

PEMROGRAMAN BERORIENTASI OBJEK

POLIMORFISME (POLYMORPHISM)

Rosa Ariani Sukamto

ROSA ARIANI SUKAMTO

Blog: <http://hariiniadalahhadiah.wordpress.com>

Facebook: <https://www.facebook.com/rosa.ariani.sukamto>

Email: rosa_if_itb_01@yahoo.com




POLIMORFISME

Polimorfisme berarti banyak bentuk

Pada pemrograman berorientasi objek, polimorfisme merupakan konsep yang menyatakan sesuatu yang sama dapat memiliki berbagai bentuk dan perilaku yang berbeda

Polimorfisme misalnya beberapa metode yang memiliki nama yang sama diijinkan dalam pemrograman berorientasi objek di dalam sebuah kelas atau berada pada kelas turunannya asalkan masih memiliki identitas yang tidak sama persis, misalnya berbeda parameter masukan metode atau berbeda nama kelas



EXAMPLES (1)

Constructor Summary

Constructors

Constructor and Description

Scanner(File source)

Constructs a new Scanner that produces values scanned from the specified file.

Scanner(File source, String charsetName)

Constructs a new Scanner that produces values scanned from the specified file.

Scanner(InputStream source)

Constructs a new Scanner that produces values scanned from the specified input stream.

Scanner(InputStream source, String charsetName)

Constructs a new Scanner that produces values scanned from the specified input stream.

Scanner(Path source)

Constructs a new Scanner that produces values scanned from the specified file.

Scanner(Path source, String charsetName)

Constructs a new Scanner that produces values scanned from the specified file.

Scanner(Readable source)

Constructs a new Scanner that produces values scanned from the specified source.

EXAMPLES (2)

String	<code>findInLine(Pattern pattern)</code> Attempts to find the next occurrence of the specified pattern ignoring delimiters.
String	<code>findInLine(String pattern)</code> Attempts to find the next occurrence of a pattern constructed from the specified string, ignoring delimiters.
String	<code>findWithinHorizon(Pattern pattern, int horizon)</code> Attempts to find the next occurrence of the specified pattern.
String	<code>findWithinHorizon(String pattern, int horizon)</code> Attempts to find the next occurrence of a pattern constructed from the specified string, ignoring delimiters.
boolean	<code>hasNext()</code> Returns true if this scanner has another token in its input.
boolean	<code>hasNext(Pattern pattern)</code> Returns true if the next complete token matches the specified pattern.
boolean	<code>hasNext(String pattern)</code> Returns true if the next token matches the pattern constructed from the specified string.
boolean	<code>hasNextBigDecimal()</code> Returns true if the next token in this scanner's input can be interpreted as a BigDecimal using the <code>nextBigDecimal()</code> method.

OVERLOADING

Overloading merupakan bentuk polimorfisme yaitu beberapa metode dapat memiliki nama yang sama dengan isi dan parameter yang berbeda di dalam sebuah kelas



IMPLEMENTASI OVERLOADING - C++ (1)

```
class Buku{
    private:
        string judul;
        int tahun;
        string pengarang;

    public:
        Buku() {
        }

        Buku(string judul, int tahun,
            string pengarang){
            this->judul = judul;
            this->tahun = tahun;
            this->pengarang = pengarang;
        }

        void setBuku(string judul){
            this->judul = judul;
        }
}
```

```
void setBuku(string judul,
    int tahun){
    this->judul = judul;
    this->tahun = tahun;
}

void setBuku(string judul,
    int tahun, string pengarang){
    this->judul = judul;
    this->tahun = tahun;
    this->pengarang = pengarang;
}

//get dan set
.....

~Buku() {
}

};
```

IMPLEMENTASI OVERLOADING - C++ (2)

```
#include <string>
#include <iostream>
using namespace std;
#include "Buku.cpp"

int main(){
    Buku obuku;
    obuku.setBuku("PBO");
    cout << "Judul: " << obuku.getJudul() << endl;
    cout << "Tahun: " << obuku.getTahun() << endl;
    cout << "Pengarang: " << obuku.getPengarang() << endl;

    obuku.setBuku("PBO2", 2011);
    cout << "Judul: " << obuku.getJudul() << endl;
    cout << "Tahun: " << obuku.getTahun() << endl;
    cout << "Pengarang: " << obuku.getPengarang() << endl;
    return 0;
}
```


BAGAIMANA DENGAN INI?

int tambah(int x, int y)

dan

int tambah(int a, int b)

Apakah dapat diimplementasikan dan dieksekusi?



JAWABNYA....

Tidak

**Karena yang dilihat adalah tipe, bukan
nama, sehingga kedua metode itu sama-
sama**

int tambah(int, int)



IMPLEMENTASI OVERLOADING - PHP (1)

```
<?php
```

```
class Buku{
```

```
    var $judul;
```

```
    var $tahun;
```

```
    var $pengarang;
```

```
    function Buku() {
```

```
    }
```

```
    function setBuku($judul){
```

```
        $this->judul = $judul;
```

```
    }
```

```
    function setBuku($judul, $tahun){
```

```
        $this->judul = $judul;
```

```
        $this->tahun = $tahun;
```

```
    }
```

```
function setBuku($judul,
```

```
    $tahun, $pengarang){
```

```
    $this->judul = $judul;
```

```
    $this->tahun = $tahun;
```

```
    $this->pengarang = $pengarang;
```

```
    }
```

```
}
```

```
?>
```

IMPLEMENTASI OVERLOADING - PHP (2)

```
<?php
    include "Buku.php";

    $obuku = new Buku();
    $obuku->setBuku("PBO");

    echo "Judul: ".$obuku->getJudul()."<br/>";
    echo "Tahun: ".$obuku->getTahun()."<br/>";
    echo "Pengarang: ".$obuku->getPengarang()."<br/>";

    $obuku->setBuku("PBO2", 2011);
    echo "Judul: ".$obuku->getJudul()."<br/>";
    echo "Tahun: ".$obuku->getTahun()."<br/>";
    echo "Pengarang: ".$obuku->getPengarang()."<br/>";

?>
```

IMPLEMENTASI OVERLOADING - JAVA (1)

```
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;
            }

        //get dan set
        .....
    }
}
```

IMPLEMENTASI OVERLOADING - JAVA (2)

```
class Main{

    public static void main(String[] args){

        Buku obuku;

        obuku = new Buku();

        obuku.setBuku("PBO");

        System.out.println("Judul: " + obuku.getJudul());

        System.out.println("Tahun: " + obuku.getTahun());

        System.out.println("Pengarang: " + obuku.getPengarang());

        obuku.setBuku("PBO2", 2011);

        System.out.println("Judul: " + obuku.getJudul());

        System.out.println("Tahun: " + obuku.getTahun());

        System.out.println("Pengarang: " + obuku.getPengarang());

    }

}
```

OVERRIDING

Overriding merupakan bentuk polimorfisme yaitu beberapa metode pada kelas orang tua dapat ditulis ulang pada kode kelas anak dalam pewarisan (*inheritance*) dengan memiliki nama yang sama dan memiliki isi ataupun parameter yang sama atau berbeda



IMPLEMENTASI OVERRIDING - C++ (1)

```
class BangunDatar{  
  
    public:  
  
        BangunDatar() {  
        }  
  
        int luas() {  
            return 0;  
        }  
  
        ~BangunDatar() {  
        }  
  
};
```

```
class Persegi : public BangunDatar{  
  
    public:  
  
        Persegi() {  
        }  
  
        int luas(int p, int l) {  
  
            return (p * l);  
        }  
  
        ~Persegi() {  
        }  
  
};
```


IMPLEMENTASI OVERRIDING - C++ (2)

```
#include <iostream>
using namespace std;
#include "BangunDatar.cpp"
#include "Persegi.cpp"

int main(){

    Persegi opersegi;

    opersegi.luas(5, 6);

    return 0;
}
```

IMPLEMENTASI OVERRIDING - PHP (1)

```
<?php

class BangunDatar{

    function BangunDatar() {
    }

    function luas() {
        return 0;
    }

}

?>
```

```
<?php

class Persegi extends BangunDatar{

    function Persegi() {
    }

    function luas($p, $l) {

        return ($p * $l);
    }

}

?>
```

IMPLEMENTASI OVERRIDING - PHP (2)

```
<?php
```

```
include "BangunDatar.php";
```

```
include "Persegi.php";
```

```
$opersegi = new Persegi();
```

```
$opersegi->luas(5, 6);
```

```
?>
```

IMPLEMENTASI OVERRIDING - JAVA (1)

```
class BangunDatar{
```

```
    BangunDatar() {  
    }
```

```
    int luas() {  
        return 0;  
    }
```

```
}
```

```
class Persegi extends BangunDatar{
```

```
    Persegi() {  
    }
```

```
    int luas(int p, int l){  
        return (p * l);  
    }
```

```
}
```

IMPLEMENTASI OVERRIDING - JAVA (2)

```
class Main{  
  
    public static void main(String[] args) {  
  
        Persegi opersegi;  
        opersegi = new Persegi();  
  
        opersegi.luas(5, 6);  
  
    }  
  
}
```

COBA DI CODING

Bapak a = new Bapak();

Anak b = a;

Atau

Anak b = new Anak();

Bapak a = b;

Yang bisa diimplementasikan dan dieksekusi

DAFTAR PUSTAKA

S, Rosa A. dan M. Shalahuddin. 2011. Modul Pembelajaran: Pemrograman Berorientasi Objek. Modula: Bandung.

