Java Core

제 17 강

Swing 실전예제

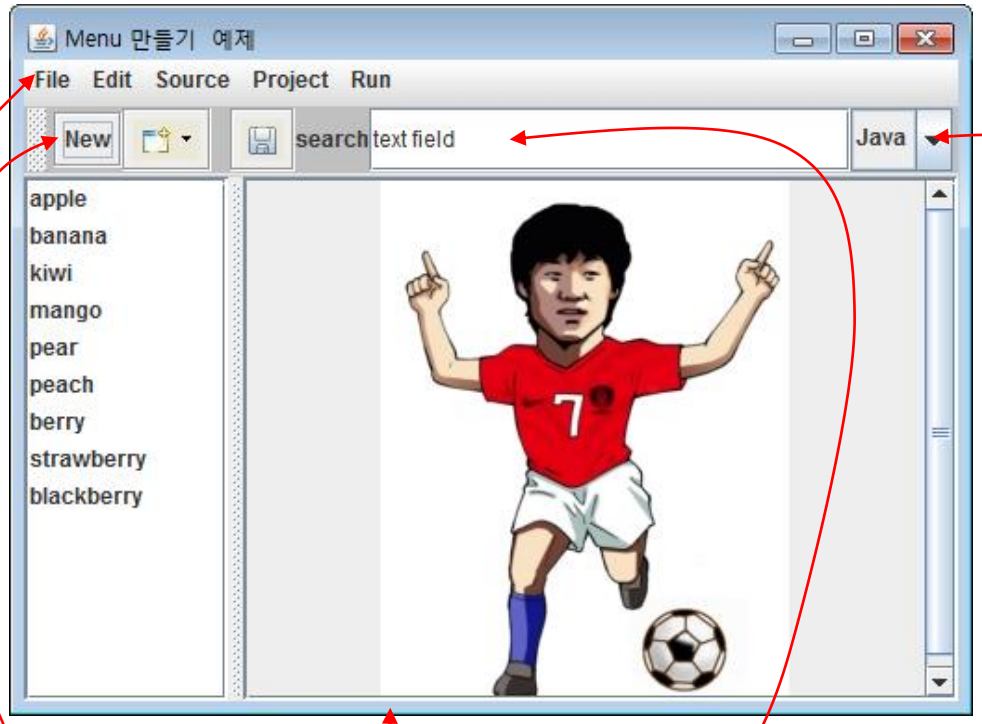
목차

4. 각종 리스너

이벤트 기반 프로그래밍

- 이벤트 기반 프로그래밍(Event Driven Programming)
  - 이벤트의 발생에 의해 프로그램 흐름이 결정되는 방식
    - 이벤트가 발생하면 이벤트를 처리하는 루틴(이벤트 리스너) 실행
    - 프로그램 내의 어떤 코드가 언제 실행될 지 아무도 모름, 이벤트의 발생에 의해 전적으로 결정
  - 반대되는 개념 : 배치 실행(batch programming)
    - 프로그램의 개발자가 프로그램의 흐름을 결정하는 방식
- 이벤트 종류
  - 사용자의 입력 : 마우스 드래그, 마우스 클릭, 키보드 누름 등
  - 센서로부터의 입력, 네트워크로부터 데이타 송수신
  - 다른 응용프로그램이나 다른 스레드로부터의 메시지
- 이벤트 기반 프로그램의 구조
  - 이벤트 처리 리스너 들의 집합
- 이벤트 처리 순서
  - 이벤트 발생(예 :마우스나 키보드의 움직임 혹은 입력)
  - 이벤트 객체 생성
    - 현재 발생한 이벤트에 대한 정보를 가진 객체
  - 이벤트 리스너 찾기
  - 이벤트 리스너 호출
    - 이벤트 객체가 리스너에 전달됨
  - 이벤트 리스너 실행

이벤트의 실제 예



마우스로 메뉴 선택
(Mouse Event, Action Event)

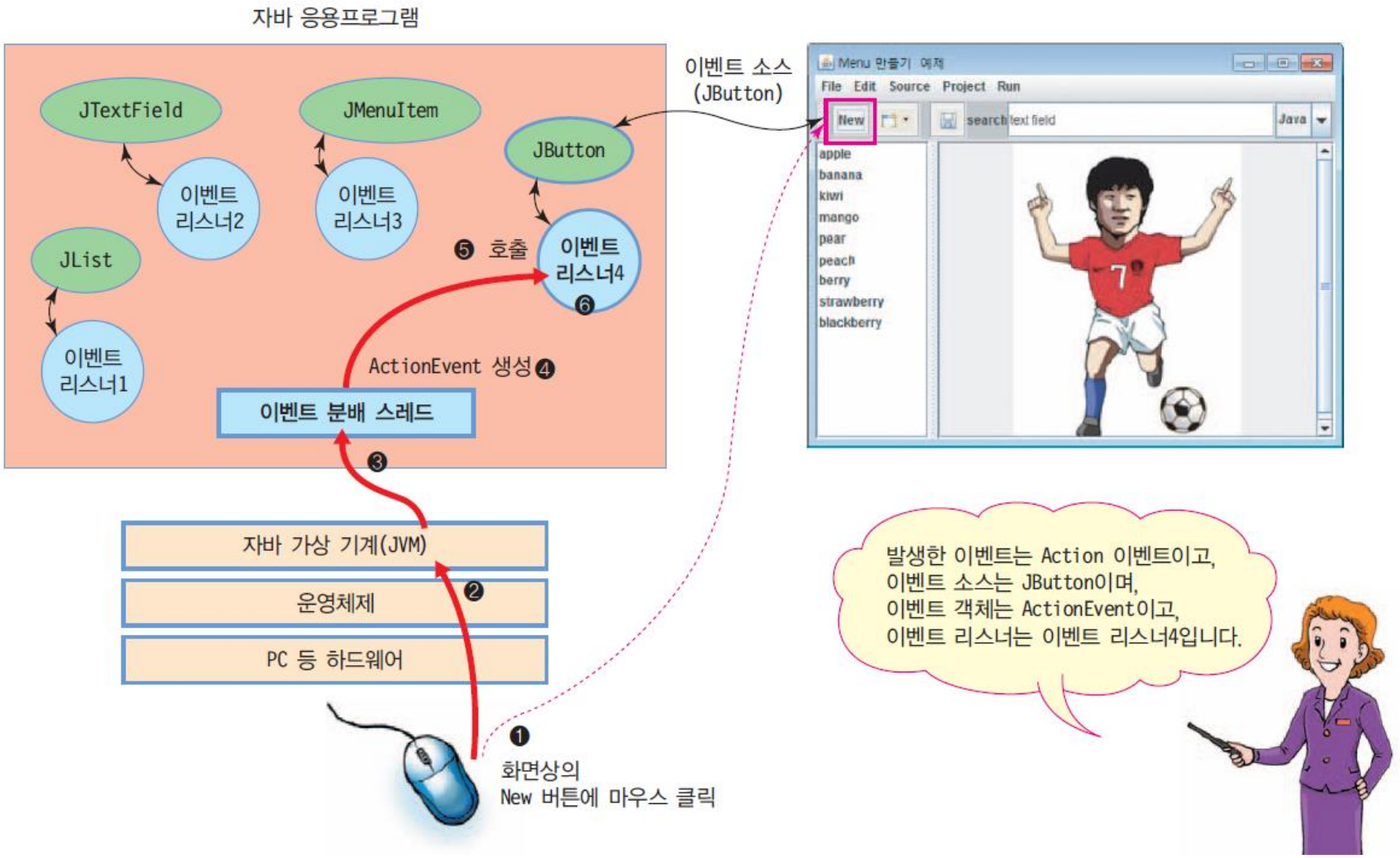
마우스로 버튼 클릭
(Mouse Event, Action Event)

마우스로 윈도우 크기 조절
(Mouse Event, Container Event)

키보드 입력
(Key Event)

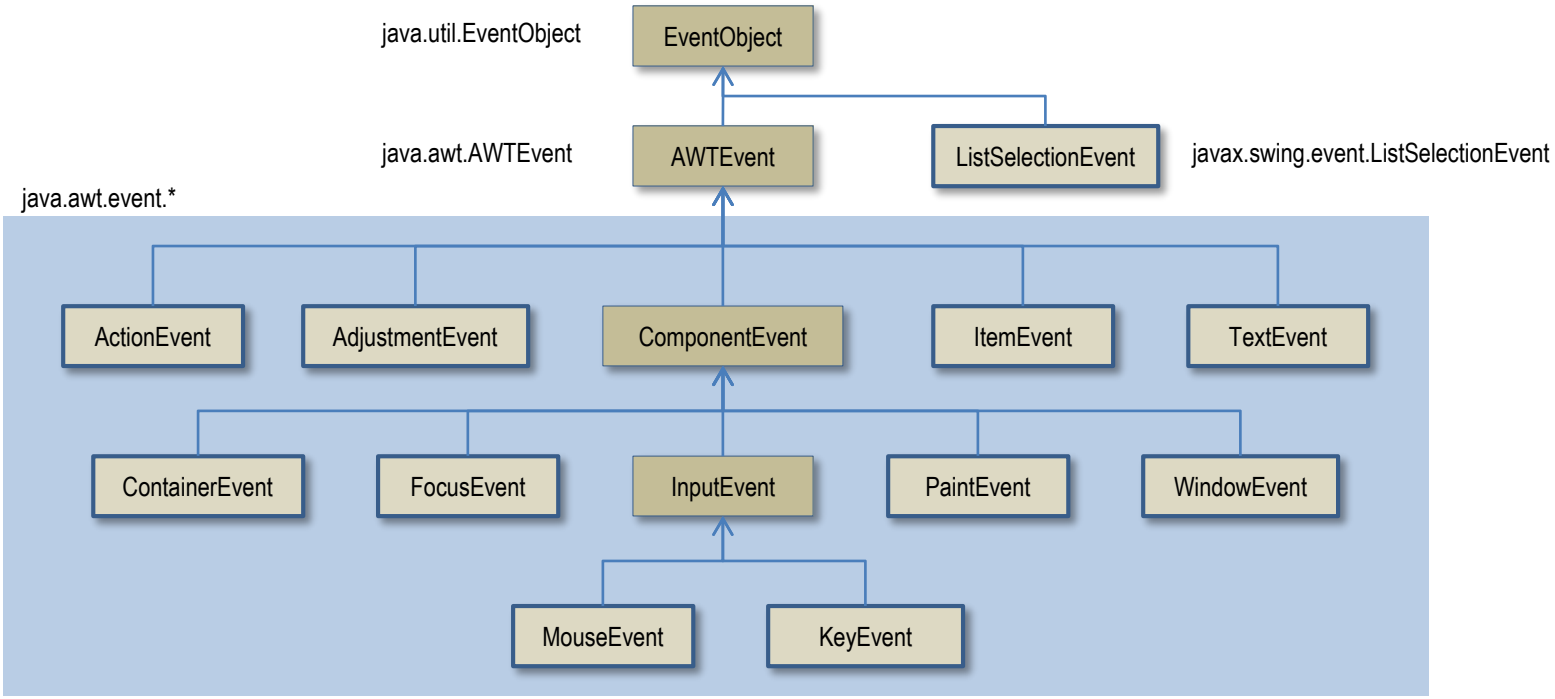
키보드로 메뉴 선택
(Key Event, Action Event)

자바의 이벤트 기반 GUI 응용프로그램 구성



이벤트 객체

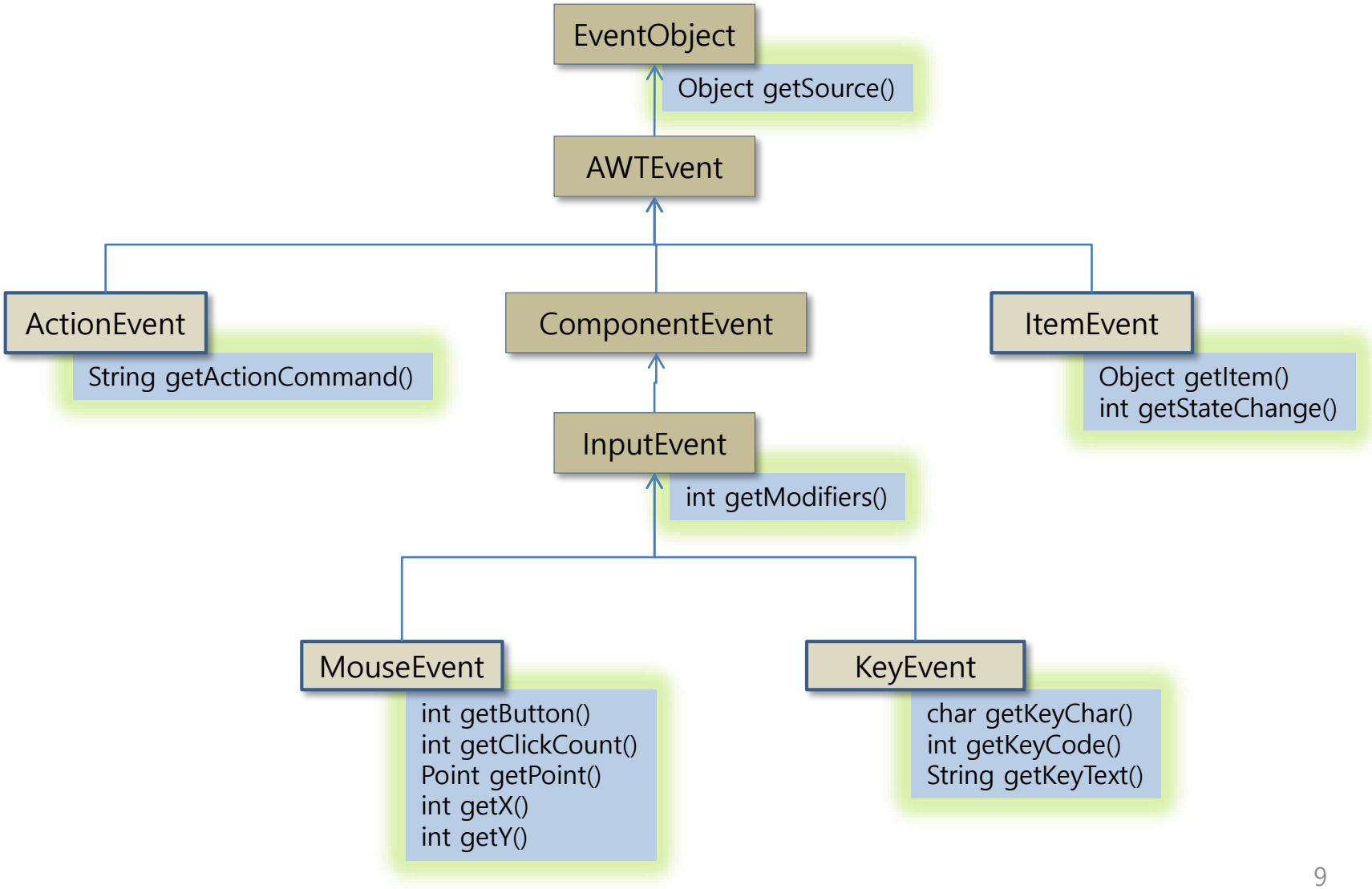
- 이벤트 객체란?
  - 이벤트가 발생할 때, 발생한 이벤트에 관한 정보를 가진 객체
  - 이벤트 리스너에 전달됨
    - 이벤트 리스너 코드에서 이벤트가 발생한 상황을 파악할 수 있게 함
- 이벤트 객체의 종류



이벤트 객체에 포함된 정보

- 이벤트 객체가 포함하는 정보
  - 이벤트 종류
  - 이벤트 소스
  - 이벤트가 발생한 화면 좌표
  - 이벤트가 발생한 컴포넌트 내 좌표
  - 버튼이나 메뉴 아이템에 이벤트가 발생한 경우 버튼이나 메뉴 아이템의 문자열
  - 클릭된 마우스 버튼 번호
  - 마우스의 클릭 횟수
  - 키가 눌러졌다면 키의 코드 값과 문자 값
  - 체크박스, 라디오버튼 등과 같은 컴포넌트에 이벤트가 발생하였다면 체크 상태
- 이벤트에 따라 조금씩 다른 정보 포함
  - ActionEvent 객체 : 액션 문자열
  - MouseEvent 객체 : 마우스의 위치 정보, 마우스 버튼, 함께 눌러진 키 정보 등
  - ItemEvent 객체 : 아이템의 체크 상태
- 이벤트 소스 알아 내기
  - Object EventObject.getSource()
    - 발생한 이벤트의 소스 컴포넌트 리턴
    - Object 타입으로 리턴하므로 캐스팅하여 사용
    - 모든 이벤트 객체에 대해 적용

이벤트 객체의 메소드


EventObject
Object getSource()
AWTEvent
ActionEvent
String getActionCommand()
ComponentEvent
ItemEvent
Object getItem()
int getStateChange()
InputEvent
int getModifiers()
MouseEvent
int getButton()
int getClickCount()
Point getPoint()
int getX()
int getY()
KeyEvent
char getKeyChar()
int getKeyCode()
String getKeyText()

이벤트 객체와 이벤트 소스

이벤트 객체	이벤트 소스	이벤트가 발생하는 경우
ActionEvent	JButton	마우스나 키로 버튼 선택
	JList	리스트 아이템을 더블클릭하여 리스트 아이템 선택
	JMenuItem	메뉴 아이템 선택
	JTextField	텍스트 입력 중 〈Enter〉 키 입력
ItemEvent	JCheckBox	체크박스의 선택 혹은 해제
	JCheckBoxMenuItem	체크박스 메뉴 아이템의 선택 혹은 해제
	JList	리스트 아이템 선택
KeyEvent	Component	키가 눌러지거나 눌러진 키가 떼어질 때
MouseEvent	Component	마우스 버튼이 눌러지거나 떼어질 때, 마우스 버튼이 클릭될 때, 컴포넌트 위에 마우스가 올라갈 때, 올라간 마우스가 내려올 때, 마우스가 드래그될 때, 마우스가 단순히 움직일 때
FocusEvent	Component	컴포넌트가 포커스를 받거나 잃을 때
TextEvent	TextField	텍스트 변경
	TextArea	텍스트 변경
WindowEvent	Window	Window를 상속받는 모든 컴포넌트에 대해 윈도우 활성화, 비활성화, 아이콘화, 아이콘에서 복구, 윈도우 열기, 윈도우 닫기, 윈도우 종료
AdjustmentEvent	JScrollBar	스크롤바를 움직일 때
ComponentEvent	Component	컴포넌트가 사라지거나, 나타나거나, 이동하거나, 크기가 변경될 때
ContainerEvent	Container	Container에 컴포넌트의 추가 혹은 삭제

이벤트 리스너(Event Listener)

- 이벤트 리스너란?
  - 이벤트를 처리하는 코드
  - 클래스로 작성
- JDK에서 이벤트 리스너 작성을 위한 인터페이스(interface) 제공
  - 개발자가 리스너 인터페이스의 추상 메소드 구현
    - 이벤트가 발생하면 자바 플랫폼은 리스너 인터페이스의 추상 메소드 호출
  - 예) ActionListener 인터페이스

```
interface ActionListener { // 아래 메소드를 개발자가 구현해야 함
   public void actionPerformed(ActionEvent e); // Action 이벤트 발생시 호출됨
}
```

  - 예) MouseListener 인터페이스

```
interface MouseListener { // 아래의 5개 메소드를 개발자가 구현해야 함
   public void mousePressed(MouseEvent e); // 마우스 버튼이 눌러지는 순간 호출
   public void mouseReleased(MouseEvent e); // 눌러진 마우스 버튼이 떼어지는 순간 호출
   public void mouseClicked(MouseEvent e); // 마우스가 클릭되는 순간 호출
   public void mouseEntered(MouseEvent e); // 마우스가 컴포넌트 위에 올라가는 순간 호출
   public void mouseExited(MouseEvent e); // 마우스가 컴포넌트 위에서 내려오는 순간 호출
}
```

이벤트 리스너 등록

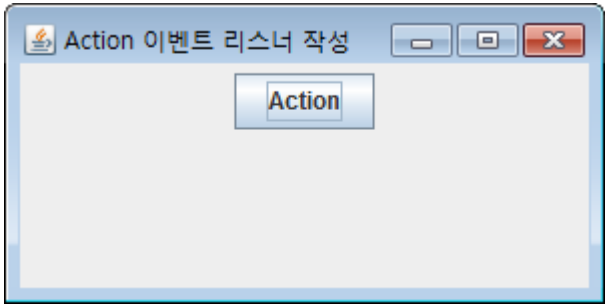
- 이벤트 리스너 등록
  - 이벤트를 받아 처리하고자 하는 컴포넌트에 이벤트 리스너 등록
- 이벤트 리스너 등록 메소드
  - Component.addXXXListener(listener)
    - xxx : 이벤트 명
    - listener : 이벤트 리스너 객체
    - 예) addMouseListener(), addActionListener(), addFocusListener() 등
- 이벤트 리스너가 등록된 컴포넌트에만 이벤트 전달
  - 이벤트 리스너가 등록된 컴포넌트만 이벤트 리스너 코드 작동

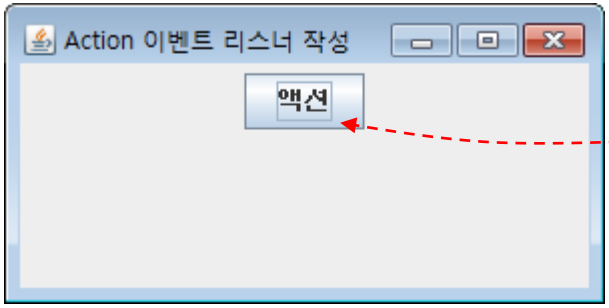
리스너 인터페이스와 메소드

이벤트 종류	리스너 인터페이스	리스너의 추상 메소드	메소드가 호출되는 경우
Action	ActionListener	void actionPerformed(ActionEvent)	Action 이벤트가 발생하는 경우
Item	ItemListener	void itemStateChanged(ItemEvent)	Item 이벤트가 발생하는 경우
Key	KeyListener	void keyPressed(KeyEvent)	모든 키에 대해 키가 눌러질 때
		void keyReleased(KeyEvent)	모든 키에 대해 눌러진 키가 떼어질 때
		void keyTyped(KeyEvent)	유니코드 키가 입력될 때
Mouse	MouseListener	void mousePressed(MouseEvent)	마우스 버튼이 눌러질 때
		void mouseReleased(MouseEvent)	눌러진 마우스 버튼이 떼어질 때
		void mouseClicked(MouseEvent)	마우스 버튼이 클릭될 때
		void mouseEntered(MouseEvent)	마우스가 컴포넌트 위에 올라올 때
		void mouseExited(MouseEvent)	컴포넌트 위에 올라온 마우스가 컴포넌트를 벗어날 때
Mouse	MouseMotionListener	void mouseDragged(MouseEvent)	마우스를 컴포넌트 위에서 드래그할 때
		void mouseMoved(MouseEvent)	마우스가 컴포넌트 위에서 움직일 때
Focus	FocusListener	void focusGained(FocusEvent)	컴포넌트가 포커스를 받을 때
		void focusLost(FocusEvent)	컴포넌트가 포커스를 잃을 때
Text	TextListener	void textValueChanged(TextEvent)	텍스트가 변경될 때
Window	WindowListener	void windowOpened(WindowEvent)	윈도우가 생성되어 처음으로 보이게 될 때
		void windowClosing(WindowEvent)	윈도우의 시스템 메뉴에서 윈도우 닫기를 시도할 때
		void windowIconfied(WindowEvent)	윈도우가 아이콘화될 때
		void windowDeiconfied(WindowEvent)	아이콘 상태에서 원래 상태로 복귀할 때
		void windowClosed(WindowEvent)	윈도우가 닫혔을 때
		void windowActivated(WindowEvent)	윈도우가 활성화될 때
		void windowDeactivated(WindowEvent)	윈도우가 비활성화될 때
Adjustment	AdjustmentListener	void adjustmentValueChanged(AdjustmentEvent)	스크롤바를 움직일 때
Component	ComponentListener	void componentHidden(ComponentEvent)	컴포넌트가 보이지 않는 상태로 될 때
		void componentShown(ComponentEvent)	컴포넌트가 보이는 상태로 될 때
		void componentResized(ComponentEvent)	컴포넌트의 크기가 변경될 때
		void componentMoved(ComponentEvent)	컴포넌트의 위치가 변경될 때
Container	ContainerListener	void componentAdded(ContainerEvent)	컴포넌트가 컨테이너에 추가될 때
		void componentRemoved(ContainerEvent)	컴포넌트가 컨테이너에서 삭제될 때

이벤트 리스너 작성 예

Mouse 이벤트 리스너 객체 생성
Mouse 이벤트 리스너 등록





버튼의 문자열 변경

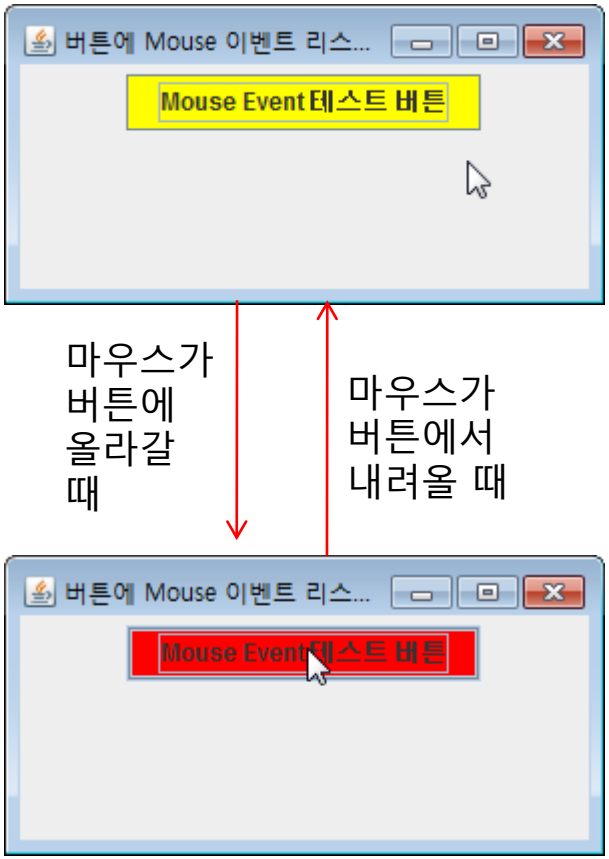
```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class ListenerSample extends JFrame {
  ListenerSample () {
    setTitle("Action 이벤트 리스너 작성");
    setLayout(new FlowLayout());
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    JButton btn = new JButton("Action");
    MyActionListener listener = new MyActionListener ();
    btn.addActionListener(listener );
    add(btn);
    setSize(300,150);
    setVisible(true);
  }
  public static void main(String [] args) {
    new ListenerSample ();
  }
}
class MyActionListener implements ActionListener {
  public void actionPerformed(ActionEvent e) {
    JButton b = (JButton)e.getSource();
    if(b.getText().equals("Action"))
      b.setText("액션");
    else
      b.setText("Action);
  }
}
```

이벤트 리스너 등록

이벤트 리스너 구현

예제 : 버튼이 Mouse 이벤트를 처리하는 예제

초기 상태




```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class ListenerMouseEx extends JFrame {
    ListenerMouseEx() {
        setTitle("버튼에 Mouse 이벤트 리스너 작성");
        setLayout(new FlowLayout());
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JButton btn = new JButton("Mouse Event 테스트 버튼");
        btn.setBackground(Color.YELLOW);
        MyMouseListener listener = new MyMouseListener();
        btn.addMouseListener(listener);
        add(btn);
        setSize(300,150);
        setVisible(true);
    }
    public static void main(String [] args) {
        new ListenerMouseEx();
    }
}

class MyMouseListener implements MouseListener {
    public void mouseEntered(MouseEvent e) {
        JButton btn = (JButton)e.getSource();
        btn.setBackground(Color.RED);
    }
    public void mouseExited(MouseEvent e) {
        JButton btn = (JButton)e.getSource();
        btn.setBackground(Color.YELLOW);
    }
    public void mousePressed(MouseEvent e) {}
    public void mouseReleased(MouseEvent e) {}
    public void mouseClicked(MouseEvent e) {}
}
```

Tip : 리스너 등록 메소드가 addXXXListener인 이유?

- 컴포넌트는 다른 이벤트에 대한 리스너를 동시에 가질 수 있다.
  - JButton.addActionListener(); // Action 리스너
  - JButton.addKeyListener(); // Key 리스너
  - JButton.addFocusListener(); // Focus 리스너
- 컴포넌트는 한 이벤트에 대해 여러 개의 리스너를 동시에 가질 수 있다.
  - JButton.addActionListener(new MyButtonListener1());
  - JButton.addActionListener(new MyButtonListener2());
  - JButton.addActionListener(new MyButtonListener3());
  - 이때, 리스너는 등록된 반대 순으로 모두 실행된다.

예제: 버튼에 ActionEvent 예

- ActionPerform1.java
- ActionPerform2.java

예제: MouseListenr 예

- MouseListen1.java

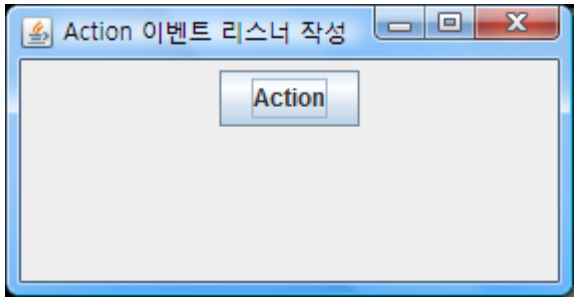
예제: JOptionPane 예

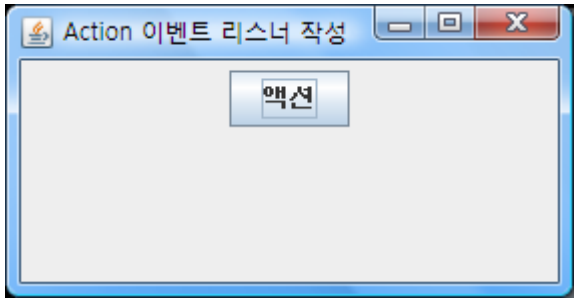
- JOptionPanGuessNum.java
- JOptionPanePractice.java

이벤트 리스너 작성 방법

- 3 가지 방법 (인터페이스 상속방법 제외)
  - 독립 클래스로 작성
    - 이벤트 리스너를 완전한 클래스로 작성
    - 이벤트 리스너를 여러 곳에서 사용할 때 적합
  - 내부 클래스(inner class)로 작성
    - 클래스 안에 멤버처럼 클래스 작성
    - 이벤트 리스너를 특정 클래스에서만 사용할 때 적합
  - 익명 클래스(anonymous class)로 작성
    - 클래스의 이름 없이 간단히 리스너 작성
    - 클래스 조차 만들 필요 없이 리스너 코드가 간단한 경우에 적합

독립 클래스로 리스너 작성

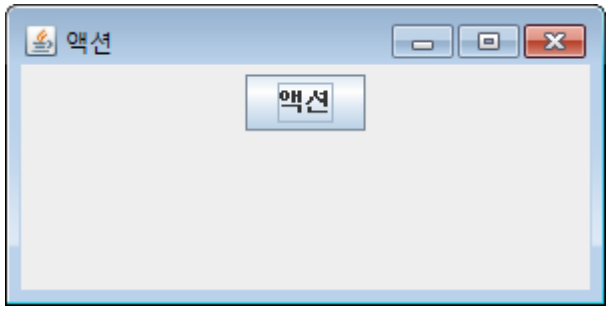


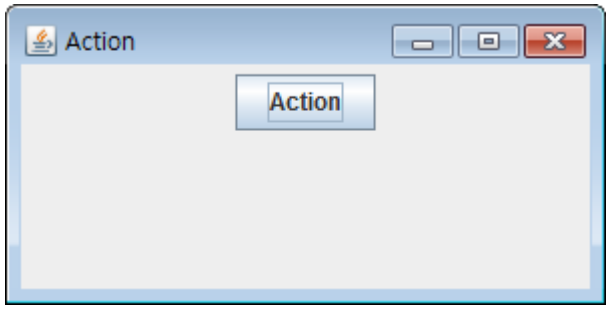


- 독립된 클래스로 Action 이벤트 리스너 작성
- 이 클래스를 별도의 MyActionListener.java파일로 작성하여도 됨

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class IndepClassListener extends JFrame {
   IndepClassListener() {
      setTitle("Action 이벤트 리스너 작성");
      setLayout(new FlowLayout());
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setSize(300,150);
      setVisible(true);
      JButton btn = new JButton("Action");
      MyActionListener listener = new MyActionListener();
      btn.addActionListener(listener);
      add(btn);
   }
   public static void main(String [] args) {
      new IndepClassListener();
   }
}
class MyActionListener implements ActionListener {
   public void actionPerformed(ActionEvent e) {
      JButton b = (JButton)e.getSource();
      if(b.getText().equals("Action"))
         b.setText("액션");
      else
         b.setText("Action");
   }
}
```

내부 클래스로 리스너 작성





- Action 이벤트 리스너를 내부 클래스로 작성
- private으로 선언하여 InnerClassListener의 외부에서 사용할 수 없게 함

- 리스너에서 InnerClassListener의 멤버에 대한 접근 용이

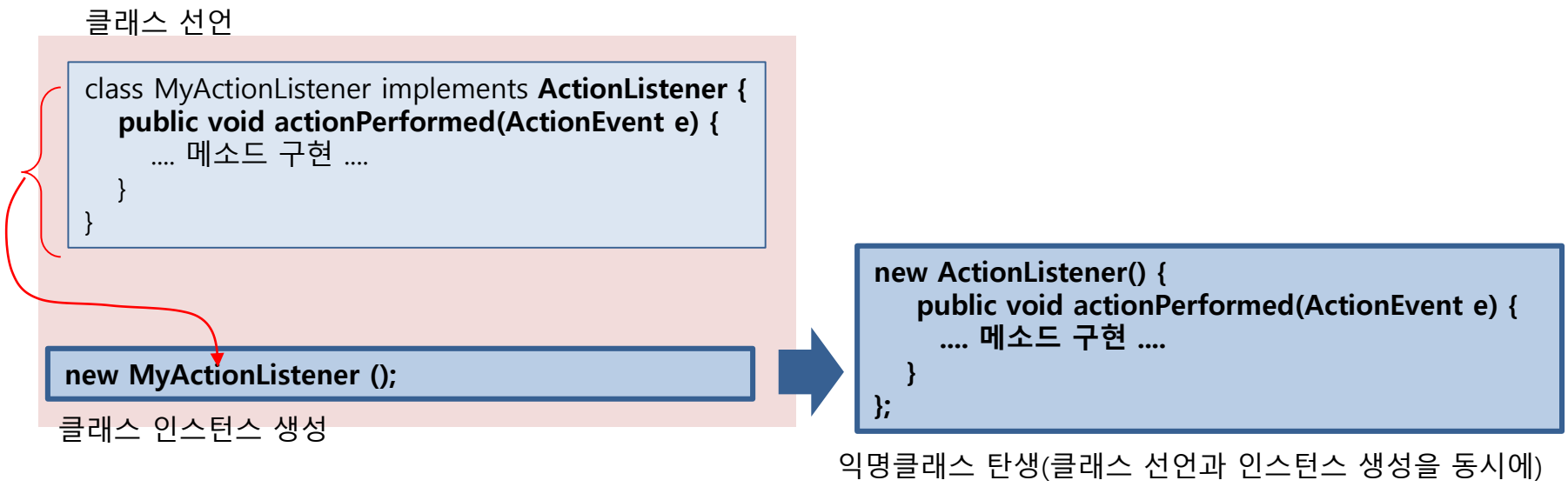
```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class InnerClassListener extends JFrame {
    InnerClassListener() {
        setTitle("Action 이벤트 리스너 작성");
        setLayout(new FlowLayout());
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300,300);
        setVisible(true);
        JButton btn = new JButton("Action");
        btn.addActionListener(new MyActionListener());
        add(btn);
    }
    private class MyActionListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            JButton b = (JButton)e.getSource();
            if(b.getText().equals("Action"))
                b.setText("액션");
            else
                b.setText("Action");

            // InnerClassListener의 멤버나 JFrame의 멤버를 호출할 수 있음
            setTitle(b.getText()); // JFrame.setTitle() 호출
        }
    }
    public static void main(String [] args) {
        new InnerClassListener();
    }
}
```

익명 클래스로 이벤트 리스너 작성

- 익명 클래스란?
  - (클래스 정의 + 인스턴스 생성)을 한번에 작성

```
new 익명클래스의수퍼클래스이름(생성자의 인자들) {
        .....................
        클래스 정의
        .....................
};
```

  - ActionListener를 구현하는 익명의 이벤트 리스너 작성 예



익명 클래스 &
이벤트 리

익명 클래스로 다시 작성된 결과

```
btn.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        JButton b = (JButton)e.getSource();
        if(b.getText().equals("Action"))
            b.setText("액션");
        else
            b.setText("Action");
            // AnonymousClassListener의 멤버나
            // JFrame의 멤버를 호출할 수 있음
            setTitle(b.getText());
    }
}
);
```

```
import javax.swing.*;
import java.awt.event.*;
import java.awt.*;

public class AnonymousClassListener extends JFrame {
    AnonymousClassListener() {
        setTitle("Action 이벤트 리스너 작성");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());
        setSize(300,300);
        setVisible(true);
        JButton btn = new JButton("Action");
        add(btn);
        btn.addActionListener(new MyActionListener()  );
    }
    private class MyActionListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            JButton b = (JButton)e.getSource();
            if(b.getText().equals("Action"))
                b.setText("액션");
            else
                b.setText("Action");

            // InnerClassListener의 멤버나 JFrame 멤버 호출 가능
            setTitle(b.getText());
        }
    }
    public static void main(String [] args) {
        new AnonymousClassListener ();
    }
}
```

- 간단한 리스너의 경우 익명 클래스 사용 추천.
- 메소드의 개수가 1, 2개인 리스너(ActionListener, ItemListener)에 대해 주로 사용

예제: ActionListener 사용 종류

- ActionEvent1.java
  - 독립클래스로 만듬 (다른 이벤트에서는 복잡함)
- ActionEvent2.java
  - 내부클래스로 만듬
- ActionEvent3.java
  - 익명의클래스로 만듬

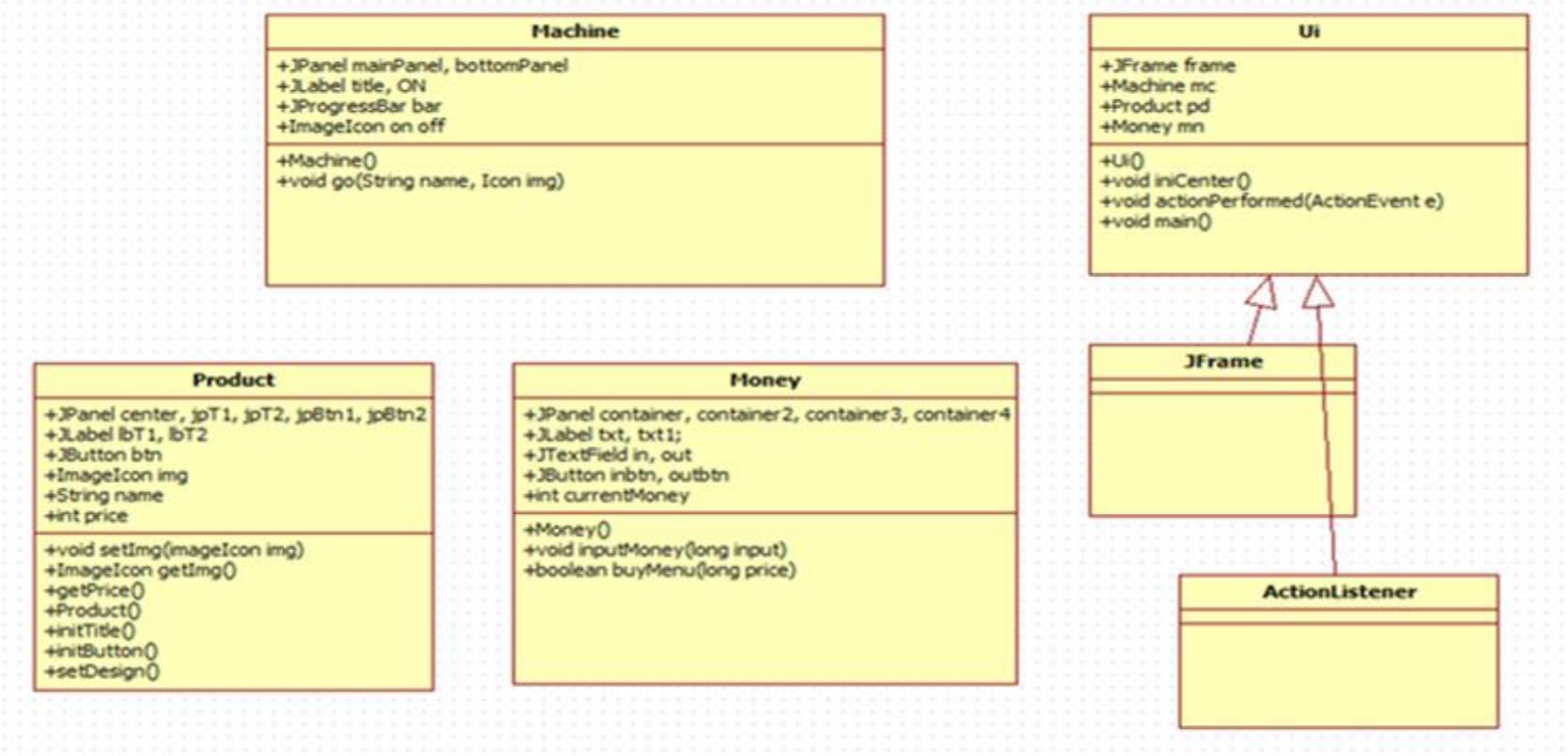
예제: Mouse이벤트 사용 종류

- MouseListener1.java
  - 인터페이스로 만듬
- MouseListener2.java
  - 내부클래스로 만듬
- MouseListener3.java
  - 익명의클래스로 만듬
- MouseListener4.java
  - 독립클래스로 만듬 (복잡함)

5. Awt Swing 실전예제

윈도우 자판기

클래스 다이어그램


Machine
+JPanel mainPanel, bottomPanel
+JLabel title, ON
+JProgressBar bar
+ImageIcon on off
+Machine()
+void go(String name, Icon img)
Ui
+JFrame frame
+Machine mc
+Product pd
+Money mn
+Ui()
+void iniCenter()
+void actionPerformed(ActionEvent e)
+void main()
Product
+JPanel center, jpT1, jpT2, jpBtn1, jpBtn2
+JLabel lbT1, lbT2
+JButton btn
+ImageIcon img
+String name
+int price
+void setImg(imageIcon img)
+ImageIcon getImg()
+getPrice()
+Product()
+initTitle()
+initButton()
+setDesign()
Money
+JPanel container, container2, container3, container4
+JLabel txt, txt1;
+JTextField in, out
+JButton inbtn, outbtn
+int currentMoney
+Money()
+void inputMoney(long input)
+boolean buyMenu(long price)
JFrame
ActionListener

주요화면 및 기능(1)

- 좌측에 메뉴를 선택할 수 있게 배열
- 우측에 금액을 관리할 수 있는 구역
- 하단에 진행상황을 볼수있도록 구현

주요화면 및 기능(2)

- 메뉴를 선택시 금액이 차감되고 생성 시작
- 하단에 진행상황이 표시
- 완성되기 다른 메뉴를 선택하면 경고 메시지 출력

주요화면 및 기능(3)

- 메뉴 완성시 팝업 메시지 출력
- 팝업 메시지 종료시 기본 상태로 되돌아감