Pemrograman Berorientasi Objek

Abstract Class

Abstract Class

Abstract Class

- Superclass mendeklarasikan struktur tanpa menyediakan implementasi yang jelas kepada setiap method.
- Superclass menyediakan bentuk umum yang akan “dibagikan” kepada subclass.
- Subclass akan menentukan detail yang ditentukan superclass

Superclass tanpa body method

Abstract Class

- Abstract class didefinisikan sebagai class yang tidak bisa digunakan untuk membuat object.
- Abstract class dibuat ketika adanya hal yang cukup kompleks terjadi di saat pendefinisian di tahap awal pembangunan program.
- Sub class yang memperluas (atribut dan method) dari abstract class ini bisa dibuatkan object namun masih tetap memperhatikan modifier yang digunakan di abstract class tersebut.

Abstract Class

- Kelas abstrak merupakan suatu bentuk khusus dari suatu
- Sifat:
  - Tidak dapat diinstansiasi
  - Dapat digunakan untuk diturunkan ke dalam bentuk kelas konkret
  - Dapat digunakan untuk diturunkan ke kelas abstrak berikutnya
  - Dideklarasikan menggunakan keyword abstract.
- Simbol: Sama seperti class, tapi dengan penulisan nama dengan *style italic* (untuk class dan method)

Kondisi Penggunaan

- Mendeklarasikan struktur dari suatu abstraksi tanpa memberikan implementasi
- Mendefenisikan generalized form yang akan di-share ke setiap subclass
- Memberikan kebebasan pada subclass untuk memberikan prosedur details

Penggunaan Abstract Class

- Hanya class abstract yang memiliki method abstract
- Method abstract tidak memiliki body method
- Method abstract harus di-overriden oleh subclass

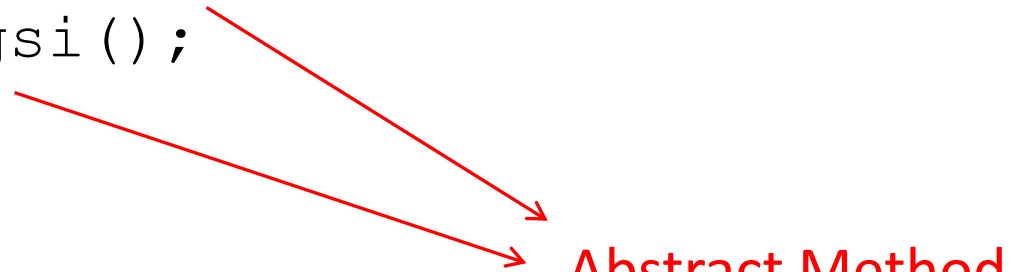
Penggunaan “ABSTRACT”

Jika abstract class
tidak dapat dibentuk objeknya,

Apakah abstract class
dapat dituliskan konstruktornya?

Deklarasi

```
public abstract class namaClass{
   public abstract void namaProsedur();
   public abstract int namaFungsi();
   public void namaMethod(){
      //algoritma
   }
}
```



Contoh Penggunaan

```
// Using abstract methods and classes.
abstract class Figure {
  double dim1;
  double dim2;

  Figure(double a, double b) {
    dim1 = a;
    dim2 = b;
  }

  // area is now an abstract method
  abstract double area();
}
```

```
class Rectangle extends Figure {
  Rectangle(double a, double b) {
    super(a, b);
  }

  // override area for rectangle
  double area() {
    System.out.println("Inside Area for Rectangle.");
    return dim1 * dim2;
  }
}
```

```
class Triangle extends Figure {
  Triangle(double a, double b) {
    super(a, b);
  }

  // override area for right triangle
  double area() {
    System.out.println("Inside Area for Triangle.");
    return dim1 * dim2 / 2;
  }
}
```

Method Abstrak harus diturunkan oleh subclass

Contoh Penggunaan

```
class AbstractAreas {
  public static void main(String args[]) {
  // Figure f = new Figure(10, 10); // illegal now
    Rectangle r = new Rectangle(9, 5);
    Triangle t = new Triangle(10, 8);
    Figure figref; // this is OK, no object is created

    figref = r;
    System.out.println("Area is " + figref.area());

    figref = t;
    System.out.println("Area is " + figref.area());
  }
}
```

Ilegal karna objek class abstract tidak mungkin dibuat

Class Abstract sebagai referensi tipe objek

Akhir Presentasi