Date/Time API (JSR 310)

The Date/Time API (JSR 310), introduced in Java 8, addresses the shortcomings of the previous date and time handling classes (java.util.Date, java.util.Calendar, etc.). The new API is found in the **java.time** package and offers a more comprehensive, consistent, and flexible way to handle dates and times.

Here are some key features and classes of the Date/Time API:

Key Features

1. **Immutability**: All classes in the java.time package are immutable and thread-safe.
2. **Fluent API**: Methods are designed for easy readability and chaining.
3. **Time Zones and Offsets**: Improved support for time zones and offsets.
4. **Comprehensive**: Covers dates, times, instants, durations, and periods.

Important Classes

1. **LocalDate**

    - Represents a date (year, month, day) without time or time zone.

    Example:

    ```
    LocalDate today = LocalDate.now();
    LocalDate birthDate = LocalDate.of(1990, Month.JANUARY, 1);
    ```

2. **LocalTime**

    - Represents a time (hour, minute, second, nanosecond) without date or time zone.

    Example:

```
LocalTime now = LocalTime.now();
LocalTime meetingTime = LocalTime.of(14, 30);
```

3. **LocalDateTime**

   - Combines LocalDate and LocalTime, representing both date and time without time zone.

   ```
   LocalDateTime now = LocalDateTime.now();
   LocalDateTime appointment = LocalDateTime.of(2024,
   Month.JULY, 22, 14, 30);
   ```

4. **ZonedDateTime**

   Represents a date and time with a time zone.

   ```
   ZonedDateTime zonedDateTime = ZonedDateTime.now();
   ZonedDateTime parisTime =
   ZonedDateTime.now(ZoneId.of("Europe/Paris"));
   ```

5. **Instant**

   - Represents a point in time (timestamp) with nanosecond precision.

   Example:

   ```
   Instant now = Instant.now();
   ```

6. **Duration**

   - Represents a duration or amount of time in seconds and nanoseconds.

   Example:

   ```
   Duration duration = Duration.ofHours(5);
   Duration between = Duration.between(startTime, endTime);
   ```

7. **Period**

    - Represents a date-based amount of time (years, months, days).

   Example:

   ```
   Period period = Period.ofYears(1);
   Period betweenDates = Period.between(startDate, endDate);
   ```

Parsing and Formatting

- The java.time.format package provides classes to parse and format dates and times.
- Example:

  ```
  DateTimeFormatter formatter =
  DateTimeFormatter.ofPattern("dd/MM/yyyy");
  LocalDate date = LocalDate.parse("22/07/2024", formatter);
  String formattedDate = date.format(formatter);
  ```