ABSTRACT AND INTERFACE CLASS

:: Materi – 7 ::

Section 01

ABSTRACT

Konsep

Menurut kamus, **abstraksi** adalah kualitas menangani ide, bukan peristiwa.

Misalnya:

> Ketika mempertimbangkan kasus email, detail rumit seperti apa yang terjadi setelah mengirim email, protokol yang digunakan server email disembunyikan dari pengguna. Oleh karena itu, untuk mengirim email hanya perlu mengetikkan isinya, menyebutkan alamat penerima dan klik kirim.

Konsep

Dalam pemrograman berorientasi objek, abstraksi adalah proses menyembunyikan detail implementasi dari pengguna, hanya fungsionalitasnya yang akan diberikan kepada pengguna.

Dengan kata lain, pengguna hanya akan mempunyai informasi tentang apa yang dilakukan objek dan bukan bagaimana objek tersebut melakukannya.

Achieve Abstraction

Ada dua cara untuk mencapai *abstraction* di java, yaitu:

Contoh Kehidupan Nyata

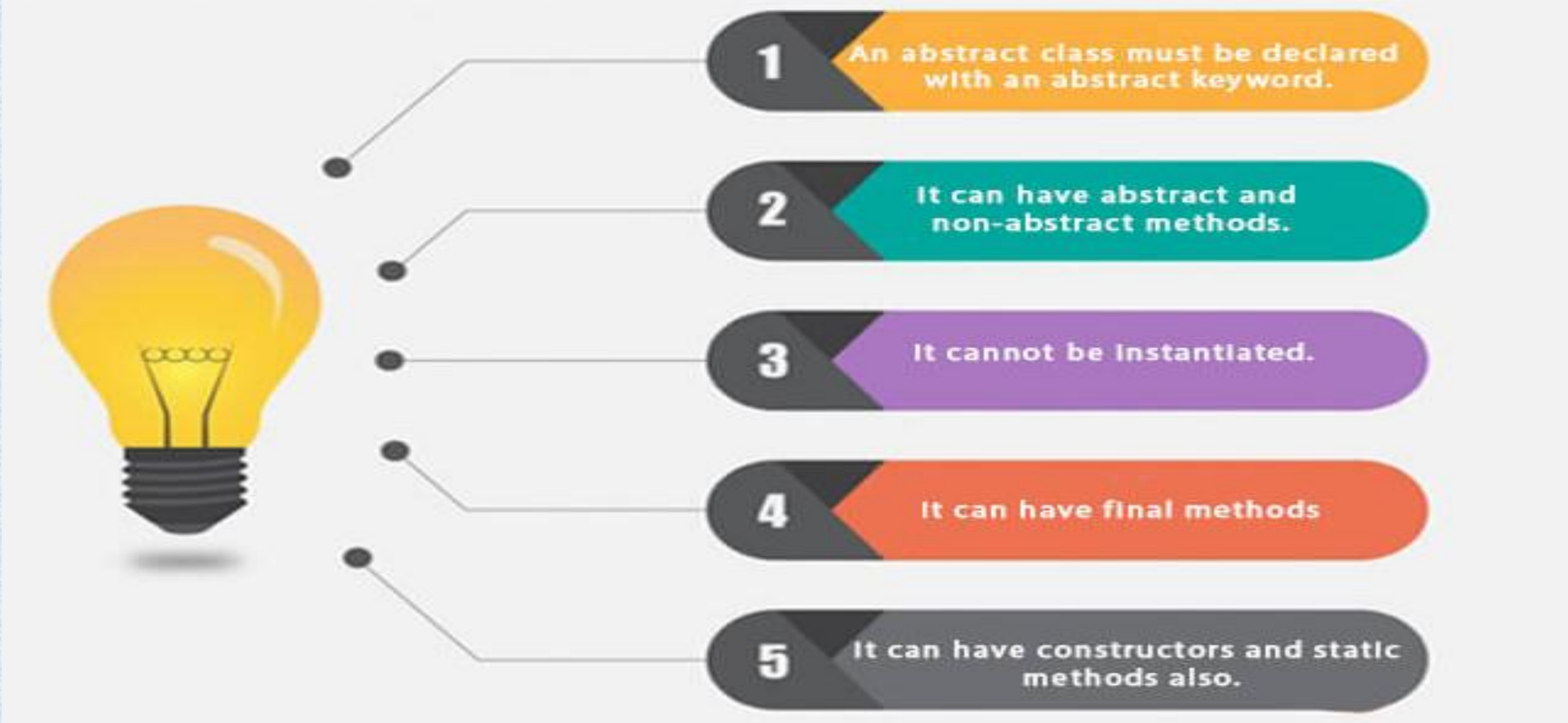
Seorang pria ataupun wanita yang mengendarai sebuah mobil.

Orang tersebut hanya mengetahui bahwa menekan pedal gas akan menambah kecepatan mobil atau mengerem akan menghentikan mobil.

namun dia tidak mengetahui tentang mekanisme bagian dalam mobil tersebut atau penerapan pedal gas, rem, dan lainnya pada mobil tersebut.

Inilah yang dimaksud dengan abstraksi.

Rules for java abstract class



Example

Abstract class that has an abstract method

- ✓ Dalam contoh di samping, Bike adalah kelas abstrak yang hanya berisi satu metode abstrak yang dijalankan.
- ✓ Implementasinya disediakan oleh kelas Honda.

```
abstract class Bike {

    abstract void run();
}

public class Honda extends Bike {

    @Override
    void run() {
        System.out.println("running safely");
    }

    public static void main(String args[]) {
        Bike obj = new Honda();
        obj.run();
    }
}
```

Section 02

INTERFACE

Konsep

- ✓ *Interface* adalah sebuah blueprint dari suatu kelas.
- ✓ *Interface* memiliki konstanta statis dan metode abstrak.
- ✓ *Interface* adalah mekanisme untuk mencapai abstraksi. Hanya ada metode abstrak di *Interface* pada Java yang digunakan untuk mencapai abstraksi dan pewarisan berganda atau ***multiple inheritance***.

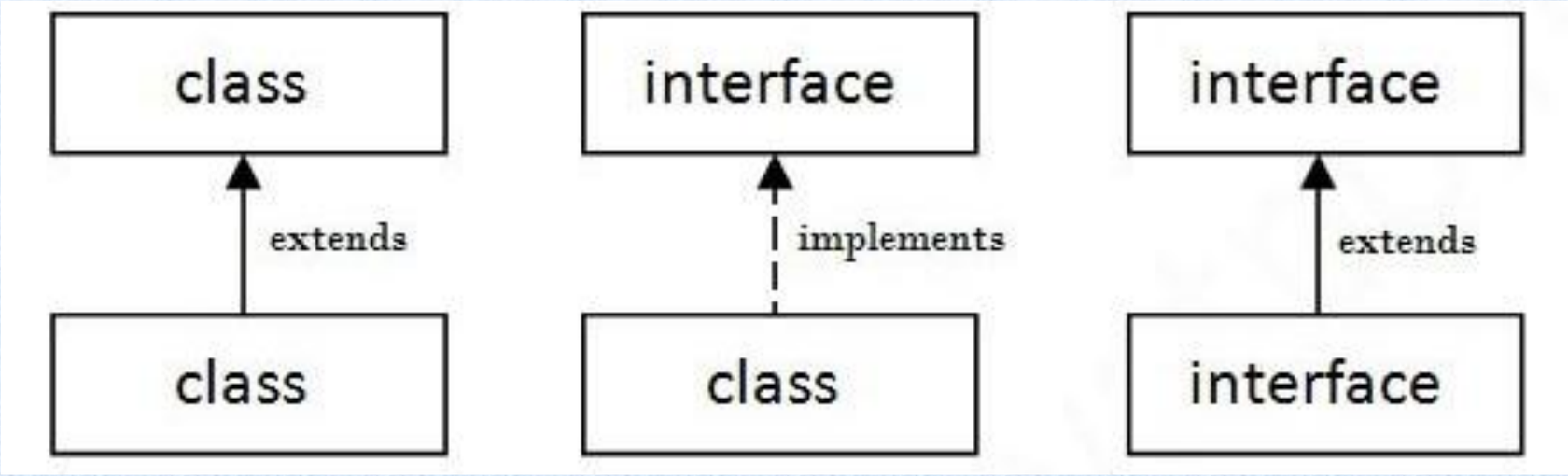
Mengapa menggunakan antarmuka Java?

How to declare an interface?

- ✓ Sebuah interface dideklarasikan dengan menggunakan kata kunci interface.
- ✓ Ini memberikan abstraksi secara total, berarti semua metode dalam interface dideklarasikan dengan isi kosong dan semua atribut bersifat public, static dan final secara default.
- ✓ Sebuah kelas yang mengimplementasikan sebuah interface harus mengimplementasikan semua metode yang dideklarasikan dalam interface.

Hubungan antara class dan interface?

Seperti yang ditunjukkan pada gambar di bawah, sebuah kelas memperluas kelas lain, sebuah *interface* memperluas interface lain, tetapi sebuah kelas mengimplementasikan sebuah *interface*.

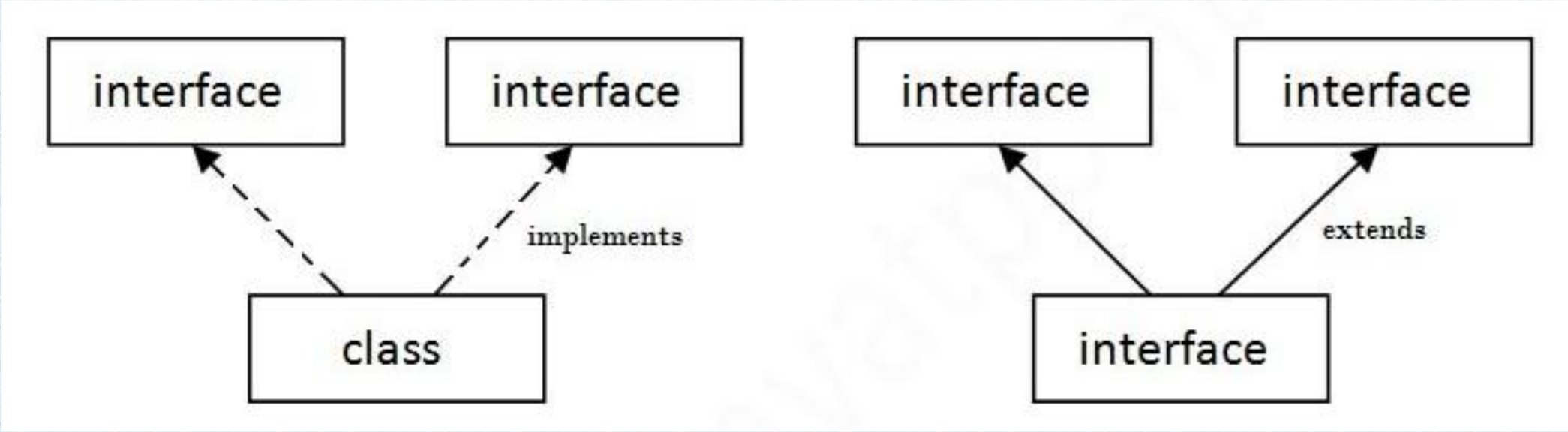


Example


```
interface printable {

    void print();
}

public class TI implements printable {

    @Override
    public void print() {
        System.out.print("Teknik Informatika");
        System.out.println(" Politeknik Hasnur");
    }

    public static void main(String args[]) {
        TI obj = new TI();
        obj.print();
    }
}
```

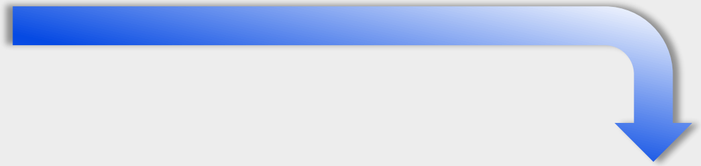
Multiple inheritance in Java by interface

Jika suatu class mengimplementasikan *multiple interface* atau suatu *interface* ***extends*** *multiple interface maka,* ini dikenal sebagai ***multiple inheritance***.



Example


```
interface printable1 {

    void print();
}

interface Showable {

    void show();
}

public class TI1 implements printable1, Showable {

    @Override
    public void print() {
        System.out.println(":: Politeknik Hasnur ::");
    }
```



```
    @Override
    public void show() {
        System.out.println("Teknik Informatika");
        System.out.println("Teknik Otomotif");
        System.out.println("Budidaya Tanaman Perkebunan");
    }

    public static void main(String args[]) {
        TI1 obj = new TI1();
        obj.print();
        obj.show();
    }
}
```

That’s all. Thank you very much! ☺

Any Questions?