**Java – Static Class, Block, Methods and Variable**

In this tutorial we will discuss the use of **static keyword in Java**. It can be used along with Class name, Variables, Methods and block.

1. static class  
2. static block  
3. [**static methods**](http://beginnersbook.com/2013/05/static-vs-non-static-methods/)  
4. [**static variables**](http://beginnersbook.com/2013/05/static-variable/)

**Static Class**

A Class can be made **static** only if it is a nested Class. The nested static class can be accessed without having an object of outer class.

**Example 1:**

class Example1{

//Static class

static class X{

static String str="Inside Class X";

}

public static void main(String args[])

{

X.str="Inside Class Example1";

System.out.println("String stored in str is- "+ X.str);

}

}

Output:

String stored in str is- Inside Class Example1

**Example 2: Compile time Error!!**

class Example2{

int num;

//Static class

static class X{

static String str="Inside Class X";

num=99;

}

public static void main(String args[])

{

Example2.X obj = new Example2.X();

System.out.println("Value of num="+obj.str);

}

}

Output:  
Compile time error. [**Static inner class**](http://beginnersbook.com/2013/05/inner-class/) cannot access instance data of outer class.

**Static Block**

Static block is mostly used for changing the default values of static variables.This block gets executed when the class is loaded in the memory.  
A class can have multiple Static blocks, which will execute in the same sequence in which they have been written into the program.

**Example 1: Single static block**

class Example3{

static int num;

static String mystr;

static{

num = 97;

mystr = "Static keyword in Java";

}

public static void main(String args[])

{

System.out.println("Value of num="+num);

System.out.println("Value of mystr="+mystr);

}

}

Output:

Value of num=97

Value of mystr=Static Keyword in Java

**Example 2: Multiple Static blocks**

class Example4{

static int num;

static String mystr;

//First Static block

static{

System.out.println("Static Block 1");

num = 68;

mystr = "Block1";

}

//Second static block

static{

System.out.println("Static Block 2");

num = 98;

mystr = "Block2";

}

public static void main(String args[])

{

System.out.println("Value of num="+num);

System.out.println("Value of mystr="+mystr);

}

}

**Output:**

Static Block 1

Static Block 2

Value of num=98

Value of mystr=Block2

**Static Methods**

Static Methods can access class variables without using object of the class. It can access non-static methods and non-static variables by using objects. Static methods can be accessed directly in static and non-static methods.

**Example 1: public static void main itself is a static method**

class Example5{

static int i;

static String s;

public static void main(String args[]) //Its a Static Method

{

Example5 obj=new Example5();

//Non Static variables accessed using object obj

System.out.println("i:"+obj.i);

System.out.println("s:"+obj.s);

}

}

**Output:**

i:0

s:null

**Example 2: Static method display()**

class Example6{

static int i;

static String s;

//Static method

static void display()

{

//Its a Static method

Example6 obj1=new Example6();

System.out.println("i:"+obj1.i);

System.out.println("i:"+obj1.i);

}

void funcn()

{

//Static method called in non-static method

display();

}

public static void main(String args[]) //Its a Static Method

{

//Static method called in another static method

display();

}

}

Output:

i:0

i:0

**4. Static Variables**

* Static variables are also known as Class Variables.
* Such variables get default values based on the data type.
* Data stored in static variables is common for all the objects( or instances ) of that Class.
* Memory allocation for such variables only happens once when the class is loaded in the memory.
* These variables can be accessed in any other class using class name.
* Unlike **non-static variables**, such variables can be accessed directly in static and non-static methods.

**Example 1: Static variables can be accessed without reference in Static method**

class Example7{

static int var1;

static String var2;

//Its a Static Method

public static void main(String args[])

{

System.out.println("Var1 is:"+Var1);

System.out.println("Var2 is:"+Var2);

}

}

**Output:**

Var1 is:0

Var2 is:null

As you can see in the above example that both the variables are accessed in void main method without any object(reference).

**Example 2: Static variables are common for all instances**

package beginnersbook.com;

class Example8{

static int Var1=77; //Static integer variable

String Var2;//non-static string variable

public static void main(String args[])

{

Example8 ob1 = new Example8();

Example8 ob2 = new Example8();

ob1.Var1=88;

ob1.Var2="I'm Object1";

ob2.Var2="I'm Object2";

System.out.println("ob1 integer:"+ob1.Var1);

System.out.println("ob1 String:"+ob1.Var2);

System.out.println("ob2 integer:"+ob2.Var1);

System.out.println("ob2 STring:"+ob2.Var2);

}

}

Output:

ob1 integer:88

ob1 String:I'm Object1

ob2 integer:88

ob2 String:I'm Object2

In above example String variable is non-static and integer variable is Static. So you can see that String variable value is different for both objects but integer variable value is common for both the instances as all the objects share the same copy of a static variable.