

Temu-06

Inheritance

Ali Tarmuji, S.T., M.Cs.

alitarmuji@tif.uad.ac.id

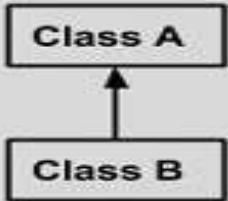
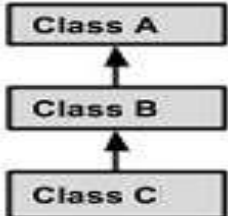
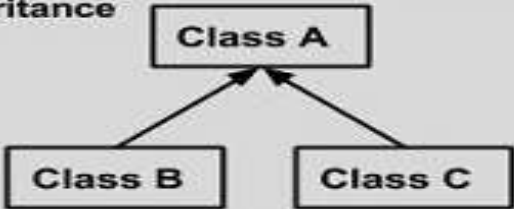
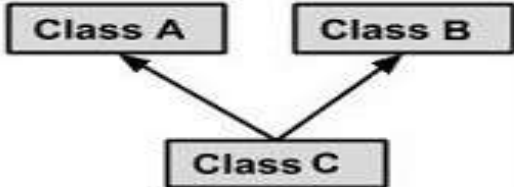
Apa itu Inheritance/Pewarisan?

- Setiap kelas dapat menurunkan seluruh atribut, prosedur, dan fungsi ke kelas lain melalui pewarisan.
- Setiap kelas dapat mewariskan ke lebih dari satu kelas.
- Namun setiap kelas hanya dapat mewarisi satu kelas.
- Di Java inheritance/ pewarisan menggunakan kata kunci **extends**.

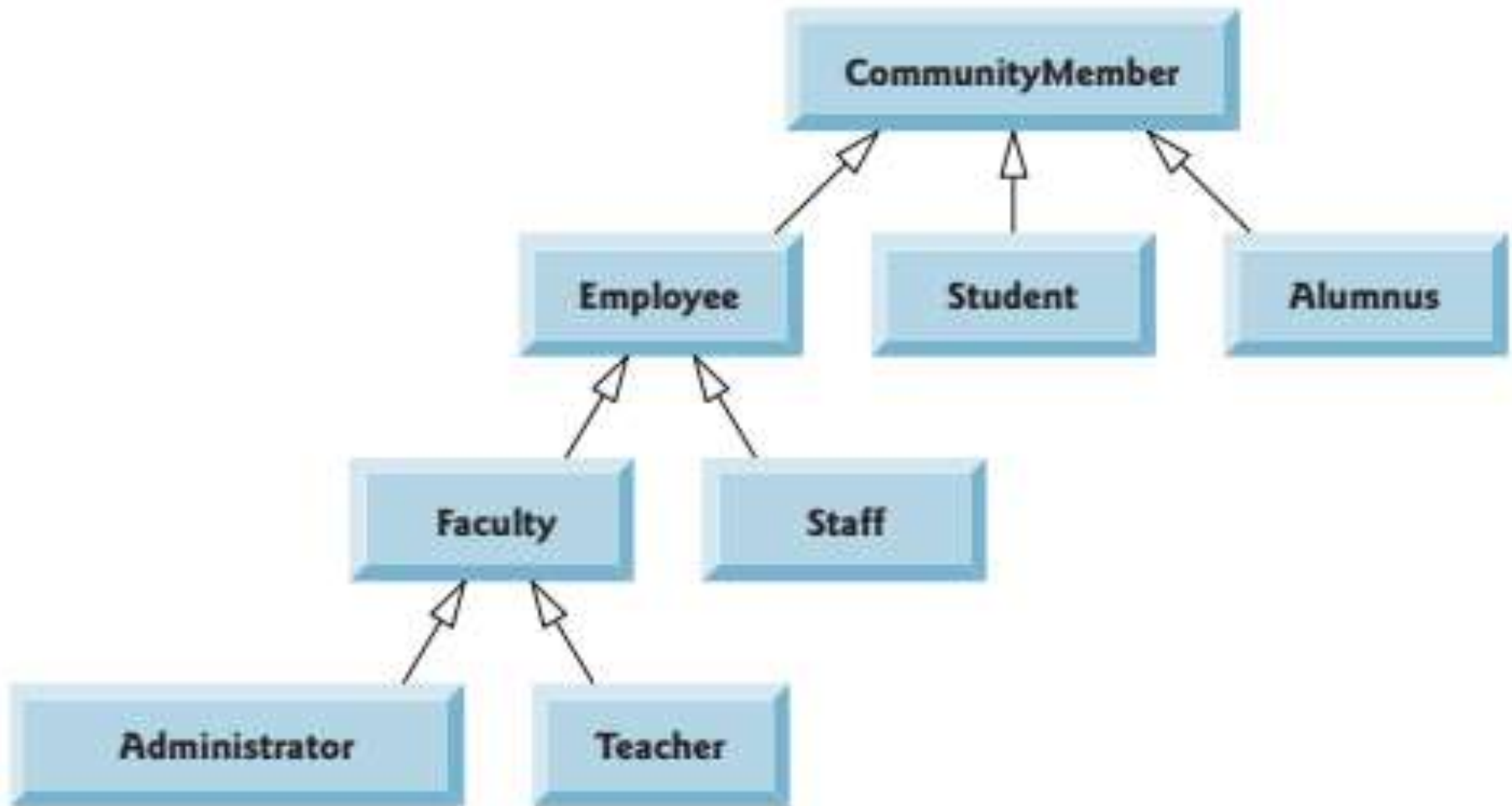
Apa itu Inheritance/Pewarisan?

- Inheritance dapat didefinisikan sebagai proses di mana satu objek mengakuisisi properti lain.
- Dengan menggunakan Inheritance, informasi yang dibuat dikelola dalam urutan hirarkis.
- Ketika kita berbicara tentang warisan, kata kunci yang paling sering digunakan adalah **extends** dan **implements**.
- Kata-kata ini akan menentukan apakah suatu tipe objek bagian (IS-A) dari objek jenis lain.
- Dengan menggunakan kata kunci tersebut dapat membuat satu objek mengakuisisi properti dari objek lain.

Tipe Pewarisan

Single Inheritance	 <pre>graph BT; B[Class B] --> A[Class A]</pre>	<pre>public class A { } public class B extends A { }</pre>
Multi Level Inheritance	 <pre>graph BT; C[Class C] --> B[Class B]; B --> A[Class A]</pre>	<pre>public class A {} public class B extends A {.....} public class C extends B {.....}</pre>
Hierarchical Inheritance	 <pre>graph BT; B[Class B] --> A[Class A]; C[Class C] --> A</pre>	<pre>public class A {} public class B extends A {.....} public class C extends A {.....}</pre>
Multiple Inheritance	 <pre>graph BT; C[Class C] --> A[Class A]; C --> B[Class B]</pre>	<pre>public class A {} public class B {.....} public class C extends A,B { } // Java does not support mutple Inheritance</pre>

Contoh hirarki pewarisan



IS-A Relationship

- IS-A Relationship adalah cara untuk menyebutkan: bahwa objek ini adalah tipe dari objek itu.

```
public class Animal{  
    .....  
}
```

```
public class Mammalia extends Animal{  
    .....  
}
```

```
public class Reptile extends Animal{  
    .....  
}
```

```
public class Cat extends Mammalia{  
    .....  
}
```

- Berdasarkan contoh tersebut, pada konsep *Object Oriented* pernyataan berikut adalah benar:
 - **Animal** adalah superclass untuk class **Mammalia**
 - **Animal** adalah superclass untuk class **Reptile**
 - **Mammalia** dan **Reptile** adalah subclass untuk class **Animal**
 - **Cat** adalah subclass untuk class **Mammalia** and **Animal**.

- Jika menggunakan IS-A relationship:
 - Mammalia IS-A Animal
 - Reptile IS-A Animal
 - Cat IS-A Mammalia
 - Karena itu : Cat IS-A Animal as well (Cat juga Animal)
- Dengan menggunakan keyword **extends**, subclass akan mampu mewarisi/inherit semua properti (variabel dan method) dari superclass kecuali properti dengan modifier private.

Contoh Pewarisan (1)

```
public class Parent {  
    private String nama;  
  
    public String getNama() {  
        return this.nama;  
    }  
    public void setNama(String nama) {  
        this.nama = nama;  
    }  
    public void tulisNama() {  
    }  
}
```

Contoh Pewarisan (2)

```
public class Child extends Orang {  
    public void tulisNama () {  
        system.out.println(  
            'Nama saya ' +nama);  
    }  
}
```

Contoh Pewarisan (3)

```
Child anak = new Child();
```

```
anak.setNama( "Eko" );
```

```
anak.tulisNama();
```

atau

```
System.out.println( anak.getNama() );
```

Kata Kunci super?

- Kata kunci **super** digunakan untuk memanggil kelas parent (kelas yang diwarisi).
- Penggunaan kata kunci **super**, hampir sama dengan kata kunci **this**.

Contoh super (1)

```
public class Parent {  
  
    public void mulaiJalan() {  
  
        System.out.println( "Jalan!!!" );  
    }  
  
}
```

Contoh super (2)

```
public class Child extends Parent {  
  
    public void mulaiLari() {  
        super.mulaiJalan();  
  
        System.out.println( "Lari!!!" );  
    }  
  
}
```

Contoh super(3)

```
Child anak = new Child();  
anak.mulaiLari();
```

```
// output
```

```
Jalan!!!
```

```
Lari!!!
```

Visibility Protected

- Protected hanya dapat dilihat oleh **kelas itu sendiri, kelas dalam paket yang sama, dan kelas turunannya.**
- Protected tidak dapat diakses dari kelas yang berada dalam paket yang berbeda.

Contoh Protected

```
public classs NamaKelas {  
  
    protected String namaAtribut;  
  
    protected void namaProsedur() {  
    }  
  
    protected TipeData namaFungsi() {  
        return hasil;  
    }  
  
}
```