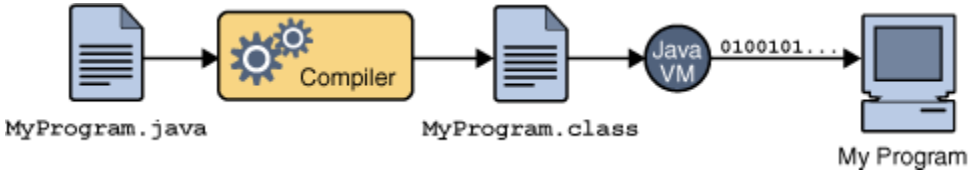


PEMROGRAMAN JAVA

- Petunjuk Penulisan Program
- Token
- Aturan Penamaan Identifier
- Lingkungan /Scope dari variabel
- Tipe Data (i)



Yoannita

Petunjuk Penulisan Program (i)

1. *Pada saat pembuatan blok, Anda dapat meletakkan kurung kurawal buka pada baris dengan pernyataan seperti contoh sebagai berikut ,*

```
public static void main( String[] args ){
```

atau Anda dapat meletakkan kurung kurawal pada baris selanjutnya, seperti,

```
public static void main( String[] args )
{
```

2. *Anda harus memberi jarak (indent) pernyataan selanjutnya setelah awal dari blok , seperti contoh berikut,*

```
public static void main( String[] args ){
    System.out.println("Hello");
    System.out.println("world");
}
```

Petunjuk Penulisan Program (ii)

1. *Untuk pemberian nama dari class Java,*
   diberikan huruf kapital untuk huruf pertama pada nama class.
   Untuk nama method dan variabel, huruf pertama dari kata harus dimulai dengan huruf kecil.
   Sebagi contoh:

```
ThisIsAnExampleOfClassName
thisIsAnExampleOfMethodName

class  BinatangAir
String  ikanAirTawar
void  membacaKoran()
```

2. *Pada kasus untuk identifier lebih dari satu kata, gunakan huruf kapital untuk mengindikasikan awal dari kata kecuali kata pertama.*

   Sebagai contoh charArray, fileNumber, ClassName.

3. *Sebaiknya hindari menggunakan underscores pada awal identifier seperti _read atau _write.*

Token

- **Token merupakan elemen terkecil di program yang mempunyai arti bagi kompilator.**

- Token Java dibagi 5, yaitu:
  - *Identifier*
  - *Keyword*
  - *Literal*
  - *Operator*
  - *Separator*

- Token : **Identifier**
  - *Identifier* adalah token yang merepresentasikan nama.
  - Dalam Java, *identifier* adalah nama yang diberikan untuk variable, class, atau method.
- Token : Reserved words/**Keywords**
  - Kata-kata yang dikenal oleh Java dan mempunyai arti khusus dlm program.
  - tidak boleh digunakan sebagai identifier (nama variabel, kelas, method, dll).

Ketentuan Identifier

- *Identifier* harus dimulai/diawali dengan huruf, underscore (_) atau tanda dollar ($).
- Untuk selebihnya dapat menggunakan karakter apapun, kecuali karakter yang digunakan sebagai operator oleh java.
- *Identifier* adalah *case sensitive* (membedakan huruf besar/ kecil) dan tidak ada batas maksimum.
- Bukan merupakan *keywords* yang dikenal Java :

abstract	default	if	package	synchronized
boolean	do	implements	private	this
break	double	import	protected	throw
byte	else	instanceof	public	throws
case	extends	int	return	transient
catch	final	interface	short	try
char	finally	long	static	void
class	float	native	super	volatile
continue	for	new	switch	while

Valid Identifier

- Valid
  - No
  - Kd_barang
  - kodeBarang
  - $barang
  - bahASA_PEmogramaN_JaVa
  - Contoh valid : int kd_barang; string $barang; class No

- Invalid
  - 1_harga - tidak boleh diawali dengan angka
  - %barang - simbol % tidak boleh digunakan
  - Nama pelanggan - tidak boleh menggunakan spasi
  - Kd+barang - tidak boleh menggunakan tanda operator

Contoh invalid: int 1_harga; int super; double void; class throws

Valid Identifier

- Sebagai pengganti bentuk penulisan tradisional untuk identifier, yang menggunakan huruf kecil dengan garis-bawah sebagai pemisah kata
  - nama_karyawan
  - Nama_Karyawan

- Beberapa identifier dituliskan dengan huruf kapital pada awal kata, seperti :
  - string namaKaryawan atau
    class NamaKaryawan

Token : Literal

Token : Literal

- Penulisan besaran untuk variabel.
- Literal Java terdiri dari angka, karakter, dan string. Angka terdiri dari bilangan bulat (*integer*), bilangan mengambang (*floating point*), dan *boolean*.
- Nilai *boolean* untuk *true* dan *false* direpresentasikan sebagai 1 dan 0.

Token : Operator

- Operator menspesifikasikan evaluasi atau komputasi terhadap objek.
- Operand yang dioperasikan dapat berupa literal, variabel, atau nilai yang dikirim oleh metode atau fungsi.

Token : Separator

- *Separator* digunakan untuk menginformasikan ke kompilator Java mengenai adanya pengelompokkan di kode program.

Simbol	Nama	Penggunaan
( )	Kurung	Untuk menghimpun parameter dalam definisi dan pemanggilan method, juga digunakan untuk menyatakan tingkat pernyataan, menghimpun pernyataan, untuk pengaturan alur program, dan untuk menyatakan tipe cast (cast types)
{ }	Kurung Kurawal	Untuk menghimpun nilai yang otomatis dimasukkan ke dalam array, digunakan juga untuk mendefinisikan blok program, untuk cakupan class, method, dan lokal.
[ ]	Kurung Siku	Untuk menyatakan tipe array dan untuk membedakan nilai array.
;	Titik Koma	Pemisah Pernyataan.
,	Koma	Pemisah urutan identifier dalam deklarasi variabel.
.	Titik	Untuk memisahkan nama paket dari sub-paket dan class dan untuk memisahkan variabel atau method dari variabel referensi.

Lingkupan (Scope) dari Variabel

```
class Scope
{
    static int a = 2;  // deklarasi variabel dalam blok class
    public static void main (String[] args)
    {
        int x = 10;  // x dikenal di seluruh method main()
        {            // awal dari blok baru
            int y = 5;
            // y hanya dikenal dalam blok kode ini saja
        }
    }
}
```

Variabel

- → lokasi penyimpanan yang ada di memori
- Setiap variabel memiliki kemampuan menyimpan suatu informasi sesuai dengan tipe data yang dideklarasikan untuk variabel tersebut.
  - tipe-data nama-variabel;
  - tipe-data nama-variabel = <pengisian nilai variabel>;

```
String teks;          // pendeklarasian variabel
                      // kode di atas mendeklarasikan suatu variabel bernama teks
                            dengan tipe data string
                      //variabel teks hanya dapat menampung data berupa String
String teks = “Mine”; // pendeklarasian dan pemberian nilai pd variabel
```

```
Char karakter;
Char karakter = ‘A’;
```

```
int counter;
Int counter = 3;
```

Tipe Data

Tipe data diperlukan agar kompiler tahu *operasi apa yang valid dan seberapa banyak memory yang diperlukan* oleh sebuah nilai yang akan disimpan atau dioperasikan.

Variabel :

- menampung suatu nilai
- Pasti memiliki tipe data
- Harus didefinisikan terlebih dahulu sebelum digunakan

- Tipe Data Primitif
  - Integer
  - Floating – Point
  - Karakter
  - Boolean
  - ….
- Tipe Data Referensi

Tipe Data

Tipe Data Integer (Bilangan Bulat)

Tipe	Nilai Minimum	Nilai maksimum
Byte	-128	127
Short	-32,768	32,767
Int	-2,147,483,648	2,147,483,647
Long	-9,223,372,036,854,775,808	9,223,372,036,854,775,808

Tipe Data Boolean

Nilai : *true* atau *false*

Tipe Data Real/ Floating Point (Bilangan Pecahan)

Tipe	Nilai Minimum	Nilai maksimum
Float	1.40239846e-45f	3.40282347e+38f
Double	4.94065645841246544e-324d	1.7976931348623157e+308d

Tipe Data

Tipe Data Karakter

- **char** :
  - hanya dapat menampung satu karakter saja,
  - Menggunakan tanda petik satu
- **String** :
  - dapat menampung kalimat.
  - Menggunakan tanda petik dua

```
class SalahCharString
{
    public static void main(String args[])
    {
        char c1 = 'A';
        char s1 = "Belajar Java menyenangkan";
    }
}
```

Seharusnya String

Berikut pesan kesalahan saat kompilasi :

SalahCharString.java [11:1] incompatible types
found : java.lang.String
required: char
char s1 = "Belajar Java menyenangkan";
^
1 error
Errors compiling SalahCharString

Contoh kode program

Program dengan Variabel

```
// nama file: data1.java
// deskripsi: program menampilkan nilai
// kompilasi: javac data1.java
// eksekusi: java data1

class data1 {
    public static void main (String [ ] args) {
        double nilai1 = 13.14;
        char nilai2;
        nilai2 = 'B';
        System.out.println ("tipe data double : " + nilai1);
        System.out.println ("tipe data char   : " + nilai2);
    }
}
```

Konstanta

- Merupakan variabel yang memiliki nilai tetap dan tidak dapat diubah saat program sedang berjalan. Untuk menjadikan sebagai variabel konstanta, cukup menambahkan kata tercadang *final*

```
// nama file : konstanta2.java
// deskripsi : perubahan konstanta
// kompilasi : javac konstanta2.java
// eksekusi : java konstanta2

class konstanta2
{
    public static void main(String args[])
    {
        final int hrg = 3500;
        int jlh = 10;
        System.out.println("total = " + (jlh * hrg));
        hrg = 250;
        System.out.println("total = " + (jlh * hrg));
    }
}
```

contoh usaha untuk mengubah nilai konstanta :

Berikut ini tampilan saat kompilasi

```
konstanta2.java [13:1] cannot assign a value to final
        variable hrg
hrg = 250;
^
1 error
Errors compiling konstanta2.
```

Terjadi kesalahan saat kompilasi, yaitu pada baris 13 terjadi usaha untuk mengubah nilai variabel hrg.

Latihan

Mendeklarasikan dan mencetak variabel

- Diberikan tabel dibawah ini, deklarasikan variabel yang terdapat didalamnya dengan tipe data yang sesuai dan berikan nilai inisialisasi(nilai awal). Tampilkan hasil outputnya yaitu nama variabel dan nilainya.

Nama Variabel	*Tipe Data*	*Nilai Awal*
number	integer	10
letter	character	a
result	boolean	true
str	String	hello

Berikut ini merupakan tampilan yang diharapkan sebagai hasil eksekusi program:

```
number = 10
letter = a
result = true
str = hello
```