**Java 8 Date**

Java 8 Date Time API is [JSR-310](https://jcp.org/en/jsr/detail?id=310) implementation. It is designed to overcome all the flaws in the legacy date time implementations. Some of the design principles of new Date Time API are:

1. **Immutability**: All the classes in the new Date Time API are immutable and good for multithreaded environments.
2. **Separation of Concerns**: The new API separates clearly between human readable date time and machine time (unix timestamp). It defines separate classes for Date, Time, DateTime, Timestamp, Timezone etc.
3. **Clarity**: The methods are clearly defined and perform the same action in all the classes. For example, to get the current instance we have now() method. There are format() and parse() methods defined in all these classes rather than having a separate class for them.

All the classes use [Factory Pattern](http://www.journaldev.com/1392/factory-design-pattern-in-java) and [Strategy Pattern](http://www.journaldev.com/1754/strategy-design-pattern-in-java-example-tutorial) for better handling. Once you have used the methods in one of the class, working with other classes won’t be hard.

1. **Utility operations**: All the new Date Time API classes comes with methods to perform common tasks, such as plus, minus, format, parsing, getting separate part in date/time etc.
2. **Extendable**: The new Date Time API works on ISO-8601 calendar system but we can use it with other non ISO calendars as well.
3. Java Date Time classes are not defined consistently, we have Date Class in both java.util as well as java.sql packages. Again formatting and parsing classes are defined in java.text package.
4. java.util.Date contains both date and time, whereas java.sql.Date contains only date. Having this in java.sql package doesn’t make sense. Also both the classes have same name, that is a very bad design itself
5. There are no clearly defined classes for time, timestamp, formatting and parsing. We have java.text.DateFormat abstract class for parsing and formatting need. Usually SimpleDateFormatclass is used for parsing and formatting
6. Date class doesn’t provide internationalization, there is no timezone support. So java.util.Calendarand java.util.TimeZone classes were introduced

**Java 8 Date Time API Packages**

Java 8 Date Time API consists of following packages.



1. **java.time Package**: This is the base package of new Java Date Time API. All the major base classes are part of this package, such as LocalDate, LocalTime, LocalDateTime, Instant, Period, Duration etc. All of these classes are immutable and thread safe. Most of the times, these classes will be sufficient for handling common requirements.
2. **java.time.chrono Package**: This package defines generic APIs for non ISO calendar systems. We can extend AbstractChronology class to create our own calendar system.
3. **java.time.format Package**: This package contains classes used for formatting and parsing date time objects. Most of the times, we would not be directly using them because principle classes in java.time package provide formatting and parsing methods.
4. **java.time.temporal Package**: This package contains temporal objects and we can use it for find out specific date or time related to date/time object. For example, we can use these to find out the first or last day of the month. You can identify these methods easily because they always have format “withXXX”.
5. **java.time.zone Package**: This package contains classes for supporting different time zones and their rules.

**Java 8 Date Time API Examples**

We have looked into most of the important parts of Java Date Time API. It’s time now to look into most important classes of Date Time API with examples.

1. **LocalDate**

LocalDate is an immutable class that represents Date with default format of yyyy-MM-dd. We can use now() method to get the current date. We can also provide input arguments for year, month and date to create LocalDate instance. This class provides overloaded method for now() where we can pass ZoneId for getting date in specific time zone. This class provides the same functionality as java.sql.Date. Let’s look at a simple example for it’s usage.