Riza Satria Perdana, S.T., M.T.

Teknik Informatika - STEI ITB

Class and Object in Java

Class Feature

Pemrograman Berorientasi Objek

Garbage Collection

- The Java runtime environment deletes objects when it determines that they are no longer being used
- An object is eligible for garbage collection when there are no more references to that object
  - variable goes out of scope
  - explicitly drop an object reference by setting the variable to the special value null

Information Hiding

```java
public class Point {
   private int x = 0;
   private int y = 0;
   // a constructor!
   public Point(int a, int b) {
      x = a; y = b; }
   public int getX() {
      return x; }
   public int getY() {
      return y; }
   public void setX(int ax) {
      x = ax; }
   public void setY(int ay) {
      y = ay; }
}
```

Bicycle class

```
public class Bicycle {
   private int cadence = 0;
   private int speed = 0;
   private int gear = 1;

   public void changeCadence(int newValue) {
      cadence = newValue; }
   public void changeGear(int newValue) {
      gear = newValue; }
   public void speedUp(int increment) {
      speed = speed + increment; }
   public void applyBrakes(int decrement) {
      speed = speed - decrement; }
   public void printStates() {
      System.out.println("cadence:"+cadence+
            "speed:"+speed+" gear:"+gear); }
}
```

Bicycledemo class

```
public class BicycleDemo {
    public static void main(String[] args) {
        // Create two different Bicycle objects
        Bicycle bike1 = new Bicycle();
        Bicycle bike2 = new Bicycle();
        // Invoke methods on those objects
        bike1.changeCadence(50);
        bike1.speedUp(10);
        bike1.changeGear(2);
        bike1.printStates();
        bike2.changeCadence(50);
        bike2.speedUp(10);
        bike2.changeGear(2);
        bike2.changeCadence(40);
        bike2.speedUp(10);
        bike2.changeGear(3);
        bike2.printStates();
    }
}
```

Class and Object Benefits

- Modularity: The source code for an object can be written and maintained independently of the source code for other objects. Once created, an object can be easily passed around inside the system.
- Information-hiding: By interacting only with an object's methods, the details of its internal implementation remain hidden from the outside world.

Class and Object Benefits

- Code re-use: If an object already exists (perhaps written by another software developer), you can use that object in your program.
- Pluggability and debugging ease: If a particular object turns out to be problematic, you can simply remove it from your application and plug in a different object as its replacement.

Static Attribute (class var)

```
public class Bicycle {
   private int cadence = 0;
   private int speed = 0;
   private int gear = 1;

   private static int numberOfBicycles = 0;
   ...
}
```

- **Hanya ada 1 copy untuk seluruh objek dan diakses menggunakan nama kelas**

```
   Bicycle.numberOfBicycles
```

Static Method

```
public class Bicycle {
   ...
   private static int numberOfBicycles = 0;
   public static int getNumberOfBicycles() {
      return numberOfBicycles; }
   ...
}
```

- **Dimiliki oleh kelas (bukan objek) dan diakses menggunakan nama kelas**

```
Bicycle.getNumberOfBicycles()
```

Reference ‘this’

- **Reference ke current object: this**

```
public class Point {
   private int x = 0;
   private int y = 0;

   public Point(int x, int y) {
      this.x = x;
      this.y = y;
   }
   ...
}
```

Reference ‘this’

```
public class Rectangle {
   ...
   public Rectangle() {
      this(0,0,0,0);
   }
   public Rectangle(int x, int y, int width, int height) {
      this.x = x;
      this.y = y;
      this.width = width;
      this.height = height;
   }
   ...
}
```

Terima Kasih