Function

Fungsi pada Java dideklarasikan di dalam sebuah kelas. Fungsi pada Java bertugas untuk mengembalikan nilai. Sedangkan untuk fungsi yang tidak mengembalikan nilai (void) disebut sebagai prosedur. Baik fungsi maupun prosedur di dalam suatu kelas kita sebut sebagai metode. Metode merupakan aspek yang penting di dalam Java. Dalam sebuah kelas bisa terdapat banyak metode sesuai dengan kegunaannya masing-masing.

Mendeklarasikan Fungsi

Cara membuat sebuah fungsi di dalam kelas adalah sebagai berikut :

```
1. modifier returnType nameOfFunction(parameters) {
2.
3. }
```

- modifier menunjukkan sifat yang dimiliki pada suatu fungsi seperti public, private, protected.
- returnType merupakan nilai balik yang diberikan oleh fungsi. Apabila fungsi tidak memiliki nilai balik maka menggunakan void.
- nameOfFunction adalah nama dari sebuah fungsi.
- parameters bersifat opsional, Suatu fungsi dapat mempunyai banyak parameters atau pun tidak sama sekali.

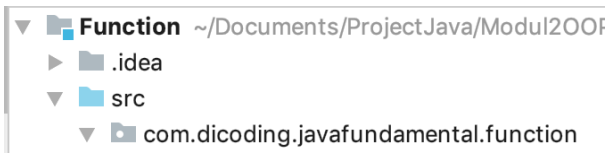
Memanggil Fungsi

Berikut adalah contoh cara mengimplementasikan fungsi secara sederhana di dalam sebuah kelas secara `statis`. Artinya penggunaan fungsi ini hanya dilakukan pada kelas tersebut tanpa membuat objek.

Codelab Memanggi Fungsi

Mari kita coba memanggil fungsi di dalam program.

1. Buatlah proyek baru dengan nama Function dengan nama package `com.dicoding.javafundamental.function` di dalamnya:



2. Buatlah sebuah kelas baru di dalamnya dengan nama `CallFunction`, kemudian tambahkan kode berikut:

```java
package com.dicoding.javafundamental.function;

public class CallFunction {

    public static void main(String[] args) {
        // memanggil fungsi
        cobaFungsi();
    }

    public static void cobaFungsi() {
        System.out.println("Ini merupakan bagian dari fungsi");
    }
}
```

3. Selanjutnya jalankanlah kode di atas pada IDE yang kalian gunakan. Bila sukses, seharusnya Console akan menampilkan output seperti ini.

```
Ini merupakan bagian dari fungsi
```

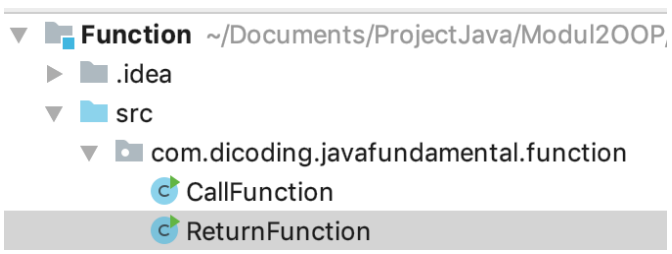
Fungsi dengan nilai balik

Jika contoh kode sebelumnya tanpa nilai balik, lalu bagaimana untuk fungsi yang mengirimkan/menghasilkan nilai balik? Fungsi yang yang mengirimkan/menghasilkan nilai balik selalu mendeklarasikan tipe data kemudian diakhiri dengan `return`.

Codelab Fungsi dengan Nilai Balik

Mari kita coba memanggil fungsi dengan nilai balik di dalam program.

1. Bukalah kembali proyek Function dan buatlah kelas baru dengan nama `ReturnFunction`.



2. Masukkan kode berikut ke dalam kelas `ReturnFunction`:

```java
package com.dicoding.javafundamental.function;

public class ReturnFunction {
    public static void main(String[] args) {
        double p = 7;
        double l = 6.5;
        double hasil = hitungLuas(p, l); //memanggil fungsi
        System.out.println("Hasilnya adalah = " + hasil);
    }

    //fungsi dengan nilai balik
    public static double hitungLuas(double panjang, double lebar){
        double luas = panjang * lebar;
        return luas;
    }
```

3. Jalankan kode di atas maka hasilnya akan jadi seperti ini:

```
Hasilnya adalah = 45.5
```

Catatan : Penamaan suatu fungsi dapat disesuaikan sesuai kebutuhan. Namun, secara umum penamaan sebuah fungsi diawali dengan huruf kecil dan diikuti dengan huruf besar setelahnya. Hal ini biasa disebut dengan notasi ”punuk unta” seperti contoh:

```
luas
hitungKeliling
hapusUser
rataNilaiMahasiswa
jumlahKomentarNetizen
jumlahLaguYoungLex
```

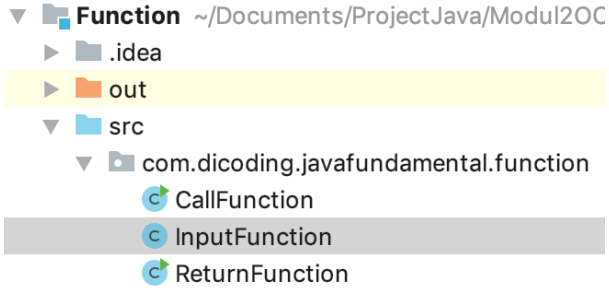
Fungsi dengan parameter

Fungsi ini bertujuan untuk menghitung luas persegi panjang dengan menggunakan parameter.

Codelab Fungsi dengan Parameter

Mari kita praktikan agar semakin paham tentang pemanggilan fungsi dengan parameter.

1. Bukalah kembali proyek `Function` dan buatlah kelas baru dengan nama `InputFunction`.



2. Masukkan kode berikut ke dalam kelas `InputFunction`:

```
1.  package com.dicoding.javafundamental.function;
2.
3.  public class InputFunction {
4.      public static void main(String[] args) {
5.          // memanggil fungsi
6.          hitungLuas(7, 6.5);
7.      }
8.
9.      public static void hitungLuas(double panjang, double lebar) {
10.         double luas;
11.         luas = panjang * lebar;
12.         System.out.println(luas);
13.     }
14. }
```

3. Jalankan kode di atas maka hasilnya akan jadi seperti ini:

```
45.5
```

Bedah Kode Fungsi dengan Parameter

Perhatikan kode berikut:

```
1.  public static void hitungLuas(double panjang, double lebar) {
2.      double luas;
3.      luas = panjang * lebar;
4.      System.out.println(luas);
5.  }
```

Pada kode di atas kita membuat fungsi `hitungLuas` dengan menggunakan paramater serta tidak memberikan nilai balik. Maka dari itu kita menggunakan `void`. Kemudian fungsi dipanggil dengan memberikan nilai pada masing-masing parameter serta dipisahkan oleh tanda koma (,).

Method Overloading

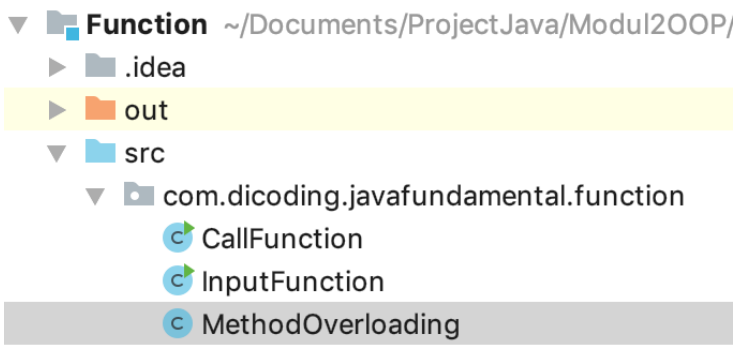
Java mengizinkan menggunakan dua atau lebih fungsi dengan nama yang sama dalam satu kelas. Namun, yang membedakan adalah parameternya. Konsep ini disebut dengan *method overloading*. Sebagai contoh kita membuat fungsi dengan nama yang sama pada kode sebelumnya dengan membedakan tipe datanya menjadi

int.

Codelab Method Overloading

Mari kita praktikan agar semakin paham tentang Method Overloading.

1. Bukalah kembali proyek Function dan buatlah kelas baru dengan nama `MethodOverloading`.



2. Masukkan kode berikut ke dalam kelas `MethodOverloading`:

```
1.  package com.dicoding.javafundamental.function;
2.
3.  public class MethodOverloading {
4.      public static void main(String[] args) {
5.          double p = 7;
6.          double l = 6.5;
7.          double hasil = hitungLuas(p, l);
8.          System.out.println("Hasilnya adalah = " + hasil);
9.          int pn = 7;
10.         int lb = 6;
11.         int hsl = hitungLuas(pn, lb);
12.         System.out.println("Hasilnya adalah = " + hsl);
13.     }
14.
15.     public static double hitungLuas(double panjang, double lebar) {
```

3. Jalankan kode di atas maka hasilnya akan jadi seperti ini:

```
Hasilnya adalah = 45.5
Hasilnya adalah = 42
```

Kini kita telah mengetahui cara mendeklarasikan dan menggunakan sebuah fungsi. Dengan menggunakan fungsi maka kita bisa jauh lebih efisien dalam menulis kode . Tidak lagi menulis secara berulang-ulang. Pada tahap selanjutnya kita akan belajar membuat fungsi pada kelas dan memanggilnya dengan menggunakan objek.

Sebelumnya

Selanjutnya

Dicoding Space
Jl. Batik Kumeli No.50, Sukaluyu,
Kec. Cibeunying Kaler, Kota Bandung
Jawa Barat 40123

Penghargaan

image
click bila
belum muncul

image
click bila
belum muncul

Decode Ideas
Discover Potential

Tentang Kami

Blog
Reward
Showcase

Hubungi Kami
FAQ