



AWT 이벤트

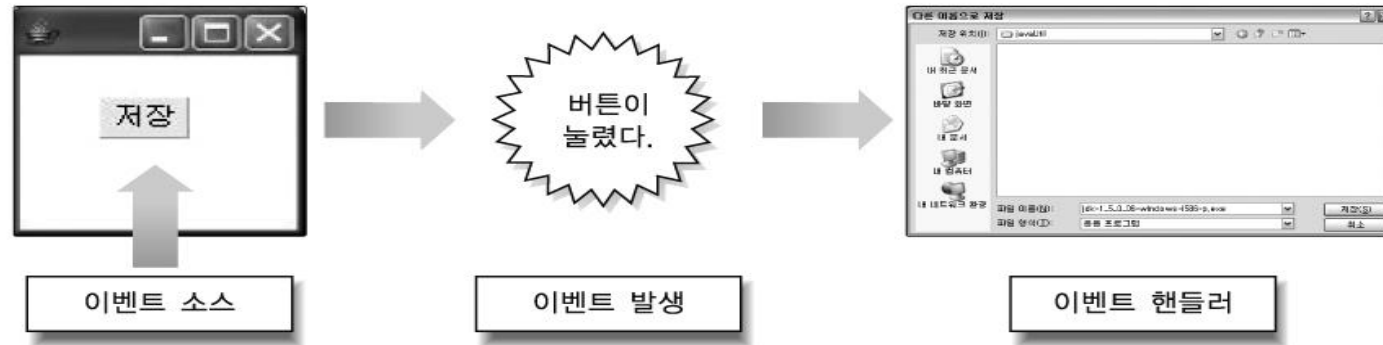
학습목표

- AWT 이벤트의 개념에 대해 알아본다.
- 이벤트 클래스 구조에 대해 알아본다.
- 이벤트 처리 방식에 대해 알아본다.

이벤트의 정의

- 이벤트(Event)라는 것은 윈도우 프로그래밍에서 어떤 특정한 행동이 발생한 그 자체를 의미한다.
- 예를 들어 메뉴를 선택했다는가, 아니면 마우스를 클릭하거나, 윈도우의 크기를 조절하거나 등의 행위를 뜻하는 것이다.
- 이러한 방식의 프로그래밍을 이벤트 중심의 프로그래밍이라고 하는데 윈도우 프로그래밍에서 중요한 개념중에 하나이다.

이벤트의 정의

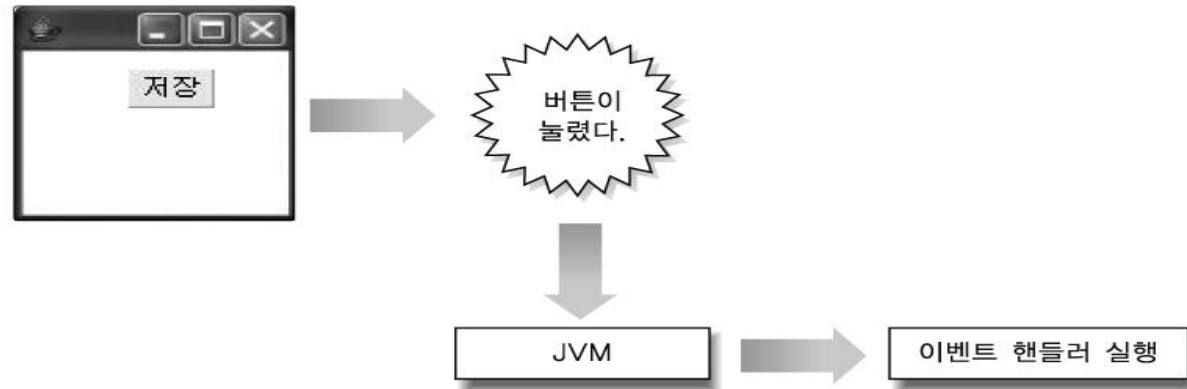


[그림 10-1] 이벤트 처리 구조

- 이벤트 소스(Event Source)는 이벤트가 발생할 수 있는 대상을 의미하고 그 대상으로부터 이벤트가 발생하면 발생된 이벤트를 처리해서 결과를 낼 수 있도록 해주는 것을 이벤트 핸들러(Event Handler)라 한다.

이벤트의 정의

- 자바에서의 이벤트 처리
 - 프로그램이 실행중에 운영체제(OS)가 해당 프로그램에서 이벤트가 발생이 되는지를 검사한다.
 - 이때 이벤트가 발생되면 운영체제가 JVM에게 이벤트를 전달하고 JVM은 발생된 이벤트를 처리하기 위하여 이벤트 객체를 생성한다.
 - 그후 이벤트 처리하기 위하여 이벤트 객체를 가지고 핸들러를 호출한다.



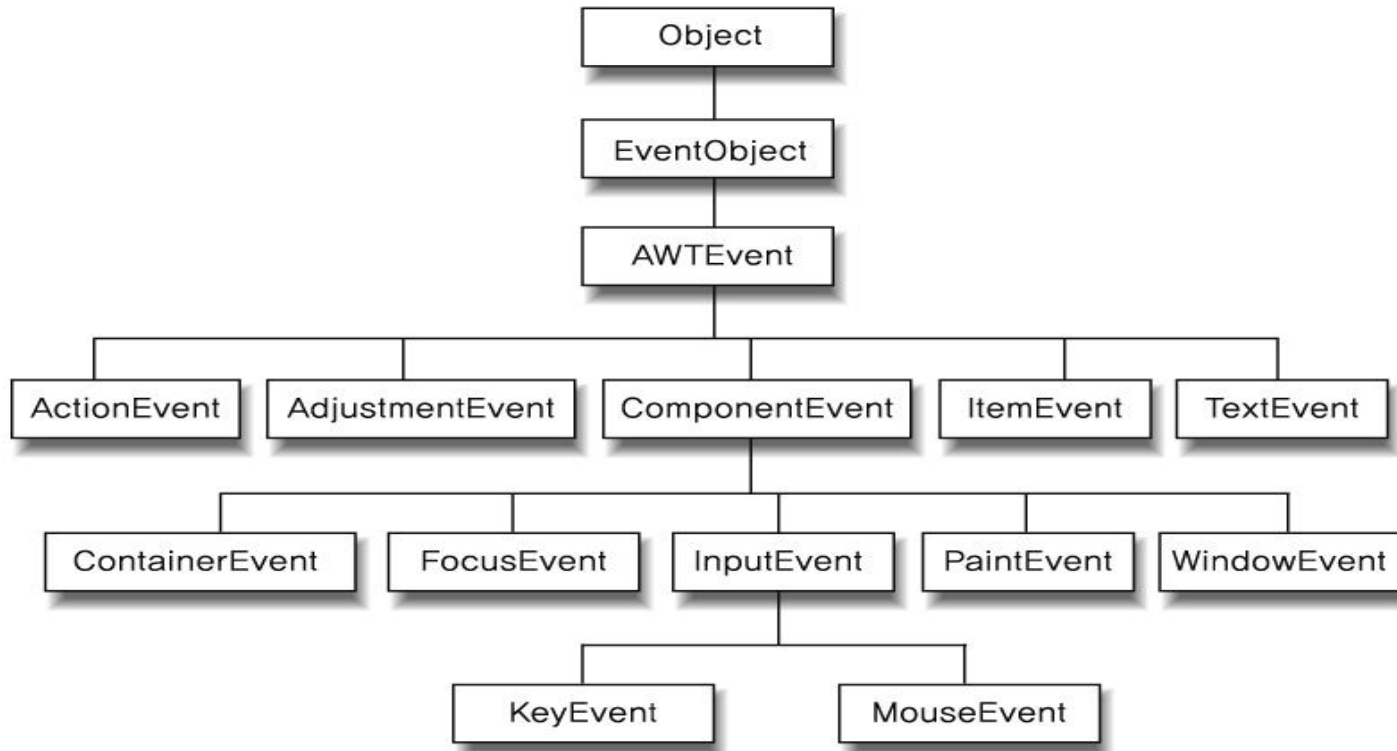
[그림 10-2] 자바에서의 이벤트 처리 구조

이벤트의 정의

- 이벤트 소스(Event Source)
 - 이벤트 소스는 이벤트가 발생하는 컴포넌트를 말한다. 즉, 버튼, 체크박스, 리스트, 프레임, 마우스 등과 같은 컴포넌트들이 이벤트 소스이다.
- 이벤트 리스너(Event Listener)
 - 이벤트 소스에서 이벤트가 발생하는지를 검사하고 있다가 이벤트가 발생이 되면 실제적으로 이벤트를 처리할 수 있도록 만든 인터페이스이다.
- 이벤트 핸들러(Event Handler)
 - 이벤트 리스너에 전달된 이벤트를 실제로 처리할 수 있도록 이벤트 리스너에 포함되어있는 메서드로 발생된 이벤트 객체를 받아와서 실제적으로 처리해주는 기능을 가지고 있다.

이벤트 클래스 구조

- 모든 이벤트 클래스는 `java.util.EventObject` 클래스로부터 상속을 받고 있으며 이 클래스에는 이벤트를 발생시킨 객체를 알려주는 `getSource()` 메서드가 존재한다. 이 메서드는 여러 이벤트가 발생할 때 이벤트를 발생시키는 객체를 구별할 목적으로 사용한다.



[그림 10-3] 이벤트 클래스 구조

이벤트 클래스 구조

- 이벤트 종류 및 설명

이벤트	개 요
ActionEvent	버튼, 리스트, 메뉴 등의 컴포넌트가 눌리거나 선택이 되었을 때 발생하는 이벤트
AdjustmentEvent	스크롤바와 같은 조정 가능한 컴포넌트에서 조정이 일어나면 발생하는 이벤트
ComponentEvent	컴포넌트의 모습이나 이동, 크기가 변화될 때 발생하는 이벤트
ItemEvent	리스트와 같은 선택항목이 있는 컴포넌트에서 선택항목이 선택될 때 발생하는 이벤트
TextEvent	텍스트 컴포넌트에서 값이 입력될 때 발생하는 이벤트
ContainerEvent	컨테이너에 컴포넌트가 추가되거나 제거될 때 발생하는 이벤트
FocusEvent	컴포넌트에 초점(Focus)이 들어 올 때 발생하는 이벤트
PaintEvent	컴포넌트가 그려져야할 때 발생하는 이벤트
WindowEvent	윈도우가 활성화되거나 비활성화 될 때, 최소, 최대, 종료 될 때 발생하는 이벤트
KeyEvent	키보드로부터 입력이 될 때 발생하는 이벤트
MouseEvent	마우스가 눌러지거나 움직일 때, 마우스 커서가 컴포넌트 영역에 들어가거나 벗어날 때 발생하는 이벤트

이벤트와 리스너 종류




- **ActionEvent**

- ActionEvent는 버튼이 눌렸다는가, 리스트, 메뉴 등의 컴포넌트가 선택이 되었을 때 발생하는 이벤트이다.
- 텍스트 필드에서 엔터를 쳤을 때도 발생하는 이벤트이다.

[표 10-1] ActionEvent 클래스의 주요 멤버필드

자료형	필드명	해당 키
static int	ALT_MASK	
	CTRL_MASK	
	SHIFT_MASK	

[표 10-2] 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.

이벤트와 리스너 종류

- ItemEvent
 - 체크박스, 리스트, 초이스 컴포넌트에서 항목을 선택되거나 선택이 해제 되었을 때 발생하는 이벤트이다.

[표 10-3] ItemEvent 클래스의 주요 멤버필드

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

[표 10-4] ItemEvent 클래스의 주요 메서드

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

이벤트와 리스너 종류

- TextEvent
 - 텍스트 컴포넌트(TextField, TextArea)에서 키가 입력이 되어 내용이 바뀌었을 때 발생하는 이벤트이다.
 - 내용이 바뀔때마다 발생하므로 주의해서 사용해야 된다.
 - 사용자가 입력할때마다 처리해야 되는 경우가 있을경우에 사용하는 이벤트이다.

이벤트와 리스너 종류

- KeyEvent
 - 사용자가 키보드와 같은 입력장치를 통해서 키 입력을 했을 때 발생하는 이벤트이다.

[표 10-5] KeyEvent 클래스의 주요 멤버필드

자료형	필드명	해당 키
static int	VK_0~VK_9, VK_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	Shift , Alt , Control 을 의미한다.
static int	VK_ENTER, VK_SPACE, VK_BACK_SPACE	Enter , Space , Back Space 를 의미한다.
	KEY_PRESSED	키가 눌러진 이벤트를 의미한다.
	KEY_RELEASED	키가 눌렀다 놓아진 이벤트를 의미한다.
	KEY_TYPED	키 입력 이벤트를 의미한다.

[표 10-6] 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.

이벤트와 리스너 종류

- MouseEvent
 - 마우스 관련 이벤트는 마우스 이벤트와 마우스 모션 이벤트 두가지가 있다.
 - MouseEvent는 마우스가 눌러지거나 컴포넌트 영역내에 들어오거나 벗어날 때 발생하는 이벤트이다.
 - MouseMotionEvent는 컴포넌트의 영역 내에서 마우스를 움직였을 때 발생하는 이벤트이지만 자체적으로 처리해 주는 클래스는 존재하지 않으며 MouseEvent 클래스를 그대로 사용한다.
 - MouseMotionEvent는 마우스가 자주 이동하기 때문에 필요한 경우만 이벤트를 처리하는 것이 좋다.

이벤트와 리스너 종류

- MouseEvent 클래스의 주요 멤버필드

필드	
자료형	필드명
static int	MOUSE_CLICKED 마우스 버튼이 클릭된 경우 발생하는 이벤트
static int	MOUSE_ENTERED 마우스 커서가 컴포넌트 영역으로 들어왔을 때 발생하는 이벤트
static int	MOUSE_EXITED 마우스 커서가 컴포넌트 영역 밖으로 나가면 발생하는 이벤트
static int	MOUSE_PRESSED 마우스 버튼이 눌러졌을 때 발생하는 이벤트
static int	MOUSE_RELEASED 마우스 버튼이 눌렀다 떼어졌을 때 발생하는 이벤트

이벤트와 리스너 종류

- 마우스 모션 이벤트와 관련 있는 멤버 필드

필드	
자료형	필드명
static int	MOUSE_DRAGGED 마우스 버튼이 클릭된 상태에서 동할 때 발생하는 이벤트
static int	MOUSE_MOVED 마우스 커서가 움직일 때 발생하는 이벤트

이벤트와 리스너 종류

- MouseEvent 클래스의 주요 메서드

메서드	
반환형	메서드
Int	getClickCount() 마우스가 눌러진 횟수를 얻어온다.
point	getPoint() 마우스 이벤트가 발생한 좌표를 얻어온다.
int	getX() 마우스 이벤트가 발생한 X좌표를 얻어온다.
int	getY() 마우스 이벤트가 발생한 Y좌표를 얻어온다.
void	translatePoint(int x, int y) 이벤트가 발생한 좌표에 주어진 값을 더해서 좌표를 변환한다.
boolean	isPopupTrigger() 마우스 이벤트가 팝업 메뉴를 부르는 것인지 알려준다.

이벤트와 리스너 종류

- WindowEvent
 - 윈도우와 관련되어 윈도우가 활성화, 아이콘화, 비활성화 및 창이 닫힐 때 발생하는 이벤트이다.
 - AWT에서는 프레임의 종료버튼을 눌러도 아무런 변화가 없는 것을 볼 수 있을 것이다. 그것이 바로 종료버튼을 눌렀을 때 아무런 이벤트를 처리하지 않았기 때문이다.

[표 10-9] 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

[표 10-10] WindowEvent 클래스의 주요 메서드

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

이벤트와 리스너 종류

- ActionListener
 - ActionEvent를 처리하는 이벤트 리스너가 ActionListener이다.

[표 10-11] ActionListener 인터페이스의 주요 메서드

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

이벤트와 리스너 종류

- ItemListener
 - ItemEvent를 처리하는 이벤트 리스너가 ItemListener이다.

[표 10-12] ItemListener 인터페이스의 주요 메서드

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

이벤트와 리스너 종류

- TextListener
 - TextEvent를 처리하는 이벤트 리스너가 TextListener이다.

[표 10-13] TextListener 인터페이스의 주요 메서드

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

이벤트와 리스너 종류

- KeyListener
 - KeyEvent를 처리하는 이벤트 리스너가 KeyListener이다.

[표 10-14] 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.

이벤트와 리스너 종류

- MouseListener
 - 마우스와 관련 있는 이벤트 중 MouseEvent를 처리하는 이벤트 리스너가 MouseListener이다.

[표 10-15] MouseListener 인터페이스의 주요 메서드

반환형	메서드	설명
void	mouseClicked(MouseEvent e)	마우스로 컴포넌트를 클릭했을 때 호출된다.
	mouseEntered(MouseEvent e)	마우스 커서가 컴포넌트 영역에 들어 오면 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.

이벤트와 리스너 종류

- `MouseMotionListener`
 - 마우스와 관련 있는 이벤트 중 `MouseEvent`를 처리하는 이벤트 리스너가 `MouseMotionListener`이다.

[표 10-16] `MouseMotionListener` 인터페이스의 주요 메서드

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

이벤트와 리스너 종류

- WindowListener
 - WindowEvent를 처리하는 이벤트 리스너가 WindowListener이다.

[표 10-17] 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.

이벤트 처리

- 컴포넌트 별 이벤트 종류

[표 10-19] 컴포넌트별 이벤트 종류

이벤트 컴포넌트	Action	Com ponent	Con tainer	Focus	Item	Key	Mouse	Mouse motion	Text	Win dow
Button	0	0		0		0	0	0		
Canvas		0		0		0	0	0		
Checkbox		0		0	0	0	0	0		
Choice		0		0	0	0	0	0		
Component		0		0		0	0	0		
Container		0	0	0		0	0	0		
Frame		0	0	0		0	0	0		0
Label		0		0		0	0	0		
List	0	0		0	0					
MenuItem	0					0	0	0		
Scrollbar		0		0		0	0	0		
TextArea		0		0		0	0	0	0	
TextField	0	0		0		0	0	0	0	

이벤트 처리

- 컴포넌트 이벤트 처리 3단계
 - 이벤트 소스 결정 : 하나의 윈도우에는 여러 개의 컴포넌트가 존재할 수 있으므로 실제로 이벤트가 발생되면 처리할 컴포넌트를 결정한다.
 - 이벤트 리스너 작성 : 이벤트를 실제로 처리할 수 있도록 해당 이벤트를 처리할 이벤트 리스너 인터페이스를 이용해서 이벤트 리스너 클래스를 작성한다.
 - 이벤트 소스와 이벤트 리스너 연결 : 이벤트 리스너가 작성이되면 리스너와 이벤트 소스와 연결을 하여 이벤트 소스에서 실제로 이벤트가 발생이 되면 처리할 수 있도록 addXXXXListener()함수를 통해 연결을 시켜준다. XXXX부분은 해당 컴포넌트에 붙일 수 있는 리스너 이름을 의미한다. 예를 들어 버튼에 ActionEvent을 처리하기 위하여 버튼에다가 addActionListener를 붙이는 경우이다.

이벤트 어댑터 클래스

- 지금까지 우리는 이벤트를 처리하기 위하여 이벤트 리스너를 등록하여 처리를 하였다.
- 리스너가 인터페이스로 되어 있어 리스너에 선언되어 있는 추상메서드를 모두 오버라이드를 시켜야 사용이 가능한 것을 알았다.
- 즉, 처리하지 않는 메서드까지도 오버라이드하여 처리를 해야하니 굉장히 번거로운 작업이라고 생각할 것이다.
- 그래서 API에는 이러한 작업을 좀 더 쉽게 처리할 수 있도록 Adapter라는 클래스가 존재한다.
- Adapter 클래스는 이벤트 리스너 인터페이스들 중에서 추상메서드가 2개 이상 존재하는 인터페이스를 구현한 추상 클래스이다.
- 인터페이스에 있는 모든 메서드를 빈(Empty) 메서드로 재정의 하였기 때문에 인터페이스를 구현하여 불필요한 메서드를 재정의하는 수고를 덜어준다.
- Adapter 클래스를 상속받은 클래스에서는 자신이 필요한 메서드만을 재정의 하면 된다.

이벤트 어댑터 클래스

[표 10-19] Adapter 클래스의 종류

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

이벤트 어댑터 클래스

- Adapter 클래스 활용
 - Adapter 클래스를 사용할 때는 예제에서 봤던 것 처럼 Adapter클래스로부터 상속받는 클래스를 생성하여 처리를 하였다.
 - 이 방법보다 좀 더 효율적으로 처리할 수 있도록 Anonymous 클래스를 이용하는 방법과 Inner 클래스를 이용하는 방법이 있다.