자바 윈도우 프로그램 환경

AWT, SWING

이클립스 RCP(Rich Client Platform)

JAVAFX

기타 등등

스윙

스윙 API: 자바기반 프론트엔드(데스크탑) 어플리케이션 개발을 쉽게 해주는 확장 GUI(Graphic User Interface) 컴포넌트 집합

스윙 컴포넌트: MVC(Model-View-Controller) 아키텍처

1. MVC 아키텍처 모델

모델: 컴포넌트의 데이터를 표현

뷰: 컴포넌트 데이터에 대한 보기 표현

컨트롤러: 뷰 상에서 유저입력으로 컴포넌트의 데이터의 변경을 반영

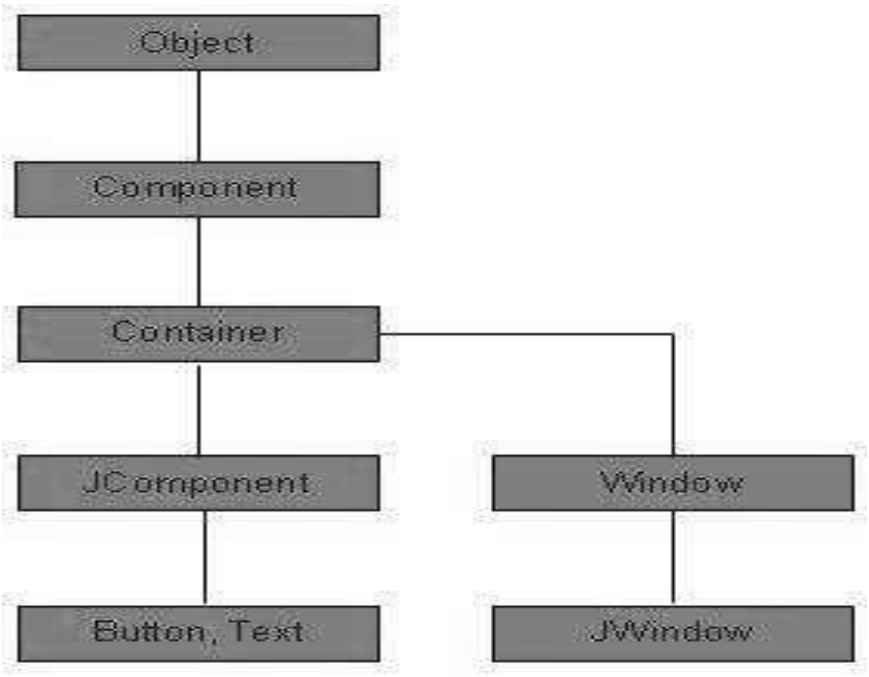
2. 유저 인터페이스 고려사항

UI 요소들: 유저와 상호작용하는 비주얼 요소

레이아웃: UI 요소들을 화면상에서 배치하는 방법

행동(Behavior): 유저가 UI 요소들과 상호작용할 때 발생하는 이벤트

3. 스윙 컴포넌트 계층도



컴포넌트: 스윙의 유저 인터페이스용 컨트롤에 대한 추상 기반 클래스

컨테이너: 다른 스윙 컴퍼넌트를 담을 수 있는 컴퍼넌트

JComponent: 스윙의 모든 UI 컴포넌트에 대한 기반 클래스

4. 스윙 UI 엘리먼트

JLabel: 글자

```
package swingcontrol;

import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

import javax.swing.JFrame;
import javax.swing.JLabel;
```

```
import javax.swing.JPanel;

public class JLabelDemo {
        private JFrame mainFrame;
        private JLabel headerLabel;
        private JLabel statusLabel;
        private JPanel controlPanel;

        public static void main(String[] args){
                LabelDemo label = new LabelDemo();
                label. showLabel ();
        }

        private void showLabel (){
                mainFrame = new JFrame("Java Swing Label Control");
                mainFrame.setSize(400,400);
                mainFrame.setLayout(new GridLayout(3, 1));

                mainFrame.addWindowListener(new WindowAdapter() {
                  public void windowClosing(WindowEvent windowEvent){
                       System.exit(0);
                  }
                });
                headerLabel = new JLabel("", JLabel.CENTER);
                statusLabel = new JLabel("",JLabel.CENTER);
                statusLabel.setSize(350,100);
                controlPanel = new JPanel();
                controlPanel.setLayout(new FlowLayout());

                mainFrame.add(headerLabel);
                mainFrame.add(controlPanel);
                mainFrame.add(statusLabel);
                headerLabel.setText("Control in action: JLabel");
                JLabel label   = new JLabel("", JLabel.CENTER);
```

```
                label.setText("Welcome to Swing Label");
                label.setOpaque(true);
                label.setBackground(Color.GRAY);
                label.setForeground(Color.WHITE);
                controlPanel.add(label);
                mainFrame.setVisible(true);
        }
}
```

JButton

JCheck Box

JRadioButton

JComboBox

JTextField

JTextArea

JList

5. 이벤트

오브젝트의 상태변화

즉, 그래픽 유저 인터페이스 컴포넌트로 사용자와의 상호작용의 결과로 발생

예) 버튼 클릭, 마우스 움직임, 키보드를 통한 문자 입력, 리스트 중 한 아이템을 선택,

페이지 스크롤: 이벤트를 발생시키는 활동

6. 이벤트 유형

포그라운트 이벤트: 이벤트의 예

백그라운드 이벤트: 타이머, 운영체제 인터럽트 등.

7. 이벤트 처리

소스: 이벤트가 발생한 오브젝트 -> 소스오브젝트(예: JButton)

리스너: 이벤트 처리 오브젝트

8. 이벤트 처리 절차

Step 1 - 유저가 버튼 클릭 -> 이벤트 발생

Step 2 - 관련 이벤트 클래스의 오브젝트가 자동으로 생성 -> 소스와 이벤트에 대한 정보를 같은 오브젝트 내에서 얻음.

Step 3 - 이벤트 오브젝트가 등록된 리스너 클래스의 메소드로 넘어간다.

Step 4 - 해당 메소드가 실행되고 결과를 되돌려 준다.

9. 이벤트 클래스

이벤트를 표현

10. 이벤트 클래스들

AWTEvent

ActionEvent

InputEvent

KeyEvent

MouseEvent

WindowEvent

11. 이벤트 리스너

액션이벤트 리스너

키리스너

마우스 리스너

아이템 리스너

12. 이벤트 어댑터

이벤트를 리스너로 할 때는 인터페이스 속한 모든 메소드를 구현해야 하나,

대표적인 메소드만 구현할 수 있도록 간소화된 이벤트 처리기

키 어댑터

마우스 어댑터

13. 컨테이너: Panel, Frame