**LAB NO: 7 Date:**

# CLASSES-ACCESS CONTROL, STATIC KEYWORD,FINAL ,WRAPPER CLASS

**Objectives:**

1. To know the different access specifiers used with the class

2. To learn the concept of nested, inner and wrapper classes

3. To study the variations of static keyword when used with methods and variables

4. To write Java programs using the concepts of access specifiers, nested, inner,

wrapper class, usage of static keyword with methods and variables.

* 1. **Access Specifiers**

Java provides a number of access modifiers to set access levels for classes, variables, methods, and constructors. The four access levels are −

* Visible to the package, the default. No modifiers are needed.
* Visible to the class only (private).
* Visible to the world (public).
* Visible to the package and all subclasses (protected).

Default Access Modifier - No Keyword

* 1. **Variation of using static keyword**

The static keyword is used in java mainly for memory management. It is used with variables, methods, blocks and nested class. It is a keyword that are used for share the same variable or method of a given class. This is used for a constant variable or a method that is the same for every instance of a class. The main method of a class is generally labeled static.

No object needs to be created to use static variable or call static methods, just put the class name before the static variable or method to use them. Static method can not call non-static method.

In java language static keyword can be used for following

* variable (also known as class variable)
* method (also known as class method)
* block
* nested class

**Static variable**

Any variable declared as static is known as **static variable**.

Static variable is used to fulfill the common requirement. For Example company name of employees, college name of students etc. Name of the college is common for all students.

The static variable allocate memory only once in class area at the time of class loading.

**Advantage of static variable**

Using static variable a program memory can be made efficient (i.e it saves memory).

**When and why to use static variable**

For example to store record of all employee of any company, in this case employee id is unique for every employee but company name is common for all. When a static variable such as a company name is created then only once memory is allocated otherwise it allocate a memory space each time for every employee.

**Syntax to declare static variable:**

*public static variableName;*

Syntax for declare static method:

public static void methodName()

{

.......

.......

}

**Syntax for access static methods and static variable:**

*className.variableName=10;*

*className.methodName();*

*Example*

*public static final double PI=3.1415;*

*public static void main(String args[])*

*{*

*......*

*......*

*}*

**Difference between static and final keyword**

static keyword always fixed the memory that means that will be located only once in the program where as final keyword always fixed the value that means it makes variable values constant.

* 1. **Nested,Inner,Wrapper Class**

1. Nested & Inner Class

The Java programming language allows it to define a class within another class. Such a class is called a nested class and is illustrated here:

class OuterClass {

...

class NestedClass {

...

}

}

**Terminology:** Nested classes are divided into two categories: static and non-static. Nested classes that are declared static are called static nested classes. Non-static nested classes are called inner classes.

class OuterClass {

...

static class StaticNestedClass {

...

}

class InnerClass {

...

}

}

A nested class is a member of its enclosing class. Non-static nested classes (inner classes) have access to other members of the enclosing class, even if they are declared private. Static nested classes do not have access to other members of the enclosing class. As a member of the OuterClass, a nested class can be declared private, public, protected, or package private. (Recall that outer classes can only be declared public or package private.)

**Why Use Nested Classes?**

Compelling reasons for using nested classes include the following:

It is a way of logically grouping classes that are only used in one place: If a class is useful to only one other class, then it is logical to embed it in that class and keep the two together. Nesting such "helper classes" makes their package more streamlined.

It increases encapsulation: Consider two top-level classes, A and B, where B needs access to members of A that would otherwise be declared private. By hiding class B within class A, A's members can be declared private and B can access them. In addition, B itself can be hidden from the outside world.

It can lead to more readable and maintainable code: Nesting small classes within top-level classes places the code closer to where it is used.

1. Wrapper Class

Wrapper classes are used to convert any data type into an Object type. The primitive data types are not objects. They do not belong to any class. They are defined in the language itself.

All the wrapper classes (Integer, Long, Byte, Double, Float, Short) are subclasses of the abstract class Number.

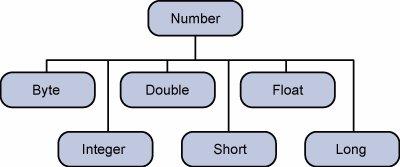


Fig 8.1 The hierarchy of the class Number.

Converting primitive data types into object is called boxing, and this is taken care by the compiler. Therefore, while using a wrapper class pass the value of the primitive data type to the constructor of the Wrapper class to convert primitive data into an Object of its owen type.

The Wrapper object can be converted back to a primitive data type, and the process is called unboxing. The Number class is part of the java.lang package.

Following is an example of boxing and unboxing −

Example

*public class Test {*

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

*Integer x = 5; // boxes int to an Integer object*

*x = x + 10; // unboxes the Integer to a int*

*System.out.println(x);*

*}*

*}*

o/p:15

**Lab exercises:**

1. Write a java program to store student record of college named “MIT”. Class Student\_Detail should contain name, id, college\_name as its members.display\_details() method should display the details of the students.
2. Write a counter program to count the number of objects created.
3. Write a java program to illustrate autoboxing and unboxing by considering all data types.

**Additional Exercises:**

1. **Write output of the following and analyze the code.**

|  |  |  |
| --- | --- | --- |
| 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);  }  } | 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);  }  } | 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);  }  } |
| 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);  }  } | 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);  }  } | 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();  }  } |