KUSUMA WEB ACADEMY

DASAR-DASAR JALUR JAVA

Modul 11: Pilar OOP 1 - Enkapsulasi & Keamanan Data

Konsep Enkapsulasi: Pembungkusan Data & Informasi

Apa itu Enkapsulasi (Encapsulation)?

- Mekanisme menyembunyikan data sensitif (variabel internal) dari akses luar langsung, sekaligus menyatukannya dengan method pengolahnya.
- **Filosofi:** Melindungi data internal agar tidak dimodifikasi secara sembarangan atau dirusak dari luar sistem secara tidak sah.

Cara Kerja di Java

1. Deklarasikan variabel class menggunakan modifier akses **`private`** agar tidak dapat disentuh oleh kelas lain.
2. Sediakan gerbang akses terkontrol berupa method publik bernilai **Getter** (membaca data) dan **Setter** (mengubah data).

Keuntungan Enkapsulasi

Meningkatkan keamanan data, fleksibilitas pemeliharaan, serta kemudahan dalam mengubah implementasi internal tanpa memutus aplikasi eksternal.

Memahami Tingkat Hak Akses (Access Modifiers)

Terdapat
4 jenis hak akses bawaan di Java yang mengatur visibilitas Class, Variabel, dan Method:

Modifier	Class yang Sama	Package yang Sama	Subclass (Anak)	Dun
`private`	Ya	Tidak	Tidak	
no modifier (Default)	Ya	Ya	Tidak	
`protected`	Ya	Ya	Ya	
`public`	Ya	Ya	Ya	

Prinsip Hak Akses Minimum

Selalu gunakan level akses yang paling ketat terlebih dahulu (`private`) kecuali jika Anda memiliki alasan khusus untuk membukanya.

Implementasi Getter & Setter dengan Validasi Data

Mengapa Setter Lebih Aman?

- Di dalam method Setter, Anda dapat menyisipkan logika penyaringan (validasi) untuk mencegah masuknya data sampah atau data ilegal ke memori sistem.

Aturan Format Penulisan:

- **Getter:** `public tipeData getVariabel()`
- **Setter:** `public void setVariabel(tipe parameter)`

```
public class Akun {
    private int saldo; // Tersembunyi

    // Getter
    public int getSaldo() {
        return this.saldo;
    }

    // Setter dengan filter keamanan
    public void setSaldo(int saldoBaru)
         {
        if (saldoBaru >= 0) {
            this.saldo = saldoBaru;
        } else {
            System.out.println("Error:
                Saldo minus!");
        }
    }
}
```

Contoh Kasus Penggunaan Enkapsulasi di Dunia Nyata

```java
public class RekeningBank {
    private String namaPemilik;
    private double saldo;

    public RekeningBank(String namaPemilik, double saldoAwal) {
        this.namaPemilik = namaPemilik;
        this.saldo = (saldoAwal > 0) ? saldoAwal : 0;
    }

    public void setorUang(double jumlah) {
        if (jumlah > 0) this.saldo += jumlah;
    }

    public double getSaldo() {
        return this.saldo; // Hanya bisa dibaca, tidak bisa diganti paksa
    }
}
```