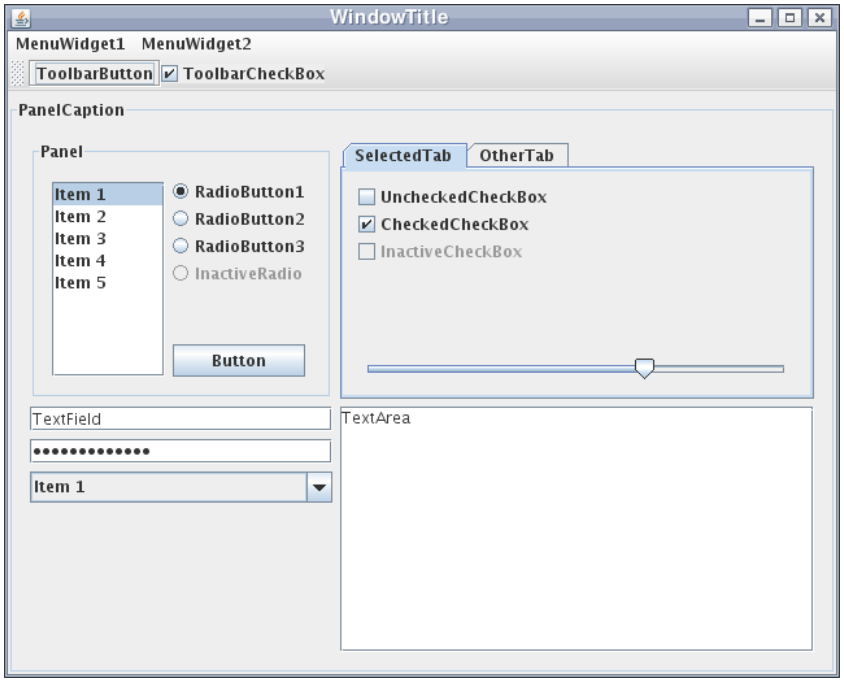
객체지향 프로그래밍 및 실습

10주차. Swing GUI Components

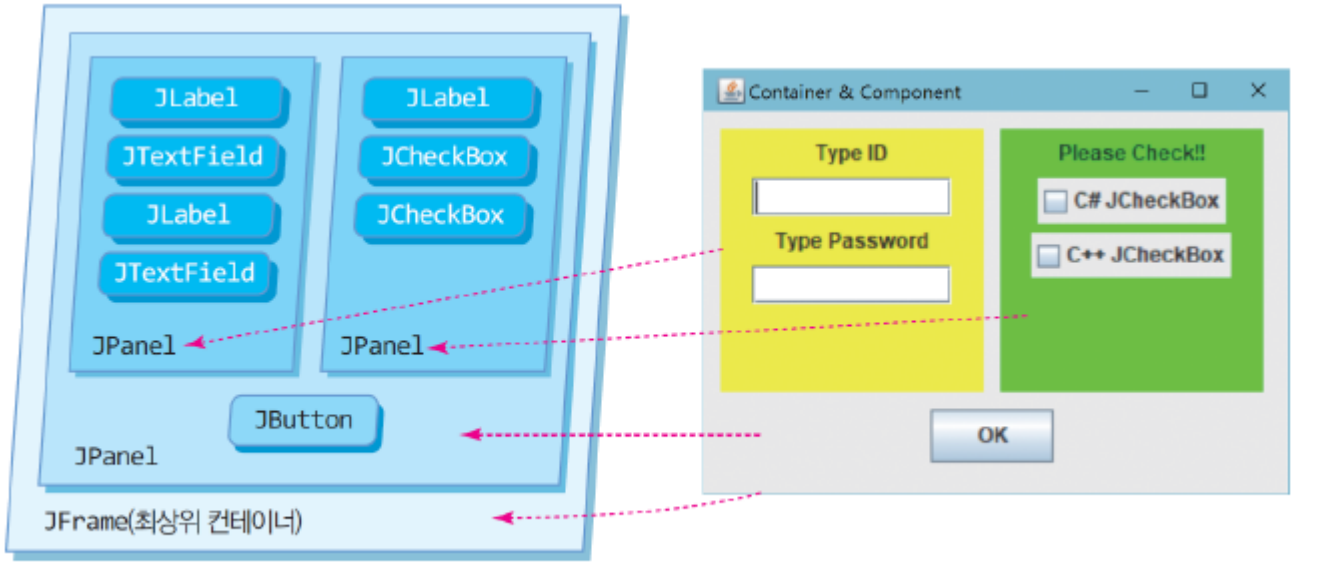
1. Introduction to Swing

- What is Swing?
  - Swing은 자바에서 GUI를 구현하기 위해 JDK에서 기본적으로 제공하는 개발 툴킷으로 sun microsystems의 자바 기반 클래스의 일부이다.



1. Introduction to Swing

- Swing의 구조
  - JFrame 인스턴스가 최상위 컨테이너 역할을 한다. 창 역할
  - JFrame는 java.awt.Container를 상속한다. => Jlabel, JCheckBox같은 Swing Component를 담을 수 있는 컨테이너의 역할을 수행
  - 직접 JFrame을 상속해서 본인만의 컨테이너를 개발할 수도 있다!



스윙의 컨테이너와 컴포넌트의 포함 관계 스윙 GUI 프로그램

1. Introduction to Swing

- 간단한 창 띄우기
  - 직접 만든 클래스에 JFrame을 상속한다.
  - setTitle() , …은 JFrame에 정의된 메서드

```java
package swingtest;

import javax.swing.JFrame;

public class SwingTest extends JFrame {
    public SwingTest() {
        setTitle("This is test");
        setSize(300, 300);
        setVisible(true);
    }

    public static void main(String[] args) {
        SwingTest frame = new SwingTest();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

1. Introduction to Swing

- 창에 텍스트 레이블을 표시해보자
  - Jframe에 TextLabel을 추가할 수 있다.

```java
package swingtest;

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

public class SwingTest extends JFrame {
    public SwingTest() {
        setTitle("This is test");
        setSize(300, 300);
        setVisible(true);
        add(new JLabel("Hi this is label test"));
    }

    public static void main(String[] args) {
        SwingTest frame = new SwingTest();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

1. Introduction to Swing

- 창에 버튼을 표시해보자
  - Jframe에 Button을 추가할 수 있다.

```java
package swingtest;

import javax.swing.JButton;
import javax.swing.JFrame;
import java.awt.Container;

public class SwingTest extends JFrame {
    public SwingTest() {
        setTitle("This is test");
        setSize(300, 300);
        setVisible(true);

        Container pane = getContentPane();
        pane.add(new JButton("Button A"));
    }

    public static void main(String[] args) {
        SwingTest frame = new SwingTest();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

1. Introduction to Swing

- 다양한 방법으로의 구현
  - JFrame을 상속하기

```java
package swingtest;

import javax.swing.JButton;
import javax.swing.JFrame;
import java.awt.Container;

public class SwingTest extends JFrame {
    public SwingTest() {
        setTitle("This is test");
        setSize(300, 300);
        setVisible(true);

        Container pane = getContentPane();
        pane.add(new JButton("Button A"));
    }

    public static void main(String[] args) {
        SwingTest frame = new SwingTest();
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
}
```

1. Introduction to Swing

- 다양한 방법으로의 구현
  - JFrame을 인스턴스화 하기

```java
package swingtest;

import javax.swing.JButton;
import javax.swing.JFrame;
import java.awt.Container;

public class SwingMain {

    public static void main(String[] args) {
        JFrame frame = new JFrame();
        frame.setTitle("This is test");
        frame.setSize(300, 300);
        frame.setVisible(true);

        Container pane = frame.getContentPane();
        pane.add(new JButton("Button A"));
    }

}
```

2. Layout

- JFrame에 레이아웃 지정하기
  - 컴포넌트를 배치하는 방식을 레이아웃이라고 부른다.
  - 우리가 레이아웃을 지정하여 컴포넌트를 어떻게 배치할지 지정할 수 있다!

  - https://examples.javacodegeeks.com/java-swing-layouts-example/

2. Layout

- JFrame에 레이아웃 지정하기

```java
package swingtest.layout;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import java.awt.GridLayout;

public class GridTest extends JFrame {
    public GridTest() {
        setTitle("Grid layout test");
        setSize(500, 500);
        JButton jb1 = new JButton("Button 1");
        JButton jb2 = new JButton("Button 2");
        JButton jb3 = new JButton("Button 3");
        JButton jb4 = new JButton("Button 4");
        JButton jb5 = new JButton("Button 5");

        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(3, 2));
        panel.add(jb1);
        panel.add(jb2);
        panel.add(jb3);
        panel.add(jb4);
        panel.add(jb5);
        getContentPane().add(panel);
    }

    public static void main(String[] args) {
        GridTest app = new GridTest();
        app.setVisible(true);
        app.setDefaultCloseOperation(GridTest.EXIT_ON_CLOSE);

    }
}
```

3. Component 분리하기

- 필요에 따라 컴포넌트 분리
  - 반복되는 요소가 많은 경우. Container를 간접 또는 직접적으로 상속하여 나만의 컴포넌트를 개발할 수 있다.

```java
package swingtest.layout;

import javax.swing.JLabel;
import javax.swing.JPanel;
import java.awt.GridLayout;

public class StudentPanel extends JPanel {
    private String id;
    private String name;

    public StudentPanel(String id, String name) {
        this.id = id;
        this.name = name;

        setLayout(new GridLayout(2, 1));
        add(new JLabel(id));
        add(new JLabel(name));
    }
}
```

3. Component 분리하기

- Java Doc 읽고 쓰는 법
  - Swing을 다루면서 모든 클래스와 모든 메서드의 기능을 다 알 수는 없다.
  - 따라서 JavaDoc의 참조가 필수!
  - Javadoc은 /** */ 주석으로 작성되어 있다.
  - Javadoc 에서는 파라미터와 리턴값 메서드 설명 등 개발자의 이해를 돕도록 설명을 작성할 수 있다.

  - https://docs.oracle.com/en/java/javase/11/

3. Component 분리하기

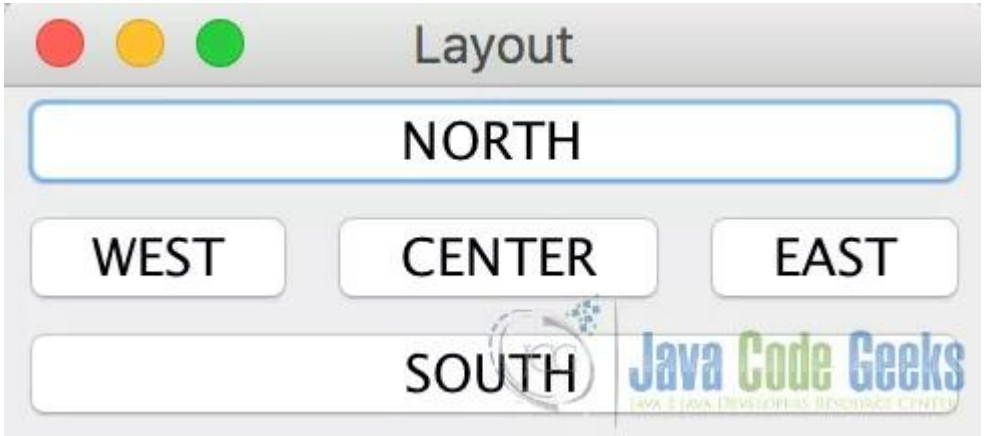
- Java Doc 읽고 쓰는 법
  - Swing을 다루면서 모든 클래스와 모든 메서드의 기능을 다 알 수는 없다.
  - 따라서 JavaDoc의 참조가 필수!
  - Javadoc은 /** */ 주석으로 작성되어 있다.
  - Javadoc 에서는 파라미터와 리턴값 메서드 설명 등 개발자의 이해를 돕도록 설명을 작성할 수 있다.

  - https://docs.oracle.com/en/java/javase/11/

  - 다같이 이클립스에서 메서드 또는 클래스 위에 Javadoc 주석을 작성해보자.

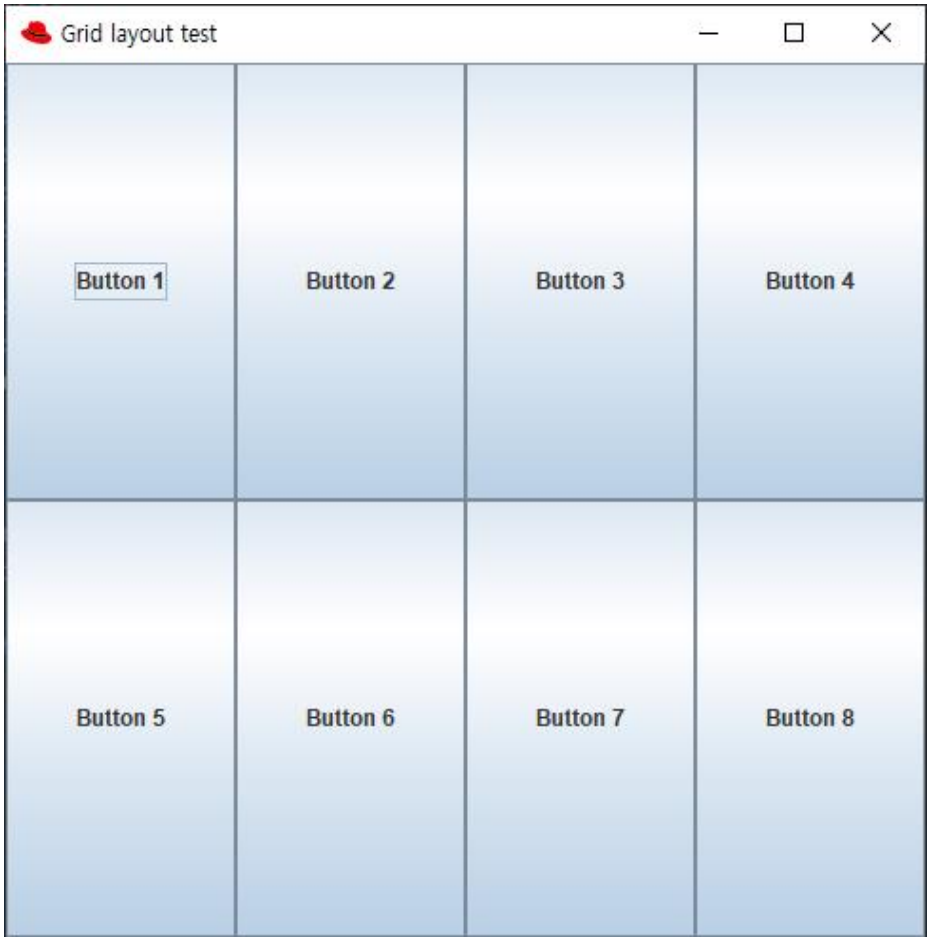
3. Component 분리하기

- **실습 문제 1**
  - BorderLayout과 JButton을 통해서 아래 그림처럼 디자인해보자.



3. Component 분리하기

- **실습 문제 2**
  - GridLayout과 JButton을 통해서 아래 그림처럼 디자인해보자.



4. Event Listener

- Event
  - 마우스를 누르거나
  - 마우스를 떼거나
  - 키보드를 누르거나
  - 소켓에 데이터가 도착하거나
  - 이런 일련의 상황을 Event라고 칭함

What is an *event*?

An *event* is a signal received by a program from the operating system as a result of some action taken by the user, or because something else has happened. Here are some examples:

- The user clicks a mouse button.
- The user presses a key on the keyboard.
- The user closes a window.
- Some data from the Internet arrives at one of the computer's "ports".

4. Event Listener

- Mouse Action Listener

```java
package event.button;

import javax.swing.JButton;
import javax.swing.JFrame;

public class Beep extends JFrame {
    public Beep() {
        super("Beep Button Program");

        setSize(300, 300);

        JButton beepButton = new JButton();
        beepButton.setText("Beep");
        beepButton.addMouseListener(new BeepListener());
        add(beepButton);
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }

    public static void main(String[] args) {
        Beep beep = new Beep();
    }
}
```

4. Event Listener

- Mouse Action Listener

```java
public class BeepListener implements MouseListener {

    @Override
    public void mouseClicked(MouseEvent e) {
        System.out.println("Mouse Clicked");
        System.out.println(e.getX());
        System.out.println(e.getY());
    }

    @Override
    public void mousePressed(MouseEvent e) {
        System.out.println("Mouse Pressed");
    }

    @Override
    public void mouseReleased(MouseEvent e) {
        System.out.println("Mouse Released");
    }

    @Override
    public void mouseEntered(MouseEvent e) {
        System.out.println("Mouse Entered");
    }

    @Override
    public void mouseExited(MouseEvent e) {
        System.out.println("Mouse Exited");
    }

}
```

4. Event Listener

- 익명클래스를 통한 구현법

```java
beepButton.addMouseListener(new MouseListener() {
  @Override
  public void mouseClicked(MouseEvent e) {
      System.out.println("Mouse Clicked");
      System.out.println(e.getX());
      System.out.println(e.getY());
  }

  @Override
  public void mousePressed(MouseEvent e) {
      System.out.println("Mouse Pressed");
  }

  @Override
  public void mouseReleased(MouseEvent e) {
      System.out.println("Mouse Released");
  }

  @Override
  public void mouseEntered(MouseEvent e) {
      System.out.println("Mouse Entered");
  }

  @Override
  public void mouseExited(MouseEvent e) {
      System.out.println("Mouse Exited");
  }
});
```

4. Event Listener

- Event Object
  - Event가 발생했다면, EventListener에서 어떤 이벤트가 발생한 건지.
  - 어느 컴포넌트에 발생한 건지
  - 만약에 마우스 이벤트라면, 어느 좌표를 클릭했는지
  - 등을 알 수 있다. 아래 코드의 파라미터 e 가 해당 역할을 수행

```java
@Override
public void mouseClicked(MouseEvent e) {
    System.out.println("Mouse Clicked");
    System.out.println(e.getX());
    System.out.println(e.getY());
}
```

4. Event Listener

- **실습 문제 3**
  - Button이 눌려져 있을 때, 버튼에 쓰여져 있는 글자가 “Pressed”
  - 눌려져 있지 않을 때, “Unpressed”가 표시되도록 해보자

  - Event Object에서 컴포넌트를 꺼내서…

4. Event Listener

- Text 이벤트를 통해서 올바른 전화번호를 입력하도록 강제 해보기
  - TextEvent - TextListener

4. Event Listener

- Adapter Class를 이용하기
  - EventListener 인터페이스를 구현하기에는 너무나도 많은 메서드를 오버라이드 해야 한다. 그 중에서도 내가 사용할 이벤트만 구현할 수는 없을까?

```java
package event.adapter;

import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

public class AdapterHandler extends MouseAdapter {
    @Override
    public void mouseClicked(MouseEvent e) {
        System.out.println("마우스 버튼 클릭");
    }
}
```