Is Java a Pure Object Oriented Programming Language?

Hello guys, today, I am going to discuss one of the oldest questions related to Java, *whether Java is a pure object-oriented language or not? The short* answer is *no.* My answer is based on the fact that in a pure object-oriented language everything is an object and there are many things in Java that are not objects like primitive data types

like `boolean`, `char`, `short`, `int`, `long`, `float`, `double`, different kinds of arithmetic, logical and bitwise operator like `+`, `-`. `*`, `/`, `&&`, `||` etc. There are only a few pure OO programming languages are **Smalltalk** and **Eiffel,** If there is more, I may not know but Smalltalk is often touted as the purest form of an object-oriented language.

Though Java is probably the most successful Object-oriented programming language, which also got some functional programming touch in Java 8, it has been never considered 100% or pure object-oriented programming language. If it were, all its primitives would be objects.

It actually moves halfway in this direction with `String` (and perhaps `Array`), but it doesn't quite go far enough. Actually, one could argue that as String and Array aren't inheritable, that makes those parts of Java at best object-based.

Though, if you have to choose an object-oriented programming language for Software development, I would argue to choose Java because of its immense popularity, features, and community support. If you want to learn Java, I also recommend you to check out **The Complete Java Masterclass** by Tim Buchalaka on Udemy, one of the most up-to-date courses on Java.

Is Java is Pure Object-Oriented language?

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

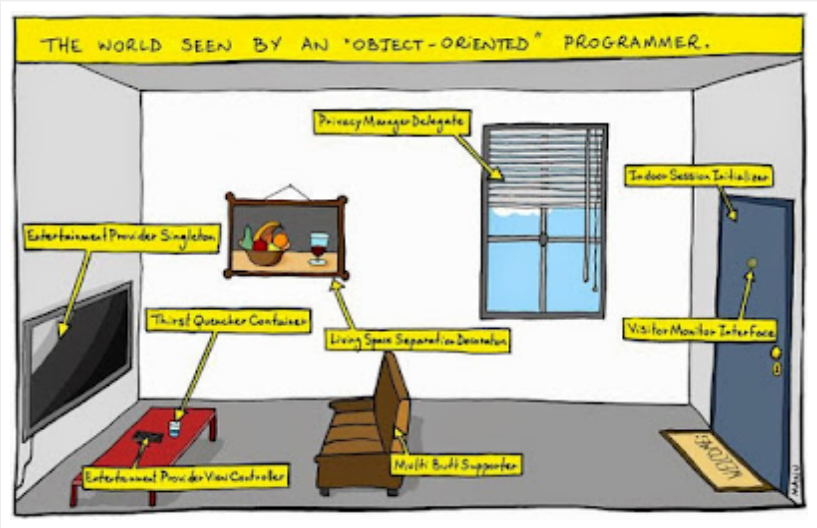
6. All operations are performed by sending messages to objects

7. All user-defined types are objects.

If you look at these seven qualities, Java does satisfy most of them. Java supports Encapsulation at class and package level, It supports Abstraction, Inheritance, and Polymorphism, and all user-defined types are also objects.

What it doesn't support is #5, all predefined types are not objects in Java, because you can define primitive types. This means it also violates #6. That's why Java is not a pure object-oriented language.

If you want to learn more about Object-Oriented Design in Java, you can further check out **Java Fundamentals: Object-oriented Design** course on Pluralsight, it's free for one month in April, make most of that time to go through this course.



Why Java is not a Pure Object-Oriented language?

`Smalltalk` is often considered one of the purest Object-oriented languages and comparing Java with `Smalltalk` will give you sufficient reasons, why Java is not 100% object-oriented language. The following point makes sense to me.

1) Primitive data types are either stored directly in fields or on the stack rather than on the heap. This is the *reason Java is not considered a pure OO programming language*

2) If we have to classify in "pure OO" or "non-pure OO" we have a problem. It’s better to talk

about "purity levels". `Smalltalk` has a higher purity level than Java. `"Primitive types"` in Smalltalk are actually `"Primitive Classes"` and in `Smalltalk,` all `"procedures"` or `"functions"` are really messages

On closing notes, You can make your program pure object-oriented by using Autoboxing, but the Java compiler supports primitive data types, so Java cannot be Pure object-oriented unless it makes everything in terms of objects.

In short, Java is not a pure object-oriented programming language because it supports primitive data types and everything is not an object in Java.