

CLASS ABSTRACT & INTERFACE



APA.....?????

- Mendeklarasikan karakteristik yang umum dari subclass.
- Dideklarasikan secara abstrak.
- Tidak dapat dibuat objeknya dengan operator **new**.
- Hanya digunakan sebagai superclass dari kelas-kelas lainnya dan berupa bentuk abstrak.
- Dideklarasikan dengan *keyword* **abstract**.
- Superclass harus kongkrit agar bisa di abstrak-kan ke subclass

- 
- Sebuah template atau design untuk subclass dibawahnya.
 - Menyediakan fungsi yang abstrak juga.(*abstract class*)
 - Fungsi di *override* di subclass.
 - Sebuah objek harus dapat mengimplementasikan semua *abstract method* yang ada di *abstract class*

PEMAHAMAN..😊

Setiap benda 2 dimensi, misal persegi panjang dan lingkaran, keduanya punya luas dan keliling, Tetapi, cara menghitung luas dan keliling pada kedua benda tersebut berbeda.

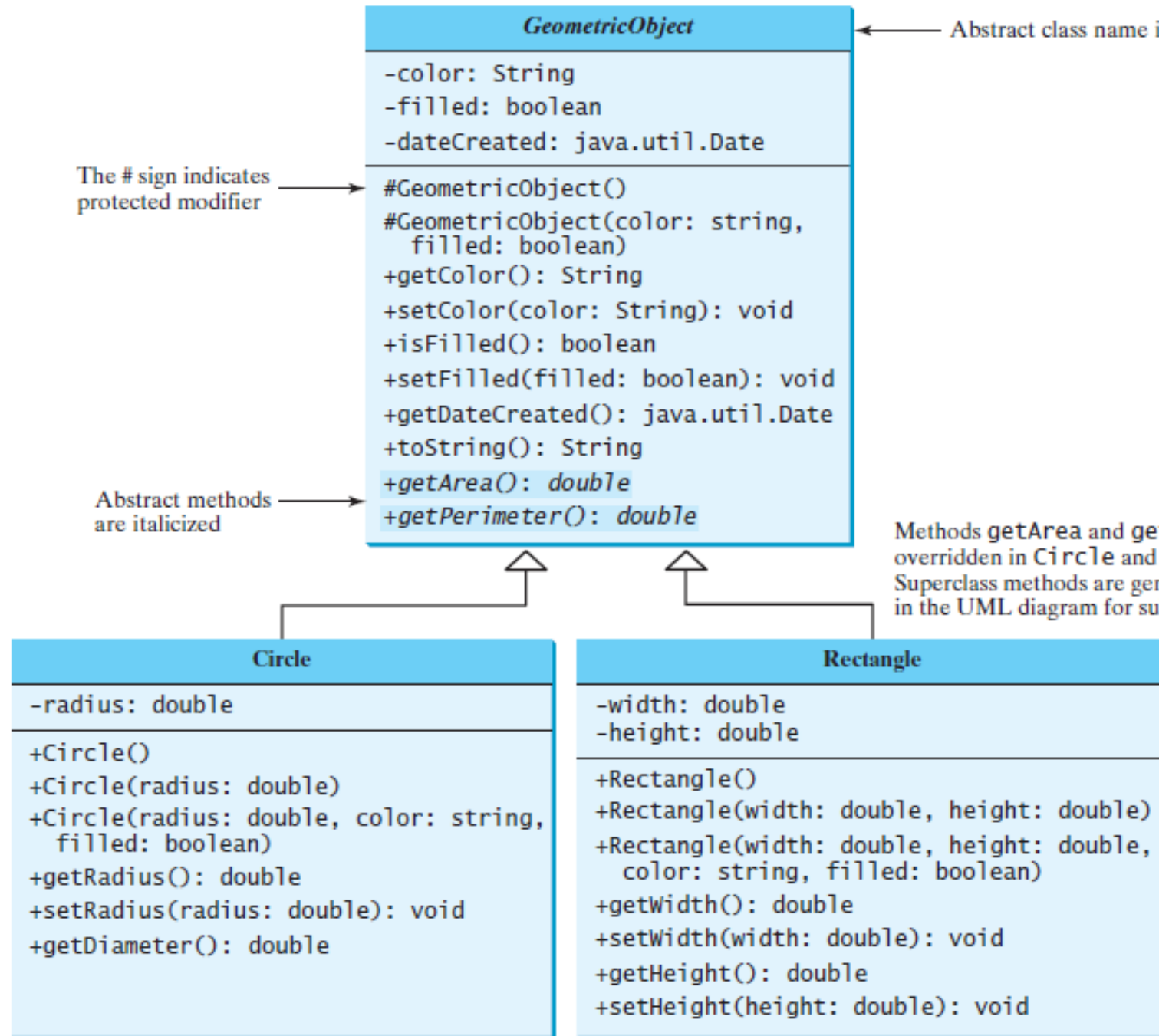
Method hitungLuas dan hitungKeliling adalah template, seharusnya ditempatkan pada kelas abstract dan menjadi superkelas, nantinya akan di turunkan ke kelas bawahnya.

The # sign indicates protected modifier

Abstract methods are italicized

Abstract class name is italicized

Methods `getArea` and `getPerimeter` are overridden in `Circle` and `Rectangle`. Superclass methods are generally omitted in the UML diagram for subclasses.



```
public abstract class cBentuk2D {  
    private String warna;  
  
    protected cBentuk2D () {  
        warna = "transparan";  
    }  
    protected cBentuk2D (String w) {  
        warna = w;  
    }  
    public void setWarna (String w) {  
        warna = w;  
    }  
    public String getWarna () {  
        return warna;  
    }  
}
```

```
public abstract double hitungLuas ();  
public abstract double hitungKeliling ();
```

```
public void info () {  
    System.out.println("Warna benda :"+getWarna());  
    System.out.println("Luas benda :"+hitungLuas());  
    System.out.println("Keliling benda :"+hitungKeliling());  
}
```

```
class lingkaran extends cBentuk2D{
    private double jari;

    public lingkaran() {
        super();
        jari = 0;
    }
    public lingkaran(double r, String w) {
        super(w);
        jari = r;
    }
    public double hitungLuas() {
        return Math.PI*jari*jari;
    }
    public double hitungKeliling() {
        return 2*Math.PI*jari;
    }
}
```

CONTOH ABSTRACT CLASS AND METHOD

■ Contoh :

Waktu deklarasi Abstract class dan Method-nya → Deklarasi abstract class

```
abstract class Shape {  
    public String color;  
    public Shape() {}  
    public void setColor(String c) {  
        color = c;  
    }  
    public String getColor() {  
        return color;  
    }  
    abstract public double area();  
}
```

→ Deklarasi abstract Method

CONTOH ABSTRACT CLASS AND METHOD (LANJUTAN ...)

■ Pada saat digunakan di Subclass

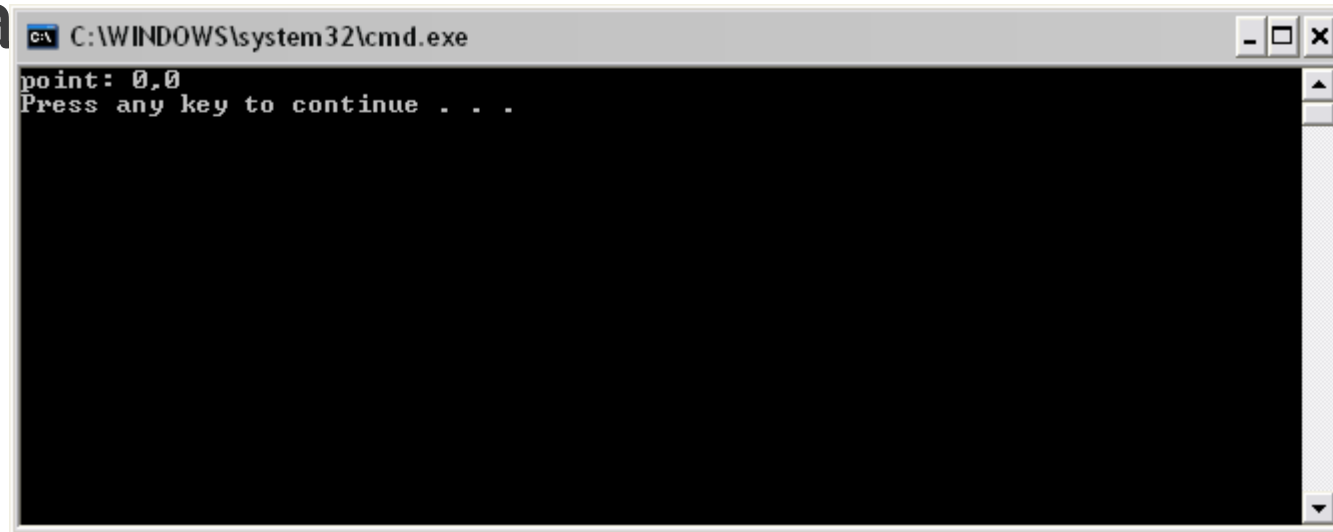
```
public class Point extends Shape {  
    static int x, y;  
    public Point() {  
        x = 0;  
        y = 0;  
    }  
    public double area() {  
        return 0;  
    }  
    public double perimeter() {  
        return 0;  
    }  
    public static void print() {  
        System.out.println("point: " + x + "," + y);  
    }  
    public static void main(String args[]) {  
        Point p = new Point();  
        p.print();  
    }  
}
```

Class Point merupakan turunan dari Class Shape yang abstrak

Fungsi area() yang di-override dari superclassnya.

CONTOH ABSTRACT CLASS AND METHOD (LANJUTAN ...)

■ Outputnya



A screenshot of a Windows command prompt window. The title bar at the top reads "C:\WINDOWS\system32\cmd.exe". The command prompt area is black with white text. The first line of output is "point: 0,0". The second line is "Press any key to continue . . .". The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

```
C:\WINDOWS\system32\cmd.exe
point: 0,0
Press any key to continue . . .
```

INTERFACES

- Hanya terdiri dari konstanta dan *abstract method*.
- Tidak dapat dibuat objeknya dengan operator **new**.
- Membuat satu subclass memiliki lebih dari satu superclass (solusi dari multiple inheritance)
- Tidak diturunkan namun di implementasi.
- Dideklarasikan dengan *keyword* **interface**.
- Pada *subclass* menggunakan *keyword* **implements**.

INTERFACES (LANJUTAN ...)

- Setiap interfaces *di-compile* ke dalam baris bytecode, seperti class biasa.
- Semua metode yang dideklarasikan di *interface* harus *di-override* oleh class yang mengimplementasikannya.
- Bentuk interfaces :

```
modifier interface Nama_Interface{  
    /** Deklarasi konstanta **/  
    /** Abstract Method **/  
}
```

Contoh :

```
public interface Tes{  
    public static final int k = 1;  
    public abstract void p();  
}
```



```
public interface Tes{  
    int k = 1;  
    void p();  
}
```

CONTOH INTERFACES

■ Contoh :

Waktu deklarasi interfaces

`interface Shape {`

`public double area();`

`public double volume();`

`}`

Deklarasi kelas interface

Abstract Method

CONTOH INTERFACES (LANJUTAN ...)

- Pada saat digunakan di Subclass

```
public class Point implements Shape {
```

```
    static int x, y;
```

```
    public Point() {
```

```
        x = 0;
```

```
        y = 0;
```

```
    }
```

```
    public double area() {
```

```
        return 0;
```

```
    }
```

```
    public double volume() {
```

```
        return 0;
```

```
    }
```

```
    public static void print() {
```

```
        System.out.println("point: " + x + "," + y);
```

```
    }
```

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

```
        Point p = new Point();
```

```
        p.print();
```

```
    }
```

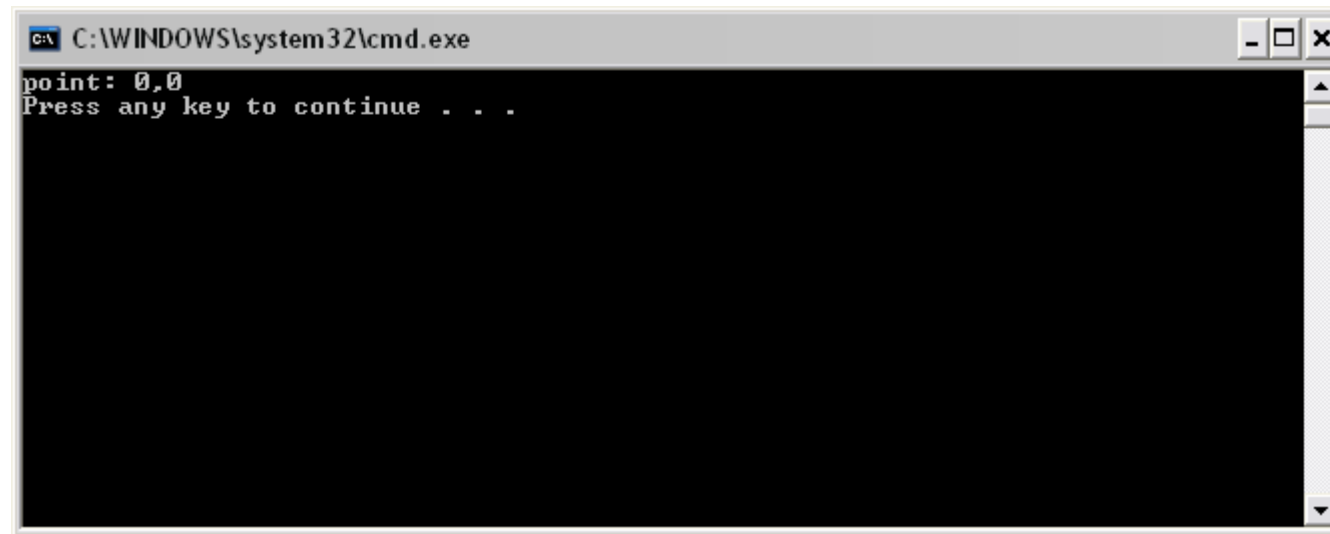
```
}
```

Menggunakan implement jika ingin memakai kelas interface dari Shape

2 Fungsi yang ada di kelas interface Shape wajib di-override di subclass Point

CONTOH INTERFACES (LANJUTAN ...)

■ Outputnya



A screenshot of a Windows command prompt window. The title bar at the top reads "C:\WINDOWS\system32\cmd.exe". The window has standard minimize, maximize, and close buttons on the right. The main area is black with white text. The first line of output is "point: 0,0". The second line is "Press any key to continue . . .". A vertical scrollbar is visible on the right side of the window.

```
C:\WINDOWS\system32\cmd.exe
point: 0,0
Press any key to continue . . .
```

- Perbedaan *interface* dan *abstract class* cukup terlihat dari pemakaiannya. *interface* itu diimplementasikan dan *abstract class* itu diturunkan (diwariskan).

	<i>Variables</i>	<i>Constructors</i>	<i>Methods</i>
Abstract Class	Bebas, tidak ada batasan, no restriction	Konstruktor dapat dipanggil melalui subclass melalui rantai konstruktor. Tidak dapat dibuat objeknya.	Bebas, tidak ada batasan, no restriction
Interfaces	Semua variable harus dideklarasikan <code>public static final</code>	Tidak ada konstruktor. Tidak dapat dibuat objeknya.	Semua fungsi harus dideklarasikan <code>public abstract</code>

INTERFACE VS ABSTRACT CLASS (LANJUTAN ...)

- Java hanya mengizinkan *single inheritance* untuk class, tetapi dapat *multiple* untuk *interfaces*.

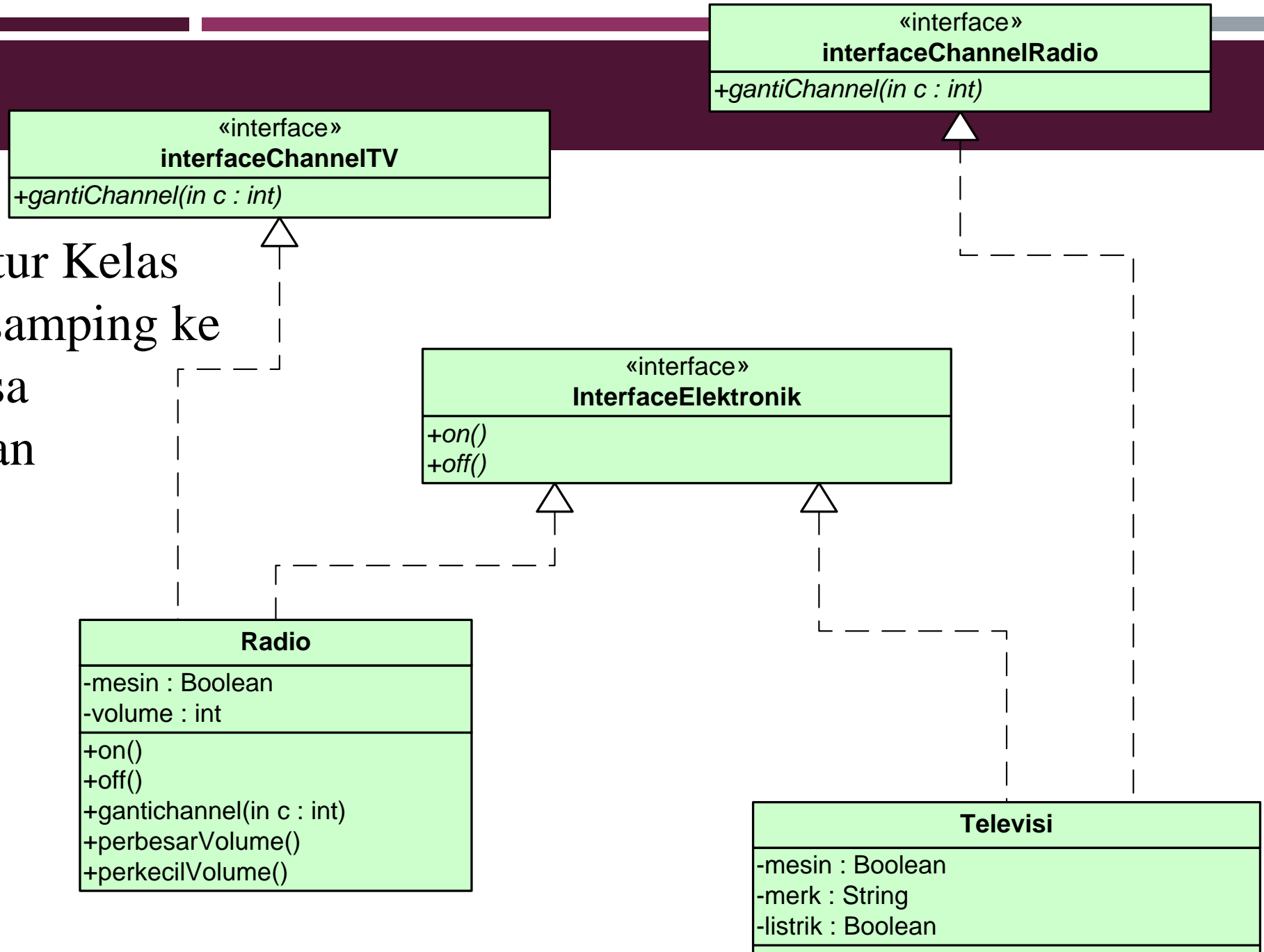
```
public class NewClass extends BaseClass implements Interface I, ..., InterfacesN {  
    // ....  
}
```

- Sebuah interfaces dapat diturunkan dari kelas lainnya dengan menggunakan keyword **extends**.

```
public interface NewClass extends Interface I, ..., InterfacesN {  
    // konstanta dan abstract method  
}
```

LATIHAN

Ubahlah struktur Kelas
Diagram disamping ke
dalam bahasa
pemrograman



LATIHAN

Untuk latihan, ubahlah struktur kelas diagram disamping ke dalam bahasa pemrograman

