3️⃣

Modul 3 - Modifier dan Encapsulation

Durasi Modul = 2-5 menit

Capaian pembelajaran:

Setelah mempelajari modul ini, mahasiswa diharapkan mampu:

- Memahami konsep dan penggunaan access modifier dalam pemrograman berorientasi objek
- Menjelaskan perbedaan antara public, private, protected, dan default access modifier
- Mengimplementasikan access modifier yang tepat pada class, method, atribut, dan constructor
- Memahami konsep encapsulation dan pentingnya dalam pemrograman berorientasi objek
- Menerapkan prinsip encapsulation dalam pembuatan class dengan menggunakan private atribut dan public getter/setter methods

Deskripsi Pembelajaran

Modul ini membahas dua konsep penting dalam pemrograman berorientasi objek: modifier dan encapsulation.

Modifier:

1. Pengenalan tentang access modifier dalam Java dan fungsinya dalam mengontrol akses ke elemen-elemen program.
2. Pembahasan detail tentang empat jenis access modifier: public, private, protected, dan default (package-private).
3. Penjelasan tentang penggunaan modifier pada class, method, atribut, dan constructor.

4. Contoh-contoh kode untuk mengilustrasikan penggunaan modifier yang berbeda dan dampaknya terhadap aksesibilitas.

Encapsulation:

1. Pengenalan konsep encapsulation dan pentingnya dalam pemrograman berorientasi objek.
2. Penjelasan tentang cara mengimplementasikan encapsulation menggunakan private atribut dan public getter/setter methods.
3. Diskusi tentang manfaat encapsulation dalam menjaga integritas data dan menyembunyikan detail implementasi.
4. Contoh-contoh praktis penggunaan encapsulation dalam pembuatan class.

Melalui pembelajaran ini, mahasiswa akan memperoleh pemahaman mendalam tentang bagaimana menggunakan modifier dan encapsulation untuk menciptakan kode yang lebih aman, terstruktur, dan mudah dipelihara.

3.1. Access Modifier

Access modifier mengatur level akses sebuah class, method, atribut hingga constructor. Berikut penjelasan mengenai access modifier dan level aksesnya:

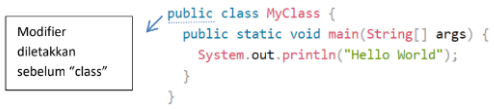
Modifier	Class	Package	Subclass	World
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no modifier	Y	Y	N	N
private	Y	N	N	N

3.1.1. Access Modifier – Class

▼ Untuk **Class**, hanya ada 2 jenis access modifier, yaitu:

a. *Public Modifier*
Jika modifier sebuah class adalah "public", maka class ini dapat diakses oleh semua level, hingga class yang berada di luar package-nya.

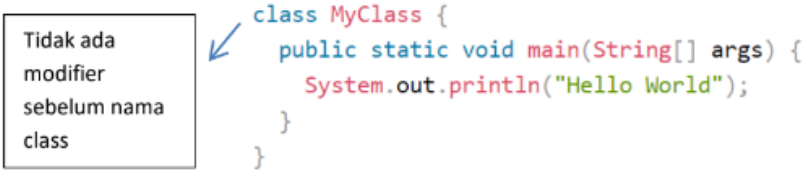
Contoh penggunaan modifier public pada sebuah class:



Mengacu pada program di samping, class MyClass memiliki modifier public, sehingga class ini akan dapat diakses dan digunakan oleh class-class lain di level apapun.

b. *No Modifier atau Default Modifier*
Jika modifier sebuah class adalah **default atau tidak terdapat modifier,** maka class ini hanya dapat diakses oleh class-class yang berada package yang sama.

Contoh penggunaan modifier default pada sebuah class:



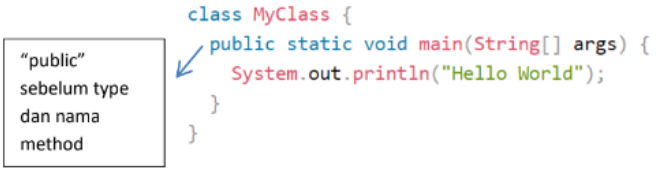
3.1.2. Access Modifier – Method, Attribut & Constructor

Untuk method, atribut dan constructor, ada 3 jenis modifier yang dapat digunakan, yaitu

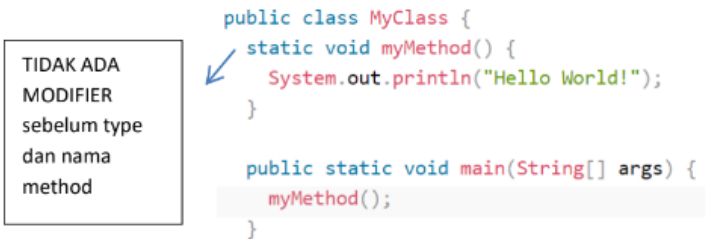
public, default modifier, protected dan private. Masing-masing level akses untuk modifier ini
telah dijelaskan seperti gambar bagian Level di atas.

▼ 3.1.2.1. Method

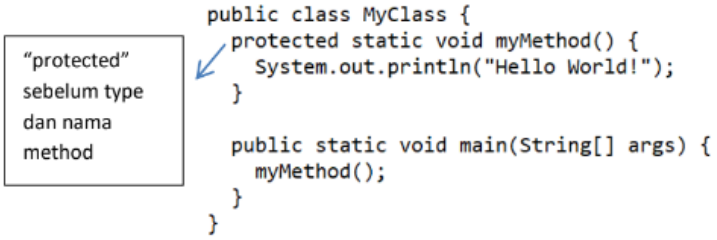
a. Public



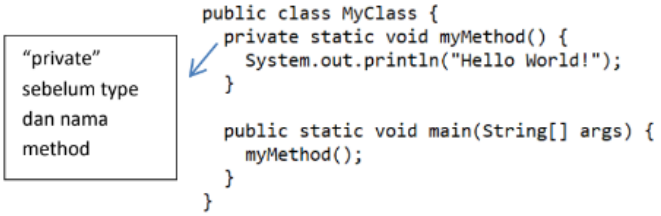
b. Default Modifier



c. Protected

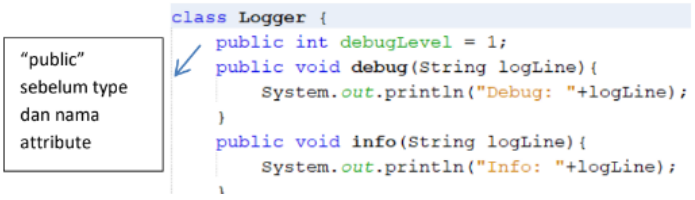


d. Private

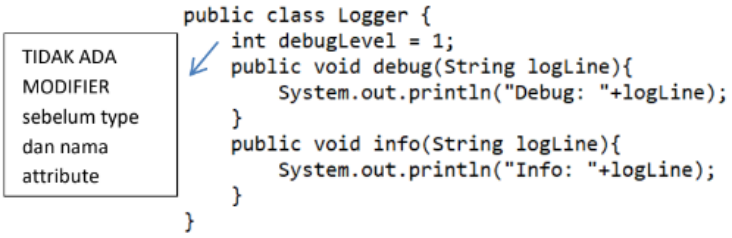


▼ 3.1.2.2. Attribute

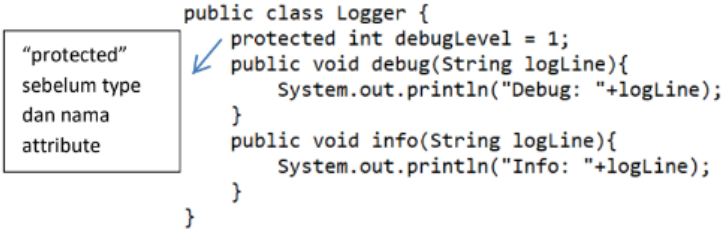
a. Public



b. Default Modifier



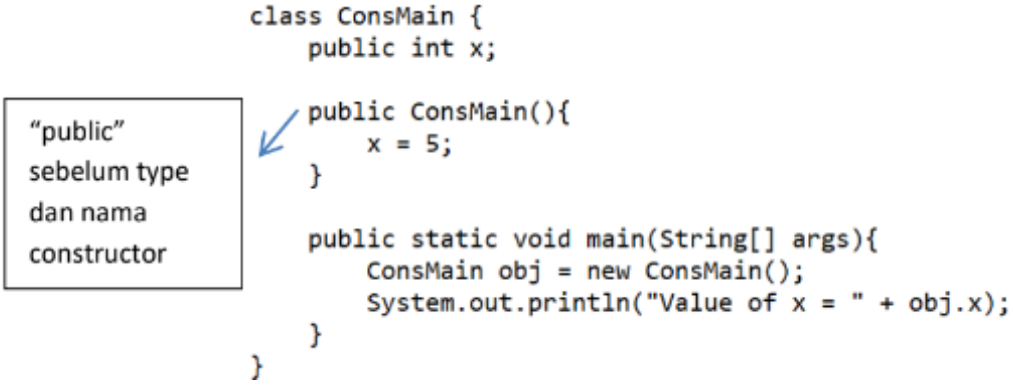
c. Protected



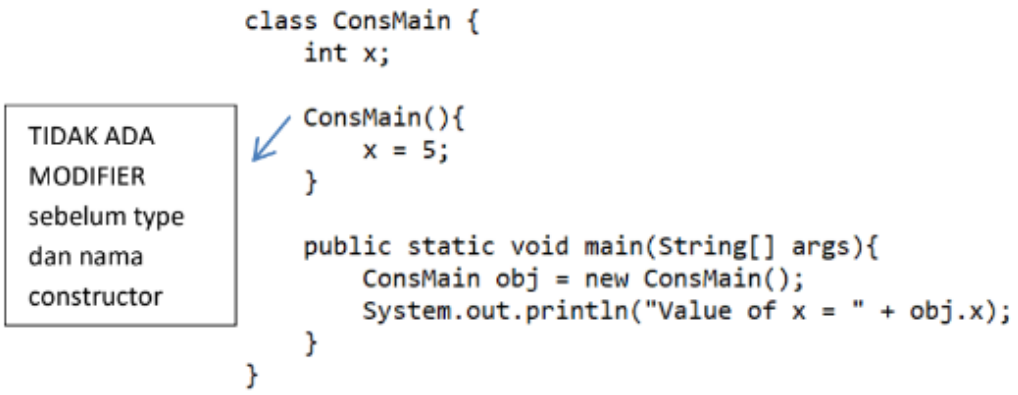
d. Private

▼ 3.1.2.3. Constructor

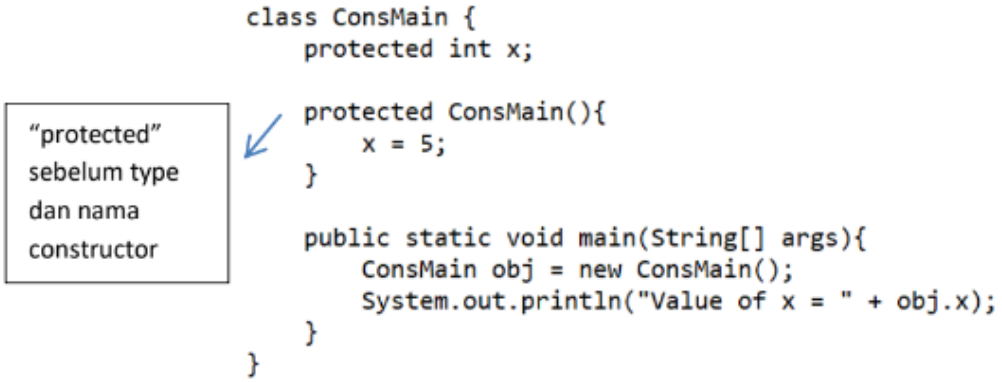
a. Public



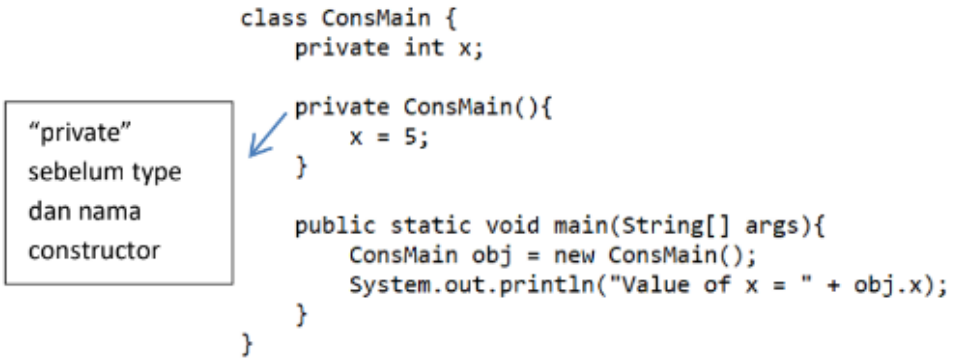
b. Default Modifier



c. Protected



d. Private



3.2. Encapsulation

Encapsulation adalah sebuah cara untuk membuat data/atribut yang "sensitif" tersembunyi dan tidak dapat diakses oleh class lain.

📌 Untuk dapat membuat sebuah class yang di-encapsulate, berikut beberapa hal yang harus ada pada class tersebut:
a. Atribut-atribut class yang dideklarasikan sebagai **“private”**
b. Menyediakan **“public”** *set* dan *get method*. **Set method** untuk memberi nilai pada private atribut
dan **get method** untuk mengambil nilai atribut private.

💡 ***Method Mutator (setter)***
Method Mutator (Setter)
adalah sebuah method yang digunakan untuk mengisi nilai/isi sebuah attribute kelas yang bersifat private.

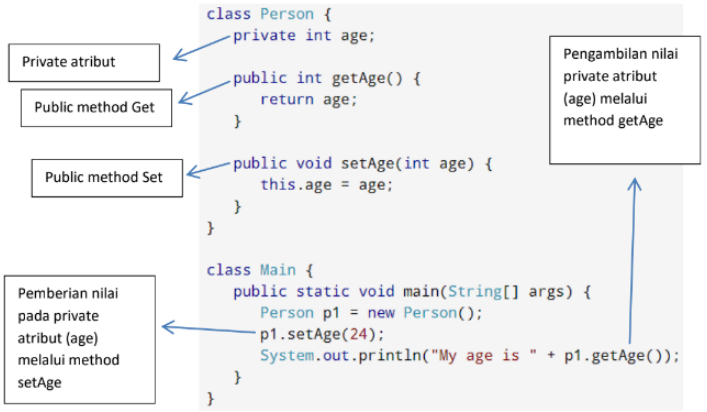
▼ Ciri-ciri method setter:

- Memiliki access modifier yang dapat diakses dari luar kelas tersebut.
- Memiliki sebuah parameter sebagai value untuk attribute yang diset.
- Nama method diawali dengan “set” diikuti namaattribute yang diawali huruf besar(umumnya).

💡 ***Method Assesor (getter)***
Method Assesor (getter) adalah sebuah method yang digunakan untuk mengambil nilai/isi sebuah attribute kelas yang bersifat private.

▼ Ciri-ciri mehod getter :

- Memiliki access modifier yang dapat diakses dari luar kelas tersebut.
- Memiliki return value.
- Nama method diawali dengan “get” diikuti nama attribute yang diawali huruf besar (umumnya).



Manfaat dari enkapsulasi:

- **Penyembunyian Informasi *(information hiding)***

  Manfaat utamanya adalah bagian internal dapat berubah tanpa mempengaruhi bagian-bagian program yang lain.

- **Modularitas**

  Modularitas berarti objek dapat **dikelola secara independen**. Karena kode sumber bagian internalobjek dikelola secara terpisah dari antarmuka, maka **Kita bebas melakukan modifikasi yang tidak menyebabkan masalah pada bagian-bagian lain dari sistem.** Manfaat ini mempermudah mendistribusikan objek-objek dari sistem.

> **Salah satu bentuk dari enkapsulasi** adalah *field encapsulation*. *Field encapsulation* merupakan **salah satu teknik OOP untuk menghindarkan pengaksesan secara langsung terhadap isi dari attribute/field sebuah kelas tertentu dari kelas lainnya.** Encapsulasi jenis ini dapat dilakukan dengan memberikan hak akses private pada setiap field/attribute yang akan dienkapsulasi, sehingga tidak dapat diakses dari luar kelas tersebut. **Jika attribute diberikan hak akses private, maka bagaimana mengaksesnya?** Untuk mengakses attribute yang di private dapat dilakukan dengan menggunakan method setter – getter atau dengan method toString (yang diberikan modifier public). **Dalam OOP, getter dan setter merupakan implementasi dari enkapsulasi.**

📚REFERENSI

Aziefah, & Nurmasari, J. (2013). Pemrograman Berorientasi Objek. Pekanbaru: Politeknik Caltex Riau.

Barclay, K., & Savage, J. (2004). Object Oriented Design with UML and Java. Elsevier.

Deitel, P., & Deitel, H. (2012). Java: How to Program (9 ed.). US: Prentice Hall.