Inheritance

Pokok Pembahasan

- Definisi Inheritance
- Super Class
- Child Class / Sub Class
- Override Method
- Extends Keyword
- Super Keyword

Tujuan Belajar

Dengan praktikum ini mahasiswa diharapkan dapat:

- Mengerti konsep dari pada Inheritance pada Pemograman Berorientasi Objek
- Memahami perbedaan dari pada Super Class dan Sub Class
- Mengerti dengan cara Overriding Method pada PBO
- Paham dengan istilah-istilah yang terkait dengan Iheritance, seperti Extends, Super, dll

Iheritance Class dapat didefinisikan dengan referensi pada class yang lain yang telah terdefinisi. ***Inheritance*** merupakan pewarisan atribut dan method pada sebuah class yang diperoleh dari class yang telah terdefinisi tersebut. Setiap ***subclass*** akan mewarisi ***state*** (variabel- variabel) dan ***behaviour*** (method- method) dari ***superclass***-nya. ***Subclass*** kemudian dapat menambahkan ***state*** dan ***behaviour*** baru yang spesifik dan dapat pula memodifikasi (***override***) ***state*** dan ***behaviour*** yang diturunkan oleh ***superclass***-nya.

Keuntungan dari inheritance adalah :

Subclass menyediakan ***state/behaviour*** yang spesifik yang membedakannya dengan ***superclass***, hal ini akan memungkinkan programmer Java untuk menggunakan ulang ***source code*** dari ***superclass*** yang telah ada.

Istilah dalam ***inheritance*** yang perlu diperhatikan :

Extends
Keyword ini harus kita tambahkan pada definisi class yang menjadi subclass.

Superclass
Superclass digunakan untuk menunjukkan hirarki class yang berarti class dasar dari subclass/class anak.

Subclass
Subclass adalah class anak atau turunan secara hirarki dari superclass.

Super
Keyword ini digunakan untuk memanggil konstruktor dari superclass atau menjadi variabel yang mengacu pada superclass. Methode ***Overriding***

Pendefinisian ulang method yang sama pada subclass.

Dalam ***inheritance***, method ***overriding*** berbeda dengan method ***overloading***. Kalau method ***overriding*** adalah mendefinisikan kembali method yang sama, baik nama method maupun signature atau parameter yang diperlukan dalam subclass, kalau method ***overloading*** adalah mendefinisikan method yang memiliki nama yang sama, tetapi dengan signature yang berbeda dalam definisi kelas yang sama

Percobaan Inheritance.java

```
package main;


class A {
    //Membar variable dari parent class
    int x;
    int y;

    void TampilkanNilaixy() {
        System.out.println("Nilai x : " + x + ", y : " + y);
    }

    void sum(int x, int y) {
        int result = x * y;
        System.out.println("Hasil Perkalian parent class :" + result);
    }
}
```

```
class B extends A {
    //member variable dari child class
    int z;
    int hasil;

    void TampilkanJumlah() {
        /*
        subclass dapat mengakses member dari superclass. member superclass
        bisa di akses dari child class baik di tulis secara langsung nama member
nya
        ataupun menggunakan keyword super. Misalnya super.x dan super.y
        */

        /*
        Mengakses member dari parent class secara langsung tanpa menggunakan
        keyword super
        */
        hasil = x + y + z;
        System.out.println("Hasil Penjumlahan dari child class : " + hasil);
    }
```

```
    @Override
    void sum(int x, int y) {
        /*
        ini adalah cara untuk meng-override( mengubah implementasi ) fungsi dari
parrent class
        yakni dengan menggunakan keyword @Override
        */

        //Memanggil fungsi dari parent class dengan keyword super
        super.sum(x, y);

        //hasil pembagian
        System.out.println("Hasil Pembagian child class: " + (hasil / 5));
    }

}


class Inheritance {

    public static void main(String[] args) {

        //buat instance masing-masing class
        A VarsuperOb = new A();
        B VarsubOb = new B();
        System.out.println("SuperClass");

        //inisialisasi nilai dari properti super class
        VarsuperOb.x = 10;
        VarsuperOb.y = 20;

        VarsuperOb.TampilkanNilaixy();

        System.out.println("SubClass");
        //member superclass dapat diakses dari child class nya
        VarsubOb.x = 5;
        VarsubOb.y = 4;

        VarsubOb.TampilkanNilaixy();

        System.out.println("Child Class Jumlah");
        //member tambahan yang hanya ada dalam child class
        VarsubOb.z = 30;
        VarsubOb.TampilkanJumlah();
        System.out.println("Child Class");

        VarsubOb.x = 15;
        VarsubOb.y = 14;
        VarsubOb.TampilkanNilaixy();

        System.out.println("Super Class");
```

```
        VarsuperOb.x = 10;
        VarsuperOb.y = 20;

        VarsuperOb.TampilkanNilaixy();

        System.out.println("Child Class Jumlah");
        VarsubOb.z = 60;

        VarsubOb.TampilkanJumlah();

        //Memanggil fungsi sum melalui child class (B)
        VarsubOb.sum(10, 20);
    }

}
```

```
SuperClass
Nilai x : 10, y : 20
SubClass
Nilai x : 5, y : 4
Child Class Jumlah
Hasil Penjumlahan dari child class : 39
Child Class
Nilai x : 15, y : 14
Super Class
Nilai x : 10, y : 20
Child Class Jumlah
Hasil Penjumlahan dari child class : 89
Hasil Perkalian parent class :200
Hasil Pembagian child class: 17
```