

Temu04

Object, Class,
Instance, Message

Ali Tarmuji, S.T., M.Cs.

alitarmuji@tif.uad.ac.id

Bahan Diskusi

- Deskripsi PBO
- Object
- Class
- Instance
- Message

Apa itu PBO?

- Pemrograman Berorientasi Objek merupakan sebuah paradigma pemrograman yang berorientasikan kepada objek.
- Semua atribut, prosedur dan fungsi dibungkus dalam objek.
- Interaksi tiap data, prosedur dan fungsi dilakukan melalui objek.

Apa Itu Objek

- Sebuah Objek merupakan sebuah entitas dengan boundary yang terdefinisi dengan baik dan identitas yang meneng kapsulasi state dan behaviour
- Objek merupakan segala sesuatu yang ada didunia ini, yaitu manusia, hewan, tumbuhan, rumah, kendaraan, dan lain sebagainya.
- Setiap Objek memiliki dua karakteristik, yaitu keadaan/state dan tingkahlaku/behavior.

Object

- Suatu entitas yang mempunyai state dan behavior.
- Biasanya diartikan sebagai entitas dengan kata nyata
- Dapat berupa logika ataupun fisik
- Contoh: kursi, motor, mahasiswa, dosen karyawan



State

- Digunakan untuk menyimpan informasi objek
- Dalam java, state disebut juga dengan attribute atau field.
- Misalnya objek mahasiswa memiliki state: nim, nama, alamat.

Behavior

- Digunakan untuk menentukan kerja apa saja yang dapat dilakukan objek tersebut.
- Dalam java, disebut juga dengan method/metode
- Contoh method: melakukan KRS, cetak KHS

Apa itu Class

- Class merupakan cetak biru (blue print) dari objek atau dengan kata lain sebuah Class menggambarkan ciri - ciri objek secara umum.
- Class merupakan kerangka kode yang berisikan struktur atribut, struktur prosedur dan struktur fungsi yang digunakan untuk membuat objek.
- Sekumpulan object yang memiliki struktur data dan behaviour yang sama

class

- Kumpulan dari banyak objek
- Berupa entitas logika
- Dalam sebuah class, state diimplementasikan menjadi atribut/variabel dan behavior diimplementasikan menjadi method baik dalam bentuk prosedur maupun fungsi

Contoh Class dan Objek

- 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)
- Method untuk mengakses data pada atributnya (misal fungsi untuk menginputkan data merk, tipe, berat, dsb serta fungsi untuk mencetak data merk, tipe, berat, dsb).

Membuat Class

- Nama Class harus sama dengan nama File.
- Nama Class tidak boleh mengandung huruf unik (@, #, \$, %, &, dll) dan huruf whitespace (spasi, enter, tab, dll)
- Deklarasi class

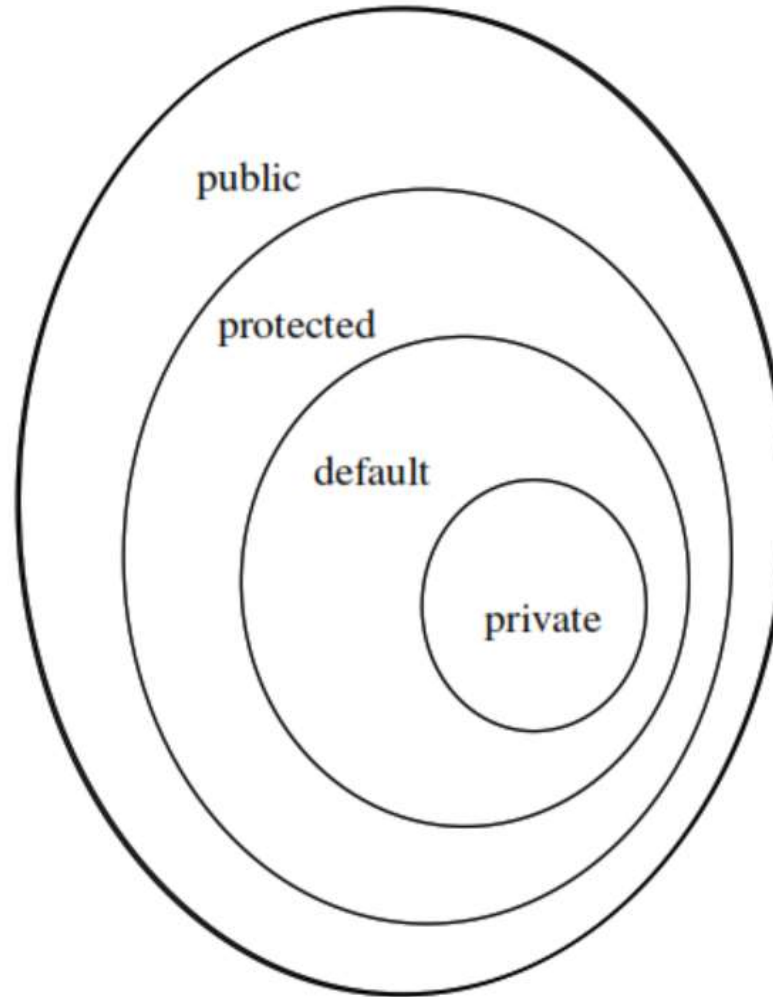
```
<modifier> class <nama_class> {  
    [deklarasi_atribut]  
    [deklarasi_konstruktor]  
    .....  
    [deklarasi_metode]  
}
```

```
Contoh: public class Siswa {  
    ...  
}
```

Scope dari modifier akses

Wilayah Akses	<i>public</i>	<i>protected</i>	<i>default</i>	<i>private</i>
Di kelas yg sama	√	√	√	√
Beda kelas, di package yg sama	√	√	√	x
Beda kelas, beda package, di kelas turunan	√	√	x	x
Beda kelas, beda package, tidak di kelas turunan	√	x	x	x

Scope dari modifier akses



- Public→menyatakan bahwa class/ method/ attribute tersebut dapat diakses oleh class lain di manapun
- Protected→menyatakan bahwa class/ method/ attribute tersebut dapat diakses oleh Class lain yang berada dalam satu package atau Class lain tersebut merupakan turunannya
- Default→mengikuti status superiornya
- Private
 - menyatakan bahwa Class tersebut tidak dapat diakses sama sekali oleh Class lain bahkan juga tidak dapat diturunkan.
 - Attribute-attribute yang private hanya dapat diakses oleh method-method dalam Class yang sama, Class lain masih dapat mengakses melalui method-method tersebut asal modifiernya public.
 - Pertimbangan suatu attribute dideklarasikan private :
 - 1. Bila Class lain tak memerlukan attribute tersebut.
 - 2. Melindungi suatu attribute dari kemungkinan nilainya diubah oleh method lain dari class lain

Atribut

- Atribut merupakan data atau sifat yang dimiliki oleh sebuah Class.
- Atribut dibuat layaknya sebuah variabel.

deklarasi atribut

```
<modifier> <type> <nama_atribut> ;
```

Contoh:

```
public class Siswa {  
    public int nrp;  
    public String nama;  
}
```


Manipulasi Atribut

- Atribut dapat diakses lewat Class.
- Untuk mengakses atribut, dapat menggunakan tanda . (titik)

```
NamaClass objek = new NamaClass();  
objek.namaAtribut = ...;
```

Deklarasi method

```
<modifier> <return_type> <nama_metode> ([daftar_argumen])  
{  
    [<statement>]  
}
```

Contoh:

```
public class Siswa {  
    public int nrp;  
    public String nama;  
    public void info() {  
        System.out.println("Ini siswa");  
    }  
}
```

- Instance dari class → untuk akses anggota objek

```
public class Siswa {  
    public static void main(String args[]) {  
        Siswa it=new Siswa();  
        it.nrp=5;  
        it.nama="Andi";  
        it.info();  
    }  
}
```

Membuat Objek

- Untuk membuat objek, di Java dapat menggunakan perintah **new**.
- Atribut, metode dan fungsi yang dimiliki oleh objek, hanya yang dideklarasikan dalam Class.
- Objek merupakan variabel yang memiliki tipe data Class
- Kriteria pembuatan objek sama seperti pembuatan variabel.

NamaClass object = new NamaClass();

Konstanta

- Sebuah atribut dapat dijadikan sebagai konstanta.
- Caranya sama dengan membuat variabel konstanta, yaitu dengan menambahkan perintah **final**.

Membuat Konstanta

```
class NamaClass {
```

```
    TipeData namaAtribut;
```

```
    final TipeData namaKonstanta = ...;
```

```
}
```

Konstanta

- Sebuah atribut dapat dijadikan sebagai konstanta.
- Caranya sama dengan membuat variabel konstanta, yaitu dengan menambahkan perintah **final**.

Membuat Konstanta

```
class NamaClass {
```

```
    TipeData namaAtribut;
```

```
    final TipeData namaKonstanta = ...;
```

```
}
```


Prosedur

- Prosedur merupakan sebuah kode program yang dapat digunakan untuk menjalankan instruksi program dalam sebuah Class.
- Sebuah Class dapat memiliki lebih dari satu prosedur.
- Nama prosedur tidak boleh sama dengan prosedur lain.

Membuat Prosedur

```
class NamaClass {  
  
    void namaProsedur(){  
        // isi prosedur  
    }  
  
}
```

Menggunakan Prosedur

- *NamaClass objek = new NamaClass();*
- *objek.namaProsedur();*

Fungsi

- Fungsi merupakan kode program yang digunakan untuk menghasilkan sesuatu.
- Untuk mengembalikan hasil, dapat menggunakan perintah **return** diikuti dengan data yang dihasilkan.
- Fungsi hanya dapat menghasilkan sebuah nilai.
- Sebuah Class dapat memiliki lebih dari satu fungsi.
- Nama fungsi tidak boleh sama dengan fungsi yang lain.

Membuat Fungsi

```
class NamaClass {
```

```
    TipeData namaFungsi(){
```

```
        // kode program
```

```
        return hasil;
```

```
    }
```

```
}
```

Menggunakan Fungsi

NamaClass objek = new NamaClass();

TipeData hasil = objek.namaFungsi();

Parameter

- Parameter merupakan data yang dapat disisipkan ke dalam prosedur dan fungsi.
- Jumlah parameter dalam prosedur dan fungsi dapat lebih dari satu.

Membuat Parameter

- *class NamaClass {*
- *void NamaProsedur(**TipeData parameter**) {*
- *// kode program*
- *}*
- *TipeData namaFungsi(**TipeData param1,***
 ***TipeData param2**) {*
- *// kode program*
- *return hasil;*
- *}*
- *}*

Menggunakan Parameter

NamaClass objek = new NamaClass();

objek.namaProsedur(“String ”);

TipeData hasil = objek.namaFungsi(100, 300);

TipeData variabel = ...;

objek.namaProsedur2(variabel);

Kata Kunci this

- Kata kunci **this** digunakan untuk mengakses Class itu sendiri.
- Biasanya kata kunci this digunakan ketika mengakses sebuah atribut Class, dimana nama atribut tersebut sama dengan nama parameter.

Menggunakan this

```
class NamaClass {
```

```
    String nama;
```

```
    void ubahNama(String nama){
```

```
        this.nama = nama;
```

```
    }
```

```
}
```

On The Laptop 1

```
1  package com.alitarmuji;
2
3  class ClassA {
4      int i = 5;
5  }
6
7  class ClassEx1 {
8      public static void main(String args[]) {
9          System.out.println("*** A Simple class with 2 objects-obA And obB ***");
10         ClassA obA = new ClassA();
11         ClassA obB = new ClassA();
12         System.out.println("obA.i =" + obA.i);
13         System.out.println("obB.i =" + obB.i );
14     }
15 }
16
```

output

*** A Simple class with a method returning an integer ***

Sum of 10 and 20 is : 30

Process finished with exit code 0

On The Laptop 2

```
1  package com.alitarmuji;  
2  
3  class MyClass {  
4      protected MyClass() {  
5          System.out.println("I am a no argument constructor");  
6          System.out.println("I Can have additional logic");  
7      }  
8  }  
9  
10 public class ExperimentWithConstructorEx1 {  
11     public static void main(String args[]) {  
12         System.out.println("*** Experiment with constructors ***");  
13         MyClass myOb = new MyClass();  
14     }  
15  
16 }
```

output

*** Experiment with constructors ***

I am a no argument constructor

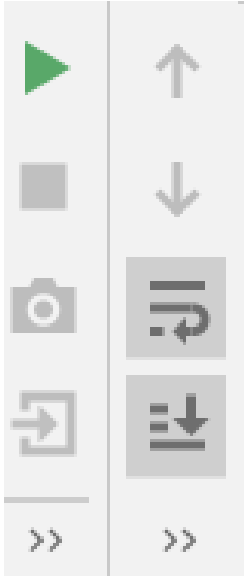
I Can have additional logic

Process finished with exit code 0

On The Laptop 3

```
1  package com.alitarmuji;|
2
3  class ClassA3 {
4      int i;
5
6  @   ClassA3(int i) {
7      |     this.i = i;
8      |
9      | }
10
11  class ClassEx3 {
12  public static void main(String args[]) {
13      System.out.println("*** A Simple class with 2 objects-obA And obB ***");
14      System.out.println("*** obA.i And obB.i are different here ***");
15      ClassA3 obA = new ClassA3(i: 20);
16      ClassA3 obB = new ClassA3(i: 30);
17      System.out.println("obA.i =" + obA.i);
18      System.out.println("obB.i =" + obB.i);
19  }
20 }
```


Output

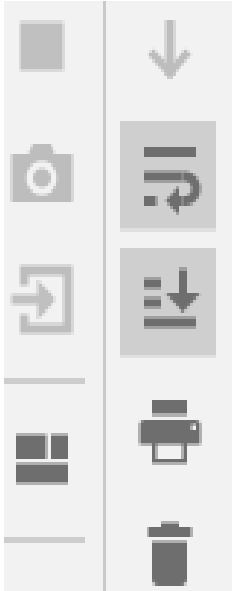


```
*** A Simple class with 2 objects-obA And obB ***  
*** obA.i And obB.i are different here ***  
obA.i =20  
obB.i =30  
  
Process finished with exit code 0  
█
```

On The Laptop 4

```
1 package com.alitarmuji;
2
3 class ClassA4 {
4     int i;
5
6     @ ClassA4() {
7         this.i = 7;
8     }
9
10    @ ClassA4(int i) {
11        this.i = i;
12    }
13 }
14
15 class ClassEx4 {
16     public static void main(String args[]) {
17         System.out.println("*** A Simple class with 2 objects-obA And obB ***");
18         System.out.println("*** Different type of constructors are used here ***");
19         ClassA4 obA = new ClassA4();
20         ClassA4 obB = new ClassA4(i: 25);
21         System.out.println("obA.i =" + obA.i);
22         System.out.println("obB.i =" + obB.i);
23     }
24 }
```

output



```
*** A Simple class with 2 objects-obA And obB ***  
*** Different type of constructors are used here ***  
obA.i =7  
obB.i =25  
  
Process finished with exit code 0
```

On The Laptop 5

```
1  package com.alitarmuji;
2  class Demo5 {
3      int sum(int x, int y) {
4          return x + y;
5      }
6  }
7
8  public class ClassEx5 {
9      public static void main(String args[]) {
10         System.out.println("*** A Simple class with a method returning an integer ***");
11         Demo5 ob = new Demo5();
12         int result = ob.sum(x: 10, y: 20);
13         System.out.println("Sum of 10 and 20 is : " + result);
14     }
15 }
```

Output

*** A Simple class with a method returning an integer ***

Sum of 10 and 20 is : 30

Process finished with exit code 0