Week 14 : JNI, AWT

Part1. JNI

The interface that connects to Java libraries and non-Java libraries.

- Using C language functions in a Java program

Native Method : Function written in C language.

[Step1] Create a Java class that declares the Native method.

-Greeting.java

```
class Greeting
{
        static{
                System.loadLibrary("jni");
        }
        public native void printHello();
}
```

jni means libjni.so(library)

- Main.java

```
class Main{
        public static void main(String [] args){
                Greeting greeting = new Greeting();
                greeting.printHello();
        }
}
```

[Step2] Compile created in Step1.

[Step3] Create a header file for Native Method using Javah

```
dcslab@compute1:~/java$ javac Greeting.java
dcslab@compute1:~/java$ javac Main.java
dcslab@compute1:~/java$ javah -jni Greeting
dcslab@compute1:~/java$ ls
Greeting.class  Greeting.h  Greeting.java  Main.class  Main.java
```

- Generated header file

```
/* DO NOT EDIT THIS FILE - it is machine generated */
#include <jni.h>
/* Header for class Greeting */

#ifndef _Included_Greeting
#define _Included_Greeting
#ifdef __cplusplus
extern "C" {
#endif
/*
 * Class:     Greeting
 * Method:    printHello
 * Signature: ()V
 */
JNIEXPORT void JNICALL Java_Greeting_printHello
  (JNIEnv *, jobject);

#ifdef __cplusplus
}
#endif
#endif
```

[Step4] Implement Native Method using C

-jni.c

```
#include <stdio.h>
#include "Greeting.h"

JNIEXPORT void JNICALL Java_Greeting_printHello(JNIEnv *env, jobject obj)
{
                fprintf(stdout, "Hello jni\n");
}
```

[Step5] Compile C code and header files

```
dcslab@compute1:~/java$ gcc jni.c -o libjni.so -shared -fPIC -I /usr/local/java/jdk1.8.0_40/include/ -I /u
sr/local/java/jdk1.8.0_40/include/linux/
```

[Step6] Execute Java program

① 라이브러리로 만든 so 파일 복사후 환경변수로 등록해서 실행하거나

cp libjni.so /usr/lib

export LD_LIBRARY_PATH=/usr/lib

② 혹은 컴파일시 옵션으로 라이브러리를 등록해서 수행

java -Djava.library.path=/usr/lib Main

자바에서 c언어로 구현한 함수가 출력됨을 확인할 수 있다.

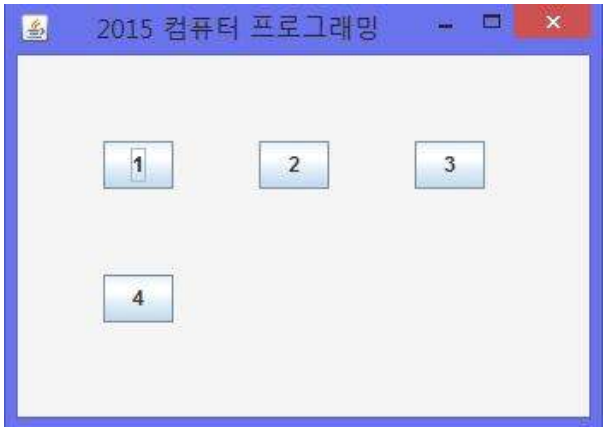
```
dcslab@compute1:~/java$ java Main
Hello jni
```

Part2. AWT

Container can have a several component. a deployment manager decide component's placement and size. When Container's size changed, a deployment manager relocate it.

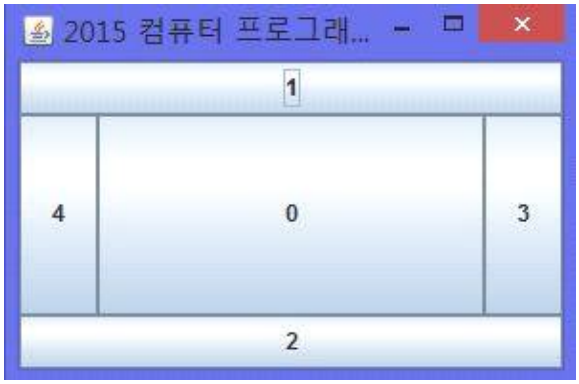
FlowLayout : Locating elements from left to right in the container

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

public class FlowLayout1extends JFrame {
        FlowLayout1() {
        setTitle("2015 Computer Programming");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout(FlowLayout.LEFT, 50, 50));
        add(new JButton("1"));
        add(new JButton("2"));
        add(new JButton("3"));
        add(new JButton("4"));
        setSize(350, 250);
        setVisible(true);
        }

        public static void main(String[] args) {
        new FlowLayout1();
        }
}
```



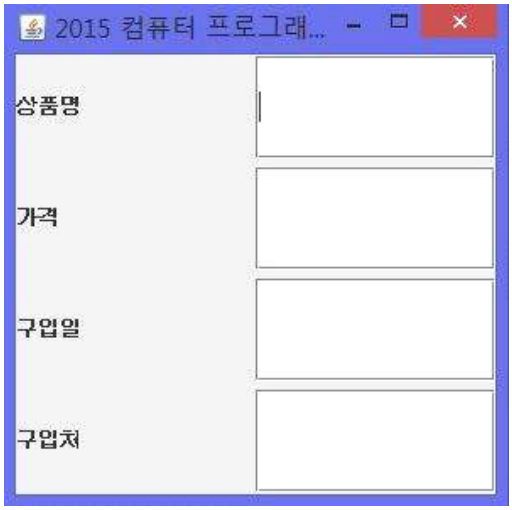
BorderLayout : Splitting container space to five blocks(East, West, South, North, Center) and locating components.

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

public class BorderLayout2extends JFrame {
        BorderLayout2() {
        setTitle("2015 Computer Programming");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        add(new JButton("1"), BorderLayout.NORTH);
        add(new JButton("2"), BorderLayout.SOUTH);
        add(new JButton("3"), BorderLayout.EAST);
        add(new JButton("4"), BorderLayout.WEST);
        add(new JButton("0"), BorderLayout.CENTER);
        setSize(300, 200);
        setVisible(true);
        }

        public static void main(String[] args) {
        new BorderLayout2();
        }
}
```



GridLayout : Splitting container space to same 4 rectangular and locating components

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

public class GridLayout3extends JFrame {
        GridLayout3() {
         setTitle("2015 Computer Programming");
         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
         GridLayout grid = new GridLayout(4, 2);
         grid.setVgap(5);
         setLayout(grid);
         add(new JLabel("Product Name"));
         add(new JTextField(""));
         add(new JLabel("Price"));
         add(new JTextField(""));
         add(new JLabel("Buying date"));
         add(new JTextField(""));
         add(new JLabel("Buying place"));
         add(new JTextField(""));
         setSize(300, 300);
         setVisible(true);
         }
        public static void main(String[] args) {
        new GridLayout3();
        }
}
```



Part5. Event Listener

Handler codes that handles event.

You make a event listener as a class and register a event handler at component.

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

public class EventListennerextends JFrame {
        EventListenner () {
        setTitle("Event Listener");
        setLayout(new FlowLayout());
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JButton btn = new JButton("Click");
        MyActionListener listener = new MyActionListener ();
        btn.addActionListener(listener);
        add(btn);
        setSize(300,150);
        setVisible(true);
 }

        public static void main(String[] args) {
        new EventListenner ();
```

```
        }
}

class MyActionListener implements ActionListener {
        public void actionPerformed(ActionEvent e) {
         JButton b = (JButton)e.getSource();
         if(b.getText().equals("Click"))
        b.setText("Clicked");
         else
        b.setText("Click");
         }
}
```

☞ ActionListener is implemented by MyActionListenner class. A 'btn' has listener, so it can handle a event.

[Exercise]

Make manipulating a font program. The initial screen show 'FONT' string on the center. Add the menu about 'color, kinds of font, location'. Consider several kinds of colors available.

- hint

JMenuBar, JMenu, JColorChooser

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

public class FontAction extends JFrame {
        Container contentPane;
        JLabel label = new JLabel("FONT");

        FontAction() {
                setTitle("FONT");
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                contentPane = getContentPane();
                label.setHorizontalAlignment(SwingConstants.CENTER);
                contentPane.add(label, BorderLayout.CENTER);
                createMenu();
                setSize(250,250);
```

```
                setVisible(true);
        }

        void createMenu() {
                /* add menu */
                // item : Color, font, top, bottom
                // use : JMenuBar, JMenu

                /* BLANK */

        }

        class MenuActionListener implements ActionListener {
                public void actionPerformed(ActionEvent e) {
                        /* add menu listener */
                        // change color, font, and location
                        String cmd = e.getActionCommand();

                        /*BLANK*/


                }
        }

        public static void main(String [] args) {
                new FontAction();
        }
}
```