**Static fields**

Classes contain field declarations of the form

*modifiers Type identifier (=initialization-code);*

We use the modifier *static* to declare that a field is a class level property. This means that the field exists independently of any instances of a class and the value of the field is the same across all instance of the class. I.e. changes to static fields are visible to all instances.

public class Test {  
 public static void main(String[] args) {  
 System.*out*.println("The value of i,j is : " + A.*i* + "," + A.*j*);  
 A a1 = new A(1000);  
 A a2 = new A(2000);  
 System.*out*.println("The value of i,j according to a1 is : " + a1.*i* + "," + a1.*j*);  
 System.*out*.println("The value of i,j according to a2 is : " + a2.*i* + "," + a2.*j*);  
 System.*out*.println("The value of id according to a1,a2 is : " + a1.id + "," + a2.id);  
 a1.*i* = 5;  
 a2.*j* = 17;  
  
 a1.id = 3000;  
 System.*out*.println("The value of i,j according to a1 is : " + a1.*i* + "," + a1.*j*);  
 System.*out*.println("The value of i,j according to a2 is : " + a2.*i* + "," + a2.*j*);  
 System.*out*.println("The value of id according to a1,a2 is : " + a1.id + "," + a2.id);  
 }  
}  
class A {  
 public static int *i* = 10;  
 static public int *j* = 23;  
 int id = 0;  
 public A(int i){  
 id = i;  
 }  
}

Outputs:

The value of i,j is : 10,23

The value of i,j according to a1 is : 10,23

The value of i,j according to a2 is : 10,23

The value of id according to a1,a2 is : 1000,2000

The value of i,j according to a1 is : 5,17

The value of i,j according to a2 is : 5,17

The value of id according to a1,a2 is : 3000,2000

**Initialization code**

The initialization code for **static** fields gets executed at class object construction time.

The initialization code for **non-static** fields gets executed at object construction time.

**Static final**

To give a field that is declared as both final and static a value it must be initialized when the class object is constructed.

Static final are usually called constants.

**Static methods**

A static method is invoked on the class rather than on an object instance and thus cannot depend on the values of fields stored in any instance or use the keyword this.