

OOPS Concepts in Java

If you're interested in mastering Object-Oriented Programming (OOPs) concepts in Java, check out our latest blog post!



Java OOPs Concepts

1. [Object-Oriented Programming](#)
2. [Advantage of OOPs over Procedure-oriented programming language](#)
3. [Difference between Object-oriented and Object-based programming language.](#)

In this page, we will learn about the basics of OOPs. Object-Oriented Programming is a paradigm that provides many concepts, such as **inheritance**, **data binding**, **polymorphism**, etc.

Simula is considered the first object-oriented programming language. The programming paradigm where everything is represented as an object is known as a truly object-oriented programming language.

What is OOPs concept in Java?

- ▯ In this article, we will get the fundamentals of **OOPs**. Object Oriented Development is a model that provides many ideas such as inheritance, data binding, polymorphism etc.
- ▯ Simula is known as the first object-oriented programming language. The programming model where everything is showed as a product, is known as truly object-oriented programming terminology.

Objects and Classes in Java

1. [Object in Java](#)
2. [Class in Java](#)
3. [Instance Variable in Java](#)
4. [Method in Java](#)
5. [Example of Object and class that maintains the records of student](#)
6. [Anonymous Object](#)

In this page, we will learn about Java objects and classes. In object-oriented programming technique, we design a program using objects and classes.

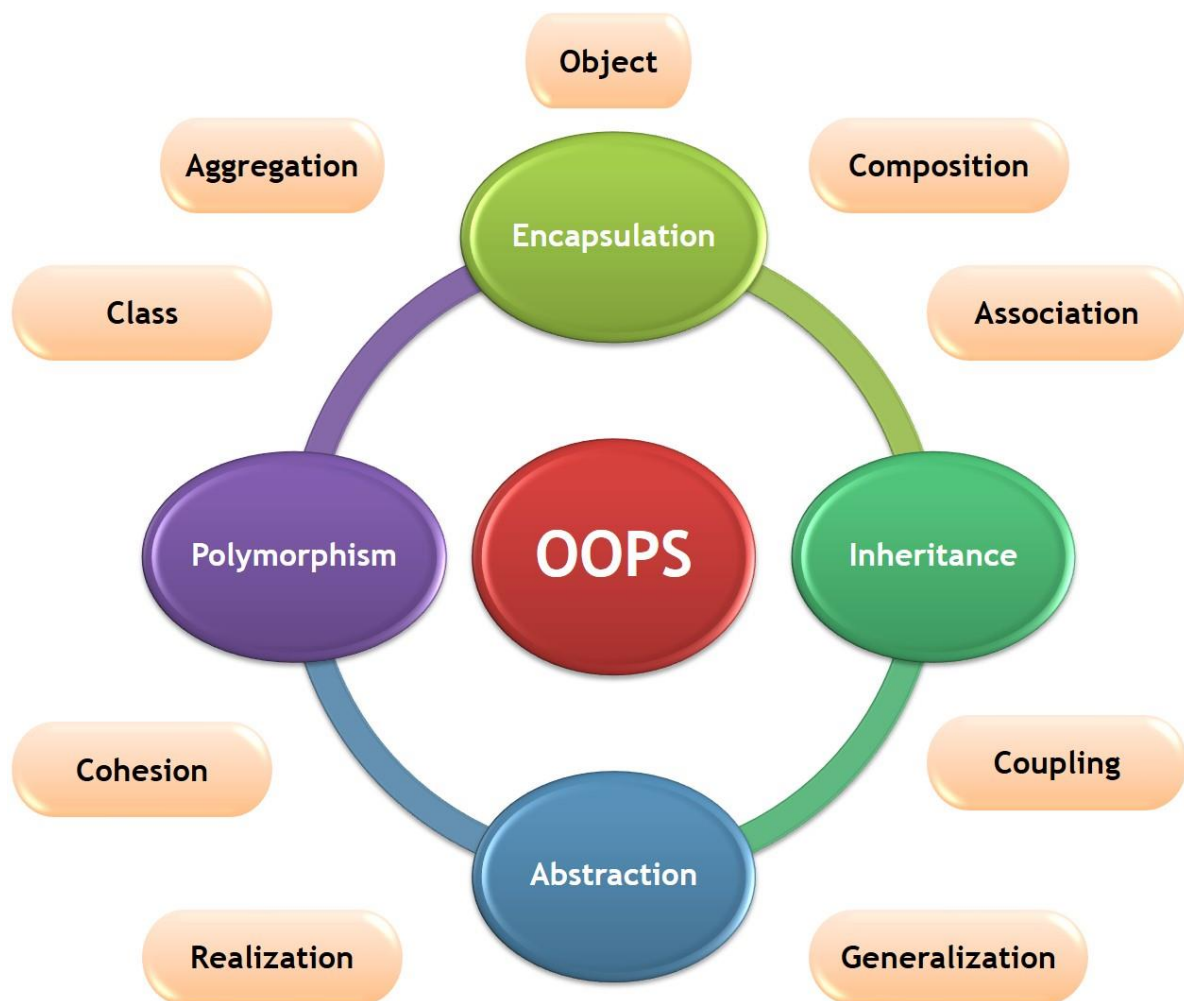
An object in Java is the physical as well as a logical entity, whereas, a class in Java is a logical entity only.

What is an object in Java

Objects: Real World Examples



An entity that has state and behavior is known as an object e.g., chair, bike, marker, pen, table, car, etc. It can be physical or logical (tangible and intangible). The example of an intangible object is the banking system.



Method Overloading in Java

1. [Different ways to overload the method](#)
2. [By changing the no. of arguments](#)
3. [By changing the datatype](#)
4. [Why method overloading is not possible by changing the return type](#)
5. [Can we overload the main method](#)
6. [method overloading with Type Promotion](#)

If a [class](#) has multiple methods having same name but different in parameters, it is known as **Method Overloading**.

If we have to perform only one operation, having same name of the methods increases the readability of the [program](#).

Suppose you have to perform addition of the given numbers but there can be any number of arguments, if you write the method such as `a(int,int)` for two parameters,

and b(int,int,int) for three parameters then it may be difficult for you as well as other programmers to understand the behavior of the method because its name differs.

So, we perform method overloading to figure out the program quickly.

Java OOPs Concepts

Abstraction

Hiding internal details and showing functionality is known as an abstraction. For example: phone call, we don't know the internal processing.

- In java, we use abstract class and interface to achieve abstraction.

Encapsulation

Binding (or wrapping) code and data together into a single unit is known as encapsulation. For example: capsule, it is wrapped with different medicines.

- A java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

Exceptions in Java

- Checked vs. Unchecked exceptions
 - **Checked** exceptions
 - If a method does NOT handle these, the method MUST state that it throws them
 - Done in a **throws clause** in the method header
 - These include IOException, and InterruptedException (and their subclasses)
 - **Unchecked** exceptions
 - Method not required to explicitly "throw" these
 - These include RuntimeException and Error