BAB IV

PERBEDAAN FINAL, STATIC, PUBLIC, PRIVATE ,PROTECTED

DALAM VARIABLE DAN METHODE

A. FINAL

1. Dalam Variable

Jika sebuah variable menggunakan kata **final** maka variable tersebut akan berubah menjadi sebuah **konstanta** dan tidak dapat di set / di isi dengan nilai yang baru.

Berikut contoh programnya :

```
package Final;

public class final_variable {   //kelas induk

   final int angka = 23;
   int bilangan = 4;

   public final void MethodeFinal(){
         System.out.println("ini methode MethodeFinal dari kelas final_variable");
   }
   public void VariableFinal() { // konstraktor
         // angka = 9;                /*kalau Di Aktifin Akan terjadi error*/
         // bilangan = 8;            /*kalau Di Aktifin tidak Akan terjadi error*/
         System.out.println("ini methode VariableFinal dari kelas final_variable");
   }

   public static void main(String[] args) {
         final_variable f = new final_variable();
         f.VariableFinal();
         System.out.println("Angka : " + f.angka);
         System.out.println("Bilangan : " + f.bilangan);
   }
}
```

Coba running sendiri yaaa . . . . .!!!!! ☺☺ hasilnya... pok....pok....pokk..pokk..pok..

2. **Dalam Method**

Jika sebuah methode menggunakan akses **final** maka methode tersebut tidak bisa di **override* method nya** oleh kelas anaknya / subkelas.

*) cari tau sendiri, pelajari pewarisan;

Kita masih menggunakan kelas **diatas,** kelas diatas sebagai induknya dan coba perhatikan **MethodeFinal( )** diatas**.** Methode tersebut kan menggunakan final jadi tidak bisa di **override** oleh kelas anaknya di bawah ini.

```
package Final;
public class final_methode extends final_variable {    //kelas anak

   @Override
   public void VariableFinal() {            // methode dari kelas anak
      //  super.MethodeFinal();             /*memanggil methode induknya*/
      System.out.println("ini methode VariableFinal dari kelas final_methode");
   }

   // @Override // kalau dibuka komennya akan error
   // public void MethodeFinal() {
   //      System.out.println("ini methode MethodeFinal dari kelas final_methode");
    // }

   public static void main(String[] args) {
            final_methode f = new final_methode();
            // f.MethodeFinal();
            f.VariableFinal();
   }
}
```

Dalam kelas anak diatas (kelas anak di tandai **extends**) tidak bisa meng**Override** method **MetodeFinal()** dari kelas induknya yakni kelas **final_variable,** karena di kelas induknya methode tersebut menggunakan akses **final.** Sebaliknya methode **VariableFinal()** dapat diOverride di kelas anaknya karena tidak menggunakan final.

B. STATIC

1. Dalam Variable

Jika sebuah variable menggunakan **static** maka variable tersebut akan menjadi variable kelas. nilainya akan selalu berubah jika ada perubahan di objeknya.

```
package Static;
public class VariableStatic {
   static int bilangan = 0;
   int angka = 0;

   public void perubahan() {
         bilangan += 10;
         angka += 10;
         System.out.println("Nilai BILANGAN : " + bilangan);
         System.out.println("Nilai ANGKA    : " + angka);
   }

   public static void main(String[] args) {
         VariableStatic vs1 = new VariableStatic();
         vs1.perubahan();
         VariableStatic vs2 = new VariableStatic();
         vs2.perubahan();
         VariableStatic vs3 = new VariableStatic();
         vs3.perubahan();
         VariableStatic vs4 = new VariableStatic();
         vs4.perubahan();
   }
}
```

Coba di run... sendiri yaaa.....!!! hasilllnyaaaa.. pokkk.. pokkk.. pokk.. ☺ ☺ ☺

Mantaplahhh.....!@#$#@$@?@#$%^*

2. **Dalam Methode**

   Jika sebuah methode menggunakan **static** maka untuk mengaksesnya tidak perlu menggunakan objek / instansiasi kelas nya dulu cukup **NamaKelas.NamaMethode.** terus juga variable yang ada didalamnya haruslah static tidak boleh menggunakan variable biasa, kecuali variable dari instansiasi dan variable yang baru dideklarasikan dimethode tersebut.

```
package Static;
public class MethodStatic {
  int a = 10;
  public static void biasa() {
        // a = 0;      // variable "a" tidak bisa di pakai kecuali pakai objek
        int b = 5;
        System.out.println("Niali B : " + b);

        MethodStatic ms = new MethodStatic();
        System.out.println("Nilai A : " + ms.a);
          //  untuk mengkases nilai "a" harus di instansiasi dulu
  }

  public static void main(String[] args) {
        MethodStatic.biasa();
        //untuk memanggilnya tidak perlu pake objek
        biasa();
        //langsung panggil
  }
}
```

Jadi jelaskan,!!! ☺ kenapa dalam main harus di instansiasi dulu walaupun itu satu kelas yang sama. Karena **main()** kan bentuknya **static,** main hanya dapat memanggil langsung methode yang bentuknya **static** juga dalam kelas yang sama dengan **main**.

C. PENENTU AKSES VARIABLE DAN METHODE

a. PUBLIC

Baik variable ataupun methode, Jika methode/variable tersebut menggunakan **PUBLIC**, maka methode/variable tersebut bisa dipakai oleh **kelas manapun** baik **paket(paket=package)** yang sama maupun paket yang berbeda. Tentu saja untuk memanggilnya pakai instansiasi dulu.

b. PRIVATE

Jika methode/variable menggunakan **PRIVATE**, maka methode/ variable tersebut **HANYA** bisa dipakai oleh **kelas itu sendiri** dan tidak dapat dipanggil oleh kelas lain.

c. PROTECTED

Jika methode/variable menggunakan **PROTECTED**, maka methode/variable tersebut **hanya** bisa dipakai oleh **kelas itu sendiri** dan **kelas turunannya / subkelasnya,**. Kalau tidak paham pelajari pewarisan.

d. Tanpa Penentu Akses / tidak menggunakan (Private, Public, Protected)

Jika methode/variable **tidak menggunakan ketiganya**, berbeda dengan akses **public** maka Variable dan methode tersebut memang bisa di akses oleh kelas manapun asal masih satu paket(**package**) jadi tidak boleh beda package.

Jadi tau kan perbedaan antara public dan tidak menggunakan apapun.?? ☺

*) Maaf untuk pembahasan ini Saya tidak menyertakan Contohnya karena saya yakini anda pasti nya sudah mengerti ini contohnya bukan?? ☺ ☺

☺ ☺ - - SELAMAT MENCOBA - - ☺ ☺

~ SAMPAI KETEMU DIEDISI BERIKUTNYA~

^_^

Nama : Akhmad Bilyyasif

Email : asief.asief@yahoo.com

Yogyakarta, 13 MARET 2012