드

반환형	메서드	설명
int	getExtendedState( )	프레임의 상태를 얻어온다.
static Frame[ ]	getFrames( )	애플리케이션에서 생성한 모든 프레임을 리턴한다.
MenuBar	getMenuBar( )	프레임의 메뉴바를 얻어온다.
int	getState( )	프레임의 상태를 얻어온다.
String	getTitle( )	프레임의 타이틀을 얻어온다.
void	remove(MenuComponent m)	프레임에서 지정한 메뉴바를 제거한다.
	setIconImage(Image image)	프레임이 최소화될 때 출력되는 이미지를 지정한다.
	setMenuBar(MenuBar mb)	프레임의 메뉴바를 지정한다.
	setResizable(boolean resizable)	프레임의 크기를 사용자가 변경할 수 있게 할 것인지를 지정한다.
	setState(int state)	프레임의 상태를 지정한다.
	setTitle(String title)	프레임의 타이틀을 지정한다.

Section 02 Container

Panel 클래스

- 컴포넌트들을 그룹별로 묶어서 처리할 때 주로 사용한다.
- Frame에 컴포넌트를 직접 붙이지 않고 Panel에 그룹별로 붙이고, 다시 Panel을 Frame에 붙이는 경우가 많다.
- 다른 Panel을 생성하여 자신에게 붙일 수도 있어 윈도우 프로그램을 만들때는 여러 개의 Panel을 사용하는 경우가 많다.

Panel 클래스의 생성자

[표 9-3] Panel 클래스의 주요 생성자

생성자	설명
Panel( )	디폴트의 레이아웃 매니저를 사용해 새로운 패널을 작성한다.
Panel(LayoutManager layout)	지정된 레이아웃 매니저를 가지는 새로운 패널을 작성한다.

Panel 클래스의 주요 메서드

[표 9-4] Panel 클래스의 주요 메서드

반환형	메서드	설명
void	addNotify( )	패널의 피어를 작성한다.
AccessibleContext	getAccessibleContext( )	Panel에 관련한 AccessibleContext를 얻어온다.

Dialog 클래스

- 메인 윈도우 외에 메시지를 출력하거나, 사용자로부터 데이터를 입력 받을 때 주로 사용하는 컨테이너이다.
- 보통은 Dialog 클래스로부터 상속을 받아 새로운 기능을 가진 대화상자를 만드는데 사용된다.

Dialog 클래스의 생성자

[표 9-5] Dialog 클래스의 주요 생성자

생성자	설명
Dialog(Dialog owner)	생성되는 Dialog 객체를 소유하는 객체가 owner인 Frame을 생성한다.
Dialog(Dialog owner, String title)	생성되는 Dialog 객체의 소유자를 owner라는 객체로 설정하고 타이틀을 설정한다.
Dialog(Dialog owner, String title, boolean modal)	소유자로 owner 객체를 설정하고 타이틀을 가지며, 모덜일지 모덜이 아닌지를 설정하여 Dialog 객체를 생성한다.
Dialog(Frame owner)	생성되는 Dialog 객체를 소유하는 객체가 owner라는 Frame 객체를 생성한다.

Dialog 클래스의 주요메서드

[표 9-6] Dialog 클래스의 주요 메서드

메서드	반환형	설명
void	addNotify( )	패널의 피어를 작성한다.
AccessibleContext	getAccessibleContext( )	Panel에 관련한 AccessibleContext를 얻어온다.

Section 03 Component

Component

모든 컴포넌트들의 super 클래스로서 GUI 프로그램을 구성하는 구성단위로 각 컴포넌트들에서 공통으로 사용되어지는 메서드들을 가지고 있다.

Component 클래스의 기본 메서드

[표 9-7] Component 클래스의 크기 및 위치와 관련있는 주요 메서드

반환형	메서드	설명
int	getX( )	컴포넌트의 현재의 X 좌표를 얻어온다.
	getY( )	컴포넌트의 현재의 Y 좌표를 얻어온다.
	getWidth( )	컴포넌트의 현재의 폭을 얻어온다.
	getHeight( )	컴포넌트의 현재의 높이를 얻어온다.
Dimension	getSize( )	컴포넌트의 크기를 크기 객체(Dimensioned Object)로 얻어온다.
	getMaximumSize( )	컴포넌트의 최대 크기를 크기 객체로 얻어온다.
	getMinimumSize( )	컴포넌트의 최소 크기를 크기 객체로 얻어온다.
Rectangle	getBounds( )	컴포넌트의 경계를 직사각형 객체(Rectangle Object)로 얻어온다.
void	setSize(int width, int height)	컴포넌트의 폭, 높이를 지정한다.
	setLocation(int x, int y)	컴포넌트의 새로운 위치를 지정하여 이동시킨다.
	setBounds(int x, int y, int width, int height)	컴포넌트의 위치와 크기를 지정한다.
	setBounds(Rectangle r)	새로운 경계 Rectangle r에 적합하도록 컴포넌트의 위치와 크기를 지정한다.

Component 클래스의 주요메서드

[표 9-8] Component 클래스의 색상, 폰트와 관련있는 주요 메서드

반환형	메서드	설명
Color	getBackground( )	컴포넌트의 배경색을 색상 객체(Color Object)로 얻어온다.
	getForeground( )	컴포넌트의 전경색을 색상 객체로 얻어온다.
void	setBackground(Color c)	컴포넌트의 배경색을 Color c로 지정한다.
	setForeground(Color c)	컴포넌트의 전경색을 Color c로 지정한다.
Font	getFont( )	컴포넌트의 글꼴을 글꼴 객체(Font Object)로 얻어온다.
void	setFont(Font f)	컴포넌트의 글꼴을 Font f로 지정한다.

[표 9-9] Component 클래스의 설정과 관련있는 주요메서드

반환형	메서드	설명
void	setEnabled(Boolean b)	파라미터 b값에 의해 컴포넌트의 활성화와 비활성화를 지정한다.
	setVisible(Boolean b)	파라미터 b값에 의해 컴포넌트를 출력하거나 숨기는 것을 지정한다.
String	getName( )	컴포넌트의 이름을 얻어온다.
Container	getParent( )	컴포넌트를 소유하고 있는 컨테이너를 얻어온다.
void	requestFocus( )	현 컴포넌트에 포커스를 요청한다.

기본 Component

[표 9-10] 기본 컴포넌트 종류와 기능

종류	프로그래밍 언어
Button	버튼을 만들 때 사용한다.
Canvas	비어 있는 공간으로 그래픽을 처리할 때 사용한다.
Checkbox	체크 박스나 라디오 버튼을 만들 때 사용한다.
Choice	드롭-다운 리스트를 만들 때 사용한다.
Label	고정 문자열을 표시할 때 사용한다.
List	리스트를 만들 때 사용한다.
Scrollbar	스크롤바를 만들 때 사용한다.

Section 03 Component

Button

버튼을 사용자가 눌렸을 때 특정한 액션을 실행할 수 있도록 만든 컴포넌트이다.

Button 클래스의 생성자

[표 9-11] Button 클래스의 주요 생성자

생성자	설명
Button( )	비어 있는 버튼 객체를 생성한다.
Button(String label)	label을 지정하여 버튼 객체를 생성한다.

Button 클래스의 주요 메서드

[표 9-12] Button 클래스의 주요 메서드

반환형	메서드	설명
void	addActionListener(ActionListener l)	버튼으로부터 액션 이벤트를 받기 위해 지정된 액션 리스너를 추가한다.
String	getActionCommand( )	버튼에서 발생되는 액션 이벤트의 커맨드명을 얻어온다.
	getLabel( )	버튼의 레이블을 얻어온다.
void	setLabel(String label)	버튼의 레이블을 지정한 label로 설정한다.

Checkbox

- 사용자가 여러 종류의 옵션을 선택할 것인지의 여부를 지정할 때 사용한다 .
- 여러 개의 체크박스를 묶어 하나의 그룹으로 만들어 그룹내에서는 하나만이 값을 유지할 수 있는 라디오 버튼 형태로도 사용할 수 있는 컴포넌트이다.
- 그룹으로 묶을 때는 CheckboxGroup 클래스를 사용한다.

Checkbox 클래스의 생성자

[표 9-13] Checkbox 클래스의 주요 생성자

생성자	설명
Checkbox( )	label이 없는 체크박스 객체를 생성한다.
Checkbox(String label)	지정된 label을 가지는 체크박스 객체를 생성한다.
Checkbox(String label, Boolean state)	지정된 label과 지정된 state를 넣어서 체크박스 객체를 생성한다.
Checkbox(String label, Boolean state, CheckboxGroup group)	지정된 label, 지정된 state를 넣어, 지정된 group에 속하는 체크박스 객체를 생성한다.

Checkbox 클래스의 주요 메서드

[표 9-14] Checkbox 클래스의 주요 메서드

반환형	메서드	설명
void	addItemListener(ItemListener l)	체크박스로부터 아이템 이벤트를 받기 위해 지정된 아이템 리스너를 추가한다.
	setLabel(String label)	체크박스의 레이블을 지정한다.
String	getLabel( )	체크박스의 레이블을 얻어온다.
void	setState(boolean state)	체크박스 상태를 지정된 상태로 설정한다.
boolean	getState( )	체크박스가 'On' 또는 'Off' 상태인지를 얻어온다.
void	setCheckboxGroup(CheckboxGroup g)	체크박스 그룹을 지정한다.

Choice

- List 클래스와 거의 유사한 기능을 가지고 있는 컴포넌트로 사용자가 드롭-다운 버튼을 사용하여 여러 아이템 중에 하나를 선택할 수 있는 기능을 제공한다.
- 컴포넌트를 생성한 후에 드롭-다운 리스트에 항목에 추가시켜 사용한다.

Choice 클래스의 생성자

[표 9-15] Choice 클래스의 주요 생성자

생성자	설명
Choice( )	새로운 선택 메뉴 객체를 생성한다.

Choice 클래스의 주요 메서드

[표 9-16] Choice 클래스의 주요 메서드

반환형	메서드	설명
void	add(String item)	Choice 메뉴에 항목을 추가한다.
	addItemListener(ItemListener l)	Choice 메뉴로부터 아이템 이벤트를 받기 위해 지정된 아이템 리스너를 추가한다.
	insert(String item, int index)	Choice에 지정된 위치에 항목을 삽입한다.
	remove(int position)	Choice 메뉴에 지정한 위치에 있는 항목을 제거한다.
	remove(String item)	Choice 메뉴로부터 item이 첫 번째로 발견된 항목을 제거한다.
	removeAll( )	Choice 메뉴로부터 모든 item을 제거한다.
String	getItem(int index)	Choice 메뉴에서 지정한 위치의 항목의 문자열을 얻어온다.
int	getItemCount( )	Choice 메뉴에서 항목의 개수를 얻어온다.
	getSelectedIndex( )	현재 선택된 항목의 위치를 얻어온다.
String	getSelectedItem( )	현재 선택된 항목의 문자열을 얻어온다.
void	select(int index)	지정한 위치의 항목을 선택한다.
	select(String str)	지정한 이름의 항목을 선택한다.

Label

- 사각형의 영역에 문자열을 표시할 때 사용하는 컴포넌트이다.
- 레이블은 경계선이 없고 특별한 상태를 가지지도 않는다. 그러므로 레이블을 컨테이너에 포함시키게 되면 레이블의 문자만 화면에 표시가 된다.
- 레이블의 문자열은 좌, 우, 중앙으로 정렬시킬 수 있다.

Label 클래스의 주요 멤버필드

[표 9-17] Label 클래스의 주요 멤버 필드

자료형	필드명	설명
static int	CENTER	레이블의 문자를 중앙에 정렬시킨다.
	LEFT	레이블의 문자를 왼쪽에 정렬시킨다.
	RIGHT	레이블의 문자를 오른쪽에 정렬시킨다.

Label 클래스의 생성자

[표 9-18] Label 클래스의 주요 생성자

생성자	설명
Label( )	빈 레이블을 생성한다.
Label(String text)	레이블에 지정한 text를 가지고 왼쪽 정렬이 된 상태로 생성한다.
Label(String text, int alignment)	레이블에 지정한 text를 가지고, 지정한 정렬이 된 상태로 생성한다.

Label 클래스의 주요 메서드

[표 9-19] Label 클래스의 주요 메서드

반환형	메서드	설명
String	getText( )	레이블의 텍스트를 얻어온다.
void	setText(String text)	레이블에 지정한 text로 설정한다.
	setAlignment(int align)	레이블의 텍스트를 지정한 정렬로 정렬시킨다.

List

- Choice와 유사한 기능이지만 여러 개의 항목을 보여주고 사용자가 하나 또는 여러 개의 항목을 선택할 수 있도록 지원하는 컴포넌트이다.
- 기본적으로는 하나의 항목만을 선택할 수 있지만 MultipleMode를 설정하면 한번에 여러 개의 항목을 선택할 수 있다.

List 클래스의 생성자

[표 9-20] List 클래스의 주요 생성자

생성자	설명
List( )	새로운 리스트 객체를 생성한다.
List(int rows)	지정한 숫자만큼의 항목을 보여주는 새로운 리스트 객체를 생성한다.
List(int rows, Boolean multipleMode)	지정한 숫자만큼의 항목을 보여주는 새로운 리스트 객체를 생성하며, 단일 선택 모드나 다중 선택 모드를 지정할 수 있다.

List 클래스의 주요 메서드

[표 9-21] List 클래스의 주요 메서드

반환형	메서드	설명
void	add(String item)	지정한 항목을 List의 끝에 추가한다.
	add(String item, int index)	List의 지정된 위치에 항목을 삽입한다.
	addItemListener(ItemListener l)	List로부터 아이템 이벤트를 받기 위해 지정된 아이템 리스너를 추가한다.
	remove(int position)	List에 지정한 위치에 있는 항목을 제거한다.
	remove(String item)	List로부터 item이 첫 번째로 발견된 항목을 제거한다.
	removeAll( )	List에 있는 모든 item을 제거한다.
String	getItem(int index)	List에서 지정한 위치의 항목의 문자열을 얻어온다.
String[ ]	getItems( )	List의 항목들을 문자열 배열로 얻어온다.
int	getItemCount( )	List에서 항목의 개수를 얻어온다.
	getSelectedIndex( )	현재 선택된 항목의 위치를 얻어온다.
	getSelectedIndexes( )	다중 선택 모드일 때, 현재 선택된 항목의 위치 값들을 배열로 얻어온다.
String	getSelectedItem( )	현재 선택된 항목의 문자열을 얻어온다.
String[ ]	getSelectedItems( )	다중 선택 모드일 때, 현재 선택된 항목들을 문자열 배열로 얻어온다.
void	select(int index)	지정한 위치의 항목을 선택한다.
	replaceItem(String newValue, int index)	지정한 위치의 항목을 newValue값을 바꾼다.

Canvas

- 특정한 모양을 갖고 있지 않고 단지 사각형의 영역만을 갖고 있는 컴포넌트이다.
- 이 컴포넌트는 그림을 그릴 수 있는 도화지의 역할을 하는 컴포넌트로 컨테이너에 포함되어 그래픽 처리를 할 수 있다.

Canvas 클래스의 생성자

[표 9-22] Canvas 클래스의 주요 생성자

생성자	프로그래밍 언어
Canvas( )	새로운 캔버스 객체를 생성한다.
Canvas(GraphicsConfiguration gc)	화면 장치의 GraphicsConfiguration을 이용하여 캔버스 객체를 생성한다.

Canvas 클래스의 주요메서드

[표 9-23] Canvas 클래스의 주요 메서드

반환형	메서드	설명
void	paint(Graphics g)	캔버스를 업데이트할 때 사용된다.
	update(Graphics g)	캔버스에 그림을 그릴 때 사용된다.

텍스트 Component

텍스트를 다루는 클래스의 super 클래스로 텍스트를 처리하는 각종 메서드를 가지고 있다.

독립적으로는 생성되지 못한다.

텍스트 Component 클래스의 주요 메서드

[표 9-24] 텍스트 편집과 관련있는 메서드

반환형	메서드	설명
int	getCaretPosition( )	텍스트 컴포넌트의 텍스트가 삽입될 캐럿의 현재의 위치를 얻어온다.
void	setCaretPosition(int Position)	텍스트 컴포넌트의 텍스트가 삽입될 캐럿의 위치를 지정한다.
String	getText( )	텍스트 컴포넌트가 가지고 있는 텍스트를 얻어온다.
void	setText(String t)	텍스트 컴포넌트에 표시될 텍스트를 설정한다.
	setEditable(boolean b)	텍스트 컴포넌트의 편집 가능 여부를 결정한다.

텍스트 Component 클래스의 주요 메서드

[표 9-25] 텍스트 선택과 관련있는 메서드

반환형	메서드	설명
String	getSelectedText( )	텍스트 컴포넌트에서 선택되어진 텍스트를 얻어온다.
int	getSelectionEnd( )	텍스트 컴포넌트에서 선택되어진 영역의 끝 위치를 얻어온다.
	getSelectionStart( )	텍스트 컴포넌트에서 선택되어진 영역의 시작 위치를 얻어온다.
void	setSelectionEnd(int selectionEnd)	텍스트 컴포넌트에서 선택할 영역의 끝 위치를 지정한다.
	setSelectionStart(int selectionStart)	텍스트 컴포넌트에서 선택할 영역의 시작 위치를 지정한다.
	select (int selectionStart, int selectionEnd)	지정된 시작과 끝 위치의 텍스트를 선택 상태로 만들어준다.
	selectAll()	텍스트 컴포넌트에 있는 모든 텍스트를 선택 상태로 만들어준다.

[표 9-26] TextComponent 클래스의 하위 클래스

종류	기능
TextField	문자 한 줄만 입력받을 때 사용한다.
TextArea	여러 줄의 문자를 입력받을 때 사용한다.

TextField

- 한 줄 내의 텍스트를 입력 받거나 편집할 수 있는 컴포넌트이다.
- 한 줄에 표시할 수 있는 컬럼수를 지정할 수 있고 반향 문자(Echo Character)를 지정하면 입력되는 문자대신 반향 문자로 지정한 문자로 출력된다.

TextField 클래스의 생성자

[표 9-27] TextField 클래스의 주요 생성자

생성자	설명
TextField( )	비어있는 텍스트 필드 객체를 생성한다.
TextField(int columns)	지정한 컬럼수만큼 문자를 보여줄 수 있는 크기로 텍스트 필드 객체를 생성한다.
TextField(String text)	지정한 텍스트로 초기화하여 텍스트 필드 객체를 생성한다.
TextField(String text, int columns)	지정한 텍스트로 초기화하여 출력하고, 지정한 컬럼 수만큼 문자를 보여줄 수 있는 크기로 텍스트 필드 객체를 생성한다.

TextField 클래스의 주요메서드

[표 9-28] TextField 클래스의 주요 메서드

반환형	메서드	설명
int	getColumns( )	텍스트 필드의 컬럼 수를 얻어온다.
void	setColumns(int columns)	텍스트 필드의 컬럼 수를 지정한다.
	getEchoChar( )	현재 설정되어 있는 반향 문자를 얻어온다.
	setEchoChar(char c)	텍스트 필드의 반향 문자를 지정한다.

TextArea

- 여러 줄의 텍스트를 사용자로부터 입력받거나 편집할 수 있는 컴포넌트이다.
- 화면에 출력되는 영역이 벗어나면 스크롤바 표시 방식에 따라 자동으로 스크롤바가 생성된다.
- 사용자가 필요에따라 일부 스크롤바만 나타나게 할 수도 있다.

TextArea 클래스의 주요 멤버필드

[표 9-29] TextArea 클래스의 주요 멤버 필드

반환형	메서드	설명
static int	SCROLLBARS_BOTH	수평/수직 스크롤바를 모두 표시한다.
	SCROLLBARS_HORIZONTAL_ONLY	수평 스크롤바만 표시한다.
	SCROLLBARS_VERTICAL_ONLY	수직 스크롤바만 표시한다.
	SCROLLBARS_NONE	스크롤바를 표시하지 않는다.

TextArea 클래스의 생성자

[표 9-30] TextArea 클래스의 주요 생성자

반환형	설명
TextArea( )	비어있는 텍스트 영역 객체를 생성한다.
TextArea(int rows, int columns)	지정된 행수와 컬럼수만큼 표현할 수 있는 텍스트 영역 객체를 생성한다.
TextArea(String text)	지정된 문자를 가지고 텍스트 영역 객체를 생성한다.
TextArea(String text, int rows, int columns)	지정된 문자를 가지고 초기화하여 지정된 행수와 컬럼수만큼 표현할 수 있는 텍스트 영역 객체를 생성한다.
TextArea(String text, int rows, int columns, int scrollbars)	지정된 문자를 가지고 초기화하여 지정된 행수와 컬럼수만큼 표현할 수 있는 텍스트 영역 객체를 생성한다. 그리고 스크롤바의 모습이 어떻게 나타낼 것인지를 지정한다.

TextArea 클래스의 주요 메서드

[표 9-31] TextArea 클래스의 주요 메서드

반환형	메서드	설명
void	append(String str)	지정된 문자열을 기존 내용의 끝에 추가한다.
	insert(String str, int pos)	지정된 문자열을 지정된 위치에 삽입한다.
	replaceRange(String str, int start, int end)	지정된 시작과 끝 위치의 문자열을 지정된 문자열로 바꾼다.
int	getColumns( )	텍스트 영역의 컬럼수를 얻어온다.
	getRows( )	텍스트 영역의 행수를 얻어온다.

메뉴 Component

- 메뉴는 보통 최상위 레벨의 윈도우 타이틀바 아래에 존재하는 것으로 사용자가 프로그램의 기능을 선택할 수 있도록 해주는 기능을 가지고 있는 컴포넌트이다.
- 메뉴의 구성은 MeunBar, Menu, MenuItem으로 구성된다.

메뉴 Component 클래스의 Sub 클래스

[표 9-32] MenuComponent 클래스의 하위 클래스

종류	기능
MenuBar	메뉴를 올려 놓을 수 있는 메뉴바를 만들 때 사용한다.
Meun	메뉴 바에 올려 놓을 수 있는 메뉴를 만들 때 사용한다.
MenuItem	메뉴의 하위 메뉴를 만들 때 사용한다
CheckboxMenuItem	체크박스가 들어 있는 메뉴아이템을 만들 때 사용한다.
PopupMenu	동적으로 표현할 수 있는 메뉴를 만들 때 사용한다.

메뉴 사용법

1. 메뉴바 객체를 생성한다.

```
MenuBar mb = new MenuBar();
```

2. 메뉴바에 삽입할 메뉴를 생성한 후 메뉴를 메뉴바에 붙인다.

```
Menu menu_file =  new Menu("파일" );
mb.add(menu_file);
```

3. 메뉴에 붙일 메뉴아이템을 생성한 후 해당 메뉴에 붙인다.

```
MenuItem menu_file_new = new MenuItem("새문서" );
Menu_file.add(menu_file_new);
```

4. 메뉴바를 윈도우에 붙인다.

```
setMenuBar(mb);
```

Section 04 LayoutManager

LayoutManager

- 컨테이너는 자기 자신에 컴포넌트를 붙일 때 어디에, 어떤방식으로 배치하여 붙일것인가를 이미 결정하고 있다.
- 즉, 미리 정해진 레이아웃에 따라 컴포넌트들을 자동으로 배치하는 기능을 가지고 있는 객체를 컨테이너들은 가지고 있는데 이것을 배치관리자(LayoutMananger)라 한다.
- 자바에서 사용하는 배치관리자는 FlowLayout, BorderLayout, GridLayout, GridBagLayout, CardLayout의 5가지가 있다.
- 배치관리자는 각자 다른 방식으로 배치기능을 가지고 있으며 컨테이너는 기본적으로 하나의 배치관리자를 가지고 있다.
- 사용자가 임의로 배치관리자는 다시 설정할 수 있으며 배치관리자를 제거하고 수동으로 좌표를 이용해서 배치할 수도 있다.

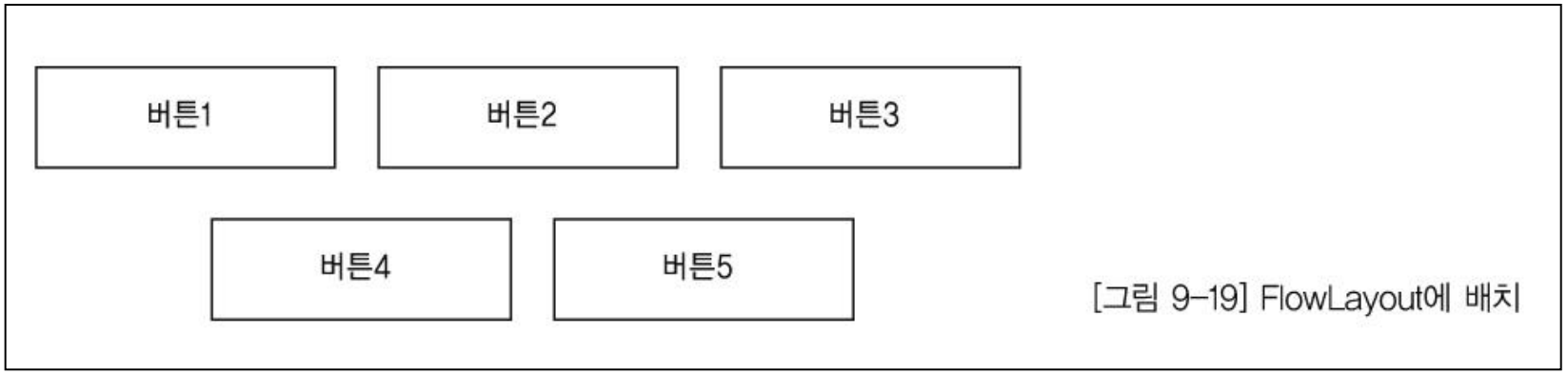
[표 9-33] 컨테이너의 기본 배치관리자

컨테이너	기본 배치관리자	컨테이너	기본 배치관리자
Frame	BorderLayout	Dialog	BorderLayout
Panel	FlowLayout	Applet	FlowLayout

Section 04 LayoutManager

FlowLayout

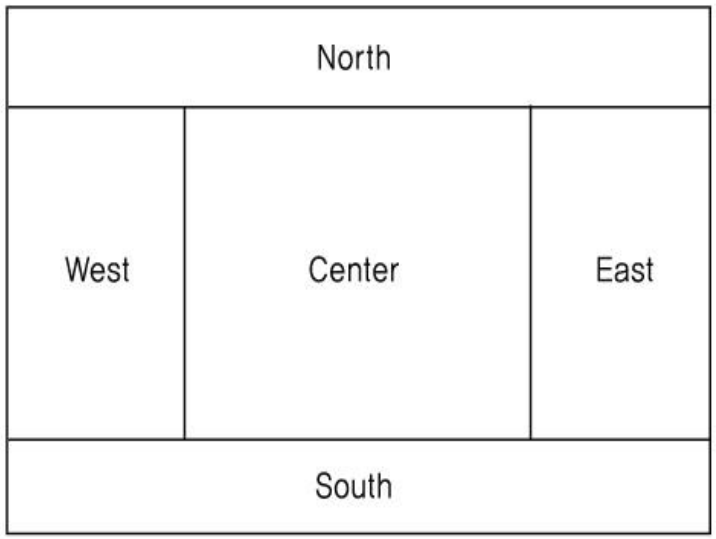
- 컴포넌트들을 수평으로 순서대로 늘어놓는 배치 기능을 가지고 있다.
- 처음에 배치를 하게되면 상단, 중앙부터 배치가 되는데 배치를 하다가 더 이상 배치할 공간이 없으면 자동으로 다음 줄로 이동하여 배치하게 된다.
- 컴포넌트를 배치할 때 컴포넌트의 간격을 갭(gap)이라고 하는데 컴포넌트들 사이의 수평, 수직간 간격을 설정할 수 있다.



[그림 9-19] FlowLayout에 배치

BorderLayout

- 컨테이너의 영역을 5개의 영역으로 분할하여 컴포넌트를 배치하는 관리자이다.
- 기본적으로 컴포넌트를 BorderLayout에 붙일 때 아무런 영역을 지정하지 않은경우는 기본적으로 CENTER영역에 붙이게 된다.
- CENTER영역은 다른영역에 아무것도 존재하지않으면 그 영역까지 포함해서 영역이 잡히게 된다.
- SOUTH, NORTH영역은 컴포넌트의 높이는 제대로 나타나지만 폭의 길이는 인정되지 않는다. WEST, EAST영역은 컴포넌트의 폭의 길이는 제대로 나타나지만 높이는 제대로 인정되지 않고 항상 그 영역의 길이만큼 잡히게 된다.



[그림 9-21] BorderLayout 경계

Section 04 LayoutManager

GridLayout

- 격자모양(모눈종이와 같은 모양)과 같이 가로와 세로가 같은 크기의 비율로 나누어 각 공간(셀)에 컴포넌트을 배치할 수 있는 관리자이다.
- GridLayout 배치 관리자를 만들 때 행과 열의 수를 지정하는데, 값은 0이상의 값으로 지하며 만약 0으로 지정하게 되면 무한대로 컴포넌트를 추가하여 붙일 수 있다.
- 행과 열의 수하고 붙이는 컴포넌트의 수가 더 많은 경우는 행의 수를 우선으로 맞춘다.

GridBagLayout

- GridLayout과 유사한 기능을 제공하는 배치 관리자로 가장 복잡한 구조를 가지고 있다.
- GirdLayout은 하나의 셀에는 하나의 컴포넌트를 가질 수 있는데 GridBagLayout은 여러 셀에 걸쳐서 서로 다른 크기와 간격으로 하나의 컴포넌트가 배치될 수 있다.
- GridBagLayout을 사용하는 경우는 GridBagConstraints 클래스를 더 사용하여 배치를 시킨다.
- GridBagConstraints 클래스는 GridLayout으로 지정된 컨테이너에 컴포넌트가 얼마만큼의 영역을 차지하여 배치할 것인가에 대한 자세한 영역 구조에 대해 지정을 한다.

[그림 9-24] GridBagLayout 배치 예

Section 04 LayoutManager

CardLayout

- 여러 개의 카드를 쌓아둔 것 처럼 컴포넌트를 하나만 보여주는 배치관리자이다.
- 맨 위의 컴포넌트만 보여주므로 한번에 하나의 컴포넌트만 볼 수 있다.
- CardLayout에는 맨 위에 위치할 컴포넌트를 지정할 수 있는 메서드가 지원되며, 또한 그 다음에 나올 컴포넌트를 이동시킬 수 있는 메서드를 지원한다.