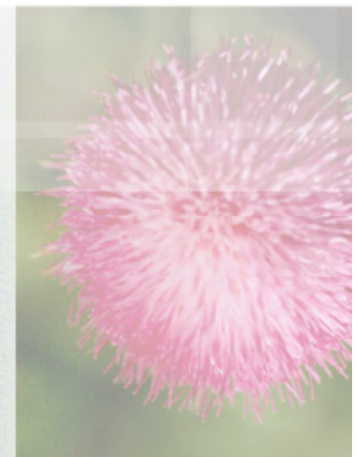




Pertemuan 9 : Class Lanjutan

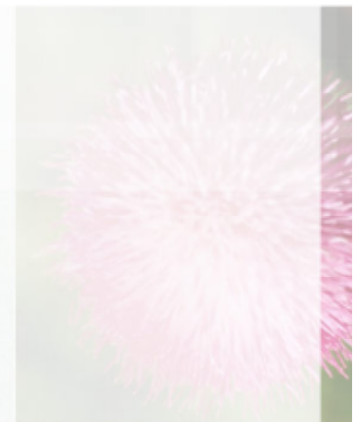
Tessy Badriyah

<http://lecturer.eepis-its.edu/~tessy>



Pembahasan

- Review
 - Bentuk Umum Class
 - Pembuatan Class sederhana
 - Menambahkan Constructors
 - Constructor Overloading
 - Keyword this
 - Local Variables dan Variable Scope
- Lanjutan
 - Instance Variables dan Instance Methods
 - Static Variables dan Static Methods
 - Method Overloading
 - Melewatkan Argument



Bentuk Umum Class

Deklarasi sebuah Class

```
Class namaClass{  
  // instance variables  
  tipeData1 namaVar1 = nilai1;  
  tipeData2 namaVar2 = nilai2;  
  .....  
  tipeDataN namaVarN = nilaiN;  
  
  //Constructors  
  namaClass(parameters1) {  
    // body constructor  
  }  
  .....  
  namaClass(parameterN) {  
    // body constructor  
  }  
  
  // Methods  
  returnTipe1 namaMethod1(parameter1) {  
    // body method  
  }  
  .....  
  returnTipeN namaMethodN(parameterN) {  
    // body method  
  }  
}
```

Pembuatan Class sederhana



Bentuk Sederhana

```
class Contoh {  
    int a;  
    int b;  
    int c;  
}
```

```
Contoh satu = new Conto  
h();
```

```
class Titik3D {  
    double x;  
    double y;  
    double z;  
}
```

```
Class ContohTitik3D {  
    public static void main(String args[]) {  
        Titik3D p = new Titik3D();  
        p.x = 1.1;  
        p.y = 3.4;  
        p.z = -2.8;
```

```
System.out.println("p.x = " + p.x);  
System.out.println("p.y = " + p.y);  
System.out.println("p.z = " + p.z);  
}  
}
```

Hasil :

```
p.x = 1.1  
p.y = 3.4  
p.z = -2.8
```


Menambahkan Constructors

```
class Titik3D {
```

```
    double x;
```

```
    double y;
```

```
    double z;
```

```
    Titik3D (double ax, double ay, double az) {
```

```
        x = ax;
```

```
        y = ay;
```

```
        z = az;
```

```
    }
```

```
}
```

Constructor

```
Class ContohTitik3D {
```

```
    public static void main(String args[]) {
```

```
        Titik3D p = new Titik3D(1.1, 3.4, -2.8);
```

```
        System.out.println("p.x = " + p.x);
```

```
        System.out.println("p.y = " + p.y);
```

```
        System.out.println("p.z = " + p.z);
```

```
    }
```

```
}
```

Hasil :

```
p.x = 1.1
```

```
p.y = 3.4
```

```
p.z = -2.8
```

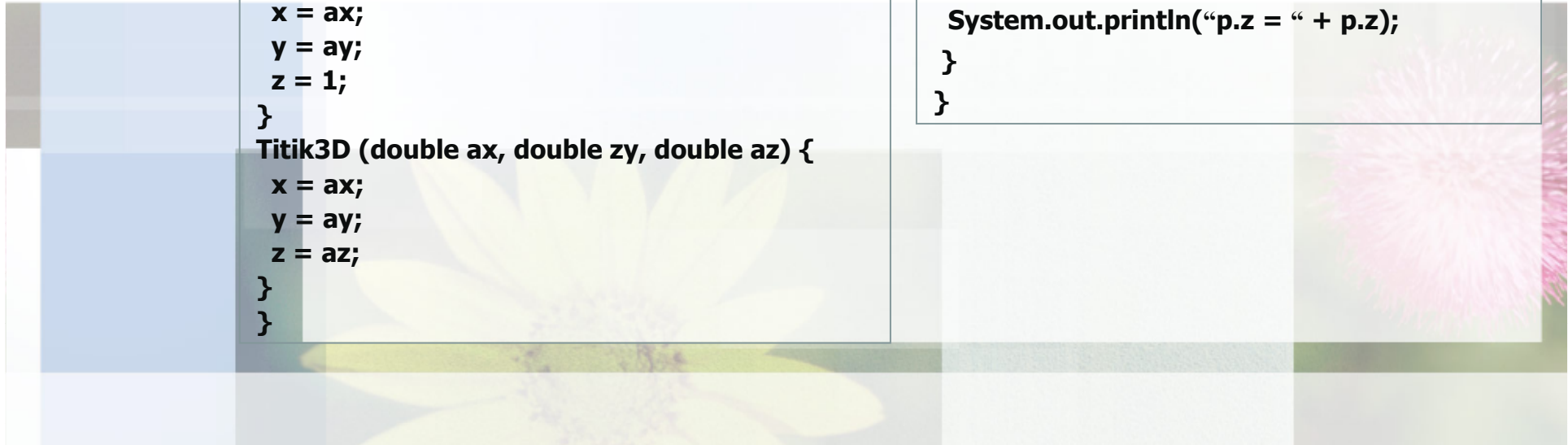
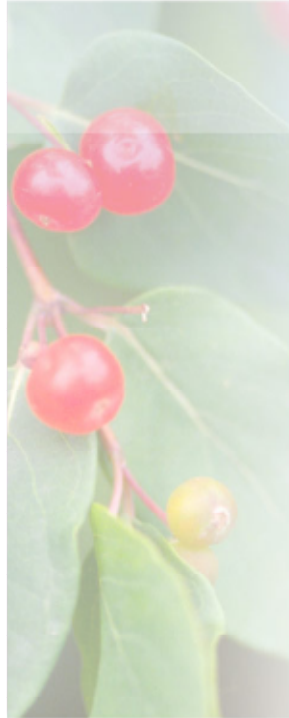
Constructor Overloading



Signature adalah informasi untuk membedakan method seperti nama method, jumlah parameter, tipe data, dan tipe return.

```
class Titik3D {  
    double x;  
    double y;  
    double z;  
    Titik3D (double ax) {  
        x = ax;  
        y = 1;  
        z = 1;  
    }  
    Titik3D (double ax, double zy) {  
        x = ax;  
        y = ay;  
        z = 1;  
    }  
    Titik3D (double ax, double zy, double az) {  
        x = ax;  
        y = ay;  
        z = az;  
    }  
}
```

```
class ContohTitik3D {  
    public static void main(String args[]) {  
        Titik3D p = new Titik3D(1.1);  
        System.out.println("p.x = " + p.x);  
        System.out.println("p.y = " + p.y);  
        System.out.println("p.z = " + p.z);  
        Titik3D p = new Titik3D(1.1, 3.4);  
        System.out.println("p.x = " + p.x);  
        System.out.println("p.y = " + p.y);  
        System.out.println("p.z = " + p.z);  
        Titik3D p = new Titik3D(1.1, 3.4, -2.8);  
        System.out.println("p.x = " + p.x);  
        System.out.println("p.y = " + p.y);  
        System.out.println("p.z = " + p.z);  
    }  
}
```



Keyword this



Keyword this

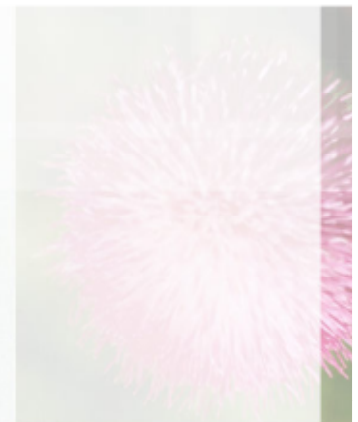
this.namaVar

Pemanggilan Constructors

this(args);

```
class Titik3D {  
    double x;  
    double y;  
    double z;  
  
    Titik3D(double x, double y, double z) {  
        this.x = x;  
        this.y = y;  
        this.z = z;  
    }  
}
```

```
class DemoThis {  
    public static void main(String args[]) {  
        Titik3D p = new Titik3D(1.1, 3.4, -2.8);  
        System.out.println("p.x = " + p.x);  
        System.out.println("p.y = " + p.y);  
        System.out.println("p.z = " + p.z);  
    }  
}
```



Local Variables and Variable Scope



```
class X {
    void f() {
        for( int j=0; j<5; j++) {
            int k=100;
            System.out.println("j= " + j + "; k= " +k);
        }
    }
}

class VariableScope {
    public static void main(String args[]) {
        X x = new X();
        x.f();
    }
}
```

Hasil :

```
j = 0; k = 100
j = 1; k = 100
j = 2; k = 100
j = 3; k = 100
j = 4; k = 100
```

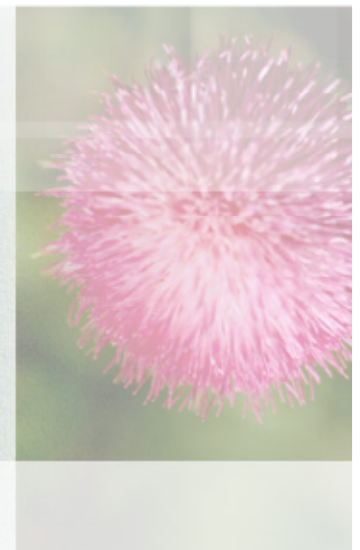
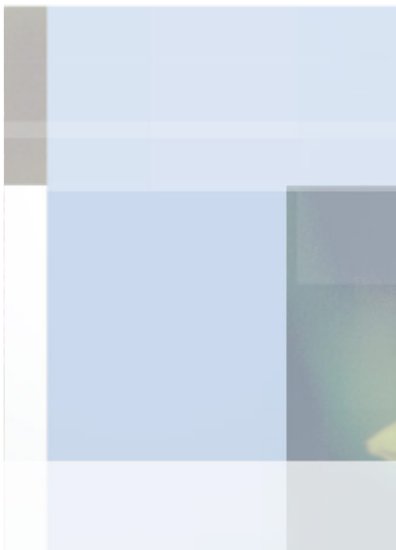
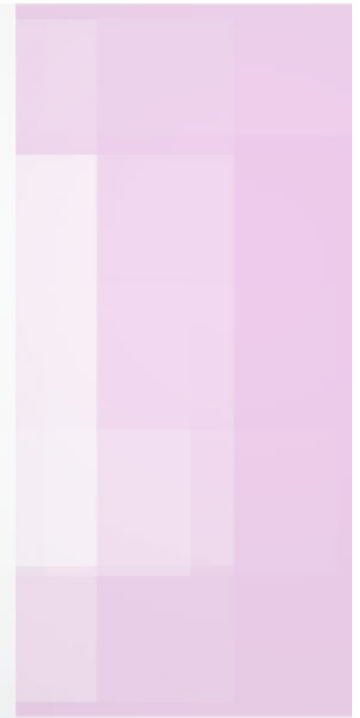
```
class Obyekku {
    static short s = 400; // variabel static
    int i = 200;
    void f() {
        System.out.println("s = " + s);
        System.out.println("i = " + i);
        short s = 300; // variabel local
        short i = 100; // variabel variable
        double d = 1E100; // variabel local
        System.out.println("s = " + s);
        System.out.println("i = " + i);
        System.out.println("d = " + d);
    }
}

class VariabelLocal {
    public static void main(String args[]) {
        Obyekku obyekku = new Obyekku();
        obyekku.f();
    }
}
```

Hasil :

```
s = 400
i = 200
s = 300
i = 100
d = 1.0E100
```


Materi Class Lanjutan



Instance Variables & Instance Methods



Deklarasi Instance Variable

```
Tipedata varName1;
```

Deklarasi Multiple Instance Variables

```
Tipedata namaVar1,namaVar2,... .. namaVarN;
```

Deklarasi dan Inisialisasi Instance Variable

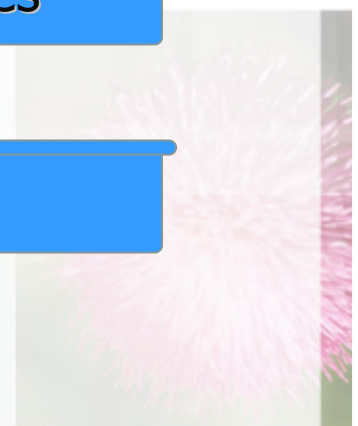
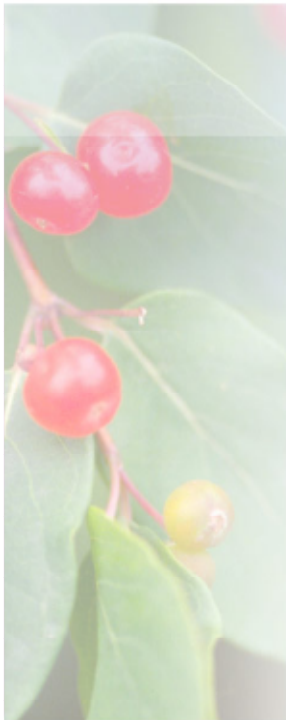
```
Tipedata namaVar1 = expr1;
```

Deklarasi dan Inisialisasi Multiple Instance Variables

```
Tipedata namaVar1=expr1, ... namaVarN = exprN;
```

Deklarasi Instance Method

```
tipeReturn namaMethod (parameters) {  
    // body method  
}
```

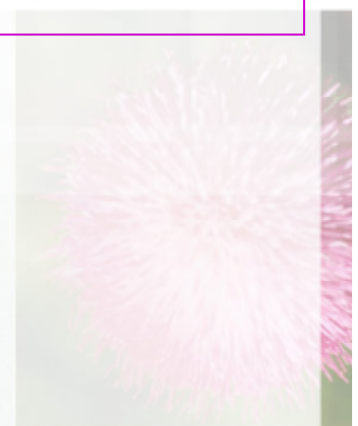


Instance Variables & Instance Methods



```
class Bag {
    boolean flag;
    int i, j=2, k=3, l, m;
    double array[] = {-3.4, 8.8e100, -9.2e-100 };
    String s1, s2= new String("Hello");
}
class BagTest {
    public static void main(String args[]) {
        Bag bag = new Bag();
        System.out.println(bag.flag);
        System.out.println(bag.i);
        System.out.println(bag.j);
        System.out.println(bag.k);
        System.out.println(bag.l);
        System.out.println(bag.m);
        for (int i=0; i < bag.array.length; i++)
            System.out.println(bag.array[i]);
        system.out.println(bag.s1);
        system.out.println(bag.s2);
    }
}
```

```
Hasil :
false
0
2
3
0
0
-3.4
8.8E100
-9.2E-100
null
Hello
```



Instance Variables & Instance Methods



```
class Titik3D {
    double x;
    double y;
    double z;

    Titik3D (double ax) {
        x = ax;
        y = 1;
        z = 1;
    }
    Titik3D (double ax, double ay) {
        x = ax;
        y = ay;
        z = 1;
    }
    Titik3D (double ax, double ay, double az) {
        x = ax;
        y = ay;
        z = az;
    }
    // Instance Method
    void pindah(double x, double y, double z) {
        this.x = x;
        this.y = y;
        this.z = z;
    }
}
```

```
Class Titik3DMethod {
    public static void main(String args[]) {
        Titik3D p = new Titik3D(1.1, 3.4, -2.8);
        System.out.println("p.x = " + p.x);
        System.out.println("p.y = " + p.y);
        System.out.println("p.z = " + p.z);
        p.pindah(5,5,5);
        System.out.println("p.x = " + p.x);
        System.out.println("p.y = " + p.y);
        System.out.println("p.z = " + p.z);
    }
}
```

Hasil :

```
p.x = 1.1
p.y = 3.4
p.z = -2.8
p.x = 5.0;
p.y = 5.0;
p.z = 5.0;
```


Static Variables & Static Methods



Deklarasi Static Variable

```
static tipeData namaVar1;
```

Deklarasi Multiple Static Variables

```
static tipeData namaVar1, namaVar2, ... namaVarN;
```

Deklarasi dan Inisialisasi Static Variable

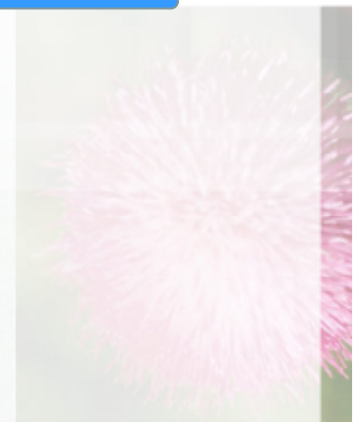
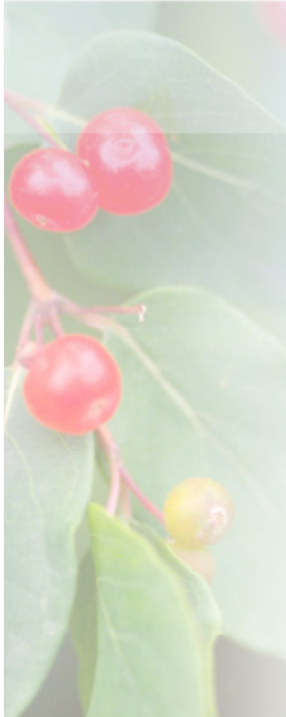
```
static tipeData namaVar1=expr1;
```

Deklarasi dan Inisialisasi Multiple Instance Variables

```
static tipeData namaVar1= expr1, ... namaVarN=exprN;
```

Static Initialization Block

```
class namaClass {  
    static {  
        // blok statement  
    }  
}
```



Static Variables & Static Methods



Deklarasi Static Method

```
static tipeReturn  
namaMethod(parameters) {  
  
// body method  
}
```

```
class StaticTas {  
    static boolean flag;  
    static int i, j=2, k=3, l, m;  
    static double array[] = {-3.4, 8.8e100, -9.2e-100};  
    static String s1, s2= new String("Halo");  
}  
class TasTest {  
    public static void main(String args[]) {  
        Tas Tas = new Tas();  
        System.out.println(StaticTas.flag);  
        System.out.println(StaticTas.i);  
        System.out.println(StaticTas.m);  
        for (int i=0; i < Tas.array.length; i++)  
            System.out.println(StaticTas.array[i]);  
        System.out.println(StaticTas.s1);  
        System.out.println(StaticTas.s2);  
    }  
}
```

```
class Bola {  
    static int count;  
    String nama;  
  
    Bola(String nama) {  
        this.nama = nama;  
        ++count;  
    }  
}  
//  
class StaticVariable {  
    public static void main(String args[]) {  
        Bola t1 = new Bola("Bola kasti");  
        System.out.println(t1.nama + " " +  
t1.count);  
        Bola t2 = new Bola("Bola Ping Pong");  
        System.out.println(t2.nama + " " +  
t2.count);  
        Bola t3 = new Bola("Sepak Bola");  
        System.out.println(t3.nama + " " +  
t3.count);  
    }  
}
```

Result :

Bola kasti 1

Bola Ping Pong 2

Sepak Bola 3

Static Variables & Static Methods



```
class X {
    static int array[];
    static {
        array = new int[6];
        for (int i = 0; i < 6; i++)
            array[i] = i;
    }
}
class InisialisasiStatic {
    public static void main(String args[]) {
        for (int i=0; i < 6; i++)
            System.out.println(X.array[i]);
    }
}
```

Hasil :

0
1
2
3
4
5

```
class PersamaanLinear {
    static double hasil(double a, double b) {
        return -b / a;
    }
}
class BlokInisialisasiStatic {
    public static void main(String args[]) {
        System.out.println(PersamaanLinear.hasil(2,2));
    }
}
```

Hasil :

-1.0

Method Overloading



Overloaded Methods

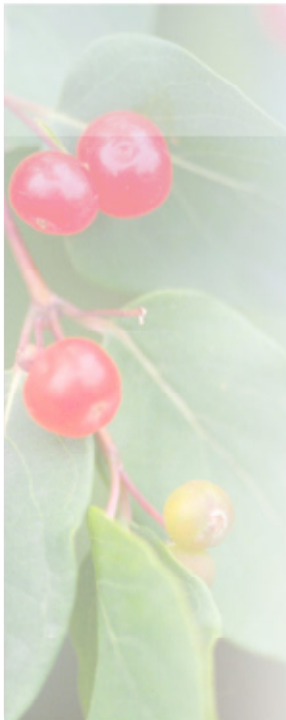
```
class Titik3D {
    double x;
    double y;
    double z;
    Titik3D (double x) {
        this(x,0,0);
    }
    Titik3D (double x, double y) {
        this(x,y,0);
    }
    Titik3D (double x, double y, double z) {
        this.x = x;
        this.y = y;
        this.z = z;
    }
    void pindah(double x, double y, double z) {
        this.x = x;
        this.y = y;
        this.z = z;
    }
    void pindah(double x, double y) {
        this.x = x;
        this.y = y;
    }
    void pindah(double x) {
        this.x = x;
    }
}
```

```
Class Titik3DOverloadMethods {
    public static void main(String args[]) {
        Titik3D p = new Titik3D(1.1, 3.4, -2.8);
        p.pindah(5);
        System.out.println("p.x = " + p.x);
        System.out.println("p.y = " + p.y);
        System.out.println("p.z = " + p.z);
        p.pindah(6,6);
        System.out.println("p.x = " + p.x);
        System.out.println("p.y = " + p.y);
        System.out.println("p.z = " + p.z);
        p.pindah(7,7,7);
        System.out.println("p.x = " + p.x);
        System.out.println("p.y = " + p.y);
        System.out.println("p.z = " + p.z);
    }
}
```

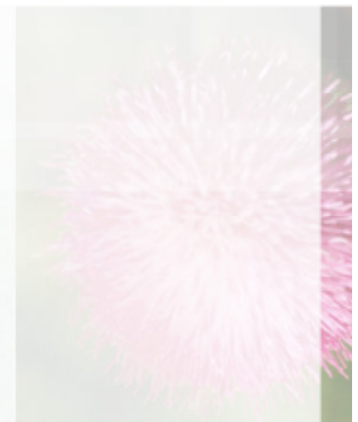
Hasil :

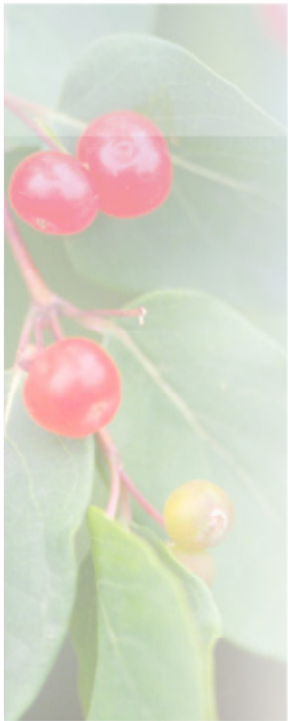
p.x = 5.0	p.x = 7.0
p.y = 3.4	p.x = 7.0
p.z = -2.8	p.x = 7.0
p.x = 6.0;	
p.y = 6.0;	
p.z = -2.8;	

Contoh penggunaan static method



```
class OuterClass {
    static class InnerClass {
        static String str;
        InnerClass(String s) {
            str = s;
        }
        void print() {                // Instance Method
            staticPrint(str);
        }
        static void staticPrint(String s) {    // Static Method
            str = s;
            System.out.println(s);
        }
    } // end of InnerClass
} // end of OuterClass
public class StaticInnerClass {
    public static void main(String[] args) {
        String s = "... without creating Outer-class object";
        OuterClass.InnerClass p = new OuterClass.InnerClass(s);
        p.print();
        OuterClass.InnerClass.staticPrint("call static method");
        p.print();
    }
}
```





selesai

5 Mei 2007

