

Java



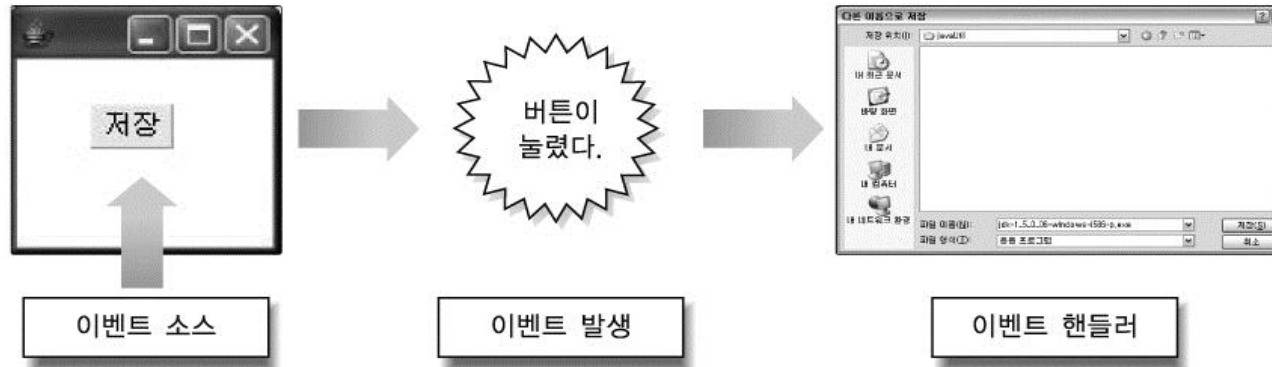
Study Point :

- AWT Event의 개념에 대해 알아본다.
- Event class 구조에 대해 알아본다.
- Event 처리 방식에 대해 알아본다.

Event Definition

- ▶ Event는 Window Programming에서 어떤 특정한 행동이 발생한 그 자체를 의미.
- ▶ 예를 들어 Menu를 선택했다든가, 아니면 Mouse를 Click하거나, Window의 크기를 조절하거나 등의 행위를 뜻하는 것.
- ▶ 이런 방식의 Programming을 Event 중심의 Programming이라고 하는데 Window Programming에서 중요한 개념 중에 하나.

Event Definition

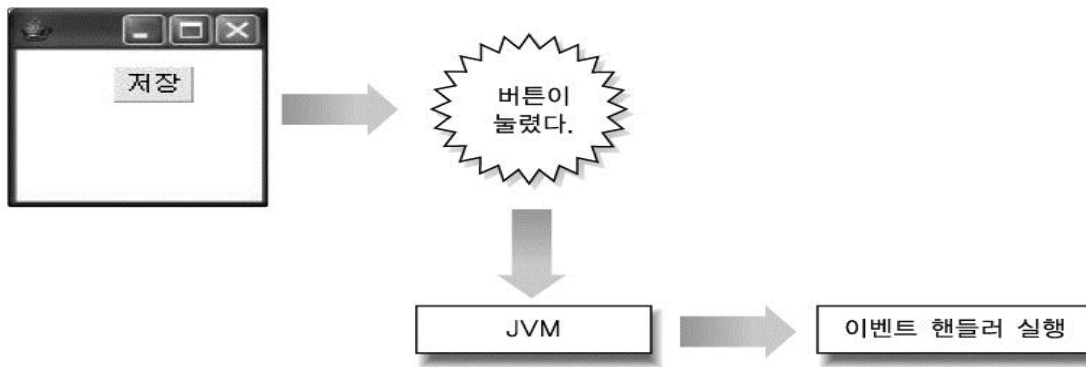


- ▶ Event Source는 Event가 발생할 수 있는 대상을 의미하고 그 대상으로부터 Event가 발생하면 발생된 Event를 처리해서 결과를 낼 수 있도록 해주는 것을 Event Handler라 한다.

Event Definition

Java에서의 Event 처리

- ▶ Program이 실행중에 OS가 해당 Program에서 Event가 발생이 되는지를 검사,
- ▶ Event가 발생되면 OS가 JVM에게 Event를 전달하고 JVM은 발생한 Event를 처리하기 위하여 Event Object를 생성,
- ▶ Event를 처리하기 위하여 Event Object를 가지고 Handler를 호출.



Event Definition

● Event Source

- ➡ Event Source는 Event가 발생하는 Component를 말한다. 즉, Button, Checkbox, List, Frame, Mouse 등과 같은 Component들이 Event Source이다.

● Event Listener

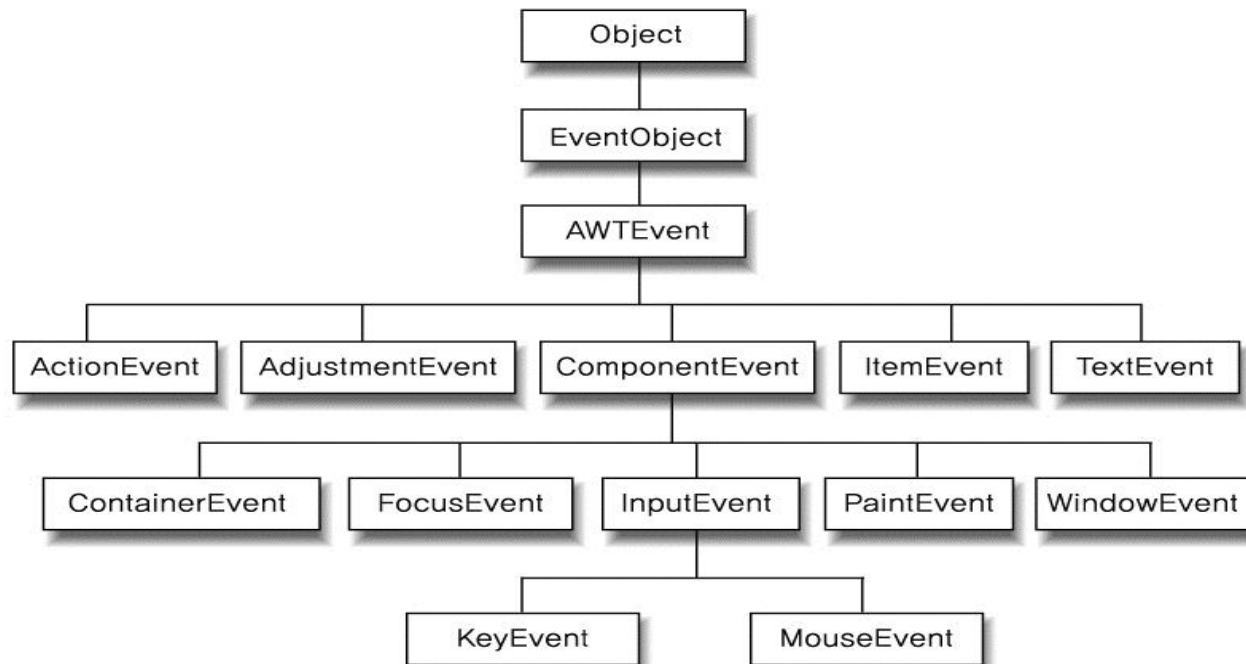
- ➡ Event Source에서 Event가 발생하는지를 검사하고 있다가 Event가 발생이 되면 실제적으로 Event를 처리할 수 있도록 만든 Interface이다.

● Event Handler

- ➡ Event Listener에 전달된 Event를 실제로 처리할 수 있도록 Event Listener에 포함되어있는 method로 발생된 Event Object를 받아와서 실제적으로 처리해주는 기능을 가지고 있다.

Event Class Structure

- 모든 Event class는 `java.util.EventObject` class로부터 상속을 받고 있으며 이 class에는 Event를 발생시킨 Object를 알려주는 `getSource()` method가 존재, 이 method는 여러 Event가 발생할 때 Event를 발생시키는 Object를 구별할 목적으로 사용.



Event Class Structure

● Event 종류 및 설명

Event	개 요
ActionEvent	Button, List, Menu 등의 Component가 눌리거나 선택이 되었을 때 발생하는 Event
AdjustmentEvent	Scrollbar와 같은 조정 가능한 Component에서 조정이 일어나면 발생하는 Event
ComponentEvent	Component의 모습이나 이동, 크기가 변화될 때 발생하는 Event
ItemEvent	List와 같은 선택항목이 있는 Component에서 선택항목이 선택될 때 발생하는 Event
TextEvent	TextComponent에서 값이 입력될 때 발생하는 Event
ContainerEvent	Container에 Component가 추가되거나 제거될 때 발생하는 Event
FocusEvent	Component에 Focus가 들어 올 때 발생하는 Event
PaintEvent	Component가 그려져야할 때 발생하는 Event
WindowEvent	Window가 활성화되거나 비활성화 될 때, 최소, 최대, 종료 될 때 발생하는 Event
KeyEvent	Keyboard로부터 입력이 될 때 발생하는 Event
MouseEvent	Mouse가 눌러지거나 움직일 때, Mouse 커서가 Component 영역에 들어가거나 벗어날 때 발생하는 Event

Event와 Listener 종류




● ActionEvent

- ➡ ActionEvent는 Button이 눌렸다는가, List, Menu 등의 Component가 선택이 되었을 때 발생하는 Event.
- ➡ TextField에서 Enter를 쳤을 때도 발생하는 Event이다.

※ ActionEvent 클래스의 주요 멤버필드

자료형	필드명	해당 키
static int	ALT_MASK	
	CTRL_MASK	
	SHIFT_MASK	

※ ActionEvent 클래스의 주요 메서드

반환형	메서드	설명
String	getActionCommand()	Action을 발생시킨 객체의 명령 문자열을 얻어온다.
int	getModifiers()	이벤트가 발생되었을 때 같이 사용된 modifier 키 ( ,  , )들을 얻어온다.

Event와 Listener 종류

ItemEvent

- ➡ Checkbox, List, Choice Component에서 항목을 선택되거나 선택이 해제 되었을 때 발생하는 Event.

※ ItemEvent 클래스의 주요 멤버필드

자료형	필드명	설명
static int	SELECTED	항목이 선택되었다는 것을 의미한다.
	DESELECTED	선택된 항목이 해제되었다는 것을 의미한다.

※ ItemEvent 클래스의 주요 메서드

반환형	메서드	설명
Object	getItem()	이벤트를 발생시킨 항목을 얻어온다.
int	getStateChange()	이벤트의 발생으로 변화된 상태를 얻어온다.

Event와 Listener 종류

● TextEvent

- ➡ Text Component(TextField, TextArea)에서 Key 입력이 되어 내용이 바뀌었을 때 발생하는 Event.
- ➡ 내용이 바뀔 마다 발생하므로 주의해서 사용해야 된다.
- ➡ User가 입력할 때마다 처리해야 되는 경우가 있을 경우에 사용하는 Event이다.

Event와 Listener 종류

● KeyEvent

- ▶ User가 Keyboard와 같은 입력장치를 통해서 Key 입력을 했을 때 발생하는 Event.

※ KeyEvent 클래스의 주요 멤버필드

자료형	필드명	해당 키
static int	VK_0~VK_9, K_A~VK_Z, VK_F1~VK_F24	각각 숫자 0~9, 영문자 A~Z, 기능키 F1~F24를 의미한다.
	VK_NUMPAD0~VK_NUMPAD9	키보드의 키패드에서의 숫자 0~9를 의미한다.
	VK_SHIFT, VK_ALT, VK_CONTROL	 ,  ,  을 의미한다.
	VK_ENTER, VK_SPACE, VK_BACK_SPACE	 ,  ,  를 의미한다.
	KEY_PRESSED	키가 눌러진 이벤트를 의미한다.
	KEY_RELEASED	키가 눌렀다 놓아진 이벤트를 의미한다.
	KEY_TYPED	키 입력 이벤트를 의미한다.

※ KeyEvent 클래스의 주요 메서드

반환형	메서드	설명
char	getKeyChar()	이벤트에 의해 입력된 문자를 얻어온다.
int	getKeyCode()	이벤트에 의해 입력된 문자에 해당하는 코드를 얻어온다.
Static String	getKeyModifiersText (int modifiers)	Modifier 키들의 상태를 보기 쉬운 텍스트로 얻어온다.
	getKeyText (int keyCode)	키 코드를 알기 쉬운 텍스트로 얻어온다.

Event와 Listener 종류

● MouseEvent

- ➡ Mouse 관련 Event은 Mouse Event와 Mouse Motion Event 두 가지,
- ➡ MouseEvent는 Mouse가 눌러지거나 Component 영역내에 들어오거나 벗어날 때 발생하는 Event,
- ➡ MouseMotionEvent는 Component의 영역 내에서 Mouse를 움직였을 때 발생하는 Event이지만 자체적으로 처리해 주는 class는 존재하지 않으며 MouseEvent class를 그대로 사용,
- ➡ MouseMotionEvent는 Mouse가 자주 이동하기 때문에 필요한 경우만 Event를 처리하는 것이 좋다,

Event와 Listener 종류

● MouseEvent class의 주요 Member Field

Field	
Data Type	Field Name
static int	MOUSE_CLICKED Mouse Button이 Click된 경우 발생하는 Event
static int	MOUSE_ENTERED Mouse Cursor가 Component 영역으로 들어왔을 때 발생하는 Event
static int	MOUSE_EXITED Mouse Cursor가 Component 영역 밖으로 나가면 발생하는 Event
static int	MOUSE_PRESSED Mouse Button이 눌러졌을 때 발생하는 Event
static int	MOUSE_RELEASED Mouse Button이 눌렀다 떼어졌을 때 발생하는 Event

Event와 Listener 종류

● Mouse 모션 Event와 관련 있는 Member Field

Field	
Data Type	Field name
static int	MOUSE_DRAGGED Mouse Button이 Click된 상태에서 동할 때 발생하는 Event
static int	MOUSE_MOVED Mouse Cursor가 움직일 때 발생하는 Event

Event와 Listener 종류

● MouseEvent class의 주요 method

method	
Return Type	method
int	<code>getClickCount()</code> Mouse가 눌러진 횟수를 얻어온다.
point	<code>getPoint()</code> Mouse Event가 발생한 좌표를 얻어온다.
int	<code>getX()</code> Mouse Event가 발생한 X좌표를 얻어온다.
int	<code>getY()</code> Mouse Event가 발생한 Y좌표를 얻어온다.
void	<code>translatePoint(int x, int y)</code> Event가 발생한 좌표에 주어진 값을 더해서 좌표를 변환한다.
boolean	<code>isPopupTrigger()</code> Mouse Event가 popup Menu를 부르는 것인지 알려준다.

Event와 Listener 종류

● WindowEvent

- ➡ Window와 관련되어 Window가 활성화, 아이콘화, 비활성화 및 창이 닫힐 때 발생하는 Event.
- ➡ AWT에서는 frame의 종료Button을 눌러도 아무런 변화가 없는 것을 볼 수 있다. 종료Button을 눌렀을 때 아무런 Event를 처리하지 않았기 때문이다.

※ WindowEvent 클래스의 멤버필드

자료형	필드명	설명
static int	WINDOW_ACTIVATED	윈도우가 활성화될 때 발생하는 이벤트
	WINDOW_DEACTIVATED	윈도우가 비활성화될 때 발생하는 이벤트
	WINDOW_CLOSED	윈도우가 닫힐 때 발생하는 이벤트
	WINDOW_CLOSING	윈도우가 사용자의 요청으로 닫힐 때 발생하는 이벤트
	WINDOW_ICONIFIED	윈도우가 아이콘화될 때 발생하는 이벤트
	WINDOW_OPENED	윈도우가 생성될 때 발생하는 이벤트

※ WindowEvent 클래스의 주요 메서드

반환형	메서드	설명
Window	getWindow()	이벤트를 발생시킨 윈도우를 얻어온다.

Event와 Listener 종류

● ActionListener

➡ ActionEvent를 처리하는 Event Listener가 ActionListener이다.

※ ActionListener 인터페이스의 주요 메서드

반환형	메서드	설명
void	actionPerformed (ActionEvent e)	컴포넌트에서 액션 이벤트가 발생했을 때 리스너에 의해 호출된다.

❖ ItemListener

▪ ItemEvent를 처리하는 Event Listener가 ItemListener이다.

※ ItemListener 인터페이스의 주요 메서드

반환형	메서드	설명
void	itemStateChanged (ItemEvent e)	컴포넌트에서 아이템의 선택 상태가 바뀌면 호출된다.

Event와 Listener 종류

● TextListener

➡ TextEvent를 처리하는 Event Listener가 TextListener이다.

※ TextListener 인터페이스의 주요 메서드

반환형	메서드	설명
void	textValueChanged(TextEvent e)	텍스트 컴포넌트의 내용이 바뀌면 호출된다.

❖ KeyListener

▪ KeyEvent를 처리하는 Event Listener가 KeyListener이다.

※ KeyListener 인터페이스의 주요 메서드

반환형	메서드	설명
void	keyPressed(KeyEvent e)	컴포넌트에 키가 눌러졌을 때 호출된다.
	keyReleased(KeyEvent e)	컴포넌트에 키가 눌러졌다가 떼어졌을 때 호출된다.
	keyTyped(KeyEvent e)	컴포넌트에 키보드를 통해 문자가 입력되었을 때 호출된다.

Event와 Listener 종류

● MouseListener

- ➡ Mouse와 관련 있는 Event 중 MouseEvent를 처리하는 Event Listener가 MouseListener이다,

※ MouseListener 인터페이스의 주요 메서드

반환형	메서드	설명
void	mouseClicked (MouseEvent e)	마우스로 컴포넌트를 클릭했을 때 호출된다.
	mouseEntered (MouseEvent e)	마우스 커서가 컴포넌트 영역에 들어오면 호출된다.
	mouseExited (MouseEvent e)	마우스 커서가 컴포넌트 영역 밖으로 나가면 호출된다.
	mousePressed (MouseEvent e)	마우스 버튼이 눌러지면 호출된다.
	mouseReleased (MouseEvent e)	마우스 버튼이 눌러졌다 떼어지면 호출된다.

Event와 Listener 종류

● MouseMotionListener

- ➡ Mouse와 관련 있는 Event 중 MouseMotionEvent를 처리하는 Event Listener가 MouseMotionListener이다.

※ MouseMotionListener 인터페이스의 주요 메서드

반환형	메서드	설명
void	mouseDragged (MouseEvent e)	마우스 버튼이 컴포넌트에서 눌러진 상태로 마우스를 이동하면 호출된다.
	mouseMoved (MouseEvent e)	마우스를 이동하면 호출된다.

Event와 Listener 종류

● WindowListener

➡ WindowEvent를 처리하는 Event Listener가 WindowListener이다.

※ WindowListener 인터페이스의 주요 메서드

반환형	메서드	설명
void	windowActivated (WindowEvent e)	윈도우가 활성화될 때 호출된다.
	windowClosed (WindowEvent e)	윈도우가 닫혀졌을 때 호출된다.
	windowClosing (WindowEvent e)	윈도우가 시스템 메뉴에 의해 닫힐 때 호출된다.
	windowDeactivated (WindowEvent e)	윈도우가 비활성화될 때 호출된다.
	windowDeiconified (WindowEvent e)	윈도우가 최소화 상태에서 원래상태로 되돌아 올 때 호출된다.
	windowIconified (WindowEvent e)	윈도우가 최소화 상태로 될 때 호출된다.
	windowOpened (WindowEvent e)	윈도우가 열릴 때 호출된다.

Event 처리

● Component Event 처리 3단계

- ▶ Event Source 결정 : 하나의 Window에는 여러 개의 Component가 존재할 수 있으므로 실제로 Event가 발생되면 처리할 Component를 결정.
- ▶ Event Listener 작성 : Event를 실제적으로 처리할 수 있도록 해당 Event를 처리할 Event Listener Interface를 이용해서 Event Listener class를 작성.
- ▶ Event Source와 Event Listener 연결 : Event Listener가 작성되면 Listener와 Event Source와 연결을 하여 Event Source에서 실제적으로 Event가 발생이 되면 처리할 수 있도록 addXXXXListener() method를 통해 연결을 시켜준다. XXXX부분은 해당 Component에 붙일 수 있는 Listener 이름을 의미.
Button에 ActionEvent를 처리하기 위하여 Button에 addActionListener를 붙이는 경우이다.

Event Adapter class

- ▶ 지금까지 우리는 Event를 처리하기 위하여 Event Listener를 등록하여 처리를 했다.
- ▶ Listener가 interface로 되어 있어 Listener에 선언되어 있는 추상 method를 모두 Override를 시켜야 사용이 가능.
- ▶ 즉, 처리하지 않는 method까지도 Override하여 처리를 해야 하니 굉장히 번거로운 작업이라고 생각할 것이다.
- ▶ 그래서 API에는 이러한 작업을 좀 더 쉽게 처리할 수 있도록 Adapter라는 class가 존재한다.
- ▶ Adapter class는 Event Listener Interface들 중에서 추상 method가 2개이상 존재하는 Interface를 구현한 추상 class이다.
- ▶ Interface에 있는 모든 method를 Empty method로 재정의 하였기 때문에 Interface를 구현하여 불필요한 method를 재정의하는 수고를 덜어준다.
- ▶ Adapter class를 상속받은 class에서는 자신이 필요한 method만을 재정의 하면 된다.

Event Adapter class

※ Adapter 클래스의 종류

이벤트	이벤트 리스너	이벤트 어댑터
ComponentEvent	ComponentListener	ComponentAdapter
ContainerEvent	ContainerListener	ContainerAdapter
FocusEvent	FocusListener	FocusAdapter
KeyEvent	KeyListener	KeyAdapter
MouseEvent	MouseListener	MouseAdapter
MouseEvent	MouseEvent	MouseEvent
MouseMotionEvent	MouseMotionListener	MouseMotionAdapter
WindowEvent	WindowListener	WindowAdapter

● Adapter class 활용

- ➡ Adapter class를 사용할 때는 예제에서 봤던 것처럼 Adapter class로부터 상속받는 class를 생성하여 처리를 하였다.
- ➡ 이 방법보다 좀 더 효율적으로 처리할 수 있도록 Anonymous class를 이용하는 방법과 Inner class를 이용하는 방법이 있다.

Study Point :

- Java에서의 GUI Programming 기법을 익히고 제작한다.
- AWT의 기본 개념 및 구조를 알아본다.
- Java에서 제공하는 Component 및 Layout Manager에 대해 알아본다.

GUI Programming :

● GUI Programming?

- ▶ GUI는 과거에 사용하였던 DOS(CUI방식)와 같은 방식의 Text 기반 OS가 아닌 Graphic을 이용하여 User와 Program 간의 상호작용을 할 수 있도록 해주는 Interface를 의미한다. Java에서 이러한 Graphic Programming을 지원하기 위해 나온 것이 바로 AWT이다.

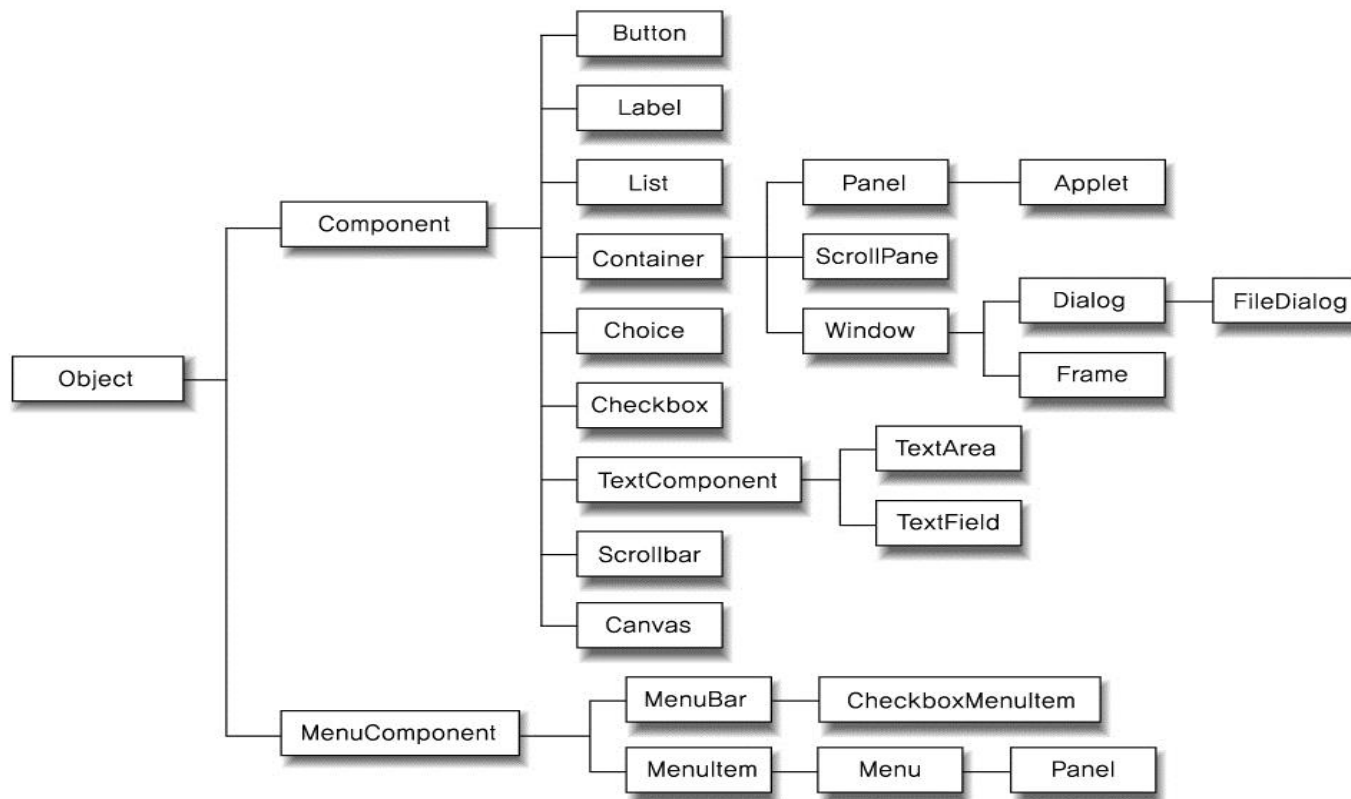
GUI Programming :

● AWT의 기본 개념

- ➡ AWT(Abstract Window Toolkit)는 GUI Programming을 제작하기 위해 Java에서 제공하는 library를 모아놓은 것이다.
- ➡ AWT는 모든 GUI Program에 사용되는 Component 및 Toolkit을 제공하고 있으며 향후에는 JFC와 같은 Swing 및 Java2D의 모태가 되는 개념.
- ➡ AWT는 OS에 구해받지 않고 쓸 수 있도록 OS의 것을 그대로 사용하지 않고 공통적이고 기본적인 Component들을 추상화시켜 제공한다.
- ➡ 실행되는 OS에 따라 다르게 보이거나 동작 방식에 차이가 있을 수 있다.
- ➡ 이러한 단점을 극복하기 위해 개발된 것이 JFC(Java Foundation Classes)이다.

GUI Programming :

• java.awt package



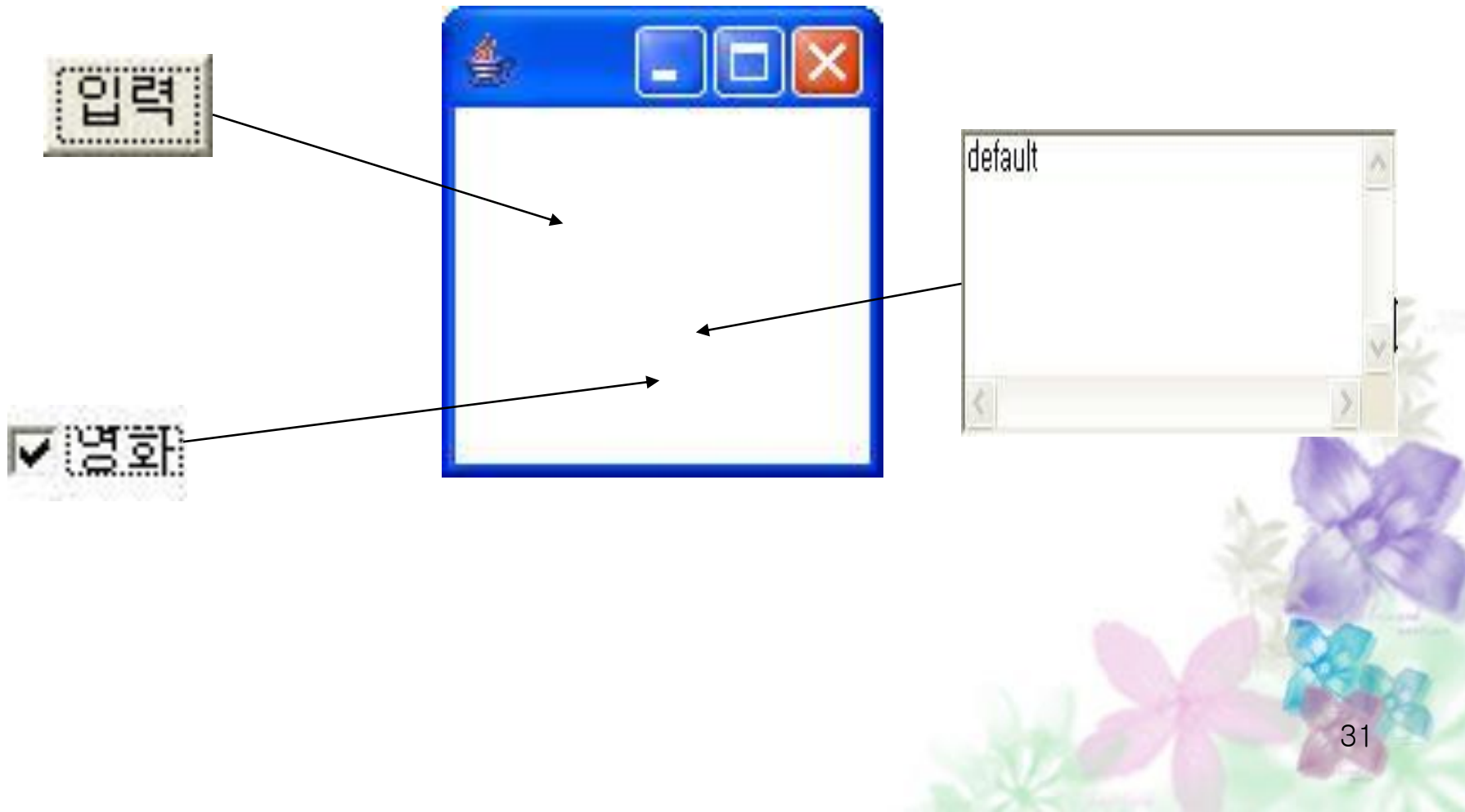
Container

● Container

- 자신의 scope에 Component를 포함시키고 관리하는 역할을 하며 Container가 다른 Container를 포함할 수도 있다.
- Component도 또한 Container에 부착시키지 않으면 독자적으로 화면에 출력될 수가 없고 반드시 Container에 부착을 시켜야만 화면에 출력이 될 수 있다.
- Container의 종류에는 Frame, Window, Panel, Applet, Dialog, FileDialog, ScrollPane이 있다.
- Container에 Component를 부착시키기 위해 add() method를 사용한다.

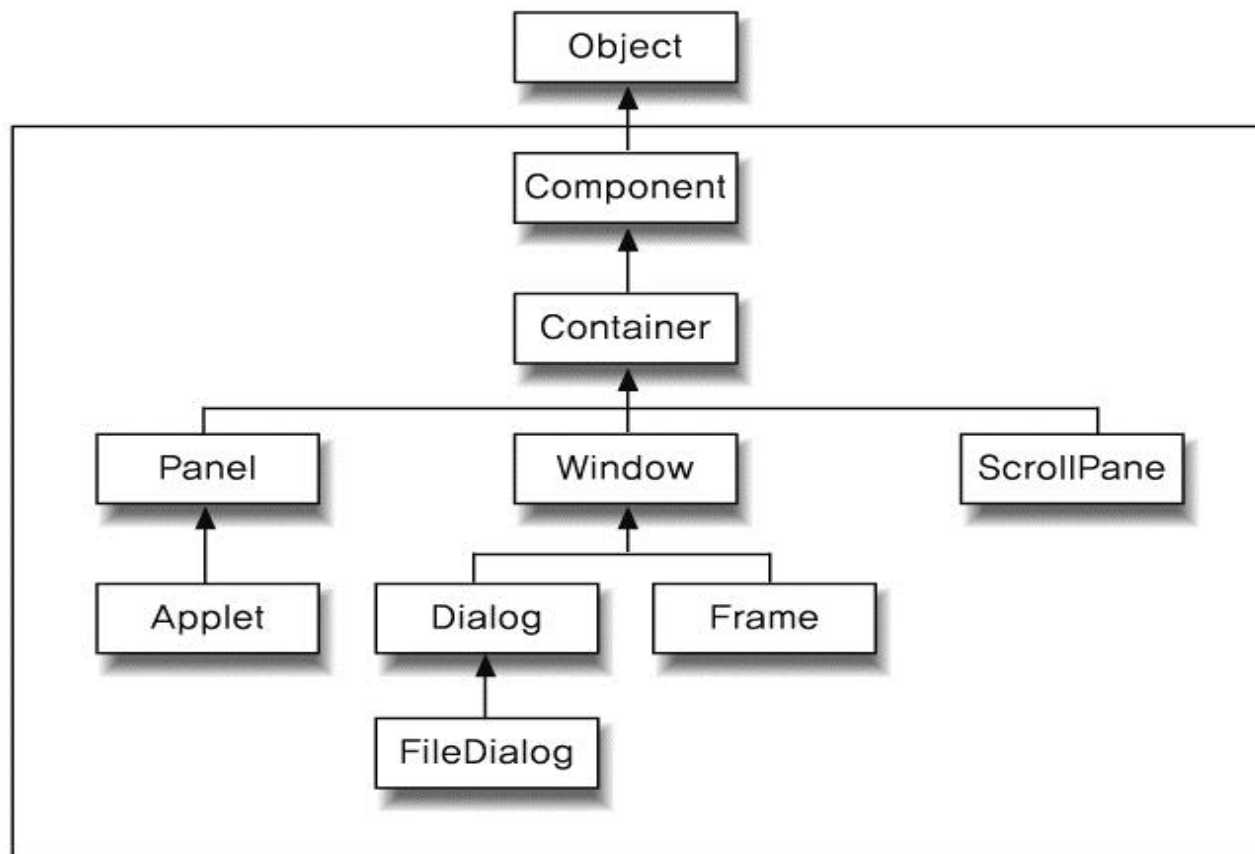
Container

- Container와 Component 관계



Container

● Container 종류 및 상속관계



Container

● Frame

- ▶ Window class의 하위 class로 일반적인 Application에서 window를 생성하기 위해 사용되는 class이다.
- ▶ Frame class의 상위 class인 Window class는 title, menu 등이 지원되지 않기 때문에 일반적으로 사용하지 않고 Frame class를 사용한다.
- ▶ Frame class는 기본적으로 Border, title, menu, System Box(최소화, 최대화, 종료 버튼) 등의 기능을 제공한다.
- ▶ Frame은 다른 window에 속해 있지 않은 window로 최상위 레벨 window라 한다.
- ▶ setSize(), setBounds() method 등을 이용해서 Window의 크기를 설정한 후 setVisible(), show() method를 통해서 화면에 출력시킬 수 있다.

Container

● Frame class의 Constructor

※ Frame 클래스의 주요 생성자

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

Container

Frame class의 주요method

반환형	메서드명	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.

Container

● Panel class

- ▶ Component들을 Group별로 묶어서 처리할 때 주로 사용한다.
- ▶ Frame에 Component를 직접 붙이지 않고 Panel에 그룹별로 붙이고, 다시 Panel을 Frame에 붙이는 경우가 많다.
- ▶ 다른 Panel을 생성하여 자신에게 붙일 수도 있어 window Program을 만들때는 여러 개의 Panel을 사용하는 경우가 많다.

❖ Panel class의 Constructor

※ Panel 클래스의 주요 생성자

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

Container

● Panel class의 주요method

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

Container

● Dialog class

- ➡ 메인 window 외에 Message를 출력하거나, User로부터 Data를 입력 받을 때 주로 사용하는 Container이다.
- ➡ 보통은 Dialog class로부터 상속을 받아 새로운 기능을 가진 대화상자를 만드는데 사용된다.

❖ Dialog class의 Constructor

※ Dialog 클래스의 주요 생성자

생성자	설명
Dialog(Dialog owner)	생성되는 Dialog 객체를 소유하는 객체가 owner인 Frame을 생성
Dialog(Dialog owner, String title)	생성되는 Dialog 객체의 소유자를 owner라는 객체로 설정하고 타이틀을 설정
Dialog(Dialog owner, String title, boolean modal)	소유자로 owner 객체를 설정하고 타이틀을 가지며, 모달인지 모달이 아닌지를 설정하여 Dialog 객체를 생성
Dialog(Frame owner)	생성되는 Dialog 객체를 소유하는 객체가 owner라는 Frame 객체를 생성

Container

● Dialog class의 주요method

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

❖ Component

- 모든 Component들의 super class로서 GUI Program을 구성하는 구성단위로 각 Component들에서 공통으로 사용되어지는 method들을 가지고 있다.

Component

● Component class의 주요method

※ 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

● Component class의 주요method

※ 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.

Component

● Component class의 주요method

※ Component 클래스의 설정과 관련있는 주요메서드

반환형	메서드	설명
void	setEnabled(boolean b)	파라미터 b값에 의해 컴포넌트의 활성화와 비활성화를 지정한다.
	setVisible(Booleann 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.

Component

● Basic Component

※ 기본 컴포넌트의 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.

Component

● Button

- 버튼을 사용자가 눌렀을 때 특정한 액션을 실행할 수 있도록 만든 Component이다.

❖ Button class의 Constructor

※ Button 클래스의 주요 생성자

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

Component

● Button class의 주요method

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

Component

❖ Checkbox

- ▶ User가 여러 종류의 옵션을 선택할 것인지의 여부를 지정할 때 사용.
- ▶ 여러 개의 Checkbox를 묶어 하나의 group으로 만들어 group내에서는 하나만이 값을 유지할 수 있는 Radiobutton 형태로도 사용할 수 있는 Component.
- ▶ group으로 묶을 때는 CheckboxGroup class를 사용.

❖ Checkbox class의 Constructor

※ 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.

Component

● Checkbox class의 주요method

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.

Component

● Choice

- ▶ List class와 거의 유사한 기능을 가지고 있는 Component로 User가 drop-down button을 사용하여 여러item중에 하나를 선택할 수있는 기능을 제공.
- ▶ Component를 생성한 후에 drop-down list에 항목에 추가시켜 사용한다.

❖ Choice class의 Constructor

※ Choice 클래스의 주요 생성자

생성자	설명
Choice()	새로운 선택 메뉴 객체를 생성한다.

Component

● Choice class의 주요method

반환형	메서드	설명
void	add(String item)	Choice 메뉴에 항목을 추가한다.
	addItemListener (ItemListener I)	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 지정된 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.

Component

Label

- ▶ 사각형의 scope에 문자열을 표시할 때 사용하는 Component이다.
- ▶ Label은 경계선이 없고 특별한 상태를 가지지도 않는다, 그러므로 Label을 Container에 포함시키게 되면 Label의 문자만 화면에 표시가 된다.
- ▶ Label의 문자열은 좌, 우, 중앙으로 정렬시킬 수 있다.

❖ Label class의 주요 Member Field

※ 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.

Component

● Label class의 Constructor

※ 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 class의 주요 method

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

Component

● List

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

❖ List class의 Constructor

※ 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.

Component

● List class의 주요method

반환형	메서드	설명
void	add(String item)	지정한 항목을 List의 끝에 추가한다.
	add(String item, int index)	List의 지정된 위치에 항목을 삽입한다.
	addItemListener (ItemListener I)	List로부터 아이템 이벤트를 받기 위해 지정된 아이템 리스너를 추가한다.
	remove(int position)	List에 지정한 위치에 있는 항목을 제거한다.
	remove(String item)	List로부터 item이 첫번째로 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 항목들의 문자열을 배열로 얻어온다.
void	select(int index)	지정한 위치의 항목을 선택한다.
	replaceItem(String newValue, int index)	지정한 위치의 항목의 newValue값을 바꾼다.

Component

● Canvas

- ▶ 특정한 모양을 갖고 있지 않고 사각형의 scope만을 갖고 있는 Component, 그림을 그릴 수 있는 도화지의 역할을 하는 Component로 Container에 포함되어 Graphic 처리를 할 수 있다.

❖ Canvas class의 Constructor

※ Canvas 클래스의 주요 생성자

생성자	설명
Canvas()	새로운 캔버스 객체를 생성
Canvas (GraphicsConfiguration gc)	화면 장치의 GraphicsConfiguration을 이용하여 캔버스 객체를 생성

❖ Canvas class의 주요method

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

Component

● TextComponent

- ➡ Text를 다루는 class의 super class로 Text를 처리하는 method를 제공.
- ➡ 독립적으로는 생성되지는 못한다.

❖ TextComponent class의 주요method

※ 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.

Component

● TextComponent class의 주요 method

※ 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.

※ TextComponent 클래스의 하위 클래스

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

Component

● TextField

- ▶ 한 줄 내의 Text를 입력 받거나 편집할 수 있는 Component.
- ▶ 한 줄에 표시할 수 있는 Column수를 지정할 수 있고 Echo Character를 지정하면 입력되는 문자대신 Echo Character로 지정한 문자로 출력된다.

❖ TextField class의 Constructor

※ 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.

Component

● TextField class의 주요method

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.

Component

● TextArea

- ▶ 여러 줄의 Text를 사용자로 부터 입력받거나 편집할 수 있는 Component.
- ▶ 화면에 출력되는 scope이 벗어나면 Scrollbar 표시 방식에 따라 자동으로 Scrollbar가 생성된다.
- ▶ 사용자가 필요에 따라 일부 Scrollbar만 나타나게 할 수도 있다.

❖ TextArea class의 주요 Member Field

※ TextArea 클래스의 주요 멤버 필드

자료형	필드명	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.

Component

● TextArea class의 Constructor

※ TextArea 클래스의 주요 생성자

생성자	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.

Component

● TextArea class의 주요method

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.

Component

■ Menu Component

- ▶ Menu는 보통 최상위 Level의 window Titlebar 아래에 존재하는 것으로 User가 Program의 기능을 선택할 수 있도록 해주는 기능을 가지고 있는 Component.
- ▶ Menu의 구성은 MenuBar, Menu, MenuItem으로 구성된다.

❖ MenuComponent class의 Sub class

※ MenuComponent 클래스의 하위 클래스

종류	기능
MenuBar	메뉴를 올려 놓을 수 있는 메뉴바를 만들 때 사용한다.
Menu	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.

Component

● 메뉴 사용법

1. MenuBar Object를 생성.

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

2. MenuBar에 삽입할 Menu를 생성한 후 Menu를 MenuBar에 붙인다,

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

3. Menu에 붙일 MenuItem을 생성한 후 해당 Menu에 붙인다.

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

4. MenuBar를 window에 붙인다.

```
setMenuBar(mb);
```

LayoutManager

● LayoutManager

- ➡ Container는 자기 자신에 Component를 붙일 때 어디에, 어떤 방식으로 배치하여 붙일것인가를 이미 결정하고 있다.
- ➡ 미리 정해진 Layout에 따라 Component들을 자동으로 배치하는 기능을 가지고 있는 Object를 Container들은 가지고 있는데 이것을 Layout Manager라 한다.
- ➡ Java 에서 사 용 하 는 Layout Manager 는 FlowLayout, BorderLayout, GridLayout, GridBagLayout, CardLayout의 5가지가 있다.
- ➡ Layout Manager는 각자 다른 방식으로 배치기능을 가지고 있으며 Container는 기본적으로 하나의 Layout Manager를 가지고 있다.
- ➡ 사용자가 임의로 Layout Manager는 다시 설정할 수 있으며 Layout Manager를 제거하고 수동으로 좌표를 이용해서 배치할 수도 있다.

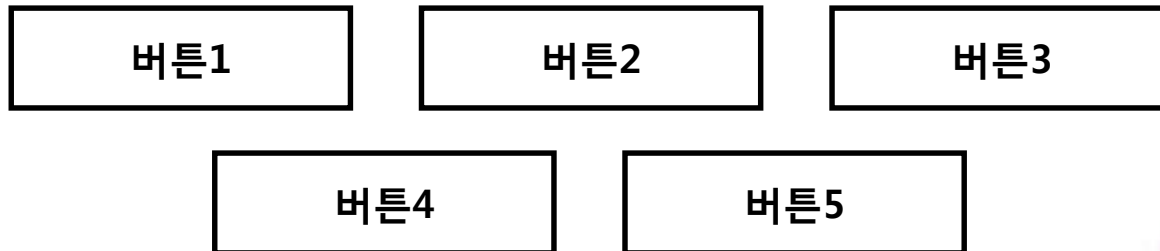
※ 컨테이너의 기본 배치관리자

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

LayoutManager

● FlowLayout

- ▶ Component들을 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.
- ▶ Component를 배치할 때 Component의 간격을 gap이라고 하는데 Component들 사이의 수평, 수직간 간격을 설정할 수 있다.

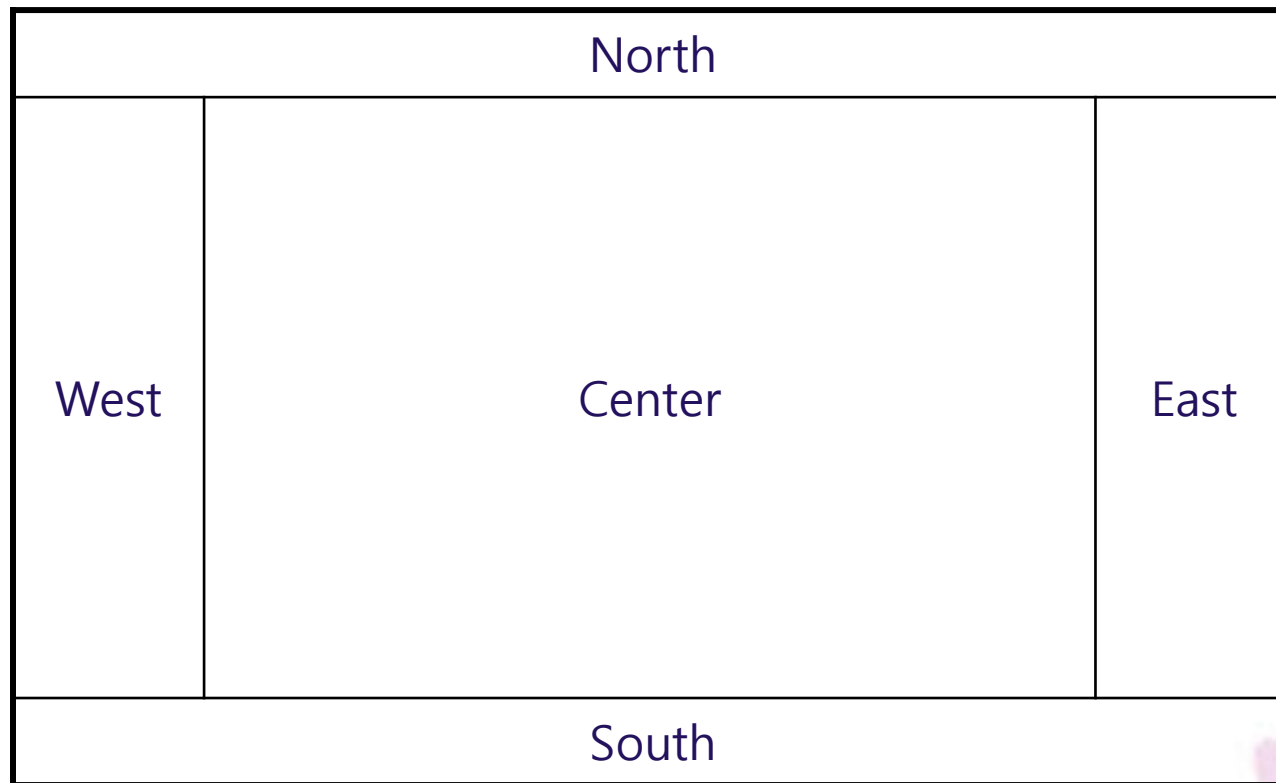


BorderLayout

- ▶ Container의 scope을 5개의 scope으로 분할하여 Component를 배치하는 관리자이다,
- ▶ 기본적으로 Component를 BorderLayout에 붙일 때 아무런 scope을 지정하지 않은경우는 기본적으로 CENTERscope에 붙이게 된다,
- ▶ CENTERscope은 다른scope에 아무것도 존재하지않으면 그 scope까지 포함해서 scope이 잡히게 된다,
- ▶ SOUTH, NORTHscope은 Component의 높이는 제대로 나타나지만 폭의 길이는 인정되지 않는다,
- ▶ WEST, EASTscope은 Component의 폭의 길이는 제대로 나타나지만 높이는 제대로 인정되지 않고 항상 그 scope의 길이만큼 잡히게 된다,

LayoutManager

● BorderLayout



※ BorderLayout 경계

LayoutManager

■ GridLayout

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

LayoutManager

■ GridBagLayout

- ▶ GridLayout과 유사한 기능을 제공하는 배치 관리자로 가장 복잡한 구조를 가지고 있다.
- ▶ GridLayout은 하나의 cell에는 하나의 Component를 가질 수 있는데 GridBagLayout은 여러 cell에 걸쳐서 서로 다른 크기와 간격으로 하나의 Component가 배치될 수 있다.
- ▶ GridBagLayout을 사용하는 경우는 GridBagConstraints class를 더 사용하여 배치를 시킨다.
- ▶ GridBagConstraints class는 GridLayout으로 지정된 Container에 Component가 얼마만큼의 scope을 차지하여 배치할 것인가에 대한 자세한 scope 구조에 대해 지정을 한다.

컴포넌트 1		컴포넌트 2
컴포넌트 3		
컴포넌트 4	컴포넌트 5	
	컴포넌트 6	

※ GridBagLayout 배치 예

LayoutManager

● CardLayout

- ▶ 여러 개의 카드를 쌓아둔 것 처럼 Component를 하나만 보여주는 Layout Manager이다.
- ▶ 맨 위의 Component만 보여주므로 한번에 하나의 Component만 볼 수 있다.
- ▶ CardLayout에는 맨 위에 위치할 Component를 지정할 수 있는 method가 지원되며, 또한 그 다음에 나올 Component를 이동시킬 수 있는 method를 지원한다.

영

어

하

시

다

^^!