

LAPORAN RESMI
MODUL II
(Constructor dan Keyword Static)
PEMROGRAMAN BERBASIS OBJEK



NAMA	: WARDATUS SHOLICHA
N.R.P	: 230441100170
DOSEN	: ACHMAD ZAIN NUR S.Kom., M.T
ASISTEN	: DEVI DWI NOVITASARI
TGL PRAKTIKUM	: 30 MARET 2024

Disetujui : MARET 2024

Asisten

DEVI DWI NOVITASARI

22.04.411.00090



LABORATORIUM BISNIS INTELIJEN SISTEM
PRODI SISTEM INFORMASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO MADURA

BAB I

PENDAHULUAN

1.1 Latar Belakang

Konstruktor di Java adalah metode khusus yang digunakan untuk menginisialisasi objek. Konstruktor dipanggil ketika objek kelas dibuat. Ini dapat digunakan untuk menetapkan nilai awal untuk atribut objek. Konstruktor juga dapat mengambil parameter, yang digunakan untuk menginisialisasi atribut. Dalam pemrograman java, keyword static digunakan untuk mengakses variable ataupun method (prosedur atau fungsi) pada class tertentu tanpa harus membuat suatu objek dari class itu. Umumnya untuk mengakses member dari kelas lain kita harus membuat objek kelas, tapi dengan menggunakan keyword static kita dapat langsung menggunakan member kelas lain.

Constructor adalah method yang secara otomatis dipanggil/dijalankan pada saat new dipakai untuk menciptakan instan kelas. Atau dengan kata lain constructor adalah method yang pertama kali dijalankan pada saat sebuah objek pertama kali diciptakan. Jika dalam sebuah class tidak terdapat constructor, maka secara otomatis Java akan membuatkan sebuah default constructor. Sama halnya dengan method, constructor dapat memiliki satu atau banyak parameter maupun tanpa parameter

1.2 Tujuan

- Mahasiswa mampu memahami konsep Constructor dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.
- Mahasiswa mampu memahami penggunaan keyword static dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.
- Mahasiswa mampu memahami penggunaan setter() dan getter() dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.

BAB II

DASAR TEORI

2.1 Constructor

2.1.1 Pengertian/Konsep

Constructor adalah method yang secara otomatis dipanggil/dijalankan pada saat *new* dipakai untuk menciptakan instan kelas. Atau dengan kata lain **constructor** adalah method yang pertama kali dijalankan pada saat sebuah objek pertama kali diciptakan. Jika dalam sebuah class tidak terdapat **constructor**, maka secara otomatis Java akan membuatkan sebuah default **constructor**. Sama halnya dengan method, **constructor** dapat memiliki satu atau banyak parameter maupun tanpa parameter. Hal mendasar yang perlu diperhatikan, yaitu :

- Nama *Constructor* sama dengan nama *Class*.
- Tidak ada *return type* yang diberikan kedalam *Constructor Signature*.
- Tidak ada *return statement*, di dalam tubuh *constructor*.

2.1.2 Contoh Program

a. *Constructor Kendaraan.java*

```
1 public class ConstructorKendaraan {
2     private String merk; // ini adalah instant variable
3     private static String pemilik; // ini adalah static variable
4
5     // ini adalah constructor tanpa parameter
6     protected ConstructorKendaraan() {
7         merk = null;
8     }
9
10    // overloading constructor
11    // ini adalah constructor dengan parameter merk
12    protected ConstructorKendaraan(String merk) {
13        this.merk = merk;
14        merk = null;
15    }
16
17    protected void setMerk(String merk) {
18        this.merk = merk;
19    }
20
21    protected String getMerk() {
22        return merk;
23    }
24
25    // ini adalah static method
26    protected static void setPemilik(String pemilik) {
27        ConstructorKendaraan.pemilik = pemilik;
28    }
29
30    // ini adalah static method
31    protected static String getPemilik() {
32        return ConstructorKendaraan.pemilik;
33    }
34
35    protected void tampil(String a)
36    {
37        System.out.println(a);
38        a = null;
39    }
40
41    protected void hapus()
42    {
43        // menghapus variable private dari memory
44        merk = null;
45        pemilik = null;
46    }
47 }
```

b. ConstructorMotor.java

```
1 // ConstructorMotor turunan dari class ConstructorKendaraan
2 public class ConstructorMotor extends ConstructorKendaraan {
3     private String warna; // ini adalah instant variable
4
5     // ini adalah constructor dengan parameter merk & warna
6     protected ConstructorMotor(String merk, String warna){
7         // memanggil constructor parent (class ConstructorKendaraan)
8         // dengan keyword super dan parameter merk
9         super(merk);
10
11         this.warna = warna;
12
13         // menghapus variable parameter dari memory
14         merk = null;
15         warna = null;
16     }
17
18     protected String getWarna() {
19         return warna;
20     }
21
22     protected void hapus()
23     { // menghapus variable private dari memory
24         warna = null;
25         // memanggil method hapus class parent (class ConstructorKendaraan)
26         // dengan keyword super
27         super.hapus();
28     }
29 }
```

c. MainConstructorMotor.java

```
1 public class MainConstructorMotor {
2     public static void main(String args[])
3     {
4         String pemilik = "Ahmad Afif";
5         String merk = "Honda";
6         String warna = "Merah";
7
8         // cara akses static variable dan static method dapat dengan
9         // memanggil class (ConstructorKendaraan) secara langsung
10        ConstructorKendaraan.setPemilik(pemilik);
11        System.out.println("Pemilik Kendaraan = "+ConstructorKendaraan.getPemilik());
12        System.out.println("=====");
13
14        // variable merk menjadi parameter Constructor pada saat
15        // instansiasi/membuat objek baru (ob)
16        ConstructorKendaraan ob = new ConstructorKendaraan(merk);
17        ob.tampil("Merk Kendaraan = "+ob.getMerk());
18        // cara akses static variable dan static method dapat juga dengan
19        // objek (ob) pada method getPemilik()
20        ob.tampil("Pemilik Kendaraan = "+ob.getPemilik());
21        System.out.println("=====");
22
23        // instansiasi/membuat objek baru (ob2) tanpa parameter
24        ConstructorKendaraan ob2 = new ConstructorKendaraan();
25        // bandingkan akses untuk menampilkan nilai pada instant variable (merk)
26        // dan static variable (pemilik) melalui method getter setelah membuat
27        // objek baru "ob2", dimana sebelumnya juga sudah membuat objek "ob".
28        // instant variable (merk) nilainya akan hilang,
29        // sedangkan static variable (pemilik) nilainya tidak hilang/berubah
30        ob2.tampil("Merk Kendaraan (instant variable) = "+ob2.getMerk());
31        ob2.tampil("Pemilik Kendaraan (static variable) = "+ob2.getPemilik());
32        System.out.println("=====");
33
34        // variable merk dan warna menjadi parameter Constructor pada saat
35        // instansiasi/membuat objek baru (ob3)
36        ConstructorMotor ob3 = new ConstructorMotor(merk, warna);
37        ob3.tampil("Merk Motor = "+ob3.getMerk());
38        ob3.tampil("Warna Motor = "+ob3.getWarna());
39        ob3.tampil("Pemilik Motor = "+ob3.getPemilik());
40
41        pemilik = null;
42        merk = null;
43        warna = null;
44        ob.hapus();
45        ob = null;
46        ob2 = null;
47        ob3 = null;
48    }
49 }
```

2.2 Keyword Static

2.2.1 Pengertian/Konsep

Dalam pemrograman java, keyword *static* digunakan untuk mengakses variable ataupun method (prosedur atau fungsi) pada class tertentu tanpa harus membuat suatu objek dari class itu. Umumnya untuk mengakses member dari kelas lain kita harus membuat objek kelas, tapi dengan menggunakan *keyword static* kita dapat langsung menggunakan member kelas lain. *Keyword static* bisa digunakan untuk variable ataupun method.

Static member adalah variable atau method yang bukan milik dari suatu object, melainkan milik dari class.

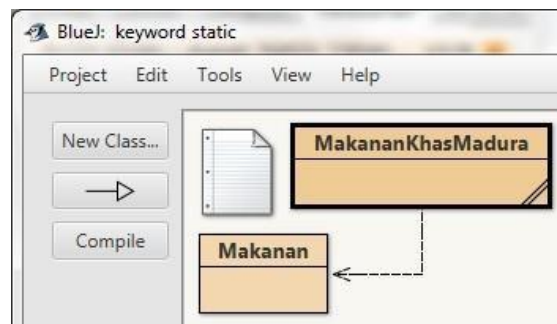
Bentuk Umum:

static void namaMethodStatic(){

Method yang dideklarasikan sebagai static memiliki aturan sebagai berikut :

- Hanya dapat dipanggil oleh method lain yang juga adalah static method.
- Hanya dapat mengakses atribut static.
- Tidak dapat menggunakan keyword *this* dan *super*, karena kedua keyword ini menunjuk ke suatu instance tertentu, bukan pada sebuah object.

2.2.2 Contoh Program

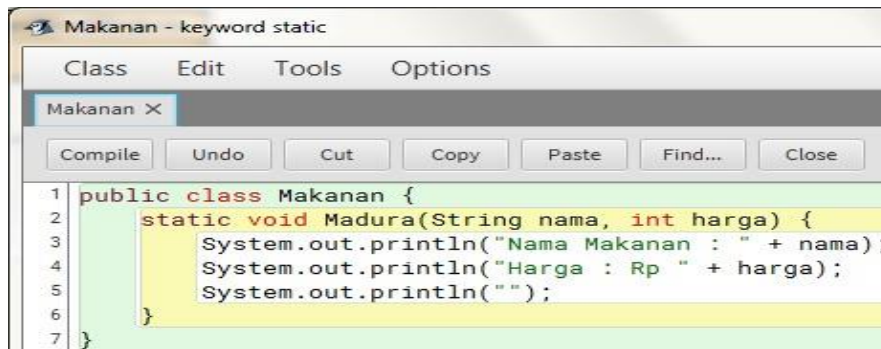


- Class Makanan
- Class MakananKhasMadura

2.2.3 Tampilan Class

- Class Makanan

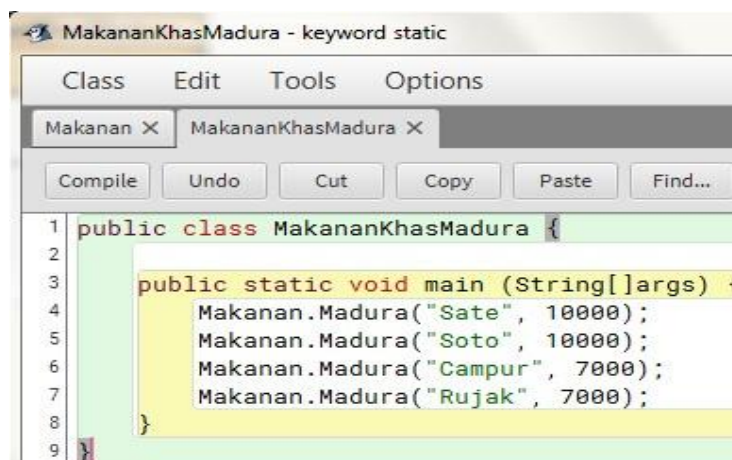
Pada class Makanan, method di set sebagai *static* dengan parameter makanan dan harga.



```
1 public class Makanan {
2     static void Madura(String nama, int harga) {
3         System.out.println("Nama Makanan : " + nama);
4         System.out.println("Harga : Rp " + harga);
5         System.out.println("");
6     }
7 }
```

b. Class MakananKhasMadura

Pada class MakananKhasMadura, method juga bersifat *static* tanpa membuat *instance* objek dari kelas Makanan.



```
1 public class MakananKhasMadura {
2
3     public static void main (String[]largs) {
4         Makanan.Madura("Sate", 10000);
5         Makanan.Madura("Soto", 10000);
6         Makanan.Madura("Campur", 7000);
7         Makanan.Madura("Rujak", 7000);
8     }
9 }
```

2.2.4. Running Program

2.2.5. Penjelasan Program

a. Class Makanan

Pada class Makanan, terdapat method static dengan parameter makanan dan harga.

- i. Baris 2, *keyword static* yang berupa method.
- ii. Baris 3-4, mencetak nama makanan dan harga.

b. Class MakananKhasMadura

Pada class MakananKhasMadura, method juga bersifat static tanpa membuat *instance* objek dari kelas Makanan.

- i. Baris 3, method static untuk menjalankan program.
- ii. Baris 4-7, nilai dari parameter yang terdapat pada class Makanan.

BAB III

TUGAS PENDAHULUAN

3.1 Soal

1. Jelaskan perbedaan Constructor dan keyword static!
2. Jelaskan setter dan getter menurut bahasa pemahamanmu!

3.2 Jawaban

1. Constructor adalah method yang pertama kali dijalankan pada saat sebuah objek pertama kali diciptakan. Di dalam constructor, nama kelas dan nama Constructor sama, tidak ada return type dan juga return statement didalamnya. Sedangkan keyword static digunakan untuk mengakses variabel ataupun method pada class tertentu tanpa harus membuat suatu objek dari class itu.

2. a. setter

Metode setter digunakan untuk mengatur (menetapkan) nilai dari sebuah atribut atau properti dalam sebuah objek. Setter memungkinkan kita untuk mengubah nilai dari atribut sesuai dengan kebutuhan. Penamaan setter dimulai dengan "Set" lalu diikuti dengan nama atribut yang akan diatur nilainya.

b. getter

Metode getter digunakan untuk mendapatkan (mengambil) nilai dari sebuah atribut atau properti dalam sebuah objek. Getter memungkinkan kita untuk mengakses nilai dari atribut tanpa perlu mengubah nilai tersebut secara langsung. Getter memiliki nama yang dimulai dengan "get" dan diikuti dengan nama atribut yang ingin diambil nilainya.

BAB IV

IMPLEMENTASI

4.1 Source Code

1. Program class Data Mahasiswa

a. Main class

```
package datamahasiswa;
import java.util.Scanner;
public class DataMahasiswa {
    public static void main(String[] args) {
        String lanjut = "Y";
        while(lanjut.equals("Y")){
            Scanner inputUser = new Scanner(System.in);
            System.out.println("===== INPUT DATA MAHASISWA =====");
            System.out.println(" ");
            System.out.print("UNIVERSITAS : ");
            String univ = inputUser.nextLine();
            System.out.print("NIM      : ");
            String NIM = inputUser.nextLine();
            System.out.print("NAMA      : ");
            String Nama = inputUser.nextLine();
            System.out.print("ALAMAT    : ");
            String Alamat = inputUser.nextLine();
            System.out.println("DAFTAR KODE JURUSAN :");
            System.out.println(" TEKNIK INFORMATIKA   [41]");
            System.out.println(" TEKNIK INDUSTRI     [42]");
            System.out.println(" TEKNIK ELEKTRO       [43]");
            System.out.println(" SISTEM INFORMASI     [44]");
            System.out.println(" TEKNIK MESIN         [48]");
            System.out.println(" TEKNIK MEKATRONIKA   [49]");
            System.out.print("KODE JURUSAN : ");
            String Jurusan = inputUser.nextLine();
            System.out.println(" ");
        }
    }
}
```



```

        DataUniv datamhs = new DataUniv(univ, NIM, Nama, Alamat, Jurusan);
System.out.println("=====DATA MAHASISWA=====");
System.out.println("UNIVERSITAS : " + DataUniv.getUniv());
System.out.println("NIM      : " + DataUniv.getNIM());
System.out.println("NAMA      : " + DataUniv.getNama());
System.out.println("ALAMAT    : " + DataUniv.getAlamat());
switch (DataUniv.getJurusan()){
    case "41":
        System.out.println("JURUSAN    : TEKNIK INFORMATIKA");
        break;
    case "42":
        System.out.println("JURUSAN    : TEKNIK INDUSTRI");
        break;
    case "43":
        System.out.println("JURUSAN    : TEKNIK ELEKTRO");
        break;
    case "44":
        System.out.println("JURUSAN    : SISTEM INFORMASI");
        break;
    case "48":
        System.out.println("JURUSAN    : TEKNIK MESIN");
        break;
    case "49":
        System.out.println("JURUSAN          :    TEKNIK
MEKATRONIKASUT");
        break;
    default :
        System.out.println("KODE JURUSAN TIDAK VALID");
}

```

```

System.out.println("=====");
    System.out.println(" ");
    System.out.println("APAKAH ANDA INGIN MEMASUKKAN DATA
LAGI?");
    System.out.println("1. YA [Y] \n2. TIDAK [T]");
    System.out.println("PILIHAN : ");
    lanjut = inputUser.nextLine();
}
System.out.println("=====TERIMAKASIH=====");
}

}

```

b. Class method

```

package datamahasiswa;

public class DataUniv {
    public static String Univ;
    public static String NIM;
    public static String Nama;
    public static String Alamat;
    public static String Jurusan;

    public DataUniv(String Univ, String NIM, String Nama, String Alamat, String
Jurusan) {
        this.Univ = Univ;
        this.NIM = NIM;
        this.Nama = Nama;
        this.Alatamat = Alamat;
        this.Jurusan = Jurusan;
    }
}

```

```
public static void setUniv(String Univ) {
    DataUniv.Univ = Univ;
}
public static void setNIM(String NIM) {
    DataUniv.NIM = NIM;
}
public static void setName(String Nama) {
    DataUniv>Nama = Nama;
}
public static void setAddress(String Alamat) {
    DataUniv.Alatnat = Alamat;
}
public static void setJurusan(String Jurusan) {
    DataUniv.Jurusan = Jurusan;
}
public static String getUniv() {
    return Univ;
}
public static String getNIM() {
    return NIM;
}
public static String getName() {
    return Nama;
}
public static String getAddress() {
    return Alamat;
}
public static String getJurusan() {
    return Jurusan;
}
```

4.2 Penjelasan

1. Penjelasan program Data Mahasiswa

Program ini adalah program sederhana untuk menginputkan data mahasiswa, menyimpannya, dan menampilkannya kembali. Berikut adalah langkah-langkah secara singkat bagaimana program ini berjalan:

- Program dimulai dengan deklarasi variabel lanjut yang diinisialisasi dengan nilai "Y" yang menandakan bahwa pengguna ingin melanjutkan memasukkan data.
- Program memasuki loop while yang akan terus berjalan selama variabel lanjut memiliki nilai "Y".
- Di dalam loop, program akan menampilkan pesan dan meminta pengguna untuk memasukkan data mahasiswa seperti universitas, NIM, nama, alamat, dan kode jurusan. Data tersebut dimasukkan melalui objek Scanner.
- Setelah data dimasukkan, objek DataUniv dibuat dengan menggunakan konstruktor untuk menyimpan data mahasiswa yang dimasukkan.
- Data mahasiswa yang telah dimasukkan kemudian ditampilkan kembali dengan menggunakan method `getUniv()`, `getNIM()`, `getNama()`, `getAlamat()`, dan `getJurusan()` dari objek DataUniv.
- Untuk menampilkan nama jurusan, program menggunakan sebuah switch statement berdasarkan kode jurusan yang dimasukkan oleh pengguna.
- Program menanyakan kepada pengguna apakah ingin memasukkan data mahasiswa lagi. Jika pengguna memilih "Y", program akan kembali ke langkah 3. Jika pengguna memilih "T", loop akan berhenti.
- Program menampilkan pesan "TERIMA KASIH" dan selesai.

4.3 Hasil

===== INPUT DATA MAHASISWA =====

UNIVERSITAS : TRUNOJOYO MADURA
NIM : 230441100170
NAMA : WARDATUS SHOLICHA
ALAMAT : JL. KETAPANG BESAR 36-A
DAFTAR KODE JURUSAN :
TEKNIK INFORMATIKA [41]
TEKNIK INDUSTRI [42]
TEKNIK ELEKTRO [43]
SISTEM INFORMASI [44]
TEKNIK MESIN [48]
TEKNIK MEKATRONIKA [49]
KODE JURUSAN : 44

=====DATA MAHASISWA=====

UNIVERSITAS : TRUNOJOYO MADURA
NIM : 230441100170
NAMA : WARDATUS SHOLICHA
ALAMAT : JL. KETAPANG BESAR 36-A
JURUSAN : SISTEM INFORMASI

=====

APAKAH ANDA INGIN MEMASUKKAN DATA LAGI?

1. YA [Y]
 2. TIDAK [T]
- PILIHAN : Y

===== INPUT DATA MAHASISWA =====

UNIVERSITAS : |

APAKAH ANDA INGIN MEMASUKKAN DATA LAGI?

1. YA [Y]
 2. TIDAK [T]
- PILIHAN : T

=====TERIMA KASIH=====

BUILD SUCCESSFUL (total time: 8 seconds)

BAB V

PENUTUP

5.1 Analisa

Dari hasil pratikum, pratikan dapat menjelaskan bahwasnya Konstruktor di Java adalah metode khusus yang digunakan untuk menginisialisasi objek. Konstruktor juga dapat mengambil parameter, yang digunakan untuk menginisialisasi atribut. Konstruktor dipanggil ketika objek kelas dibuat. Ini dapat digunakan untuk menetapkan nilai awal untuk atribut objek constructor dapat digunakan tanpa parameter, dan jika tidak menggunakan parameter, java telah menyediakan sebuah constructor yaitu constructor pada class sebelum class utama, constructor ini juga disebut dengan default constructor.

Pada pembuatan constructor, nama dari constructor harus sama dengan nama dari class yang digunakan untuk menampung constructor. Pada pembuatan constructor menggunakan parameter, pada bagian atribut harus menggunakan keyword this dikarenakan pada bagian tersebut menggunakan variabel yang sama dengan variabel yang terdapat pada class yang digunakan untuk membuat constructor.

5.2 Kesimpulan

1. Default constructor untuk class yang tidak memiliki parameter dan pernyataan pada bloknya.
2. Sedangkan untuk class yang sudah memiliki constructor, tidak akan diberikan constructor default.
3. Constructor tidak dapat diwariskan. Bila suatu method pada superclass bisa diwariskan pada subclassnya maka tidak demikian dengan constructor.
4. Menggunakan keyword static memudahkan kita untuk membuat sebuah program.
5. Keyword static digunakan pada atribut (variabel) dan method supaya program dapat dipanggil dengan menggunakan keyword static tanpa harus membuat objek baru.