Object Oriented Programming

Final Classes, Wrapper Classes, Nested & Inner Classes

Compiled By: Umair Yaqub

Final Keyword In Java

The **final keyword** in java is used to restrict the user. The java final keyword can be used in many contexts. Final can be:

1. variable
2. method
3. class

1) Java final variable: If you make any variable as final, you cannot change the value of final variable (It will be constant).

Example: There is a final variable speedlimit, we are going to change the value of this variable, but It can't be changed because final variable once assigned a value can never be changed.

```
class Bike9{
 final int speedlimit=90;//final variable
 void run(){
  speedlimit=400;
 }
 public static void main(String args[]){
 Bike9 obj=new  Bike9();
 obj.run();
 }
}//end of class
```

Output: Compile Time Error

2) Java final method

If you make any method as final, you cannot override it.

Example:

```
class Bike{
1.  final void run(){System.out.println("running");}
}

class Honda extends Bike{
  void run(){System.out.println("running safely with 100kmph");}

  public static void main(String args[]){
  Honda honda= new Honda();
  honda.run();
  }
}
```

Output: Compile Time Error

---

3) Java final class

If you make any class as final, you cannot extend it.

Example:

```
final class Bike{}

class Honda1 extends Bike{
 void run(){System.out.println("running safely with 100kmph");}

 public static void main(String args[]){
 Honda1 honda= new Honda();
 honda.run();
 }
}
```

Output: Compile Time Error

Wrapper class in Java

Wrapper class in java provides the mechanism *to convert primitive into object and object into primitive*. **Autoboxing** and **unboxing** feature converts primitive into object and object into primitive automatically. The automatic conversion of primitive into object is known as autoboxing and vice-versa unboxing.

The eight classes of *java.lang* package are known as wrapper classes in java. The list of eight wrapper classes are given below:

Primitive Type	Wrapper class
boolean	Boolean
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double

Wrapper class Example: Primitive to Wrapper

```
public class WrapperExample1{
public static void main(String args[]){
//Converting int into Integer
int a=20;
Integer i=Integer.valueOf(a);          //converting int into Integer
Integer j=a;          //autoboxing, now compiler will write Integer.valueOf(a) internally
System.out.println(a+" "+i+" "+j);
}}
```

Output: 20 20 20

Wrapper class Example: Wrapper to Primitive

```
public class WrapperExample2{
public static void main(String args[]){
//Converting Integer to int
Integer a=new Integer(3);
int i=a.intValue();              //converting Integer to int
int j=a;                         //unboxing, now compiler will write a.intValue() internally

System.out.println(a+" "+i+" "+j);
}}
```

Output: 3 3 3

Java Inner Classes

Java inner class or nested class is a class which is declared inside the class or interface.

Additionally, it can access all the members of outer class including private data members and methods.

Syntax of Inner class

```
class Java_Outer_class{
 //code
        class Java_Inner_class{
         //code
         }
}
```

Advantage of java inner classes

There are basically three advantages of inner classes in java. They are as follows:

1) Nested classes represent a special type of relationship that is **it can access all the members (data members and methods) of outer class** including private.

2) Nested classes are used **to develop more readable and maintainable code** because it logically group classes and interfaces in one place only.

3) **Code Optimization**: It requires less code to write.

Difference between nested class and inner class in Java

Inner class is a part of nested class. Non-static nested classes are known as inner classes.

Types of Nested classes

There are two types of nested classes non-static and static nested classes. The non-static nested classes are also known as inner classes.

- Non-static nested class (inner class)
    1. Member inner class
    2. Anonymous inner class
    3. Local inner class
- Static nested class

Type	Description
Member Inner Class	A class created within class and outside method.
Anonymous Inner Class	A class created for implementing interface or extending class. Its name is decided by the java compiler.
Local Inner Class	A class created within method.
Static Nested Class	A static class created within class.
Nested Interface	An interface created within class or interface.
