Pemrograman Berbasis Obyek

Overriding dan Overloading

Oleh Politeknik Elektronika Negeri Surabaya

2017

Politeknik Elektronika Negeri Surabaya
Departemen Teknik Informatika dan Komputer

Konten

- Overriding
- Overloading
- Constructor overloading

Overriding

- Subclass yang berusaha memodifikasi tingkah laku yang diwarisi dari superclass.
- Tujuan: subclass memiliki tingkah laku yang lebih spesifik.
- Dilakukan dengan cara mendeklarasikan kembali method milik parent class di subclass.

Overriding

- Deklarasi method pada subclass harus sama dengan yang terdapat di super class. Kesamaan pada:
  - Nama
  - Return type
  - Daftar parameter (jumlah, tipe, dan urutan)
- Method pada parent class disebut overriden method
- Method pada subclass disebut overriding method.

Contoh Overriding

```
public class Employee {
  protected String name;
  protected double salary;
  protected Date birthDate;

  public String getDetails() {
    return "Name: " + name + "\n" +
           "Salary: " + salary;
  }
}
```

```
public class Manager extends Employee {
  protected String department;

  public String getDetails() {
    return "Name: " + name + "\n" +
           "Salary: " + salary + "\n" +
           "Manager of: " + department;
  }
}
```

Contoh Overriding

```
public class Animal {
        public void SetVoice() {
                System.out.println("Blesepblesep");
        }
}

public class Dog extends Animal {
        public void SetVoice() {
                System.out.println("Hug hug");
        }
}
```

Aturan Overriding

- Mode akses overriding method harus sama atau lebih **luas** dari pada overriden method.
- Subclass hanya boleh meng-override method superclass satu kali saja, tidak boleh ada lebih dari satu method pada kelas yang sama yang sama persis.
- Overriding method tidak boleh throw checked exceptions yang tidak dideklarasikan oleh overridden method.

```
public class Parent {
    public void doSomething() {}
}

public class Child extends Parent {
    private void doSomething() {}
}

public class UseBoth {
    public void doOtherThing() {
            Parent p1 = new Parent();
            Parent p2 = new Child();
            p1.doSomething();
            p2.doSomething();
    }
}
```

Overloading

- Overloading adalah suatu keadaan dimana beberapa method sekaligus dapat mempunyai nama yang sama, akan tetapi mempunyai fungsionalitas yang berbeda tapi mirip
- Cara: Menuliskan kembali method dengan nama yang sama pada suatu class atau antar parent dan subclass.
- Tujuan : memudahkan penggunaan/pemanggilan method dengan fungsionalitas yang **mirip**.

Contoh Overloading

1 titik → menggambar titik

2 titik → menggambar garis

3 titik → menggambar segitiga

4 titik → menggambar persegi empat

dst…

- Jika pemrograman terstruktur, kita harus menyelesaikan kasus diatas dengan cara membuat fungsi untuk masing-masing gambar dengan nama yang harus berbeda dan melakukan tugas penggambaran masing-masing fungsi.
  - `titik(int t1)`
    1 parameter titik, untuk menggambar titik
  - `garis(int t1, int t2)`
    2 parameter titik, untuk menggambar garis
  - `segitiga(int t1, int t2, int t3`
    3 parameter titik, untuk menggambar segitiga
  - `persegiEmpat(int t1, int t2, int t3, int t4)`
    4 parameter titik, untuk menggambar persegi empat
  - dst…

Contoh method overloading

- `void gambar(int t1)`
  1 parameter titik, untuk menggambar titik
- `void gambar(int t1, int t2)`
  2 parameter titik, untuk menggambar garis
- `void gambar(int t1, int t2, int t3`
  3 parameter titik, untuk menggambar segitiga
- `void gambar(int t1, int t2, int t3, int t4)`
  4 parameter titik, untuk menggambar persegi empat
- dst…

Aturan Pendeklarasian Method Overloading

- Nama method harus sama
- Daftar parameter harus berbeda
- Return type boleh sama, juga boleh berbeda

Daftar Parameter Pada Overloading

- Perbedaan daftar parameter bukan hanya terjadi pada perbedaan banyaknya parameter, tetapi juga urutan dari parameter tersebut.
- Misalnya saja dua buah parameter berikut ini :
  - function_member(int x, String n)
  - function_member(String n, int x)
- Dua parameter tersebut juga dianggap berbeda daftar parameternya.

Daftar Parameter Pada Overloading

- Daftar parameter tidak terkait dengan penamaan variabel yang ada dalam parameter.
- Misalnya saja 2 daftar parameter berikut :
  - function_member(int x)
  - function_member(int y)
- Dua daftar parameter diatas dianggap sama karena yang berbeda hanya penamaan variabel parameternya saja.

- Signature method pada OO yaitu jumlah parameter yang ada pada method dan tipe data parameternya.
- Return type tidak termasuk signature
- Jadi kita tidak dapat melakukan overloading hanya dengan perbedaan return type.

Contoh Overloading

```
public void println(int i)
public void println(float f)
public void println(String s)
```

Contoh

```
public class Bentuk {
    …
    public void gambar(int t1) {
            // 1 parameter titik, untuk menggambar titik
            …
    }
    public void gambar(int t1, int t2) {
            // 2 parameter titik, untuk menggambar garis
            …
    }
    public void gambar(int t1, int t2, int t3) {
            // 3 parameter titik, untuk menggambar segitiga
            …
    }
    public void gambar(int t1, int t2, int t3, int t4) {
            // 4 parameter titik, untuk menggambar persegi empat
            …
    }
}
```

return type	nama method	daftar parameter
void	Gambar	(int t1)
void	Gambar	(int t1, int t2)
void	Gambar	(int t1, int t2, int t3)
void	Gambar	(int t1, int t2, int t3, int t4)
↓	↓	↓
Boleh sama atau beda	sama	berbeda

- Overloading juga bisa terjadi antara parent class dengan subclass-nya jika memenuhi ketiga syarat overload.
- Misalnya saja dari class Bentuk pada contoh sebelumnya kita turunkan sebuah class baru yang bernama WarnaiBentuk.

```
public class WarnaiBentuk extends Bentuk {
   public void gambar(String warna, int t1, int t2, int3) {
            …
   }

   public void gambar(String warna, int t1, int t2, int3, int t4) {
            …
   }
   …
}
```

```
public class buku {
        private String judul;
        private int tahun;
        private String pengarang;

        buku (){
        }
        buku (String judul, int tahun, String pengarang){
            this.judul = judul;
            this.tahun = tahun;
            this.pengarang = pengarang;
          }
        public void setBuku (String judul){
            this.judul = judul;
        }
        public void setBuku (String judul,int tahun){
            this.judul = judul;
            this.tahun = tahun;
        }
        public void setBuku (String judul,int tahun,String pengarang){
            this.judul = judul;
            this.tahun = tahun;
            this.pengarang = pengarang;
        }



    }
```

```
{
  buku obuku;
          obuku = new buku();
          obuku.setBuku ("PBO");

}
```

Constructor Overloading

- As with methods, constructors can be overloaded.
- Example:

```
public Employee(String name, double salary, Date DoB)
public Employee(String name, double salary)
public Employee(String name, Date DoB)
```

- Argument lists *must* differ.
- You can use the `this` reference at the first line of a constructor to call another constructor.

Constructor Overloading

```
1   public class Employee {
2     private static final double BASE_SALARY = 15000.00;
3     private String name;
4     private double salary;
5     private Date   birthDate;
6
7     public Employee(String name, double salary, Date DoB) {
8       this.name = name;
9       this.salary = salary;
10      this.birthDate = DoB;
11    }
12    public Employee(String name, double salary) {
13      this(name, salary, null);
14    }
15    public Employee(String name, Date DoB) {
16      this(name, BASE_SALARY, DoB);
17    }
18    public Employee(String name) {
19      this(name, BASE_SALARY);
20    }
21    // more Employee code...
22  }
```

Memanggil parent class construktor

```
1   public class Manager extends Employee {
2     private String department;
3
4     public Manager(String name, double salary, String dept) {
5       super(name, salary);
6       department = dept;
7     }
8     public Manager(String n, String dept) {
9       super(name);
10      department = dept;
11    }
12    public Manager(String dept) {
13      department = dept;
14    }
15  }
```

Tugas

- Buatlah ringkasan mengenai Overloading dan Overriding!

1. Oracle Java Documentation, The Java™ Tutorials, https://docs.oracle.com/javase/tutorial/, Copyright © 1995, Oracle 2015.
2. Tita Karlita, Yuliana Setrowati, Rizky Yuniar Hakkun, Pemrograman Berorientasi Obyek, PENS-2012
3. Sun Java Programming, Sun Educational Services, Student Guide, Sun Microsystems, 2001.
4. John R. Hubbard, Programming With Java, McGraw-Hill, ISBN: 0-07-142040-1, 2004.
5. Patrick Niemeyer, Jonathan Knudsen, Learning Java, O'reilly, CA, ISBN: 1565927184, 2000.
6. Philip Heller, Simon Roberts, Complete Java 2 Certification Study Guide, Third Edition, Sybex, San Francisco, London, ISBN: 0-7821-4419-5, 2002.
7. Herbert Schildt, The Complete Reference, Java™ Seventh Edition, Mc Graw Hill, Osborne, ISBN: 978-0-07-163177-8, 2007