Modul Pelatihan :
Pemrograman Berorientasi Obyek dengan Java
(Bagian 2)

Disusun oleh:

Nurmaya, S.Kom, M.Eng
Herika Hayurani, M.Kom
Nova Eka Diana, S.Kom, M.Eng

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS YARSI
APRIL 2015

Jl. Letjen. Suprapto, Cempaka Putih
Jakarta Pusat 10510 - Telp/Fax : (021) 4269301
Website : www.yarsi.ac.id

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

PRAKTIKUM 6
INHERITANCE (BAGIAN KEDUA)

Waktu : 30 menit

6.1 Tujuan

Tujuan dari Praktikum 6 adalah peserta mampu memahami konsep inheritance dalam Java.

6.2 Deskripsi

Inheritance (pewarisan) adalah konsep dalam pemrograman berorientasi object dimana sebuah class dapat mewarisi atribut dan method dari class lain. Class yang mewarisi disebut sebagai subclass, sedangkan class yang diwarisi disebut sebagai superclass. Saat deklarasi subclass, gunakan *keyword* `extends` setelah deklarasi class, lalu diikuti dengan nama superclass-nya.

Untuk memanggil constructor dari superclass-nya, suatu subclass harus memanggilnya dengan keyword super yang diletakkan pada statement pertama di constructor subclass.

Dengan menggunakan **Inheritance**, kita dapat melalukan

a. Overriding Method
b. Memanggil Constructor SuperClass dari Constructor Subclass

6.3 Praktikum

Pada praktikum 6 anda telah membuat class **Person**, **Student**, **Teacher**, dan **InternasionalStudent** di dalam **package mypeopleexample**. Class Student dan class Teacher merupakan subclass dari class Person. Class InternationalStudent merupakan subclass dari class Student. Class-class ini akan kita modifikasi untuk mempraktekan **Overriding Method**.

6.3.1 Overriding Method

Method subclass meng-override (menimpa) method superclass **hanya jika memiliki nama dan tipe parameter** yang sama seperti pada method superclass

- Ketika method X ada di superclass dan subclassnya, maka **overriding method** yang dieksekusi
- **Overriding method** (method yang menimpa) adalah method di subclass

- **Overriden method** (method yang ditimpa) adalah method di superclass

6.3.1.1 Class Person

Langkah-langkah yang harus dilakukan:

1. Buatlah class Person (tanpa method main)
2. Buatlah atribut/property/instance variable di class Person (deklarasikan dengan tipe yang sesuai):
    a. name → untuk menyimpan nama orang
    b. address → untuk menyimpan alamat orang
    c. hitungPenghasilan → untuk menghitung penghasilan orang
3. Buatlah method getter dan setter untuk setiap atribut Person.

Tabel 1 Class Person with additional method

```
package mypeopleexample;

public class Person {
     private String name;
     private String address;
       private double penghasilan;
     public String getName() {
          return name;
     }

     public void setName(String name) {
          this.name = name;
     }

     public String getAddress() {
          return address;
     }

     public void setAddress(String address) {
          this.address = address;
     }

     public void setPenghasilan(double penghasilan)
     {
          this.penghasilan = penghasilan;
     }
     public double getPenghasilan(){
      return penghasilan;
    }
}
```

6.3.1.2 Class Student

Langkah-langkah yang harus dilakukan:

1. Buatlah class Student (tanpa method main) yang merupakan subclass dari class Person
2. Buatlah atribut/property/instance variable di class Student (deklarasikan dengan tipe yang sesuai):
    a. school → untuk menyimpan sekolah siswa
    b. grade → untuk menyimpan nilai siswa
3. Buatlah method getter dan setter untuk setiap atribut Student.

Tabel 2 Class Student

```
package mypeopleexample;

public class Student extends Person {
     private String school;
     private double grade;

     public String getSchool() {
          return school;
     }

     public void setSchool(String school) {
          this.school = school;
     }

     public double getGrade() {
          return grade;
     }

     public void setGrade(double grade) {
          this.grade = grade;
     }
}
```

6.3.1.4 Class InternationalStudent

Langkah-langkah yang harus dilakukan:

1. Buatlah class InternationalStudent (tanpa method main) yang merupakan subclass dari class Student

2. Buatlah atribut/property/instance variable di class InternationalStudent (deklarasikan dengan tipe yang sesuai):
    a. country → untuk menyimpan asal negara
3. Buatlah method getter dan setter untuk setiap atribut InternationalStudent.

Tabel 3 Class InternationalStudent

```
package mypeopleexample;

public class InternationalStudent extends Student {
     /**
      * Creates a new instance of InternationalStudent
      */
     public InternationalStudent() {
     }

     private String country;

     public String getCountry() {
          return country;
     }

     public void setCountry(String country) {
          this.country = country;
     }
}
```

6.3.1.5 Class Teacher

Langkah-langkah yang harus dilakukan:

1. Buatlah class Teacher (tanpa method main) yang merupakan subclass dari class Person
2. Buatlah atribut/property/instance variable di class Teacher (deklarasikan dengan tipe yang sesuai):
    a. subject → untuk menyimpan subject yang diampu
    b. position → untuk menyimpan jabatan
3. Buatlah method getter dan setter untuk setiap atribut Teacher.
4. Buatlah method setPenghasilan yang terdapat di Class Person beserta parameternya ke Class Teacher

Tabel 4 Class Teacher

```
package mypeopleexample;

public class Teacher extends Person{
     private String subject;
     private int position;
     private double penghasilan;
     /*
      * position
      * 1 : dekan
      * 2 : wakil dekan
      * 3 : kps
      * 4 : no position
      * */

     public String getSubject() {
          return subject;
     }
     public void setSubject(String subject) {
          this.subject = subject;
     }
     public void setPosition(int position){
          this.position = position;
     }
     public int getPosition(){
          return position;
     }
     @Override
     public void setPenghasilan(double penghasilan) {
        if(getPosition()==1){
          penghasilan = penghasilan + 5000000;
        }
        else if(getPosition()==2){
          penghasilan = penghasilan + 3500000;
        }
        else if(getPosition()==3){
          penghasilan = penghasilan + 2500000;
        }
        else{
          this.penghasilan = penghasilan;
        }
        // TODO Auto-generated method stub
        super.setPenghasilan(this.penghasilan);
     }
}
```

6.3.1.6 Class MainTester

Langkah-langkah yang harus dilakukan:

1. Buatlah class MainTester (berisi method main)
2. Buatlah perintah untuk mencetak hasil Override Method di Class Teacher yaitu informasi penghasilan

Tabel 5 Mencetak Informasi Penghasilan di Class MainTester

```
package mypeopleexample;
import java.text.DecimalFormat;
public class MainTester {
  public static void main(String[] args) {
     // Create object instances and invoke methods.
     // Note that you can use methods defined
     // in a parent class for object
     // instances of the child class.
     Person person1 = new Person();
     person1.setName("Tom Jones");
     Student student1 = new Student();
     student1.setName("CCR");
     student1.setSchool("Lexington High");
     InternationalStudent internationalStudent1
               = new InternationalStudent();
     internationalStudent1.setName("Bill Clinton");
     internationalStudent1.setSchool("Lexington High");
     internationalStudent1.setCountry("Korea");
     Teacher teacher1 = new Teacher();
     teacher1.setName("Beatles");
     teacher1.setSubject("History");
     teacher1.setPosition(1);
```

```
        teacher1.setPenghasilan(7000000);
        /*
         * position
         * 1 : dekan
         * 2 : wakil dekan
         * 3 : kps
         * 4 : no position
         * */
        // Display name of object instances
        // using the getName() method
        // defined in the Person class.
        System.out.println("Displaying names of all object
instances");
        System.out.println(" person1.getName() = "
                + person1.getName());
        System.out.println(" student1.getName() = "
                + student1.getName());
        System.out.println(" internationalStudent1.getName() = "
                + internationalStudent1.getName());
        System.out.println(" teacher1.getName() = "
                + teacher1.getName());
        //Display penghasilan of object instances
        //using the getPenghasilan() method
        //defined in the person class
        System.out.println(" person1.getPenghasilan() = "
              + person1.getPenghasilan());
        System.out.println(" student1.getPenghasilan() = "
```

```
            + student1.getPenghasilan());

     System.out.println("
internationalStudent1.getPenghasilan() = "

            + internationalStudent1.getPenghasilan());

     System.out.println(" teacher1.getPenghasilan() = "

            + new
DecimalFormat("").format(teacher1.getPenghasilan()));

  }

}
```

PRAKTIKUM 7

INTERFACE

Waktu : 30 menit

7.1 Tujuan

Tujuan dari Praktikum 7 adalah peserta mampu memahami konsep interface dalam Java.

7.2 Deskripsi

Interface berisi sekumpulan method abstrak, yaitu method tanpa isi atau implementasi. Method-method tersebut harus diimplementasikan ke sebuah objek yang mengimplementasikan interface tersebut. Interface bersifat umum dan dapat digunakan oleh banyak kelas yang tidak saling berkaitan.

Ilustrasi :

Terdapat dua kelas yaitu sayuran dan baju yang memiliki sifat yang sama yaitu kedua-duanya perlu dicuci. Sehingga, kita bisa buat sebuah interface dengan nama Washable yang didalamnya dideklarasikan method wash. Interface tersebut akan diimplementasikan oleh Class Vegetable yaitu kelas sayuran dan Class Cloth yang merupakan kelas baju. Berikut contoh pemrogramannya.

```
public interface Washable
{
	public abstract void wash();
}

public class Vegetable implements Washable {
	...
	public void wash()  {
		// implementation
	}
}

public class Cloth implements Washable {
	...
	public void wash() {
		// implementation
	}
}
```

7.3 Praktikum

Anda akan membuat sebuah interface bernama Weight di `package mypeopleexample`, yang merupakan sebuah interface untuk menyimpan informasi berat . Dilanjutkan denga memodifikasi class Person yang merupakan kelas orang dan membuat kelas baru di `package mypeopleexample` yaitu

class Car yang merupakan kelas mobil. Tambahkan cetak berat mobil di method MainTester.

7.3.1 Interface Weight

Langkah-langkah yang harus dilakukan:

1. Buatlah sebuah interface dengan nama Weigth
2. Buatlah method setWeight dan getWeight

Tabel 6 interface Weight

```
package mypeopleexample;

public interface Weight {

     public void setWeight(int weight);

     public int getWeight();

}
```

7.3.2 Class Person Implements Interface Weight

Langkah-langkah yang harus dilakukan:

1. Implementasikan interface Weight ke class Person
2. Buatlah atribut/property/instance variable di class Person (deklarasikan dengan tipe yang sesuai):
    a. weight → untuk menyimpan berat orang
3. Override method setWeight dan getWeight yang terdapat di interface Weight

Tabel 7 interface Weight

```
public class Person implements Weight {

    private int weight

    @Override

    public void setWeight(int weight) {

     this.weight = weight;

    }
```

```
    @Override
     public int getWeight() {
          // TODO Auto-generated method stub
          return weight;
     }
}
```

7.3.3 Class Car

1. Buatlah class Car (tanpa method main)
2. Buatlah atribut/property/instance variable di class Person (deklarasikan dengan tipe yang sesuai):
    a. weight → untuk menyimpan berat mobil
3. Implementasikan interface Weight
4. Override method setWeight dan getWeight yang terdapat di interface Weight

Tabel 8 interface Weight

```
package mypeopleexample;
public class Car implements Weight{
     int weight;
     @Override
     public void setWeight(int weight) {
          this.weight = weight;
     }
     @Override
     public int getWeight() {
          return weight;
     }
}
```

7.3.4 Mencetak informasi berat di class MainTester

Tabel 9 Modifikasi Class MainTester (cetak Berat)

```
public class MainTester {

     public static void main(String[] args) {
      // Create object instances and invoke methods.
      // Note that you can use methods defined
      // in a parent class for object
      // instances of the child class.
      Person person1 = new Person();
      person1.setName("Tom Jones");
      person1.setWeight(65);
      Car car1 = new Car();
      car1.setWeight(1270);

      System.out.println(" person1.setWeight() = "
                    + person1.getWeight());
      System.out.println(" car1.setWeight = "
                    + car1.getWeight());
     }
}
```

7.3.5 Konversi Tipe Class dan Interface

Anda dapat mengkonversi dari tipe class ke tipe interface ataupun sebaliknya, dengan asumsi telah disediakan class yang “implements” interface tersebut.

Tabel 10 Konversi Tipe Class dan Interface

```
public class MainTester {

     public static void main(String[] args) {

          Weight iweight;

          //Class to Interface

          Person person1 = new Person();

          iweight = person1;

          iweight.setWeight(65);

          System.out.println(" weight.setWeight() = "

                    + iweight.getWeight());
```

```
            //Interface to Class
            Person p1 = (Person) iweight;
            p1.setWeight(70);
            System.out.println(" p1.setWeight = "
                        + p1.getWeight());
      }
}
```

PRAKTIKUM 8

POLIMORFISME dan INNER CLASS

Waktu : 30 menit

8.1 Tujuan

Tujuan dari Praktikum 8 adalah peserta mampu memahami konsep Polimorfisme dalam Java.

8.2 Deskripsi

8.2.1 POLIMORFISME

Polimorfisme adalah konsep penting dalam OOP yang banyak digunakan di Java dan bahasa pemrograman lainnya. Dua jenis polimorfisme adalah Overloading dan Overriding

8.2.1.1 Overloading

Penggunaan nama yang sama untuk lebih dari 1 method /constructor pada class yang sama. Contohnya lihat tabel

8.2.1.2 Overriding

Penggunaan nama dan tipe parameter yang sama untuk lebih dari 1 method pada superclass dan subclass-nya. Contohnya lihat **Tabel 9**.

8.2.2 INNER CLASS

Inner Class adalah suatu class yang didefinisikan di dalam Class lain layaknya seperti variabel atau method pada sebuah Class. Oleh karena itu, instance dari Inner Class dapat mengakses semua member dari outer Classnya, bahkan yang private.

8.3 Praktikum

8.3.1 Overloading

Langkah-langkah membuat overloading adalah sebagai berikut :

a. Tambahkan dua method yang memiliki nama yang sama yaitu setNIP dengan input parameter yang berbeda di class Student. Method setNIP pertama memiliki input parameter bertipe Integer. Sedangkan yang kedua bertipe

String. Kemudian buat dua buah method getNIP berdasarkan tipe data yang dikembalikan.

Tabel 11 Overloading Method di class Student

```
package mypeopleexample;

public class Student extends Person {
        private String school,snip;
        private double grade;
        private int nip;

    public void setNIP(int nip){
        this.nip = nip;
    }
    public void setNIP(String snip){
        this.snip = snip;
    }
    public int getNIP(){
        return nip;
    }
    public String getSNIP(){
        return snip;
    }
    public String getSchool() {
        return school;
    }

    public void setSchool(String school) {
        this.school = school;
    }
}
```

b. Tambahkan perintah cetak NIP (Nomor Induk Pelajar) di kelas utama yaitu MainTester

Tabel 12 Cetak Overloading Method

```
package mypeopleexample;

import java.text.DecimalFormat;

public class MainTester {

    public static void main(String[] args) {

        Student student1 = new Student();
```

```
            student1.setName("CCR");

            student1.setSchool("Lexington High");

            student1.setNIP(120300004);

            InternationalStudent internationalStudent1

            = new InternationalStudent();

            internationalStudent1.setName("Bill
Clinton");

            internationalStudent1.setSchool("Lexington
High");

            internationalStudent1.setCountry("Korea");

            internationalStudent1.setNIP("120300004X");


            System.out.println(" student1.getName() = "

                      + student1.getName());

            System.out.println(" student1.getSchool() = "

                      + student1.getSchool());

            System.out.println(" student1.getNIP() = "

                      + student1.getNIP());

            System.out.println("
internationalStudent1.getName() = "

                      + internationalStudent1.getName());

            System.out.println("
internationalStudent1.getSchool() = "

                      +
internationalStudent1.getSchool());

            System.out.println("
internationalStudent1.getSNIP() = "
```

```
                        + internationalStudent1.getSNIP());
      }
}
```

8.3.2 Inner Class

Contoh membuat Inner Class di Class MainTester adalah sebagai berikut

- Buatlah sebuah Inner Class dengan nama ChangingFormat yang berfungsi untuk mengubah format double menjadi format decimal
- Didalam Method Main panggilah Inner Class tersebut untuk mengubah format nilai penghasilan guru menjadi format decimal

Tabel 13 Membuat Inner Class di Class MainTester

```
package mypeopleexample;

import java.text.DecimalFormat;

public class MainTester {

   public static void main(String[] args) {

      MainTester mt = new MainTester();

      ChangingFormat cf = mt.new ChangingFormat();

      Student student1 = new Student();

      student1.setName("CCR");

      student1.setSchool("Lexington High");

      InternationalStudent internationalStudent1

                  = new InternationalStudent();

      internationalStudent1.setName("Bill Clinton");

      internationalStudent1.setSchool("Lexington High");

      internationalStudent1.setCountry("Korea");

      Teacher teacher1 = new Teacher();

      teacher1.setName("Beatles");
```

```
      teacher1.setSubject("History");
      teacher1.setPosition(1);
      teacher1.setPenghasilan(7000000);
      /*
       * position
       * 1 : dekan
       * 2 : wakil dekan
       * 3 : kps
       * 4 : no position
      * */
      System.out.println(" student1.getPenghasilan() = "
                          +
cf.getFormatChanging(student1.getPenghasilan()));
            System.out.println("
internationalStudent1.getPenghasilan() = "
                          +
cf.getFormatChanging(internationalStudent1.getPenghasilan())
);
            System.out.println(" teacher1.getPenghasilan() =
"
                          +
cf.getFormatChanging(teacher1.getPenghasilan()));
      }
      class ChangingFormat{
       String formatChange;
       public String getFormatChanging(double penghasilan){
                  DecimalFormat df = new DecimalFormat("");
                  formatChange = df.format(penghasilan);
```

```
                    return formatChange;

                }

            }

}
```

PRAKTIKUM 9

GRAPHICAL USER INTERFACE (GUI)

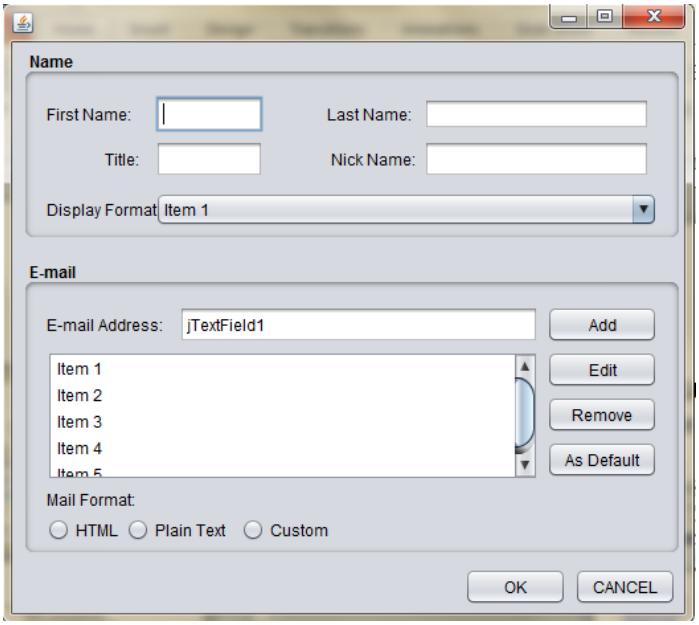
Waktu : 30 menit

9.1 Tujuan

Tujuan dari Praktikum 9 adalah peserta memahami mengenai konsep GUI dalam Java.

9.2 Deskripsi

GUI adalah sebuah interaksi antarmuka pengguna yang menggunakan metode interaksi pada perangkat elektronik secara grafis antara pengguna dan komputer. Java GUI adalah sebuah pemrograman dengan bahasa Java yang dibuat menggunakan aplikasi berbasis GUI.



Gambar 1 Contoh Aplikasi GUI

9.2.1 Element Penting di GUI

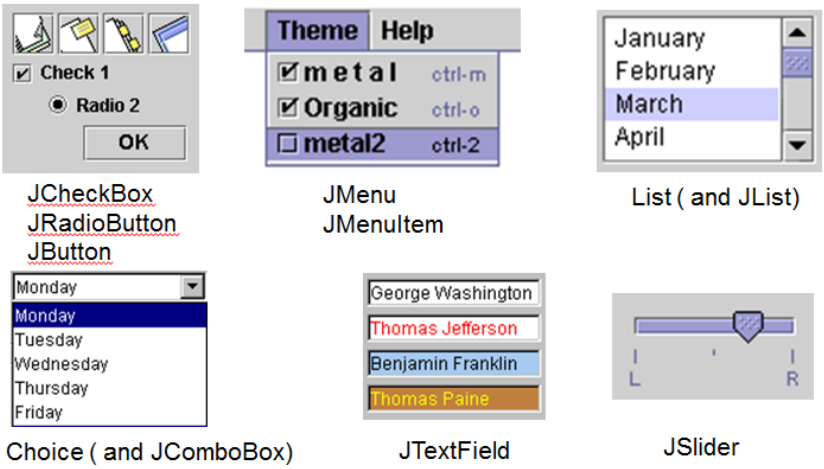
- **Component**
  Elemen-elemen yang menempati area pada layar, misalnya: button, textfield, label
- **Container**
  Area yang dapat berisi atau memegang komponen lainnya, misal: window, panel.

- **Event Handler**
  Bagaimana cara menangani event/aksi dari suatu komponen (eventAction) dan apa yang harus dilakukan (listener).

9.2.2 Java API : GUI

a. SWING (javax.swing.*)

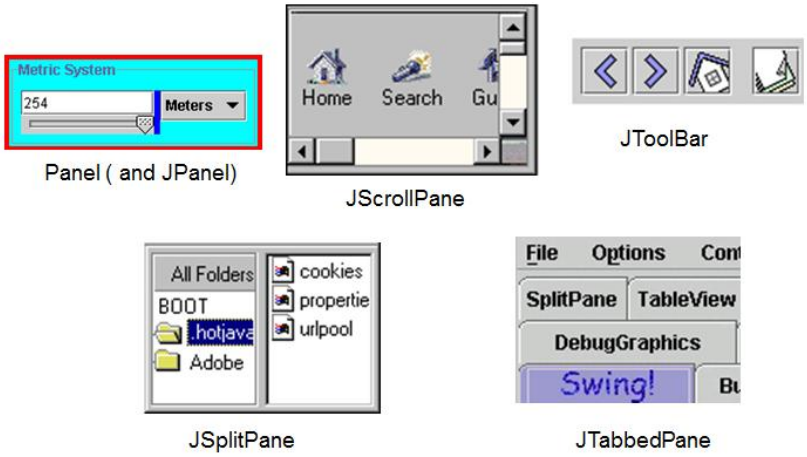
— **JComponent** : JTextField, JLabel, JButton, JComboBox, Jpanel, dan lain-lain



Gambar 2 Contoh JComponent

— **Container** : JFrame, JApplet, dan lain-lain

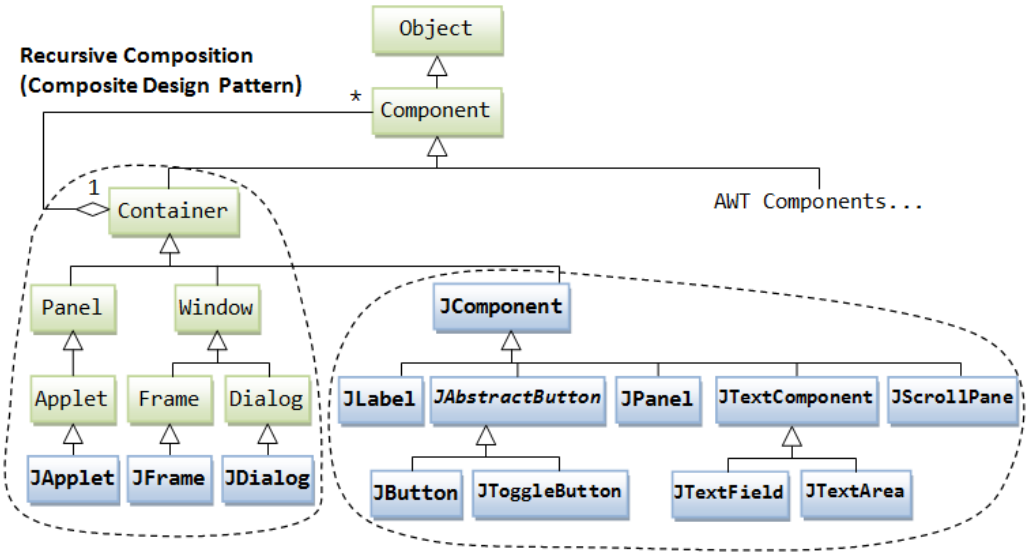
- JComponent tidak boleh langsung ditambahkan ke top-level container (JFrame, JApplet) karena merupakan *lightweight components'*
- Harus ditambahkan ke *content-pane* dari top-level container
- *Content-pane* adalah container yang bisa diset layout-nya, dan digunakan untuk mengelompokkan component.
- Ada dua cara:
  1. Get content-pane *getContentPane()* dan menambahkan komponen diatasnya (default layout : BORDER_LAYOUT)
  2. Set content-pane ke JPanel



Gambar 3 Contoh Container

b. Event AWT (java.awt.event.*)

- **Event class**: ActionEvent, MouseEvent, KeyEvent, WindowEvent
- **Event Listener interface**: ActionListener, MouseListener, KeyListener, WindowListener
- **Event Listener Adapter**: MouseAdapter, KeyAdapter, WindowAdapter



Gambar 4 API GUI

9.3 Praktikum

Langkah-langkah dalam membuat Aplikasi GUI

1. Create Window

Tahapan-tahapan dalam membuat sebuah Window.

1. Buat Class bertipe JFrame

   Buatlah sebuah kelas frame yang bernama MyFrame yang mengextends class JFrame pada package **mygui**.

2. Tampilkan window (frame) dengan size tertentu

   setSize(300, 300); // Untuk mengeset width dan height frame

   setVisible(true); // Untuk menampilkan frame

Tabel 14 Membuat Window

```
package mygui;

import javax.swing.*;

public class MyFrame extends JFrame{

     public MyFrame(){

          setSize(300,300);

          setTitle("My Empty Frame");


     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

          setVisible(true);

     }

     public static void main(String[] args) {

          MyFrame frame = new MyFrame();

     }

}
```

2. Add Component

Tambahkan component-component seperti button, textfield yang disesuaikan dengan kebutuhan

Tabel 15 Add Component

```
public class MyFrame extends JFrame {

     public MyFrame() {

       setSize(300,300);

       setTitle("My Empty Frame");

       setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

       JButton button = new JButton("Click Me");

       this.getContentPane().add(button);

       setVisible(true);

     }

     public static void main(String[] args) {

       MyFrame frame = new MyFrame();

     }

}
```

3. Event Handling & Listener

- Membuat agar tombol “Click Me” akan memberikan sebuah aksi ketika diklik
- Dua hal yang harus disediakan, yaitu:

  1. **Method** yang akan dipanggil ketika tombol diklik
  2. **Cara** untuk memanggil method tersebut

  Mengimplementasikan sebuah **interface listener** yang dapat mendengar event dari **event source** yaitu tombol “Click Me”. Setiap jenis event memiliki **interface listener** yang sesuai dengan eventnya.

  Contoh :

  a. **MouseEvent – implements MouseListener**
  b. **ActionEvent – implements ActionListener**

Adapun tahapan mengimplementasikan **Event Listener** untuk **ActionEvent**

1. Class meng-implements interface **ActionListener**
2. Registar ke komponen yang ingin didengarkan
3. Definisikan method event-handling untuk mendengarkan ActionListener interface → implementasi method **actionPerformed**

Tabel 16 Add Event Handling dan Event Listener

```
package mygui;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.*;

public class MyFrame extends JFrame implements
ActionListener{
     JButton button;

     public MyFrame(){
          setSize(300,300);
          button = new JButton("Click Me");
          button.addActionListener(this);
          this.getContentPane().add(button);
          setTitle("My Empty Frame");

     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
          setVisible(true);
     }
     public static void main(String[] args) {
          MyFrame frame = new MyFrame();
     }
     @Override
     public void actionPerformed(ActionEvent arg0) {
          JOptionPane.showMessageDialog(this, "HAI-
HAI");
          button.setText("I've been Clicked");

     }
}
```

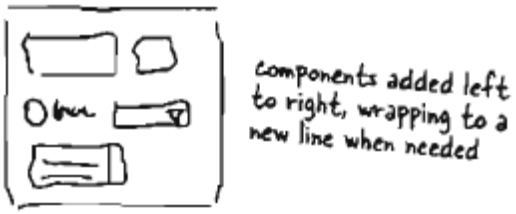
4. Layout

Untuk merapihkan letak component-componet frame dapat menggunakan Layout Manager. Berikut jenis-jenis dari Layout Manager

1. **Border Layout**
   - Membagi background component ke dalam lima daerah utama
   - Merupakan layout manager default untuk frame

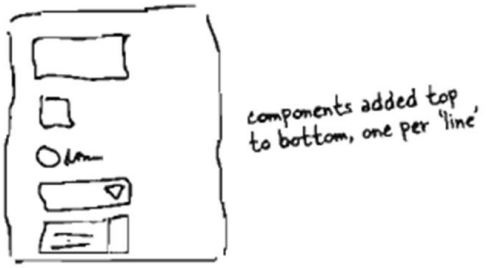
Gambar 5 Border Layout

2. **Flow Layout**
   - Menyusun komponen dari kiri ke kanan
   - Jika lebar window tidak mencukupi, maka komponen baru akan diletakkan pada baris berikutnya



Gambar 6 Flow Layout

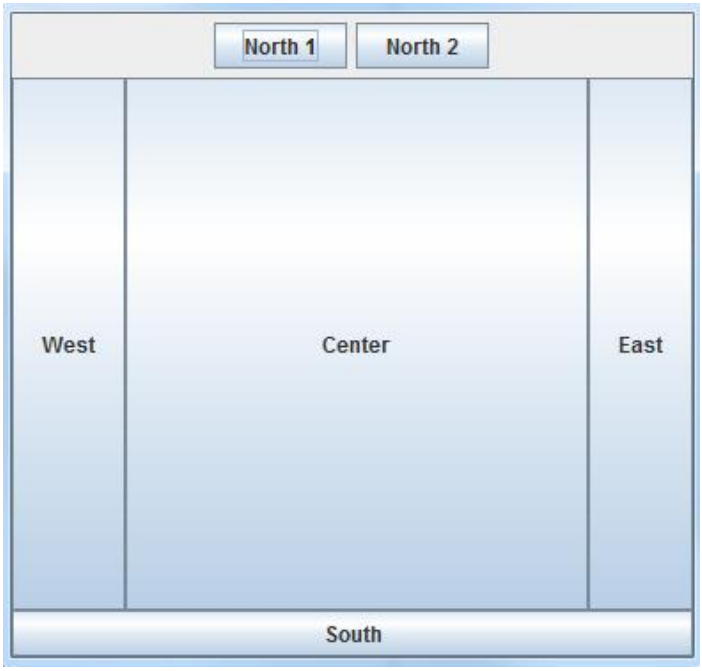
3. **Box Layout**
   - Mirip dengan FlowLayout
   - Namun, mampu menyusun komponen-komponen secara vertikal



Gambar 7 Box Layout

Latihan:

Buatlah program java untuk membuat tampilan GUI seperti dibawah ini



Gambar 8 Aplikasi GUI dengan Layout Manager

Tabel 17 Program Latihan Java GUI

```
package mygui;
import java.awt.BorderLayout;
import java.awt.Container;
import java.awt.FlowLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
public class MyFrame extends JFrame {
     JButton button;
     public MyFrame(){
      setSize(400,400);
      setTitle("Layout Manager Example");
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setVisible(true);
      Container con = getContentPane();
      con.setLayout(new BorderLayout());
      JPanel northPanel = new JPanel();
      northPanel.setLayout(new FlowLayout());
      northPanel.add(new JButton("North 1"));
      northPanel.add(new JButton("North 2"));
      con.add(northPanel,BorderLayout.NORTH);
      con.add(new JButton("South"),BorderLayout.SOUTH);
      con.add(new JButton("West"),BorderLayout.WEST);
```

```
      con.add(new JButton("East"),BorderLayout.EAST);
      con.add(new
JButton("Center"),BorderLayout.CENTER);
     }
     public static void main(String[] args) {
          MyFrame frame = new MyFrame();
     }
}
```

Untuk informasi lebih lanjut, silakan menghubungi:

Program Studi Teknik Informatika
Fakultas Teknologi Informasi
Universitas YARSI

1. Nurmaya, S.Kom, M.Eng
   E-mail : nurmaya@yarsi.ac.id
2. Herika Hayurani, M.Kom
   E-mail : herika.hayurani@yarsi.ac.id
3. Nova Eka Diana, S.Kom, M.Eng
   E-mail : nova.diana@yarsi.ac.id