**Introduction to Java 8 Date and Time API**

Java 8 introduced new classes for date and time to resolve the long-standing issues of existing APIs: java.util.Date and Java.util.Calendar. These newly introduced immutable-value classes are easy-to-use, well-documented, and thread-safe.

**Why Java 8 new Date & Time API?**

Java 8 new date and time API was launched to address the following issues with the current date and time API:

**Thread-Safety** — Both java.util.Date and java.util.Calendar classes are not thread-safe. Therefore, developers have to deal with concurrency issues by writing additional codes. The new date and time APIs are immutable and thread-safe.

**Easier to Use & Understand** — The old APIs are poorly designed with little to no direct methods for basic date operations. The new date and time APIs introduced in Java 8 provide a consistent and easier-to-user ISO-centric interface for dealing with dates, times, durations, and periods. Tons of utility methods are included in the new API for simple date and time operations.

**Better Time Zone Handling** — Old APIs do not provide any direct way to handle different timezones. Developers were forced to write additional logic to deal with timezone issues. Java 8 new date and time API separated local and zoned date times handling into two categories. Thus, making it straightforward to handle different timezones without writing extra codes.

## **Local Date & Time API**

As a part of the local date and time API, Java 8 introduced **LocalDate**, **LocalTime**, and **LocalDateTime** classes. As their names suggest, these classes are intended to use for local date and time operations without timezones. You should use them when time zones are no longer required.