

UNIVERSITAS PAHLAWAN TUANKU TAMBUSAI  
FAKULTAS TEKNIK  
PROGRAM STUDI TEKNIK INFORMATIKA

**BATAS MATERI KULIAH**

Mata Kuliah : PEMOGRAMAN BERORIENTASI OBJEK

Semester / SKS : 3 / 4

Kelas/Tahun Akd : A / 2020/2021 Ganjil

Dosen Pengampu : DEDDY GUSMAN, S.Kom, M.T.I

Dosen Pengajar :

NO	HARI/TGL	MATERI	PARAF DOSEN	P. KETUA KELAS
1	Jumat 10-09-2020	Kontrak Kuliah		Rufu
2	Jumat 02-10-2020	Konsep dasar OOP		Rufu
3	Jumat 09-10-2020	Paradigma dan tools dalam OOP		Rufu
4	Jumat 16-10-2020	Use Case Diagram		Rufu
5	Jumat 23-10-2020	Use Case Diagram		Rufu
6	Jumat 13-11-2020	Relasi Use Case Diagram		Rufu
7	Jumat 20-11-2020	OOP Praktek		Rufu
8	Jumat 04-12-2020	Programming, PHP dasar, class,		Rufu
9	11/12-2020	Method Overloading		Rufu
10	18/12-2020	Encapsulation		Rufu
11	18/12-2020	Basic concept, Declaring, accessing, accessibility		Rufu
12	25/12-2020	Package keyword		Rufu
13	3/1-2020	Interface		Rufu
14	10/1-2020	Polymorphic arguments		Rufu
15	17/1-2020	Abstract class, abstract Methods		Rufu
16	17/1-2020	UAS		Rufu

**R P S**

**PEMROGRAMAN BERORIENTASI OBJEK**

RENCANA PEMBELAJARAN SEMESTER PROGRAM STUDI S1 TEKNIK INFORMATIKA UNIVERSITAS PAHLAWAN TUANKU TAMBUSAI						
MATA KULIAH	KODE	RUMPUN MK	BOBOT (SKS)		SEMESTER	DIREVISI
Pemrograman Berorientasi Objek	INF 211		3	-	IV	
<b>Capaian Pembelajaran</b>	<b>Sikap (CPL-S)</b>					
	CPL-S-03	Menginternalisasi nilai, norma, dan etika akademik.				
	CPL-S10	Menunjukkan sikap bertanggungjawab atas pekerjaan di bidang keahliannya secara				
	<b>Keterampilan Khusus (CPL-KK)</b>					
	CPL-KK02	mampu menemukan sumber masalah rekayasa pada sistem sistem komputer, jaringan				
	CPL-KK05	mampu merancang sistem komputer, jaringan komputer, sistem tertanam, atau sistem				
	CPL-KK06	Mampu memilih sumberdaya dan memanfaatkan perangkat perancangan dan analisis				
	<b>Program Studi</b>					
	Mahasiswa memiliki keterampilan pemrograman berorientasi objek secara konsep dan aplikatif dengan menggunakan bahasa					
	<b>Mata Kuliah</b>					
1. Mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek						
2. Memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek						
<b>Deskripsi Singkat MK</b>	Mata kuliah ini adalah untuk memungkinkan siswa untuk mengasimilasi, mengidentifikasi dan menganalisa infrastruktur TI modern dan teknologi baru untuk perusahaan. Siswa akan belajar bagaimana memanfaatkan infrastruktur TI dan teknologi baru untuk melayani kebutuhan organisasi dan meningkatkan posisi kompetitif perusahaan. Kursus ini juga					
<b>Pustaka</b>	<b>Utama:</b>					
	1. Setiawan, Yudha C, 2004, Object Oriented Programming OOP, Andi Offset, Yogyakarta 2. Tim, 2014, <i>Modul Praktikum Pemrograman Java</i> , Teknokrat, Bandar Lampung					

	<b>Pendukung:</b>	
	1. Burd, B, 2011, Java for Dummies 5 <sup>th</sup> Edition, New Jersey: Wiley Publishing. 2. Deitel, P., & DeitelH, 2011, Java How to Program (9 <sup>th</sup> ed), Prentice Hall 3. Hortsman, C.S & Cornell, G. 2008. Core Java Fundamentals (8 <sup>th</sup> ed) 4. Wu, C.T, 2010, An Introduction to Object-Oriented Programming with Java 5 <sup>th</sup> Edition, New York: McGraw Hill	
<b>Media Pembelajaran</b>	<b>Software:</b>	<b>Hardware:</b>
	- Microsoft PowerPoint, Net Beans	Buku, Laptop, LCD Proyektor dan papan tulis
<b>Dosen Pengampu</b>		
<b>Mata Kuliah Syarat</b>		

Mg Ke-	CP-MK (Sesuai tahapan belajar)	Materi Pembelajaran (Pustaka)	Metode/ Strategi Pembelajaran	Assesment		
				Indikator	Bentuk	Bobot
1-2	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa	Pendahuluan	Ceramah, Tanya Jawab, Praktikum	Mahasiswa dapat menjelaskan perbedaan paradigma pemrograman terstruktur dan paradigma pemrograman berorientasi objek	Pengamatan dan Penugasan	

3-4	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Instruksi Percabangan dan Perulangan (Pemilihan kondisional dengan ekspresi boolean)	Ceramah, Tanya Jawab, Praktikum	<ul style="list-style-type: none"> <li>a. Mahasiswa memahami prinsip dasar logika pemilihan kondisional dengan menggunakan ekspresi boolean</li> <li>b. Mahasiswa mampu menggunakan instruksi percabangan dalam pengembangan program</li> <li>c. Mahasiswa memahami prinsip dasar struktur perulangan</li> <li>d. Mahasiswa mampu menggunakan instruksi perulangan dalam membuat program aplikasi</li> <li>e. Mahasiswa memahami perbedaan mendasar instruksi perulangan</li> </ul>	Pengamatan dan Penugasan	
-----	---	--	---------------------------------	---	--------------------------	--

5-6	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Kelas & Objek	Ceramah, Tanya Jawab, Praktikum	<p>a. Mahasiswa memahami tentang konsep objek dan kelas dalam pemrograman</p> <p>b. Mahasiswa memahami tipe-tipe kelas yang ada dalam OOP</p> <p>c. Mahasiswa memahami penggunaan objek dan kelas dalam pemrograman</p> <p>Mahasiswa mampu membuat aplikasi dengan bahasa pemrograman menggunakan kelas dan objek</p>	Pengamatan dan Penugasan	
7-8	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Array	Ceramah, Tanya Jawab, Praktikum	<p>a. Mahasiswa memahami tentang konsep array 1 dimensi</p> <p>b. Mahasiswa memahami tentang konsep array 2 dimensi</p> <p>c. Mahasiswa memahami tentang</p>	Pengamatan dan Penugasan	
9	<b>QUIS TEORI</b>					
10-14	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Hubungan Antar Kelas	Ceramah, Tanya Jawab, Praktikum	<p>a. Mahasiswa memahami jenis kelas dalam OOP</p> <p>b. Mahasiswa memahami pembuatan kelas</p> <p>c. Mahasiswa memahami relasi antar kelas</p>	Pengamatan dan Penugasan	
15-16	<b>UTS (PRAKTIKUM DAN TEORI)</b>					25

17-20	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki	Inheritance	Ceramah, Tanya Jawab, Praktik	Mahasiswa dapat menjelaskan konsep-konsep yang mendasari pemrograman berorientasi objek inheritance.	Pengamatan dan Penugasan	
	keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek					
21-22	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Polimorfisme	Ceramah, Tanya Jawab, Praktikum	Mahasiswa dapat menjelaskan konsep-konsep yang mendasari pemrograman berorientasi objek polimorfisme.	Pengamatan dan Penugasan	
23-24	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Encapsulation	Ceramah, Tanya Jawab, Praktikum	Mahasiswa dapat menjelaskan konsep-konsep yang mendasari pemrograman berorientasi objek encapsulation.	Pengamatan dan Penugasan	
25	<b>QUIS PRAKTIKUM</b>					
26	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Pengenalan Java GUI	Ceramah, Tanya Jawab, Praktikum	a. Mahasiswa mendesain tampilan interface menggunakan GUI. b. Mahasiswa mampu menghubungkan antar kelas dengan menggunakan GUI	Pengamatan dan Penugasan	
27-28	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Swing component dan event & exception handling	Ceramah, Tanya Jawab, Praktikum	1. Mahasiswa memahami komponen-komponen Swing component dan event 2. Mahasiswa memahami komponen-komponen exception handling	Pengamatan dan Penugasan	

29-30	Setelah menyelesaikan mata kuliah ini mahasiswa dapat memahami tentang paradigma dan konsep dasar pemrograman berorientasi objek dan memiliki keterampilan dasar mengembangkan aplikasi menggunakan bahasa pemrograman berorientasi objek	Java database connectivity	Ceramah , Tanya Jawab, Praktikum	Mahasiswa mampu membuat aplikasi berbasis windows API	Pengamatan dan Penugasan	
<b>Ujian Akhir Semester</b>						25



31-32	<b>Ujian Akhir Semester</b>					<b>25%</b>

ST-RK-1.16-082-007/R-



# Modul Praktikum

## PEMROGRAMAN BERORIENTASI OBJEK

Mahasiswa dapat mengerti dan memahami serta dapat mengimplementasikan konsep Object Oriented Programming menggunakan java



*Join Us*

**THE  
CYBER  
campus**  
<http://www.stikom.edu>

## Laboratorium Komputer



# DAFTAR ISI

DAFTAR ISI.....	1
PETUNJUK UMUM.....	3
MODUL 1 .....	5
1.1.    Class .....	6
1.2.    Instant of Class .....	6
1.3.    Keyword “this” .....	8
1.4.    Atribut & Method .....	9
1.5.    Access Modifier .....	9
1.6.    Static Variable vs Instant Variable .....	10
1.7.    Input Data .....	11
1.8.    Java Documentation.....	13
Latihan .....	14
MODUL 2 .....	17
2.1.    Apa itu Constructor ?.....	18
2.2.    Multiple Constructor .....	18
2.3.    Function Overloading.....	19
2.4.    Mengetahui Inheritance .....	20
Latihan .....	23
MODUL 3 .....	25
3.1.    Berkenalan dengan Polymorphism .....	26
3.2.    Method Overriding.....	29
Latihan .....	30
MODUL 4 .....	33
4.1.    Lebih Jauh dengan Abstract Class .....	34
4.2.    Keyword “final” .....	35
4.3.    Abstract Method.....	36
Latihan .....	37
MODUL 5 .....	39
5.1.    Deskripsi interface .....	40
5.2.    Deklarasi interface.....	40
5.3.    Implementasi interface .....	40
Latihan .....	42
MODUL 6 .....	45
6.1.    InputStream dan OutputStream .....	46
6.2.    FileInputStream dan FileOutputStream.....	46
Latihan .....	49

MODUL 7 .....	51
7.1.    JFrame sebagai Class Utama .....	52
7.2.    Komponen-komponen GUI .....	54
7.3.    Layout Management.....	56
Latihan .....	60
MODUL 8 .....	61
8.1.    Event Handler .....	62
8.2.    ActionListener .....	63
8.3.    KeyListener.....	64
8.4.    MouseListener .....	65
Latihan .....	68

# PETUNJUK UMUM

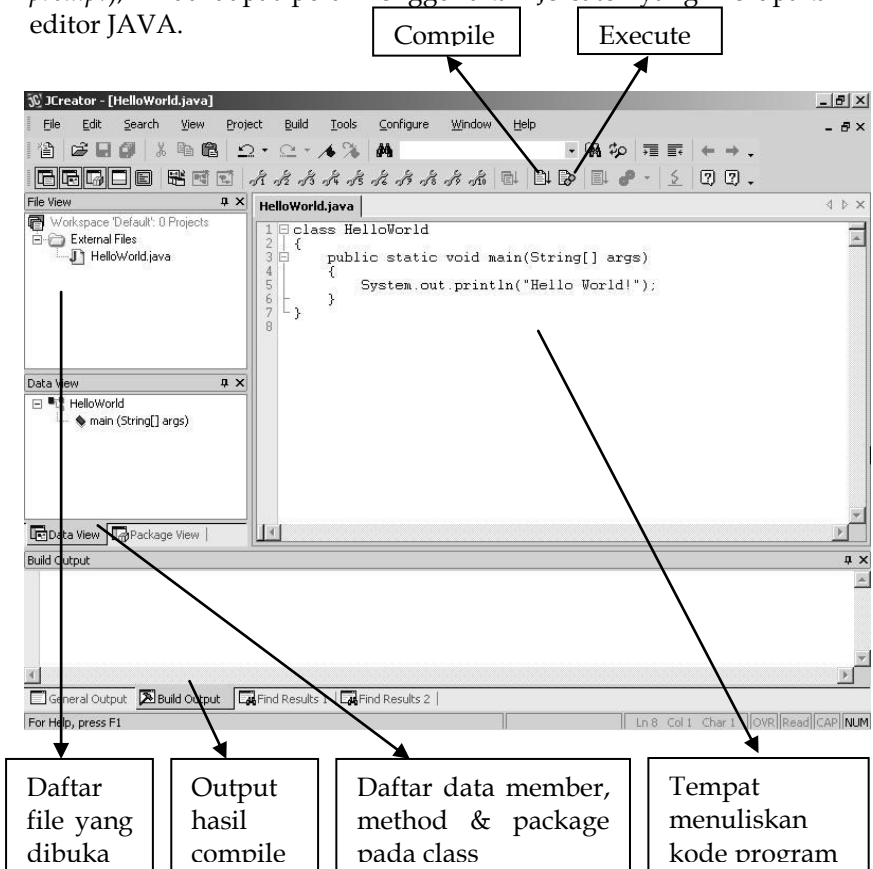
**Compile program java**, menggunakan perintah *javac*, contoh :

```
c:/>javac nama_file.java
```

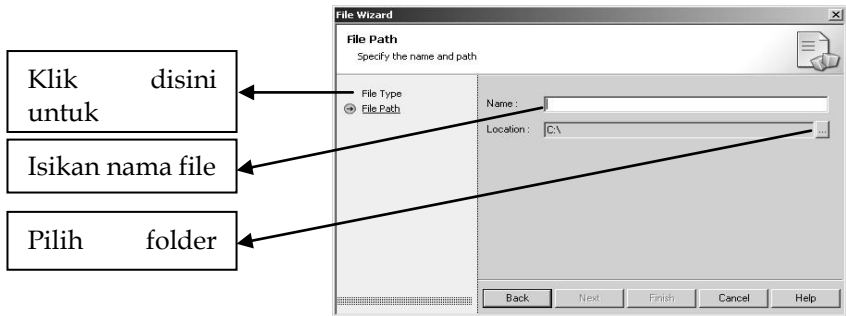
**Running program java**, menggunakan perintah *java*, contoh :

```
c:/>java nama_file
(file ini berektensi *.class)
```

Selain dengan mengcompile manual lewat DOS mode (*command prompt*), Anda dapat pula menggunakan Jcreator yang merupakan editor JAVA.



Saat memilih New File, maka Anda diminta untuk mengisi nama file pilihlah tipe file yang akan dibuat tersebut.



# MODUL 1

## PENGENALAN OBJECT & CLASS

*The beginning is  
the most important part of the work  
~ Plato ~*



### **Tujuan :**

Memahami konsep objek & kelas  
Memahami konsep Access Modifier & static variable

### **Materi :**

Object & Class, Instant of class, Attribute & Method  
Access modifier (default, public, protected, private) & Static Variable

### **Referensi :**

- <http://homepages.north.londonmet.ac.uk/~chalkp/proj/ootutor/ooconcepts.html>
- Harianto, Bambang. *Esensi-esensi Bahasa Pemrograman Java*. Informatika Bandung, 2003.
- Hermawan, Benny. *Menguasai Java 2 dan Object Oriented Programming*. Penerbit Andi, 2004.



## 1.1. Class

Class merupakan cetak biru (blue print) dari objek atau dengan kata lain sebuah Class menggambarkan ciri-ciri objek secara umum. Sebagai contoh Suzuki Smash, Yamaha VegaR, Honda SupraFit, dan Kawasaki KazeR merupakan objek dari Class sepeda motor. Suzuki Smash dan objek lainnya juga memiliki kesamaan atribut (merk, tipe, berat, kapasitas bensin, tipe mesin, warna, harga) dan method untuk mengakses data pada atributnya (misal fungsi untuk menginputkan data merk, tipe, berat, dsb serta fungsi untuk mencetak data merk, tipe, berat, dsb).

Contoh :

```
class SepedaMotor
{
    private String merk, tipe;
    private int tangki;
    private long harga;

    public void setMerk(String merk) {
        this.merk = merk;
    }
    public String getMerk(){
        return merk;
    }
}
```

## 1.2. Instant of Class

### a. Object

Objek merupakan segala sesuatu yang ada didunia ini, yaitu manusia, hewan, tumbuhan, rumah, kendaraan, dan lain sebagainya. Contoh-contoh objek yang telah disebutkan diatas merupakan contoh objek nyata pada kehidupan kita. Pada pemrograman berorientasi objek, kita akan belajar bagaimana membawa konsep objek dalam kehidupan nyata menjadi objek dalam dunia pemrograman.



Setiap objek dalam dunia nyata pasti memiliki 2 elemen penyusunnya, yaitu keadaan (*state*) dan perilaku/sifat (*behaviour*).

Sebagai contoh, sepeda memiliki keadaan yaitu warna, merk, jumlah roda, ukuran roda. Dan

perilaku/sifat sepeda adalah berjalan, berhenti, belok, menambah kecepatan, mengerem. Pada saat objek diterjemahkan ke dalam konsep PBO, maka elemen penyusunnya juga terdiri atas 2 bagian, yaitu :

- Atribut, merupakan ciri-ciri yang melekat pada suatu objek (*state*).
- Method, merupakan fungsi-fungsi yang digunakan untuk memanipulasi nilai-nilai pada atribut atau untuk melakukan hal-hal yang dapat dilakukan suatu objek (*behaviour*).

Objek dalam konsep PBO memiliki keadaan dan perilaku yang sama seperti halnya objek di dunia nyata, karena objek dalam konsep PBO merupakan representasi objek dari dunia nyata. Objek dalam PBO merepresentasikan keadaan melalui variabel-variabel (Atribut), sedangkan perilakunya direpresentasikan dengan method (yang merupakan suatu fungsi yang berhubungan dengan perilaku objek tersebut maupun berhubungan dengan atribut dari objek tersebut).

Objek yang memiliki kesamaan atribut dan method dapat dikelompokkan menjadi sebuah Class. Dan objek-objek yang dibuat dari suatu class itulah yang disebut dengan Instant of class. Untuk menginstansi (membuat) objek dari class, gunakan operator *new*.

Sintaks membuat objek dari suatu class :

```
namaClass namaObjek = new namaClass()
```

Class utama dari program :

```
class Latihan1a
{
    public static void main (String
[]args) {
        SepedaMotor motor = new
SepedaMotor();
        motor.setMerk("Suzuki");
    }
}
```

```
        System.out.println("Motor        ini  
bermerk " +  
        motor.getMerk());  
    }  
}
```

Perhatikan class Latihan1a diatas!

Nama objek (*instant of class*) dari class SepedaMotor adalah motor.

### b. Anonymous Object

Berbeda dengan object biasa, anonymous object merupakan objek yang tidak memiliki nama. Anonymous object tidak memakan resource memori. Namun kelemahannya adalah objek ini tidak bisa digunakan lagi (hanya dapat digunakan satu kali saja) karena setelah digunakan akan langsung dihapus.

Contoh :

```
public void getJenis()  
{  
    System.out.println(new  
String("Sepeda"));  
}
```

### 1.3. Keyword "this"

Perhatikan keyword "*this*" di bawah ini (lihat pada class SepedaMotor):

```
public void setMerk(String merk) {  
    this.merk = merk;  
}
```

Untuk membedakan variabel merk pada parameter dan variabel merk pada atribut dari class SepedaMotor, digunakanlah keyword "*this*". Sehingga untuk menggunakan atribut merk pada class SepedaMotor, digunakan :

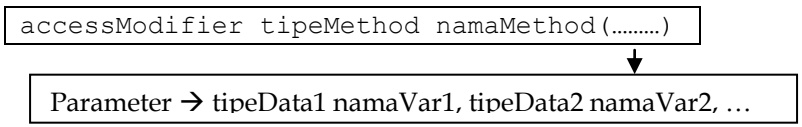
```
this.merk
```

### 1.4. Atribut & Method

Pada contoh class `Latihan1a`, atribut dari class tersebut adalah merk, tipe, tangki, dan harga, yang ini berarti bahwa setiap objek dari class sepeda motor (atau dengan kata lain setiap sepeda motor yang ada di dunia nyata) pasti memiliki merk, tipe motornya, kapasitas maksimal dari tangki BBM, dan harga jual sepeda motor tersebut di pasaran. Sedangkan contoh method dari class `Latihan1a` adalah `setMerk(String merk)` dan `getMerk()` yang berfungsi untuk mengambil nilai atribut merk.

Secara umum method (ada juga yang menyebutnya fungsi) itu ada 2 macam, yaitu method yang mengembalikan nilai dan method yang tidak mengembalikan nilai. Contoh method yang mengembalikan nilai adalah method `getMerk()` dimana hasil dari method ini adalah mengembalikan nilai string dari atribut merk. Sedangkan contoh method yang tidak mengembalikan nilai adalah method `setMerk(String merk)`, yaitu dengan ciri tipe data dari method tersebut adalah `void`.

Sintaks untuk membuat method :



Contoh :

```
public long getHarga()
```

### 1.5. Access Modifier

Yang dimaksud dengan *access modifier* adalah pengaturan hak akses class maupun method. Ada 4 akses yang tersedia, yaitu `default`, `public`, `protected`, `private`. Untuk lebih jelasnya, silahkan lihat kedua table berikut ini :



No	Modifier	Pada class dan Interface	Pada Method & Variabel
1	<i>default</i> (tak ada modifier)	Dapat diakses oleh yang sepaket.	Diwarisi oleh subkelas dipaket yang sama, dapat diakses oleh method-method yang sepaket.

2	Public	Dapat diakses dimanapun	Diwarisi oleh subkelasnya, dapat diakses dimanapun.
3	Protected	Tidak bisa diterapkan	Diwarisi oleh subkelasnya, dapat diakses oleh method-method yang sepaket.
4	Private	Tidak bisa diterapkan	Tidak dapat diakses dimanapun kecuali oleh method-method yang ada dalam kelas itu sendiri.

Aksesabilitas	public	private	protected	default
Dari kelas yang sama	Ya	Ya	Ya	Ya
Dari sembarang kelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari sembarang kelas di luar paket	Ya	Tidak	Tidak	Tidak
Dari subkelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari subkelas di luar paket	Ya	Tidak	Ya	Tidak

### 1.6. Static Variable vs Instant Variable

Static variabel adalah variabel yang didefinisikan/dideklarasikan dengan menggunakan keyword “*static*”, contoh :

```
static int jumlah;
```

Variabel-variabel yang dideklarasikan dengan tidak menggunakan keyword “*static*”, maka variabel tersebut disebut dengan instant variabel (atau variabel instant).

- Jika sebuah variable merupakan variable instant, maka masing-masing objek dari class tersebut akan memiliki variable yang sama dengan variable instant tersebut, perubahan nilai yang terjadi pada variable instant di satu objek tidak akan berpengaruh pada variable instant di objek yang berbeda.
- Jika sebuah variable merupakan variable static (pada suatu class), maka variabel static tersebut adalah variabel yang sama di semua objek dari class tersebut. Sehingga perubahan nilai

pada variabel static tersebut di suatu objek akan berpengaruh juga terhadap objek yang lainnya.

- Nilai suatu variabel static akan selalu sama untuk semua *instant of class* (atau objek) dari sebuah class.
- Sebuah variable dideklarasikan static apabila variable tersebut bersifat global bagi semua objek (*instant of class*) dari suatu class. Contohnya adalah variable yang menyimpan nilai jumlah objek yang telah dibuat.

### 1.7. Input Data

Untuk menginputkan data dari keyboard ada 2 cara, yaitu :

- Input dari mode console, yaitu dengan memanfaatkan class **BufferedReader** dan **InputStreamReader**.

Untuk bisa mengakses class **BufferedReader** maka perlu mengimpor dari package **java.io.\*** dan menambahkan statemen **throws IOException** pada header method main.

Contoh :

```
import java.io.*;

class CobaInput1
{
    public static void main (String []
args) throws IOException
    {
        BufferedReader      br      =      new
BufferedReader (
        new InputStreamReader(System.in));
        String nama, kota;
        System.out.print("Nama Anda : ");
        nama = br.readLine();
        System.out.print("Kota Asal : ");
        kota = br.readLine();
        System.out.println("Selamat      Datang
"+      nama +" dari "+ kota);
    }
}
```

- Inputan dengan memanfaatkan class **JOptionPane**.

Untuk bisa menggunakan class `JOptionPane`, maka perlu mengimpor dari package `javax.swing.*` dan gunakan method `showInputDialog()` yang terdapat pada class `JOptionPane`.

Contoh :

```
import javax.swing.*;

class CobaInput2
{
    public static void main (String []
args)
    {
        String nama, kota;
        nama =
JOptionPane.showInputDialog("Nama
Anda :");
        kota =
JOptionPane.showInputDialog("Kota
Asal :");
        System.out.println("Selamat Datang
"+ nama +" dari "+ kota);
        System.exit(0);
    }
}
```

Catatan :

Semua data yang diinputkan dianggap sebagai suatu nilai `String` meskipun data tersebut hanya terdiri atas angka saja. Untuk menampung data yang diinputkan ke dalam variabel bertipe numerik (misal : `int`, `long`, `double`), maka data harus terlebih dahulu diubah ke tipe data numerik.

Contoh :

```
String sAngka;
int a = Integer.parseInt(sAngka);
long b = Long.parseLong(sAngka);
double c = Double.parseDouble(sAngka);
```

### **1.8. Java Documentation**

Merupakan dokumentasi dari kelas-kelas yang sudah disediakan oleh java yang berupa API (*Application Programming Interface*), dan siap untuk digunakan. Dokumentasi ini hanya dapat anda lihat dari komputer lab pada <http://web-server/javadocs/api>



### Latihan

- a. Buatlah sebuah class Pelajar dengan ketentuan sebagai berikut :
- Memiliki atribut : nip, nama, nilaiUjian1, nilaiUjian2, nilaiTugas.
  - Memiliki method **nilaiAkhir** untuk menghitung nilai akhir dari pelajar tersebut dimana rumusnya adalah :  
$$NA = (\text{nilaiUjian1} * 35\%) + (\text{nilaiUjian2} * 40\%) + (\text{nilaiTugas} * 25\%)$$
  - Ada method **isLulus** yang digunakan untuk mengecek apakah seorang siswa lulus atau tidak, dimana dinyatakan lulus bila nilai akhirnya sama dengan 60 ke atas. Dan ada pula method **status** yang digunakan untuk menampilkan status keterangan lulus atau tidaknya pelajar tersebut.
  - Cobalah class Pelajar yang anda buat dengan menjalankan class InputPelajar1 berikut ini :

```
class InputPelajar1
{
    public static void main(String []a)
    {
        Pelajar p = new Pelajar();
        p.setNip("050123");
        p.setNama("Budiono");
        p.setNilaiUjian1(55.9);
        p.setNilaiUjian2(65.8);
        p.setNilaiTugas(72);
        System.out.println("Data Pelajar :
");
        System.out.println("NIP :
"+p.getNip());
        System.out.println("Nama :
"+p.getNama());
        System.out.println("N.Ujian 1 : "+
            p.getNilaiUji1());
        System.out.println("N.Ujian 2 : "+
            p.getNilaiUji2());
        System.out.println("N.Tugas : "+
            p.getNilaiTugas());
        System.out.println("N.Rata2 ujian :
"+
            p.nilaiRata2());
        System.out.println("Nilai Akhir :
"+
            p.nilaiAkhir());
        p.status();
    }
}
```

```

    }
}

```

Jika benar akan menghasilkan output sebagai berikut :

```

Data Pelajar :
NIP   : 050123
Nama  : Budiono
N.Ujian 1 : 55.9
N.Ujian 2 : 65.8
N.Tugas   : 72.0
N.Rata2 ujian : 60.85
Nilai Akhir   : 63.89
Status    : Lulus

```

Dan uji kembali dengan class InputPelajar2 :

```

class InputPelajar
{
    public static void main(String []a)
    {
        Pelajar p = new Pelajar();
        p.setData("Dian S.", "050122");
        p.setNilai(65, 53.7, 62.5);
        System.out.println("Data Pelajar :
");
        System.out.println("NIP
"+p.getNip());
        System.out.println("Nama
"+p.getNama());
        System.out.println("N.Ujian 1 : "+
            p.getNilaiUjil());
        System.out.println("N.Ujian 2 : "+
            p.getNilaiUji2());
        System.out.println("N.Tugas : "+
            p.getNilaiTugas());
        System.out.println("N.Rata2 ujian :
"+
            p.nilaiRata2());
        System.out.println("Nilai Akhir :
"+
            p.nilaiAkhir());
        System.out.println("Status : "+
            p.isLulus());
    }
}

```

Pastikan hasil yang didapat ketika diuji dengan class `InputPelajar2` adalah sebagai berikut :

```
Data Pelajar :  
NIP   : 050122  
Nama  : Dian S.  
N.Ujian 1 : 65.0  
N.Ujian 2 : 53.7  
N.Tugas   : 62.5  
N.Rata2 ujian : 59.35  
Nilai Akhir   : 59.86  
Status      : Tidak Lulus
```

- Nilai rata-rata ujian adalah nilai ujian 1 ditambah dengan nilai ujian 2 lalu dibagi dua.
- Nilai rata-rata ujian dan nilai akhir hanya ditampilkan dengan format 2 digit di belakang koma.
- Selain dari method-method yang telah ditentukan di atas, tambahkan sendiri method-method apa saja yang perlu dan harus ada menurut analisa yang anda lakukan.

Pada soal ini anda akan belajar tentang menentukan tipe dari suatu atribut, menentukan hak akses modifier yang tepat kepada atribut dan method, menentukan return tipe method (tipe data method), menentukan parameter-parameter method, serta dapat membuat dan membedakan antara method yang memiliki return type dan yang tidak memiliki return type.

- b. Jika telah selesai mengerjakan dengan benar latihan di atas, maka buatlah sebuah class `InputPelajar3` dimana data-data pelajar diinputkan oleh user.

## MODUL 2

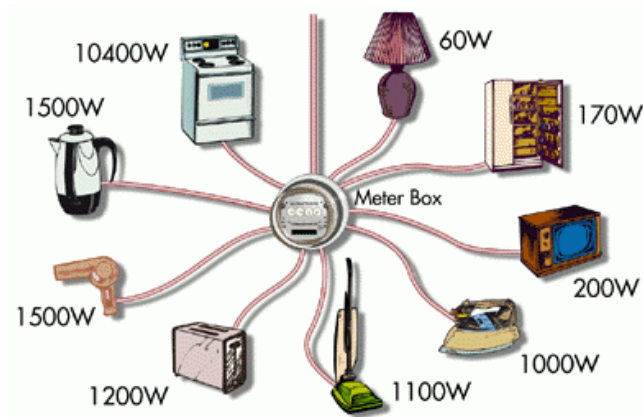
### CONSTRUCTOR & INHERITANCE

*Jangan Menunda melakukan sesuatu di hari esok apa yang dapat dikerjakan hari ini.*

*Sebab jika anda menikmati apa yang Anda lakukan hari ini,*

*Anda akan dapat menikmatinya lagi di hari esok.*

*~ James A. Michener ~*



#### **Tujuan :**

Dapat membuat objek baru dari pengembangan objek yang telah ada

#### **Materi :**

Constructor (single & multiple constructor), Inheritance (superclass, subclass)

#### **Referensi :**

- Fikri, Rijalul. *Pemrograman Java*. Penerbit Andi, 2005. Hal 97 - 102
- Hermawan, Benny. 2004. *Menguasai Java 2 dan Object Oriented Programming*. Penerbit Andi, 2004.

## 2.1. Apa itu Constructor ?

Constructor adalah method yang secara otomatis dipanggil/dijalankan pada saat sebuah class diinstansi. Jika dalam sebuah class tidak terdapat constructor maka secara otomatis Java akan membuat sebuah default constructor. Nama constructor harus sama dengan nama class dan tidak boleh memiliki tipe return value. Sama halnya dengan method, constructor dapat memiliki satu atau banyak parameter maupun tanpa parameter.

Constructor biasanya digunakan untuk memberi nilai awal dari atribut-atribut class tersebut.

Contoh :

```
class Login
{
    private String nama, paswd;
    Login() {
        this.nama = "";
        this.paswd = "";
    }
}
```

## 2.2. Multiple Constructor

Java tidak membatasi jumlah constructor dalam satu class, sehingga memungkinkan sebuah class memiliki lebih dari satu constructor. Multiple constructor adalah adanya lebih dari satu constructor untuk sebuah class. Yang membedakan antara satu constructor dengan constructor lainnya adalah pada parameternya (nama constructornya sama).

Contoh :

```
class Login
{
    private String nama, paswd;

    Login() {
        this.nama = "";
        this.paswd = "";
    }
}
```

```

Login(String nama, String paswd){
    this.nama = nama;
    this.paswd = paswd;
}
public void setNama(String nama){
    this.nama = nama
}
public void setPaswd(String paswd){
    this.paswd = paswd;
}
}
    
```

### 2.3. Function Overloading

Overloading adalah diperbolehkannya dalam sebuah class memiliki lebih dari satu function yang serupa (nama function-nya sama) tetapi deklarasi-deklarasi parameternya berbeda.

Contoh :

```

class Latihan2a
{
    private Login login1;

    public static void main (String
[]args){
        setLogin("Budi","BuDl");
        Login login2 = new Login();
        setLogin(login2);
    }

    public static setLogin(String n, String
p){
        login1 = new Login(n,p);
    }
    public static setLogin(Login log){
        log.setNama ("Wiwin");
        log.setPaswd("w1w1n");
    }
}
    
```

Perhatikan class Latihan2a!

Pada class tersebut terdapat 2 method `setLogin` yang parameternya berbeda.

## 2.4. Mengenal Inheritance

Inheritance merupakan proses pewarisan data dan method dari suatu class yang telah ada kepada suatu class baru. Class yang mewariskan disebut dengan **superclass / parent class / base class**, sedangkan class yang mewarisi (class yang baru) disebut dengan **subclass / child class / derived class**. Subclass tidak dapat mewarisi anggota private dari superclass-nya.

Dengan inheritance, class yang baru (subclass) akan mirip dengan class yang lama (superclass) namun memiliki karakteristik yang baru. Dalam Java, subclass hanya bisa memiliki satu superclass (single inheritance) sedangkan superclass bisa memiliki satu subclass atau lebih.

Untuk menerapkan inheritance, gunakan statement "*extends*".

```
namaSubclass extends namaSuperclass
{
    ..... // definisi class
}
```

Keyword "*super*" digunakan oleh subclass untuk memanggil constructor atau method yang ada pada superclass-nya.

Contoh untuk memanggil constructor milik superclass-nya :

```
super()
super(parameter)
```

Contoh untuk memanggil method milik superclass-nya :

```
super.namaMethod(parameter)
```

Contoh inheritance :

```
class Orang
{
    private String nama;
    private double tinggi;
    private double berat;

    public Orang (String nama, double
tinggi, double berat) {
```

```

        this.nama = nama;
        this.tinggi = tinggi;
        this.berat = berat;
    }
    public String toString()
    {
        return ("Nama : "+nama+"\nTinggi :
"+      tinggi + "\nBerat :      "+berat);
    }
}

class Pelajar extends Orang
{
    private String nim;
    private String asalSekolah;
    private double nilai; // range: 0-30

    public Pelajar (String nama, double
tinggi,      double berat, String nim,
String sekolah,      double nilai)
    {
        super(nama,tinggi,berat);
        this.nim = nim;
        asalSekolah = sekolah;
        this.nilai = nilai;
    }
    public String toString()
    {
        return (super.toString()+"\nNIM :
"+nim+      "\nSekolah :
"+asalSekolah+"\nNilai : "
+nilai);
    }
}

class Latihan2b
{
    public static void main(String[] args)
    {
        Pelajar      siswa      =      new
Pelajar("Musa",168,
        62,"050107","SMU Pancasila",27.8);

        System.out.println(siswa.toString());
    }
}

```



```
}  
}
```

Silahkan dicoba untuk melihat hasilnya !

**Latihan**

Sebuah rental VCD memiliki desain class master untuk data Vcd Film dan CD Musik sebagai berikut :

- Class CdFilm dengan atribut :
  - judul : judul film
  - pemain : nama-nama pemain di dilm tersebut
  - sutradara : nama sutradara film tersebut
  - publiser : yang memproduksi film tersebut
  - kategori : SU = Semua Umur, D = Dewasa, R = Remaja, A = Anak
  - stok : jumlah stok vcd film tersebut
- Class CdMusik dengan atribut :
  - judul : judul album musik
  - penyanyi : nama penyanyi di album tersebut
  - produser : nama produser album tersebut
  - publiser : studio rekaman yang memproduksi cd musik tersebut
  - top hits : lagu yang diandalkan pada album tersebut
  - kategori : C = Classic, J = Jazz, P = Pop, R = Rock, O = Other
  - stok : jumlah stok cd musik tersebut

Dari data kedua class diatas, desainlah class-class tersebut dengan konsep inheritance. Cari hubungan antar kedua class tersebut lalu tentukan superclass-nya dan subclass-nya. Setelah itu implementasikan class-class yang telah anda desain dan buatlah program sederhana yang memiliki fasilitas entri data vcd film, entri data cd musik serta melihat daftar vcd film dan cd musik yang telah dientrikan.



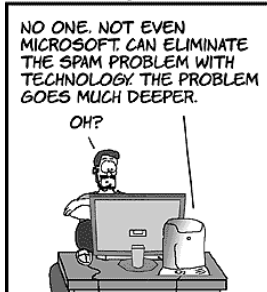
# MODUL 3

## POLYMORPHISM

*Jangan pernah takut untuk mengakui  
kesalahan Anda.*

*Ini berarti bahwa hari ini Anda lebih bijaksana  
daripada kemarin.*

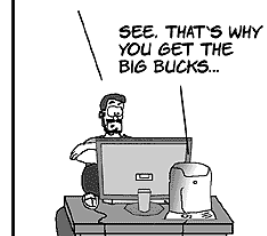
USER FRIENDLY by J.D. "Illiad" Frazer



THE PROBLEM LIES IN THE FACT THAT SPAMMING REMAINS PROFITABLE. AND THAT'S BECAUSE PEOPLE STILL CLICK ON SPAM LINKS.



SO THE SOLUTION IS TO... UH...ELIMINATE PEOPLE?



### Tujuan :

Memahami konsep polymorphism dan dapat menerapkannya dalam program

### Materi :

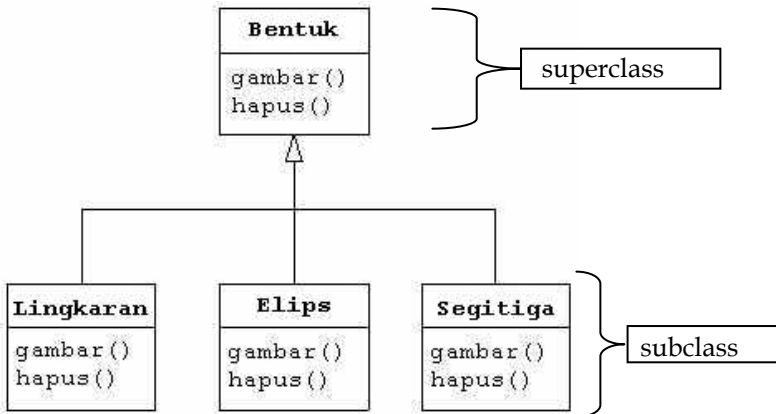
Polymorphism  
Overriding atribut & method

### Referensi :

- Fikri, Rijalul. *Pemrograman Java*. Penerbit Andi, 2005. Hal 103 - 108
- Harianto, Bambang. *Esensi-esensi Bahasa Pemrograman Java*. Informatika Bandung, 2003.
- Hermawan, Benny. *Menguasai Java 2 dan Object Oriented Programming*. Penerbit Andi, 2004.

### 3.1. Berkenalan dengan Polymorphism

Polymorphism mempunyai makna sesuatu yang memiliki banyak bentuk, yaitu memiliki nama sama, tetapi memiliki kelakuan (behaviour) yang berbeda.



Perhatikan gambar diagram di atas !

Class **Bentuk** yang merupakan class induk (*superclass*) dari class **Lingkaran**, **Elips** dan **Segitiga** mempunyai method **gambar()** dan **hapus()**. Class-class anak (*subclass*) juga mempunyai method **gambar()** dan **hapus()**. Meskipun keempat class tersebut mempunyai nama method yang sama, tetapi isi (source code/yang dilakukan/output) dari masing-masing method tersebut berbeda.

Jika kita menginginkan sebuah objek yang dapat memanggil setiap method (yaitu method gambar & hapus) yang ada pada setiap class (pada superclass maupun subclass), maka gunakanlah teknik **Polymorphism**. Polymorphism hanya berlaku pada method dan tidak berlaku untuk atribut.

Untuk mendapatkan operasi polymorphism dari suatu method, maka method tersebut haruslah merupakan method yang ada di class induk (lihat diagram diatas bahwa method gambar() dan hapus(), selain terdapat di class-class turunan class Bentuk, juga terdapat di class Bentuk).

Contoh implementasi polymorphism :

```
class Bentuk
{
    public void gambar() {
        System.out.println("Menggambar");+
    }
    public void hapus() {
        System.out.println("Menghapus
Gambar");
    }
}

class Lingkaran extends Bentuk
{
    public void gambar(){
        System.out.println("Gambar
Lingkaran");
    }
    public void hapus() {
        System.out.println("Hapus
Lingkaran");
    }
}

class Elips extends Bentuk
{
    public void gambar() {
        System.out.println("Gambar Elips");
    }
    public void hapus() {
        System.out.println("Hapus Elips");
    }
}

class Segitiga extends Bentuk
{
    public void gambar() {
        System.out.println("Gambar
Segitiga");
    }
    public void hapus() {
        System.out.println("Hapus
Segitiga");
    }
}
```

Berikut ini adalah class yang mengimplimentasikan teknik polymorphism :

```
class RandomBentuk
{
    private Random rand = new Random();
    public Bentuk next() {
        switch(rand.nextInt(3))
        {
            default:
            case 0: return new Lingkaran();
            case 1: return new Elips();
            case 2: return new Segitiga();
        }
    }
}

class Latihan3a
{
    private static RandomBentuk gen = new
        RandomBentuk();
    public static void main(String[] args)
    {
        Bentuk[] bangun = new Bentuk[3];
        for(int i = 0; i < bangun.length;
i++)
            bangun[i] = gen.next();
        for(int i = 0; i < bangun.length;
i++)
            bangun [i].gambar();
        for(int i = 0; i < bangun.length;
i++)
            bangun [i].hapus();
    }
}
```

Pada class *Latihan3a* terdapat variabel/objek *bangun* yang bertipe class *Bentuk*. Maka dapat kita katakan bahwa variabel *bangun* dapat berperan sebagai *Lingkaran*, *Elips*, atau *Segitiga*. Hal ini didasarkan bahwa pada kenyataannya setiap objek dari class Induk (*superclass*) dapat berperan sebagai class-class turunannya sebagaimana sepeda motor adalah kendaraan, pelajar dan mahasiswa adalah orang/manusia.

Perhatikan bahwa polymorphism tidak sama dengan overloading !!!

### **3.2. Method Overriding**

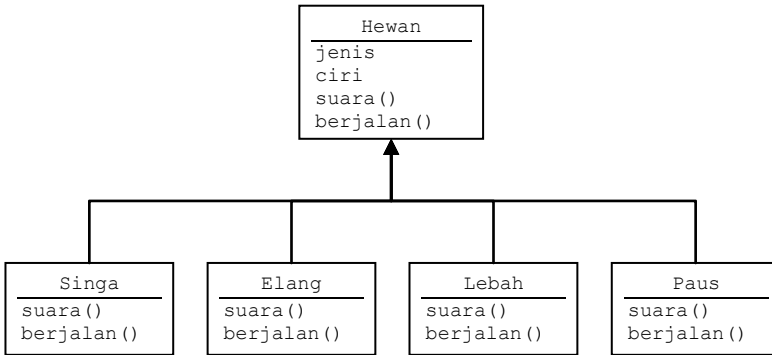
Overriding method adalah kemampuan dari subclass untuk memodifikasi method dari superclass-nya, yaitu dengan cara menumpuk (mendefinisikan kembali) method superclass-nya.

Contoh overriding method dapat dilihat pada class-class turunan dari class `Bentuk` yang mendefinisikan kembali method `gambar()` dan method `hapus()` dari class induknya.



**Latihan**

1 Terdapat class-class hewan sebagai berikut :



Penjelasan dari masing-masing atribut dan method sebagai berikut :

- jenis : apakah termasuk hewan mamalia, serangga atau burung
- ciri : ciri dari hewan tersebut
- suara() : suara hewan tersebut
- berjalan() : cara hewan tersebut berjalan/bergerak

Buatlah class-class yang mengimplimentasikan gambar diatas !

- 2 Buatlah class TestHewan yang akan mengimplementasikan class-class di atas dengan ketentuan sebagai berikut :
  - Saat pertama kali dijalankan akan ditampilkan menu pilihan, yaitu nama-nama hewan tersebut serta menu untuk keluar dari program.
  - Setelah user memilih hewan yang diinginkan, maka tampilkan suara dan cara berjalan/bergerak dari hewan yang dipilih.
  - Jika user memilih Keluar, maka program selesai.
- 3 Setelah program anda berjalan dengan benar, cobalah untuk menambahi cara masing-masing hewan tersebut bernafas, yaitu apakah menggunakan paru-paru, insang, atau yang lainnya. Sehingga pada saat class TestHewan dijalankan akan menampilkan suara, cara berjalan/bergerak dan cara bernafas dari hewan yang dipilih. Buat method ini dengan teknik

polymorphism. Buatlah kesimpulan sendiri tentang  
polymorphism.



# MODUL 4

## ABSTRACT CLASS

*Merupakan hal yang benar untuk merasa  
puas dengan apa yang Anda punya,  
Namun jangan pernah berpuas diri.  
~ John Maxwell ~*



**Tujuan :**  
Dapat membuat dan menggunakan abstract class

**Materi :**  
Deklarasi & implementasi abstract class  
Keyword "*final*"

**Referensi :**

- Fikri, Rijalul. *Pemrograman Java*. Penerbit Andi, 2005. Hal 108 – 113
- Morelli, Ralph and Walde, Ralph. *Java, Java, Java™: Object-Oriented Problem Solving*. Prentice Hall, 2005. Chapter 8.3

#### 4.1. Lebih Jauh dengan Abstract Class

Abstract class adalah class yang terletak pada posisi tertinggi dari hierarki class dan digunakan sebagai acuan (superclass) bagi penurunan class-class lainnya. Biasanya class ini hanya berisi variable-variabel umum (atribut umum) dan header-header method saja tanpa body methodnya atau bisa juga sebagian method ada body methodnya dan sebagian method yang lain tidak memiliki body method. Oleh karena itu, class-class turunannya yang mendefinisikan secara detil body method-method tersebut.

Cara untuk membuat sebuah class abstrak adalah :

```
akses_modifier      abstract      class
namaClassAbstrak
{
    ..... // definisi class
}
```

Contoh :

```
public abstract class Hewan{
    protected String jenis;

    public Hewan() { }
    public String toString() {
        return "Jenisku adalah "+jenis +
            " dan suaraku "+ suara();
    }
    public abstract String suara();
}
```

Java memiliki aturan-aturan dalam penggunaan method abstrak dan class abstrak sebagai berikut :

- Class yang di dalamnya terdapat abstract method harus dideklarasikan sebagai abstract class.
- Abstract class tidak dapat diinstansi, tetapi harus di turunkan.
- Abstract class tidak dapat diinstansi (menjadi objek dari class abstract), tetapi kita dapat mendeklarasikan suatu variable yang bertipe abstract class dan membuat instansi dari variabel tersebut yang bertipe class turunan dari abstract class tersebut (*teknik polymorphism*).
- Sebuah class dapat dideklarasikan sebagai abstract class meskipun class tersebut tidak memiliki abstract method.

- Abstract method tidak boleh mempunyai body method dan demikian juga sebaliknya bahwa method yang tidak ditulis body methodnya maka harus dideklarasikan sebagai abstract method.

## 4.2. Keyword “final”

Keyword ini digunakan untuk mencegah suatu class diturunkan atau suatu method di overriding.

Sintaks penggunaan keyword “final” pada class :

```
akses_modifier final namaClass
```

Sintaks penggunaan keyword “final” pada method :

```
akses_modifier      final      tipeMethod
namaMethod()
{
    ..... // definisi method
}
```

Contoh :

```
public final class Login
{
    private String nama, paswd;

    Login() {
        this.nama = "labkom";
        this.paswd = "l4bk0m";
    }
    public final boolean matching(String n,
String p) {
        if (n.equals(nama) &&
p.equals(paswd))
            return true;
        else
            return false;
    }
}
```

Catatan :

Class atau method yang diberi atribut *final* tidak boleh berupa class abstrak atau method abstrak. Hal ini disebabkan karena class abstrak

harus diturunkan lagi sedangkan method abstrak harus didefinisikan ulang (harus di *override*).

### 4.3. Abstract Method

Abstract method adalah method yang dideklarasikan tanpa body method. (diikuti dengan titik koma / semicolon).

Contoh :

```
abstract void move(double x, double y);
```

## Latihan

CV. Maju Mundur memiliki 40 orang pegawai, dimana ke-40 pegawainya tersebut terbagi menjadi 2 status kepegawaian, yaitu sebagian pegawai tetap dan sebagian yang lain (sisanya) adalah pegawai kontrak. Secara umum perbedaan antara keduanya adalah pegawai tetap selain mendapatkan gaji juga mendapatkan tunjangan yang besarnya 200 ribu, sedangkan pegawai kontrak hanya mendapatkan gaji saja.

Dari kasus diatas, dapat digambarkan class-class pegawai sebagai berikut :

- Class PegawaiKontrak, dengan atribut :
  - NoPeg : no pegawai kontrak (*diinputkan*)
  - Nama : nama pegawai (*diinputkan*)
  - MasaKontrak : lama kontrak pegawai (*diinputkan*)
  - Kehadiran : jumlah hari masuk dalam 1 bulan (*diinputkan*)
  - UangMakan : besarnya 6 ribu dikali jumlah kehadiran (*tidak diinputkan*)
  - Gaji : besarnya gaji pokok yang diterima tiap bulan (*diinputkan*)
  
- Class PegawaiTetap, dengan atribut :
  - NoPeg : no pegawai kontrak (*diinputkan*)
  - Nama : nama pegawai (*diinputkan*)
  - Kehadiran : jumlah hari masuk dalam 1 bulan (*diinputkan*)
  - Tunjangan : besarnya 200 ribu (*tidak diinputkan*)
  - UangMakan : besarnya 6 ribu dikali jumlah kehadiran (*tidak diinputkan*)
  - Gaji : besarnya gaji pokok yang diterima tiap bulan (*diinputkan*)

Method-method yang harus ada :

- hitungGaji() : untuk menghitung gaji bersih pegawai, dimana untuk pegawai kontrak = Uang Makan + Gaji, pegawai tetap = Tunjangan + Uang Makan + Gaji
- lihatData() : untuk menampilkan data-data pegawai secara lengkap beserta total gaji yang diterima (gaji bersih)



Dari kedua class diatas, analisa dan desainlah superclass yang tepat untuk kasus tersebut.

Setelah itu buatlah class utama yaitu class PegawaiMajuMundur (yang menggunakan class-class diatas) yang memiliki menu utama sebagai berikut:

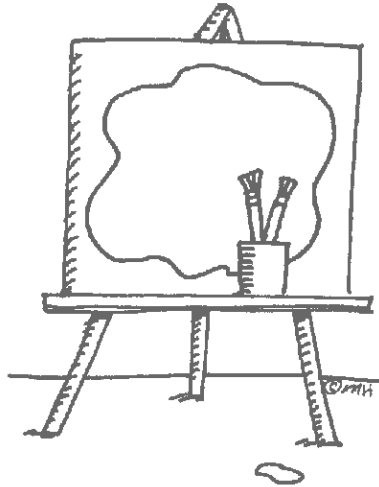
```
MENU UTAMA
1. Input Data Pegawai
2. Lihat Data Pegawai
3. Keluar
Pilihan Anda [1/2/3] ? ...
```

Tentukan pula modifier yang tepat untuk semua class diatas (termasuk superclass-nya, mana yg final class dan mana yang abstract class). Anda dapat menggunakan teknik polymorphism dalam menyelesaikan permasalahan ini.

# MODUL 5

## INTERFACE

*Menguasai kesukaran adalah kemenangan  
para pengambil peluang  
~ Winston Churchill ~*



**Tujuan :**  
Dapat membuat serta menggunakan interface dalam program

**Materi :**  
Deskripsi, deklarasi & implementasi interface  
Single & multiple interface

- Referensi :**
- Fikri, Rijalul. *Pemrograman Java*. Penerbit Andi, 2005. Hal 113 - 117
  - Harianto, Bambang. *Esensi-esensi Bahasa Pemrograman Java*. Informatika Bandung, 2003.
  - Hermawan, Benny. *Menguasai Java 2 dan Object Oriented Programming*. Penerbit Andi, 2004.

### 5.1. Deskripsi interface

*Interface* adalah sekumpulan konstanta dan atau deklarasi method tanpa menyertakan/menuliskan body methodnya. Interface biasa digunakan untuk mendeklarasikan koleksi method dan konstanta yang dapat digunakan oleh satu atau lebih class.

### 5.2. Deklarasi interface

Untuk mendeklarasikan sebuah interface gunakan sintaks :

```
interface namaInterface
{
    ..... //deklarasi konstanta dan
    method
    .....
}
```

Berikut ini adalah contoh membuat interface Speedometer :

```
public interface Speedometer
{
    public void tambahKecepatan();
    public void kurangiKecepatan();
}
```

### 5.3. Implementasi interface

Cara menggunakan suatu interface adalah dengan mengimplementasikan interface tersebut pada class yang menggunakannya. Selain itu anda juga harus mendefinisikan secara detail method-method yang ada pada interface tersebut.

Contoh :

```
class Mobil implements Speedometer
{
    public void tambahKecepatan()
    {
        System.out.println("injak kopling
        lalu, pindah ke gear yang
        lebih tinggi & Gas
        Mobilnya");
    }
    public void kurangiKecepatan()
```

```
{
    System.out.println("Rem Mobilnya &
pindah          gear          yang          lebih
rendah");
}
}

class Motor implements Speedometer
{
    public void tambahKecepatan()
    {
        System.out.println("pindah ke gear
yang          lebih          tinggi          &          Gas
Motornya");
    }
    public void kurangiKecepatan()
    {
        System.out.println("Rem          Motornya
dengan          rem belakang + depan, lalu
pindah gear          yang          lebih
rendah");
    }
}

class TestKendaraan
{
    public static void main(String[] args)
    {
        Mobil mobil = new Mobil();
        Motor motor = new Motor();
        System.out.print("Cara ngebut pake
motor: ");
        motor.tambahKecepatan();
        System.out.print("Cara berhentinya:
");
        motor.kurangiKecepatan();

        System.out.print("Cara balapan pake
mobil: ");
        mobil.tambahKecepatan();
        System.out.print("Klo udah puas: ");
        mobil.kurangiKecepatan();
    }
}
```

### Latihan

Buatlah sebuah interface login dimana pada interface tersebut terdapat 2 method, yaitu *validasi()* dan *cekData()*.

Buatlah sebuah class yang mengimplementasikan interface tersebut yaitu class *DataLogin* dengan penjelasan sebagai berikut :

- Method *validasi()* : bertipe boolean, digunakan untuk memastikan bahwa username dan password tidak boleh kosong (wajib diisi), tentukan sendiri parameternya.
- Method *cekData()* : bertipe boolean, digunakan untuk mengecek username dan password apakah cocok dengan yang terdapat di atribut class *DataLogin* (`private String namaUser1="mhs", private String pass1="mahasiswa", private String namaUser2="mahasiswa", private String pass2="praktikum"`), tentukan sendiri parameternya.

Buatlah class *TestLogin* yang menggunakan class *DataLogin* dengan aturan sebagai berikut :

- Saat program dijalankan, tampilkan menu utama sebagai berikut :

```
MENU UTAMA
-----
Menu Pilihan :
A. LOGIN
B. EXIT
-----
Pilihan Anda :
```

- User dapat memilih pilihan dengan huruf besar maupun huruf kecil.
- Jika LOGIN dipilih, maka user diminta menginputkan nama user dan password. Jika salah, maka tampilkan pesan bahwa user salah menginputkan username atau password atau keduanya dan program kembali ke menu utama. Jika benar, maka tampilkan menu pilihan sebagai berikut :

```
Selamat datang XXXXX
=====
MENU PILIHAN
-----
1. Tes Kendaraan
2. LOG OFF
```

-----  
Pilihan Anda :

- XXXXX = nama user yang sedang login.
- Jika user memilih pilihan 1, maka jalankan class TestKendaraan (lihat dibagian contoh Interface pada Modul 5) dengan memanggil fungsi main-nya. Setelah class TestKendaraan dijalankan, maka tampilkan lagi menu pilihan di atas (Tes Kendaraan & Log off).
- Jika user memilih pilihan 2, maka tampilan kembali ke menu utama (tidak keluar dari program).
- Program akan selesai jika user memilih EXIT.



# MODUL 6

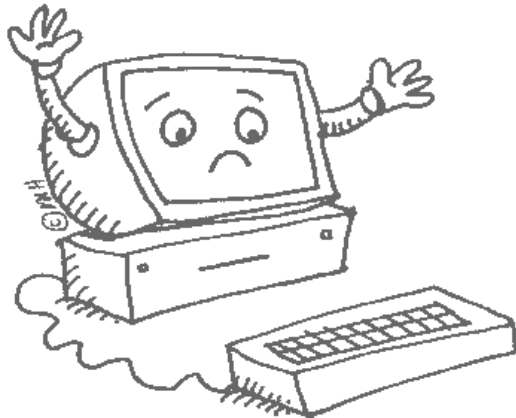
## I/O STREAM

*Saya menghargai semua yang saya peroleh.*

*Sekarang saya hanya mempunyai hari-hari  
yang baik*

*atau hari-hari yang luar biasa.*

*~ Lance Armstrong ~*



### **Tujuan :**

Dapat membuat program yang mengalirkan data melalui stream

### **Materi :**

Input & output stream

File stream

### **Referensi :**

- Rickyanto, Isak. Dasar Pemrograman Berorientasi Objek Dengan Java 2 (JDK 1.4). Penerbit Andi, 2003. Hal 221 - 246
- Morelli, Ralph and Walde, Ralph. *Java, Java, Java™: Object-Oriented Problem Solving*. Prentice Hall, 2005. Chapter 11
- Fikri, Rijalul. *Pemrograman Java*. Penerbit Andi, 2005. Hal 139 - 155



## 6.1. InputStream dan OutputStream

Class `InputStream` dan `OutputStream` merupakan class induk yang digunakan untuk menangani operasi input/output (I/O). Kedua class ini adalah class abstrak sehingga tidak dapat digunakan secara langsung, tetapi dapat digunakan class-class turunan dari kedua class ini.

Class-class turunan (subclass-subclass) dari class `InputStream` antara lain `ByteArrayInputStream`, `FileInputStream`, `FilterInputStream`, `ObjectInputStream`, `PipedInputStream`.

Class-class turunan (subclass-subclass) dari class `OutputStream` antara lain `ByteArrayOutputStream`, `FileOutputStream`, `FilterOutputStream`, `ObjectOutputStream`, `PipedOutputStream`.

## 6.2. FileInputStream dan FileOutputStream

Class `FileInputStream` dan `FileOutputStream` merupakan class yang digunakan untuk operasi baca/tulis file.

Algoritma untuk menulis data ke file :

- Koneksikan output stream ke file
- Tulis data
- Tutup file

Contoh program untuk menulis data ke file binary :

```
import java.io.*;

class TulisFile
{
    public static void main (String []args)
    throws IOException {
        String namaFile = "PBO.txt";
        String namaMhs = "Abu Zaid";
        String jurusan = "Sistem Informasi";
        int angkatan = 2004;
        FileOutputStream outFile = new
            FileOutputStream(namaFile);

        try {
            DataOutputStream outputStream = new
                DataOutputStream(outFile);
```

```

        outputStream.writeUTF(namaMhs);
        outputStream.writeUTF(jurusan);
        outputStream.writeInt(angkatan);
        outputStream.close();
    }
    catch (IOException e) {
        System.out.println("IOERROR: " +
            e.getMessage() + "\n");
    }
}
}

```

Untuk menambah data pada file yang telah ada isinya, maka baris berikut ini (pada class TulisFile):

```

FileOutputStream outFile = new
    FileOutputStream(namaFile);

```

Diubah menjadi :

```

FileOutputStream outFile = new
    FileOutputStream(namaFile, true);

```

Algoritma untuk membaca isi suatu file :

- Koneksikan input stream ke file
- Baca data
- Tutup file

Contoh program untuk membaca data dari file binary :

```

import java.io.*;

class BacaFile
{
    public static void main (String []args)
    throws IOException {
        String namaFile = "PBO.txt";
        String namaMhs, jurusan;
        int angkatan;
        try {
            FileInputStream inFile = new
                FileInputStream(namaFile);
            DataInputStream inStream = new
                DataInputStream(inFile);
            namaMhs = inStream.readUTF();

```

```
        jurusan = inStream.readUTF();
        angkatan = inStream.readInt();
        inStream.close();
        System.out.println("Nama          :
"+namaMhs+
"\nJurusan : "+jurusan+
        "\nAngkatan : "+angkatan);
    }
    catch (FileNotFoundException e) {
        System.out.println("File          "+
namaFile + "          "          Tidak
Ada !\n");
    }
    catch (IOException ex) {
        System.out.println("IOERROR:          "+
        ex.getMessage() + "\n");
    }
}
}
```

## Latihan

Terdapat class-class sebagai berikut :

- Class Manusia, dengan atribut :  
Nama, alamat, jenis kelamin
- Class Mahasiswa, dengan atribut :  
NIM, nama, alamat, jenis kelamin, program studi, jurusan

Berdasarkan kedua class diatas, desainlah hubungan yang tepat antar kedua class tersebut (dgn teknik inheritance, polymorphism, abstract, interface, final class, atau yang lainnya yang paling tepat untuk menggambarkannya). Apabila diperlukan, silahkan tambahkan class-class pembantu.

Buatlah program untuk menyimpan hasil inputan dari user ke dalam file serta menampikannya. Contoh tampilan utama program

```

=====
MENU PILIHAN
-----
1. Input Data Mahasiswa
2. Lihat Data Mahasiswa
3. Keluar Program
-----
Pilihan Anda :
    
```

Contoh tampilan untuk Input Data Mahasiswa

```

Nama :
Alamat :
Jenis Kelamin (L/P) :
    
```

Contoh tampilan Lihat Data Mahasiswa :

```

Nama : Joko
Alamat : Surabaya
Jenis Kelamin : Laki-Laki

Nama : Putri
Alamat : Malang
    
```

Jenis Kelamin : Perempuan

Data yang diinputkan akan ditambahkan (append) pada file, kemudian masing-masing data yang ada dalam file ditampilkan 1 persatu.

# MODUL 7

## GUI

*Don't Judge The Book By It's Cover*



**Tujuan :**  
Dapat membuat program berbasis grafis (GUI)

**Materi :**  
Swing (JFrame, JButton, JLabel, JTextField, JTextArea)  
Layout Management

- Referensi :**
- Fikri, Rijalul. *Pemrograman Java*. Penerbit Andi, 2005. Hal 174 - 195
  - Purnama, Rangsang. *Tuntunan Pemrograman Java, Jilid 2*. Prestasi Pustaka, 2003. Hal 125 - 175
  - <http://java.sun.com/docs/books/tutorial/uiswing/components/index.html>
  - <http://java.sun.com/docs/books/tutorial/uiswing/layout/index.html>

## 7.1. JFrame sebagai Class Utama

Untuk membuat sebuah window (atau disebut juga dengan frame) dapat dilakukan dengan 2 cara. Yang pertama adalah dengan membuat objek dari class JFrame.

Contoh :

```
import javax.swing.*;

class Latihan7a
{
    public static void main(String args[])
    {

        JFrame.setDefaultLookAndFeelDecorated(
            true);
        JFrame windowku = new JFrame("Window
            Utama");
        windowku.setDefaultCloseOperation(
            JFrame.EXIT_ON_CLOSE);
        windowku.setSize(300,150);
        windowku.setLocation(200, 150);
        windowku.setVisible(true);
    }
}
```

Yang kedua adalah dengan membuat sebuah class yang merupakan subclass (class turunan) dari class JFrame lalu membuat instansi (objek) dari subclass tersebut.

Contoh :

```
import javax.swing.*;

class MyWindow extends JFrame
{
    public MyWindow(String title)
    {
        setDefaultCloseOperation(
            JFrame.EXIT_ON_CLOSE);
        setSize(300,150);
        setLocation(200, 150);
        setTitle(title);
        setVisible(true);
    }
}
```

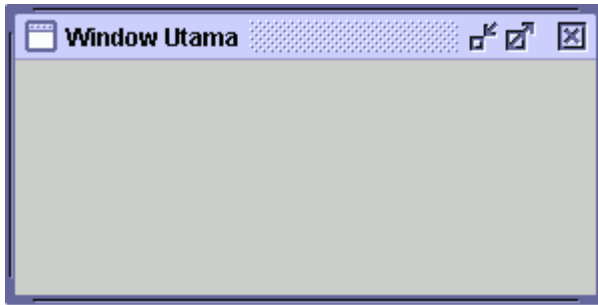
```
    }  
  }  
  
class Latihan7b  
{  
    public static void main(String args[])  
    {  
  
        JFrame.setDefaultLookAndFeelDecorated(  
            true);  
        new MyWindow("Window Utama");  
    }  
}
```

Atau bisa juga :

```
import javax.swing.*;  
  
class Latihan7c extends JFrame  
{  
    public Latihan7c(String title)  
    {  
        setDefaultCloseOperation(  
            JFrame.EXIT_ON_CLOSE);  
        setSize(300,150);  
        setLocation(200, 150);  
        setTitle(title);  
        setVisible(true);  
    }  
    public static void main(String args[])  
    {  
  
        JFrame.setDefaultLookAndFeelDecorated(  
            true);  
        new Latihan7c("Window Utama");  
    }  
}
```

Output yang dihasilkan oleh program diatas adalah





## 7.2. Komponen-komponen GUI

Apabila kita ingin menambahkan komponen (misal Button, Label, TextField, dll) ke dalam frame (bagian area window, selain bagian judul window atau disebut dengan *Content Pane*), maka kita tidak dapat melakukannya secara langsung, tetapi harus melalui perantara class Container. Class Container ini diimport dari package `java.awt.*`;

Contoh :

```
import javax.swing.*;

class Latihan7d extends JFrame
{
    private Container ctn = new
    Container();

    Latihan7d()
    {
        setDefaultCloseOperation(
            JFrame.EXIT_ON_CLOSE);
        setSize(300,150);
        setLocation(200, 150);
        setTitle("Window Utama");
        ctn = getContentPane();
        ctn.add(new JLabel("Ini Komponen
            Label"));
        setVisible(true);
    }
    public static void main(String args[])
    {
```

```

JFrame.setDefaultLookAndFeelDecorated(
    true);
    new Latihan7d();
}
}

```

Cara untuk menambahkan komponen ke dalam container :

```

Container ctn = new Container();
    //deklarasi container

....

ctn = getContentPane();
ctn.add(...);
    //menambah komponen ke container

....

```

Berikut ini beberapa komponen Swing yang perlu untuk diketahui :

- JLabel

Digunakan untuk menampilkan teks pada frame. Sintaks :

```

JLabel label1 = new JLabel("Disini
Tulisannya");

```

Beberapa method yang perlu diketahui yaitu :

getText(), setText(String), setForeground(Color), setFont(Font), setHorizontalAlignment(int), setVerticalTextPosition(int).

- JTextField

Digunakan untuk menerima inputan dari user. Sintaks :

```

JTextField edit1 = new JTextField();
JTextField edit1 = new
JTextField(String);
JTextField edit1 = new JTextField(int
kolom);

```

Beberapa method yang perlu diketahui yaitu :

getText(), setText(String), setForeground(Color), setFont(Font), setEnabled(boolean), selectAll().

- JButton

Digunakan untuk membuat tombol. Sintaks :

```
JButton tombol1 = new JButton("Tombol OK");
```

Beberapa method yang perlu diketahui yaitu :

getText(), setText(String), setForeground(Color), setFont(Font), setEnabled(boolean), setVisible(boolean).

- JTextArea

Digunakan untuk menerima inputan dari user dengan kapasitas text yang jauh lebih besar dari JTextField. Sintaks :

```
JTextArea areal = new JTextArea();  
JTextArea areal = new JTextArea(String);  
JTextArea areal = new JTextArea(int  
jmlBaris, int jmlKolom);  
JTextArea areal = new JTextArea(String,  
int brs, int kol);
```

Beberapa method yang perlu diketahui yaitu :

getText(), setText(String), append(String), getColumn(), setColumns(int), getRows(), setRows(int), setFont(Font), setEditable(boolean), selectAll(), setLineWrap (boolean), insert(String, int).

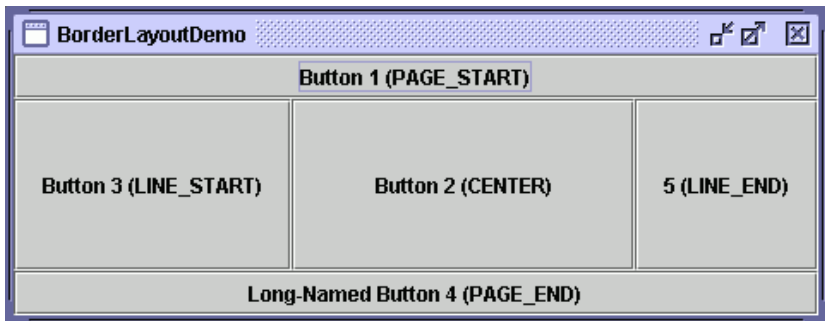
### 7.3. Layout Management

Agar komponen-komponen yang ditempelkan di window utama tertata dengan rapi, maka kita perlu mengatur layout window utama tersebut. Java menyediakan sejumlah class untuk mengatur layout dimana setiap class tersebut memiliki aturan tersendiri dan format layout yang berbeda.

Berikut ini adalah beberapa layout yang sering digunakan :

- BorderLayout

Ini adalah layout default apabila kita tidak menetapkan suatu layout dalam program yang kita buat. Tampilannya sebagai berikut :



Layout ini membagi area tampilan menjadi 5 bagian, yaitu atas, bawah, kiri, kanan, dan tengah. Posisi ini ditentukan pada saat dilakukan penambahan komponen, yaitu saat memanggil method `add()`.

Contoh :

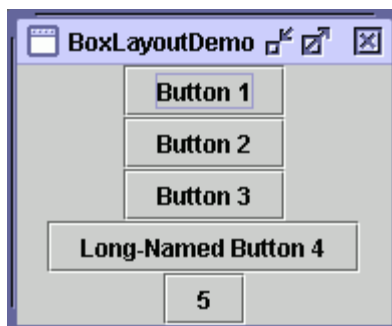
```

container.add(new JButton("Tombol 1"),
    BorderLayout.PAGE_START)
container.add(new JButton("Tombol 2"),
    BorderLayout.CENTER)
container.add(new JButton("Tombol 3"),
    BorderLayout.LINE_START)
container.add(new JButton("Tombol 4"),
    BorderLayout.PAGE_END)
container.add(new JButton("Tombol 5"),
    BorderLayout.LINE_END)

```

- **BoxLayout**

Layout ini mengatur komponen-komponen GUI agar tersusun dalam satu baris atau satu kolom. Tampilan dari `BoxLayout` :



Untuk lebih jelasnya silahkan lihat :

<http://web-server/javatut/uiswing/layout/box.html>

- **FlowLayout**

Layout ini mengatur komponen-komponen GUI agar terletak dalam satu baris saja, tetapi jika sudah mencapai batas lebar area tampilan, maka komponen berikutnya akan diletakkan pada baris berikutnya. Tampilan dari FlowLayout :



Contoh :

```
Container c;  
...  
c.setLayout(new FlowLayout());  
  
c.add(new JButton("Button 1"));  
c.add(new JButton("Button 2"));  
c.add(new JButton("Button 3"));  
c.add(new JButton("Long-Named Button 4"));  
c.add(new JButton("5"));
```

- **GridLayout**

Layout ini mengatur komponen-komponen GUI agar berada dalam posisi grid. Tampilan dari GridLayout :



Ukuran grid, yaitu jumlah baris dan kolomnya, ditentukan pada saat dilakukan setLayout.

```
Container c;  
...  
c.setLayout(new GridLayout(2,3)); //brs=2,  
kol=3
```

Contoh :

```
Container c;  
...  
c.setLayout(new GridLayout(0,2));  
    // brs=tidak ditentukan, kol=2  
c.add(new JButton("Button 1"));  
c.add(new JButton("Button 2"));  
c.add(new JButton("Button 3"));  
c.add(new JButton("Long-Named Button  
4"));  
c.add(new JButton("5"));
```

Cara untuk menetapkan suatu layout dalam program :

```
Container c;  
....  
c.setLayout(tipeLayout);
```

Contoh :

```
c.setLayout(new FlowLayout());  
c.setLayout(new GridLayout(0,2));
```

### Latihan

Untuk melatih kemampuan anda dalam memahami GUI di Java, buatlah form berikut ini dengan menggunakan text editor semisal Jcreator atau Notepad++ atau Edit Plus (bukan GUI Editor/IDE semisal NetBeans). Gunakan Layout Management yang ada selain null layout.



The image shows a Java Swing window titled "Form Biodata Mahasiswa". The window has a light gray background and a blue title bar. The title bar contains the text "Form Biodata Mahasiswa" and standard window control buttons (minimize, maximize, close). The main content area is titled "Biodata Mahasiswa" in a bold, black font. Below the title, there are several input fields and controls:

- NIM:** A single-line text input field.
- Nama:** A single-line text input field.
- Alamat:** A multi-line text input field with a vertical scrollbar on the right.
- Jenis Kelamin:** Two radio buttons labeled "Pria" and "Wanita".
- Program Studi:** A dropdown menu with "S1" selected.
- Jurusan:** A dropdown menu with "Sistem Informasi" selected.

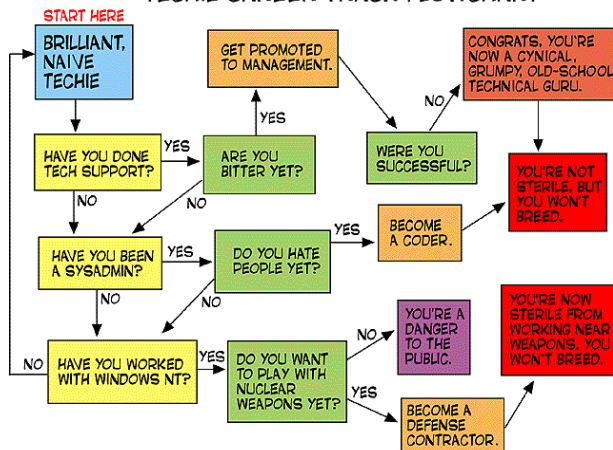
At the bottom of the window, there are three buttons: "Simpan", "Baca Data", and "Keluar".

# MODUL 8

## EVENT LISTENER

*It is the greatest of all mistakes to do nothing because you can do only a little.  
Do what you can and amaze yourself.  
~ Sydney Smith ~*

### TECHIE CAREER TRACK FLOWCHART



COPYRIGHT (C) 2000 ILLIAD

[HTTP://WWW.USERFRIENDLY.ORG/](http://www.userfriendly.org/)

#### Tujuan :

Dapat membuat program yang melibatkan pengawasan terhadap *event*

#### Materi :

ActionListener, KeyListener, MouseListener

#### Referensi :

- Fikri, Rijalul. *Pemrograman Java*. Penerbit Andi, 2005. Hal 223 – 230
- Purnama, Rangsang. *Tuntunan Pemrograman Java, Jilid 2*. Prestasi Pustaka, 2003. Hal 182 – 202
- <http://java.sun.com/docs/books/tutorial/uising/events/index.html>



### 8.1. Event Handler

Ketika user menuliskan karakter atau menekan button atau mengklik tombol mouse maka akan timbul event. Setiap objek komponen GUI dapat digunakan untuk merespon suatu event yang terjadi pada komponen tersebut, sehingga program dapat melakukan suatu proses ketika terjadi suatu event. Hal ini sangat berguna agar program yang kita buat dapat berinteraksi dengan user.

Java menerima event melalui sebuah interface. Agar suatu komponen dapat menangani event yang terjadi, maka kita harus mengimplementasikan interface yang sesuai kepada komponen tersebut.

Beberapa contoh tipe event listener yang ada di Java :

- ActionListener
- WindowListener
- MouseListener
- MouseMotionListener
- KeyListener
- FocusListener
- ItemListener
- dll

Langkah-langkah untuk mengimplementasikan event handler secara umum adalah :

- Pada deklarasi class event handler harus meng-implements sebuah interface listener atau class tersebut meng-extends sebuah class yang meng-implements sebuah interface listener.

Contoh :

```
public class MyClass implements  
ActionListener {
```

- Registrasikan instan dari class event handler sebagai listener pada satu atau beberapa komponen.

Contoh :

```
komponen.addActionListener(instanceOfMyCl  
ass);
```

- Tuliskan method yang mengimplementasikan interface listener tersebut.

Contoh :

```
public void actionPerformed (ActionEvent
e) {    ...// tuliskan kode program yang
dijalankan        ...// saat event
terjadi
}
```

## 8.2. ActionListener

Interface ini akan menangkap event yang disebabkan oleh mouse dan keyboard, antara lain :

- Klik mouse
- Penekanan tombol Enter (pada TextField)
- Memilih MenuItem

Class yang meng-implements ActionListener harus mengimplementasikan/ menuliskan method berikut ini meskipun body methodnya kosong.

```
public void actionPerformed(ActionEvent
e) {
    ... // body method
}
```

Contoh penggunaan interface ActionListener dalam program :

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

public class Latihan8a extends JFrame
implements ActionListener
{
    JButton tombol;
    int n=0;

    public Latihan8a() {
        setDefaultCloseOperation(
            JFrame.EXIT_ON_CLOSE);
        tombol = new JButton("Click Me");
        getContentPane().add(tombol);
        tombol.addActionListener(this);
    }
}
```

```
        setSize(100,100);
        setLocation(400,200);
        setVisible(true);
    }
    public static void main(String []args){
        new Latihan8a();
    }
    public void actionPerformed(ActionEvent
e) {
        n++;
        System.out.println("Klik      "+n+"
kali...");
    }
}
```

### 8.3. KeyListener

Interface ini akan menangkap event yang disebabkan oleh keyboard, yaitu saat tombol keyboard ditekan pada sebuah komponen.

Class yang meng-implements KeyListener harus mengimplementasikan/ menuliskan 3 method berikut ini meskipun body methodnya kosong.

```
public void keyTyped(KeyEvent e) {
    ... // body method
}

public void keyPressed(KeyEvent e) {
    ... // body method
}

public void keyReleased(KeyEvent e) {
    ... // body method
}
```

Contoh penggunaan interface KeyListener dalam program :

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

public class Latihan8b extends JFrame
implements KeyListener
{
```

```
private JTextField edit1;

public Latihan8b() {
    setDefaultCloseOperation(
        JFrame.EXIT_ON_CLOSE);
    setTitle("Demo KeyListener");
    setLocation(400,200);
    edit1 = new JTextField(20);
    edit1.addKeyListener(this);
    Container c = getContentPane();
    c.add(edit1);
    pack();
    setVisible(true);
}

public void keyTyped(KeyEvent e) {
    System.out.println("Key Typed : "+
        e.getKeyChar());
}

public void keyPressed(KeyEvent e) {
    System.out.println("Key Pressed : "+
        e.getKeyChar());
}

public void keyReleased(KeyEvent e) {
    System.out.println("Key Released :
"+
        e.getKeyChar());
}

public static void main(String[] args)
{
    new Latihan8b();
}
}
```

#### 8.4. MouseListener

Interface ini akan menangkap event yang disebabkan oleh mouse, yaitu pada saat pointer mouse berada di atas sebuah komponen dan terjadi penekanan tombol mouse.

Class yang meng-implements MouseListener harus mengimplementasikan/ menuliskan 5 method berikut ini meskipun body methodnya kosong.

```
public void mousePressed(MouseEvent e)
{
```

```
    ... // body method
}

public void mouseReleased(MouseEvent e)
{
    ... // body method
}

public void mouseEntered(MouseEvent e)
{
    ... // body method
}

public void mouseExited(MouseEvent e) {
    ... // body method
}

public void mouseClicked(MouseEvent e)
{
    ... // body method
}
```

Contoh penggunaan MouseListener dalam program :

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

public class Latihan8c extends JFrame
implements MouseListener
{
    private JButton tombol;
    private JLabel label1;
    private JTextField edit1;

    public Latihan8c() {
        setDefaultCloseOperation(
            JFrame.EXIT_ON_CLOSE);
        setTitle("Demo MouseListener");
        setLocation(400,200);
        label1 = new JLabel("Status ==> ");
        edit1 = new JTextField(30);
        edit1.setEnabled(false);
        tombol = new JButton("OK !");
```

```
tombol.addMouseListener(this);

Container c = getContentPane();
c.add(label1, BorderLayout.WEST);
c.add(edit1, BorderLayout.EAST);
c.add(tombol, BorderLayout.SOUTH);
pack();
setVisible(true);
}

public void mousePressed(MouseEvent e)
{
    edit1.setText("Mouse lagi ditekan");
}
public void mouseReleased(MouseEvent e)
{
    edit1.setText("Tombol      Mouse
dilepas");
}
public void mouseEntered(MouseEvent e)
{
    edit1.setText("Mouse entered");
}
public void mouseExited(MouseEvent e) {
    edit1.setText("Mouse exited");
}
public void mouseClicked(MouseEvent e)
{
    edit1.setText("Klik Mouse (" +
        e.getClickCount()+")");
}
public static void main(String[] args)
{
    new Latihan8c();
}
}
```

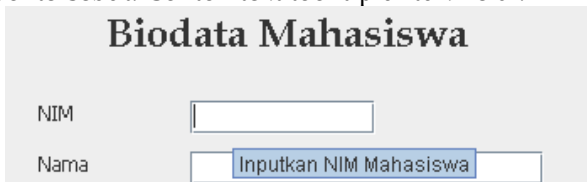
### Latihan

Melanjutkan program yang telah anda buat pada latihan sebelumnya (latihan modul 7), yaitu form dengan tampilan sebagai berikut :

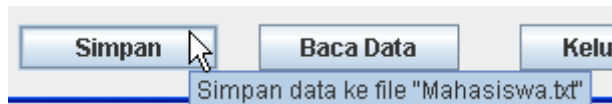


Sempurnakan program tersebut dengan beberapa ketentuan berikut ini :

- Field NIM hanya dapat diisi oleh angka (0-9).
- Pada field NIM jika dier, maka alihkan fokus kursor ke field Nama.
- Pada field Nama jika dier, maka alihkan fokus kursor ke field Alamat.
- Tambahkan Text Tool Tip untuk field NIM, Nama, Alamat, Program Studi, Jurusan, tombol Simpan, tombol Baca Data, dan tombol Keluar yang berisi penjelasan singkat tentang field atau tombol tersebut. Contoh text tool tip untuk field :



- Contoh text tool tip untuk tombol :



- Untuk jenis kelamin hanya dapat dipilih satu pilihan saja.
- Untuk pilihan Jurusan, isi fieldnya bergantung kepada pilihan apa yang dipilih oleh user pada Program Studi. Jika user memilih program studi S1, maka pada field jurusan hanya terdapat 2 pilihan yaitu Sistem Informasi dan Sistem Komputer. Sedangkan jika user memilih program studi D3, maka field jurusan hanya terdapat 2 pilihan juga yaitu Manajemen Informatika dan Komputer Akuntansi.
- Saat tombol simpan di tekan, maka lakukan pengecekan terlebih dahulu apakah data sudah terisi semua atau belum. Jika data telah terisi semua, maka simpanlah data tersebut, sedangkan jika ada yang tidak/belum diisi, maka tampilkan pesan bahwa semua field harus diisi.

Catatan :

Anda dapat melanjutkan project latihan modul 7 milik anda.