# POJO

## What is Pojo?

POJO stands for **Plain Old Java Object**. It is an ordinary Java object, not bound by any special restriction other than those forced by the Java Language Specification and not requiring any classpath. POJOs are used for increasing the readability and re-usability of a program. POJOs are now widely accepted due to their easy maintenance. They are easy to read and write. A POJO class does not have any naming convention for properties and methods. It is not tied to any Java Framework; any Java Program can use it.

The term POJO was introduced by **Martin Fowler** ( An American software developer) in 2000. it is available in Java from the EJB 3.0 by sun microsystem.

Generally, a POJO class contains variables and their Getters and Setters. The POJO classes are similar to Beans as both are used to define the objects to increase the readability and re-usability. The only difference between them that Bean Files have some restrictions but, the POJO files do not have any special restrictions.

## Example:

POJO class is used to define the object entities. For example, we can create an Employee POJO class to define its objects.

Below is an example of Java POJO class:

```
// POJO class Exmaple
package Jtp.PojoDemo;
public class Employee {
private String name;
private String id;
private double sal;
public String getName() {
   return name;
}
public void setName(String name) {
   this.name = name;
}
public String getId() {
   return id;
}
```

```
public void setId(String id) {
   this.id = id;
}
public double getSal() {
   return sal;
}
public void setSal(double sal) {
   this.sal = sal;
}
}
```

The above employee class is an example of an employee POJO class. If you are working on Eclipse, you can easily generate Setters and Getters by right click on the Java Program and navigate to **Source-> Generate Getters and Setters.**

## Properties of POJO class

Below are some properties of the POJO class:

- The POJO class must be public.
- It must have a public default constructor.
- It may have the arguments constructor.
- All objects must have some public Getters and Setters to access the object values by other Java Programs.
- The object in the POJO Class can have any access modifies such as private, public, protected. But, all instance variables should be private for improved security of the project.
- A POJO class should not extend predefined classes.
- It should not implement prespecified interfaces.
- It should not have any prespecified annotation.

# How to use POJO class in a Java Program

The POJO class is created to use the objects in other Java Programs. The major advantage of the POJO class is that we will not have to create objects every time in other Java programs. Simply we can access the objects by using the get() and set() methods.
To access the objects from the POJO class, follow the below steps:

- Create a POJO class objects
- Set the values using the set() method
- Get the values using the get() method
-

For example, create a MainClass.java class file within the same package and write the following code in it:

**MainClass.java:**

```
//Using POJO class objects in MainClass Java program
package Jtp.PojoDemo;
public class MainClass {
   public static void main(String[] args) {
      // Create an Employee class object
Employee obj= new Employee();
obj.setName("Alisha"); // Setting the values using the set() method
obj.setId("A001");
obj.setSal(200000);
System.out.println("Name: "+ obj.getName()); //Getting the values using the get() method
System.out.println("Id: " + obj.getId());
System.out.println("Salary: " +obj.getSal());
   }
}
```

**Output:**

```
Name: Alisha
Id: A001
Salary: 200000.0
```

From the above example, we can see we have accessed the POJO class properties in MainClass.java.

# DIFFERENCES BETWEEN POJO AND JAVA BEANS

| POJO | Bean |
|---|---|
| In Pojo, there are no special restrictions other than Java conventions. | It is a special type of POJO files, which have some special restrictions other than Java conventions. |
| It provides less control over the fields as compared to Bean. | It provides complete protection on fields. |
| The POJO file can implement the Serializable interface; but, it is not mandatory. | The Bean class should implement the Serializable interface. |
| The POJO class can be accessed by using their names. | The Bean class can only be accessed by using the getters and setters. |
| Fields may have any of the access modifiers such as public, private, protected. | Fields can only have private access. |
| In POJO, it is not necessary to have a no-arg constructor; it may or may not have it. | It must have a no-arg constructor. |
| There is not any disadvantage to using the POJO | The disadvantage of using the Bean is that the Default constructor and public setter can change the object state when it should be immutable. |