Date Time API

Shristi Technology Labs

Contents

- LocalTime
- LocalDate
- LocalDateTime
- ZonedDateTime
- Period
- Duration

Date Time API

- Is from **java.time** package
- Is ThreadSafe
- Supports Local and Zonal timezones
- TemporalAdjustments are possible
- Chrono units for getting day, month and year

Classes

- LocalDate
- LocalTime
- LocalDateTime
- ZonedDateTime
- ZoneId
- Period
- Duration

LocalDate

- represents **a date in ISO format (yyyy-MM-dd) without time**.
- It can be used to store dates like birthdays and paydays.
- Can get a specific day, month and year using the ***of*** method
- Can convert a date in String to date format using ***parse*** method.
- Has utility methods to add/ subtract days, month, years
- Can get the time interval(**Period)** between two days

```
LocalDate date = LocalDate.now();
```

LocalTime

- represents **time without a date**.
- An instance of *LocalTime* can be created from system clock or by using **parse** and **of** method
- Can get hour, minute and second units using the ***of*** method
- Can convert time in String to hour/minute using ***parse*** method.
- Has utitlity methods to add/ subtract hours, minute,seconds
- Can get the time interval(**Duration)**

```
LocalTime time = LocalTime.now();
```

Example

```
// current time
LocalTime time = LocalTime.now();
System.out.println("Current Time " + time);
// current date
LocalDate date = LocalDate.now();
System.out.println("Current Date " + date);
// pass date as string and get in date format
LocalDate ndate = LocalDate.parse("2017-04-02");
System.out.println("String to DATE " + ndate);

// pass int value and get in date format
LocalDate odate = LocalDate.of(2016, 12, 23);
System.out.println("Int to Date    " + odate);

LocalDate one = LocalDate.of(2016, 6, 30);
System.out.println(one);
LocalDate two = LocalDate.of(2016, 6, 1);
System.out.println(two);
System.out.println(one.isAfter(two));
```

LocalDateTime

- is used to represent **a combination of date and time**.
- is the most commonly used class
- Also has **of, parse, plus, minus** methods

```
LocalDateTime datetime = LocalDateTime.now();
```

Example

```
LocalDateTime datetime = LocalDateTime.now();
System.out.println("Current Date & Time  "+datetime);

Month month = datetime.getMonth();
System.out.println("Month "+month);
System.out.println("Year "+datetime.getYear());
System.out.println("day "+datetime.getDayOfMonth());
System.out.println("Hour "+datetime.getHour());
System.out.println("Minute "+datetime.getMinute());
System.out.println("Second "+datetime.getSecond());

//shows 17-2-2003
datetime = datetime.withDayOfYear(48).withYear(2003);
System.out.println("specific date "+datetime);
System.out.println();
```

ZonedDateTime

- ***ZonedDateTime*** *is used* with time zone specific date and time.
- ***ZoneId*** is an identifier to represent different zones. (about 40 different time zones

```
ZoneId zoneId = ZoneId.of("Europe/Paris");
ZoneId id = ZoneId.systemDefault();
ZonedDateTime zdate = ZonedDateTime.now(zoneID);
```

```
ZoneId id = ZoneId.systemDefault();
System.out.println(id);
// get all zone ids
Set<String> allIds = ZoneId.getAvailableZoneIds();
for (String zone : allIds) {
    ZoneId zoneID = ZoneId.of(zone);// zone name
    ZonedDateTime zdate = ZonedDateTime.now(zoneID);// date and time for
                                                    // the zone
    System.out.println(zoneID + "   " + zdate);
}
```

Period & Duration

Period

- Gives the time interval in terms of years, months and days
- use the ***between()*** method of the ***ChronoUnit*** class

Duration

- Gives the time interval in terms of seconds and nano seconds.

```
LocalDateTime date = LocalDateTime.now();
Period daysbetween =
        Period.between(date.toLocalDate(),futureDate.toLocalDate());
Duration duration =
        Duration.between(date.toLocalTime(), futureDate.toLocalTime());
```

Summary

- LocalTime
- LocalDate
- LocalDateTime
- ZonedDateTime
- Period
- Duration

Thank you