IPB University
Bogor Indonesia

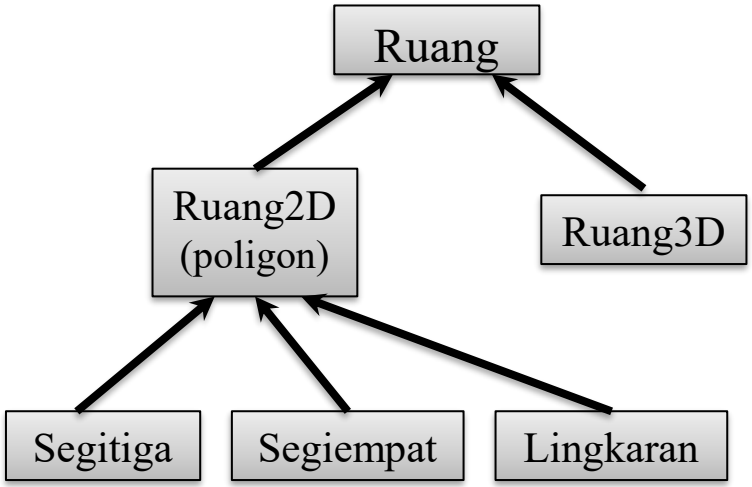
KOM120C -- PEMROGRAMAN

Object Oriented Programming

- Abstract Class
- Interface
- Multiple Inheritance

Tim Pengajar Pemrograman IPB University

Abstract Class



Class Ruang mempunyai method area() yang akan di-override oleh subclasses nya.

Ruang merupakan sifat umum dari suatu bidang dua dimensi (Segitiga, Segiempat, Lingkaran) dan tiga dimensi (Bola, Kubus).

Method dalam class Ruang tidak memiliki implementasi. Class jenis ini yang disebut dengan abstract class.

Umumnya abstract class muncul pada hirarki class pemrograman berbasis object paling atas, dan mendefinisikan keseluruhan aksi yang mungkin pada object dari seluruh subclasses dalam class.

Abstract Method

Method dalam abstract class yang tidak mempunyai implementasi dinamakan abstract method.

Only method signature, no body.

Pada C++, ditunjukkan dengan adanya method **virtual murni: virtual double area() = 0;**

Pada Java, dengan adanya deklarasi method **abstract,** Contoh:

```
public abstract class Poligon {
    public abstract double area();
}
```

Class Poligon

```
abstract class Poligon {
   protected double[] x;
   protected double[] y;
   protected int edge;

   public Poligon(int n) {
      x=new double[n]; y=new double[n]; edge=n; }

   public void set(double[] x, double[] y, int n) {
      for (int i=0; i<n; i++) {
            this.x[i]=x[i]; this.y[i]=y[i]; edge=n; }
   }

   public abstract double luas();
}
```

Interface

Interface dalam Java didefinisikan sebagai tipe abstrak untuk menentukan perilaku dari class.

Interfaces dapat memiliki abstract methods dan variables.

Interface tidak dapat dibuat obyek namun hanya dapat diimplementasi.

Sintaks:

```
interface <nameOfInterface> {
    // declare constant fields
    // declare methods that abstract
    // by default.
}
```

Contoh

```
interface Car {
    void display();
}

public class Model implements Car {
    public void display() {
        System.out.println("Car model");
    }

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

Abstract Class vs Interface

Kapan kita menggunakan **abstract class** dan kapan kita menggunakan **interface:**

Abstract class:

- Ketika hubungan yang ingin ditunjukkan adalah **is-a**
- Perlu ada beberapa metode terimplementasi pada kelas parent
- Perlu menyimpan beberapa variabel/property yang ingin diwariskan
- Tidak perlu multiple inheritance. Java tidak mensupport multiple inheritance

Interface:

- Hubungan selain **is-a**
- Tidak perlu ada metode terimplementasi, semua metode abstract pada interface
- Tidak perlu menyimpan data/variabel apa-apa. Interface tidak bisa menampung variabel.
- Perlu implementasi di beberapa kelas berbeda. Java support multiple interface.

Abstract Class vs Interface

Kapan kita menggunakan **abstract class** dan kapan kita menggunakan **interface:**

Abstract class:

```
class Vehicle {
public String BrandName;
public String RegistrationNr;
abstract public void start() {...}
                ...
}
class Car extends Vehicle {}
class Boat extends Vehicle{}
```

Interface:

```
interface MovingObjects{
 abstract public void move();
 abstract public void accelerate();
}

class Car implements
MovingObjects;
class Bird implements
MovingObjects;
```

Multiple Interface

```
interface Interface1 { public void Method1(); }

interface Interface2 { public void Method2(); }

public class Multiple implements Interface1, Interface2 {
    public void Method1() {
        System.out.println("Method 1");
    }
    public void Method2() {
        System.out.println("Method 2");
    }

    public static void main(String[] args) {
        Multiple obj = new Multiple();
        obj.Method1();
        obj.Method2();
    }
}
```

Variabel dalam Interface

Harus final atau static variables.

```
interface Rectangle {
    int height = 0;
    int width = 0;

    public int getHeight();
    public int getWidth();
    public void setHeight(int h);
    public void setWidth(int w);
}

public class Shape implements Rectangle {
    public int getHeight() { return height; }
    public int getWidth() { return width; }
    public void setHeight(int h) { height = h; }
    public void setWidth(int w) { }
    // ...
}
```

Multiple Inheritance

Java tidak mendukung multiple inheritance ➔ tidak dapat extends lebih dari 1 class.

Class dalam Java dapat implements satu atau lebih interface ➔ membantu Java mengimplementasikan konsep multiple inheritance.

Gunakan extends interface.

Contoh

```
interface Event { public void start(); }
interface Sports { public void play(); }
interface Badminton extends Sports, Event {  public void show(); }

public class Thomascup implement Badminton {
   public static void main(String[] args) {
      Thomascup obj = new Thomascup() {
         public void start() { System.out.println("Start Event"); }
         public void play() { System.out.println("Play Sports."); }
         public void show() { System.out.println("Show Badminton."); }
      };

      obj.start();
      obj.play();
      obj.show();
   }
}
```

Latihan Kelas

Buat program Java untuk mengimplementasikan struktur berikut: