**Introduction**

POJO stands for Plain Old Java Class and POJI stands for Plain Old Java Interface. Entity beans are replaced by POJO (Plain Old Java class) classes which are simply referred to as entities. In EJB3 entity beans is replaced by POJO which can run outside the EJB container and doesn’t require any special interface. The concept of POJO is to simplify EJB programming. It is a framework which provides security, transaction, and distribution.

**Why are POJO and POJI needed?**

Earlier, writing EJB was a difficult task because programmers had to write a custom interface to handle the bean’s factory support and had to define the XML metadata to specify most basic bean behavior in EJB to run a program. With the help of POJO, now the programmers have to add some simple annotation, dependency, and callback. and we can write EJB in the same way we can write a core java class. EJB is a special class which can be transported from the network to the client and client to the network.

**How can the programmer override POJO behavior?**

POJO provides intelligent use of default. POJO has taken great care that the default value of the behavior or method matches the most common usage; so the programmers don't have to override it but if they want to override a method or behavior of a particular area, they can override it with the help of annotation or XML element.

Programmers need to use @Table annotation on it when the table name does not match the entity name.

POJO framework shifts all the unnecessary code to the server and allows the program to focus on the core application. It leads to easier and clutter-free code.