transient keyword in Java

Difficulty Level : Medium Last Updated : 28 Jun, 2021

transient is a variables modifier used in serialization. At the time of serialization, if we don't want to save value of a particular variable in a file, then we use **transient** keyword. When JVM comes across **transient** keyword, it ignores original value of the variable and save default value of that variable data type.

transient keyword plays an important role to meet security constraints. There are various real-life examples where we don't want to save private data in file. Another use of **transient** keyword is not to serialize the variable whose value can be calculated/derived using other serialized objects or system such as age of a person, current date, etc.
Practically we serialized only those fields which represent a state of instance, after all serialization is all about to save state of an object to a file. It is good habit to use **transient** keyword with private confidential fields of a class during serialization.

```
// A sample class that uses transient keyword to
// skip their serialization.
class Test implements Serializable
{
    // Making password transient for security
    private transient String password;

    // Making age transient as age is auto-
    // computable from DOB and current date.
    transient int age;

    // serialize other fields
    private String username, email;
    Date dob;

    // other code
}
```

transient and static : Since **static** fields are not part of state of the object, there is no use/impact of using **transient** keyword with static variables. However there is no compilation error.

transient and final : final variables are directly serialized by their values, so there is no use/impact of declaring final variable as **transient**. There is no compile-time error though.

```
// Java program to demonstrate transient keyword
// Filename Test.java
import java.io.*;
class Test implements Serializable
{
    // Normal variables
    int i = 10, j = 20;

    // Transient variables
    transient int k = 30;

    // Use of transient has no impact here
    transient static int l = 40;
    transient final int m = 50;

    public static void main(String[] args) throws Exception
    {
        Test input = new Test();

        // serialization
        FileOutputStream fos = new FileOutputStream("abc.txt");
        ObjectOutputStream oos = new ObjectOutputStream(fos);
        oos.writeObject(input);

        // de-serialization
        FileInputStream fis = new FileInputStream("abc.txt");
        ObjectInputStream ois = new ObjectInputStream(fis);
        Test output = (Test)ois.readObject();
        System.out.println("i = " + output.i);
        System.out.println("j = " + output.j);
        System.out.println("k = " + output.k);
        System.out.println("l = " + output.l);
        System.out.println("m = " + output.m);
    }
}
```

Output :

```
i = 10
j = 20
k = 0
```

```
l = 40
m = 50
```

This article is contributed by **Gaurav Miglani**. If you like GeeksforGeeks and would like to contribute, you can also write an article using write.geeksforgeeks.org or mail your article to review-team@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.