Why Java is not a purely Object-Oriented Language?

Difficulty Level : Easy   Last Updated : 30 May, 2017

Pure Object Oriented Language or Complete Object Oriented Language are Fully Object Oriented Language which supports or have features which treats everything inside program as objects. It doesn’t support primitive datatype(like int, char, float, bool, etc.). There are seven qualities to be satisfied for a programming language to be pure Object Oriented. They are:

1. Encapsulation/Data Hiding
2. Inheritance
3. Polymorphism
4. Abstraction
5. All predefined types are objects
6. All user defined types are objects
7. All operations performed on objects must be only through methods exposed at the objects.

Example: Smalltalk

Why Java is not a Pure Object Oriented Language?

Java supports property 1, 2, 3, 4 and 6 but fails to support property 5 and 7 given above. Java language is not a Pure Object Oriented Language as it contain these properties:

- **Primitive Data Type ex. int, long, bool, float, char, etc as Objects:** Smalltalk is a “pure” object-oriented programming language unlike Java and C++ as there is no difference between values which are objects and values which are primitive types. In Smalltalk, primitive values such as integers, booleans and characters are also objects. In Java, we have predefined types as non-objects (primitive types).

```
int a = 5;
System.out.print(a);
```

- **The static keyword:** When we declares a class as static then it can be used without the use of an object in Java. If we are using static function or static variable then we can’t call that function or variable by using dot(.) or class object defying object oriented feature.
- **Wrapper Class:** Wrapper class provides the mechanism to convert primitive into object

and object into primitive. In Java, you can use Integer, Float etc. instead of int, float etc. We can communicate with objects without calling their methods. ex. using arithmetic operators.

```
String s1 = "ABC" + "A" ;
```

Even using Wrapper classes does not make Java a pure OOP language, as internally it will use the operations like Unboxing and Autoboxing. So if you create instead of int Integer and do any mathematical operation on it, under the hoods Java is going to use primitive type int only.

```
public class BoxingExample
{
    public static void main(String[] args)
    {
            Integer i = new Integer(10);
            Integer j = new Integer(20);
            Integer k = new Integer(i.intValue() + j.intValue());
            System.out.println("Output: "+ k);
    }
}
```

In the above code, there are 2 problems where Java fails to work as pure OOP:

1. While creating Integer class you are using primitive type “int” i.e. numbers 10, 20.
2. While doing addition Java is using primitive type “int”.

Related Article: Why C++ is partially Object Oriented Language?

This article is contributed by **Sangeet Anand**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@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.