Tipe Variabel

Variabel adalah ruang penyimpanan bernama yang dapat dimanipulasi oleh program kita. Setiap variabel di Java memiliki tipe tertentu, yang menentukan ukuran dan lokasi variabel pada memori; besar nilai yang dapat disimpan dalam memori tersebut; dan set operasi yang dapat diterapkan ke variabel.

Programmer harus mendeklarasikan semua variabel sebelum dapat digunakan.

Berikut adalah bentuk dasar dari deklarasi variabel :

```
data type variable [ = value][, variable [ = value] ...] ;
```

Di sini *tipe data* adalah salah satu tipe data yang dikenali di Java dan *variabel* adalah nama variabel. Untuk mendeklarasikan lebih dari satu variabel dari tipe data yang ditentukan, Anda dapat menggunakan daftar yang dipisahkan koma.

Berikut adalah contoh yang valid dari deklarasi variabel dan inisialisasi di Java -

Contoh

```
int a, b, c;          // Declares three ints, a, b, and c.
int a = 10, b = 10;   // Example of initialization
byte B = 22;          // initializes a byte type variable B.
double pi = 3.14159; // declares and assigns a value of PI.
char a = 'a';         // the char variable a iis initialized with
value 'a'
```

Bab ini akan menjelaskan berbagai tipe variabel yang tersedia dalam Bahasa Java. Ada tiga jenis variabel di Java -

- Variabel lokal
- Variabel instance
- Variabel Kelas / Statis

Variabel Lokal

- Variabel lokal dideklarasikan dalam method, konstruktor, atau blok.
- Variabel lokal dibuat ketika method, konstruktor atau blok dimasukkan dan variabel akan dihancurkan setelah keluar dari metode, konstruktor, atau blok.
- Pengubah akses tidak dapat digunakan untuk variabel lokal.
- Variabel lokal hanya terlihat dalam metode, konstruktor, atau blok yang dideklarasikan.
- Variabel lokal diimplementasikan pada tingkat tumpukan secara internal.
- Tidak ada nilai default untuk variabel lokal, jadi variabel lokal harus dideklarasikan dan nilai awal harus diberikan sebelum penggunaan pertama.

Contoh

Di sini, *age* adalah variabel lokal. Ini didefinisikan di dalam metode *pupAge ()* dan cakupannya terbatas hanya pada metode ini.

```
public class Test {
   public void pupAge() {
      int age = 0;
      age = age + 7;
      System.out.println("Puppy age is : " + age);
   }

   public static void main(String args[]) {
      Test test = new Test();
      test.pupAge();
   }
}
```

Ini akan menghasilkan hasil sebagai berikut -

Keluaran

```
Puppy age is: 7
```

Contoh

Contoh berikut menggunakan *age* tanpa melakukan inisialisasi, sehingga akan memberikan error pada saat kompilasi.

```
public class Test {
   public void pupAge() {
      int age;
      age = age + 7;
      System.out.println("Puppy age is : " + age);
   }

   public static void main(String args[]) {
      Test test = new Test();
      test.pupAge();
   }
}
```

Ini akan menghasilkan kesalahan berikut saat mengompilasinya -

Keluaran

```
Test.java:4:variable number might not have been initialized
```

```
age = age + 7;
          ^
1 error
```

Variabel Instance

- Variabel instance dideklarasikan di kelas, tetapi di luar method, konstruktor, atau blok apa pun.
- Saat sebuah ruang dialokasikan untuk objek di heap, slot untuk setiap nilai variabel instance dibuat.
- Variabel instance dibuat ketika sebuah objek dibuat dengan menggunakan kata kunci 'new' dan dihancurkan ketika objek tersebut dihancurkan.
- Variabel instance menyimpan nilai yang harus direferensikan oleh lebih dari satu method, konstruktor atau blok, atau bagian penting dari state/properties objek yang harus ada di pada kelas.
- Variabel instance dapat dideklarasikan di tingkat kelas sebelum atau setelah digunakan.
- Access Modifier dapat diberikan untuk variabel Instance.
- Variabel instance terlihat untuk semua metode, konstruktor, dan blok di kelas. Biasanya, disarankan untuk menjadikan variabel ini private (tingkat akses). Namun, visibilitas untuk subclass dapat diberikan untuk variabel ini dengan menggunakan access modifier.
- Variabel instance memiliki nilai default. Untuk angka, nilai defaultnya adalah 0, untuk Boolean adalah false, dan untuk referensi objek nilainya adalah nol. Nilai dapat ditetapkan pada saat deklarasi atau dalam konstruktor.
- Variabel instance dapat diakses secara langsung dengan memanggil nama variabel di dalam kelas. Namun, dalam metode statis (ketika variabel instance diberikan aksesibilitas), mereka harus dipanggil menggunakan nama yang sepenuhnya memenuhi syarat. *ObjectReference.VariableName* .

Contoh

Demo Langsung

```
import java.io.*;
public class Employee {

   // this instance variable is visible for any child class.
   public String name;

   // salary  variable is visible in Employee class only.
   private double salary;

   // The name variable is assigned in the constructor.
   public Employee (String empName) {
```

```
        name = empName;
    }

    // The salary variable is assigned a value.
    public void setSalary(double empSal) {
        salary = empSal;
    }

    // This method prints the employee details.
    public void printEmp() {
        System.out.println("name  : " + name );
        System.out.println("salary :" + salary);
    }

    public static void main(String args[]) {
        Employee empOne = new Employee("Ransika");
        empOne.setSalary(1000);
        empOne.printEmp();
    }
}
```

Ini akan menghasilkan hasil sebagai berikut -

Keluaran

```
name  : Ransika
salary :1000.0
```

Variabel Kelas / Static

- Variabel kelas yang juga dikenal sebagai variabel static, dideklarasikan dengan kata kunci static di dalam kelas, tetapi di luar metode, konstruktor, atau blok.
- Hanya akan ada satu salinan dari setiap variabel kelas per kelas, terlepas dari berapa banyak objek yang dibuat darinya.
- Variabel static jarang digunakan selain dideklarasikan sebagai konstanta. Konstanta adalah variabel yang dideklarasikan sebagai public / private, final, dan static. Variabel konstanta tidak pernah berubah dari nilai awalnya.
- Variabel static disimpan dalam memori statis. Sangat jarang menggunakan variabel static selain yang dinyatakan final dan digunakan sebagai konstanta public atau private.
- Variabel static dibuat saat program dimulai dan dimusnahkan saat program berhenti.

- Visibilitas mirip dengan variabel instance. Namun, sebagian besar variabel static dideklarasikan sebagai publik karena harus tersedia untuk pengguna kelas.
- Nilai default sama dengan variabel instan. Untuk angka, nilai defaultnya adalah 0; bagi Boolean, nilanya false; dan untuk referensi objek, nilainya null. Nilai dapat ditetapkan pada saat deklarasi atau dalam konstruktor. Selain itu, nilai dapat ditetapkan dalam blok penginisialisasi static khusus.
- Variabel statis dapat diakses dengan memanggil dengan nama kelas *ClassName.VariableName* .
- Saat mendeklarasikan variabel kelas sebagai final static public, maka nama variabel (konstanta) semuanya dalam huruf besar. Jika variabel static tidak public dan final, sintaks penamaan sama dengan variabel instance dan lokal.

Contoh

```
public class Employee {

   // salary  variable is a private static variable
   private static double salary;

   // DEPARTMENT is a constant
   public static final String DEPARTMENT = "Development ";

   public static void main(String args[]) {
      salary = 1000;
      System.out.println(DEPARTMENT + "average salary:" +
salary);
   }
}
```

Ini akan menghasilkan hasil sebagai berikut -

Keluaran

```
Development average salary:1000
```

Catatan - Jika variabel diakses dari kelas luar, konstanta harus diakses sebagai Employee.DEPARTMENT

Catatan :
variable:

[modifier] type namaVariable;

variable, berdasarkan letaknya bisa dibedakan menjadi:

- local variable / temporary / stack
    didefine di dalam method/constructor (baik sebagai parameter atau di dalam body method)
    hanya dapat digunakan di dalam method dimana variable didefine
    TIDAK DAPAT memiliki access modifier
    memerlukan inisialisasi secara explisit sebelum digunakan

- instance variable
    didefine di dalam class (di luar method)
    minimal dapat diakses dari dalam class yang sama (tergantung access modifier)
    otomatis akan diinisialisasi ke nilai default dari type-nya
        - byte,short,int,long 0
        - float,double 0.0
        - char ' '
        - reference null

variable berdasarkan type-nya, dapat dibedakan menjadi:
- primitive
    - integer: byte,short,int,long
    - floating: float,double
    - textual: char
    - logical: boolean
- reference
    String, Account, Wombat, ...

```
long ccNumber = 1111_1111_1111_1111L;
long phoneNumber = 62_817_644_8177;
double x = 1_000.00;
```