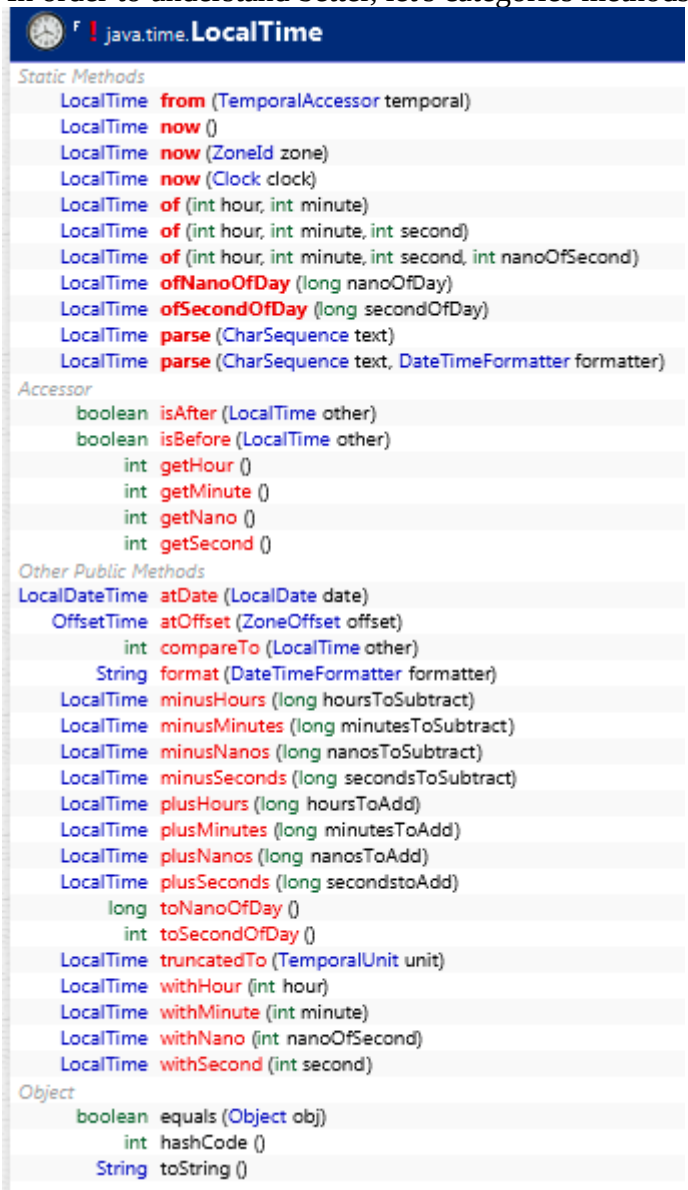
Java 8 - LocalTime Class API Guide

The `LocalTime` class is similar to the other classes whose names are prefixed with Local, but deals in time only. This class is useful for representing a human-based time of day, such as movie times, or the opening and closing times of the local library.
The java.time.LocalTime class is an immutable class which represents a time without time-zone information such as '28:28:38.111'.

In this post, we will discuss the important APIs of LocalTime Class with examples.
In order to understand better, let's categories methods into their usage and explained with examples.



LocalTime Class Methods/APIs

1. Methods to create a LocalTime object

- `static LocalTime now()` - Obtains the current time from the system clock in the default time-zone.

- `static LocalTime now(Clock clock)` - Obtains the current time from the specified clock.
- `static LocalTime now(ZoneId zone)` - Obtains the current time from the system clock in the specified time-zone.

2. Methods to get Hour, Minute, Second from LocalTime

- `int getHour()` - Gets the hour-of-day field.
- `int getMinute()` - Gets the minute-of-hour field.
- `int getNano()` - Gets the nano-of-second field.
- `int getSecond()` - Gets the second-of-minute field.

3. Methods to add or subtract hours, minutes and seconds to LocalTime

- `LocalTime plusHours(long hoursToAdd)` - Returns a copy of this LocalTime with the specified number of hours added.
- `LocalTime plusMinutes(long minutesToAdd)` - Returns a copy of this LocalTime with the specified number of minutes added.
- `LocalTime plusNanos(long nanosToAdd)` - Returns a copy of this LocalTime with the specified number of nanoseconds added.
- `LocalTime plusSeconds(long secondstoAdd)` - Returns a copy of this LocalTime with the specified number of seconds added.
- `LocalTime minusHours(long hoursToSubtract)` - Returns a copy of this LocalTime with the specified number of hours subtracted.
- `LocalTime minusMinutes(long minutesToSubtract)` - Returns a copy of this LocalTime with the specified number of minutes subtracted.
- `LocalTime minusNanos(long nanosToSubtract)` - Returns a copy of this LocalTime with the specified number of nanoseconds subtracted.
- `LocalTime minusSeconds(long secondsToSubtract)` - Returns a copy of this LocalTime with the specified number of seconds subtracted.

4. Methods to compare LocalTime objects in Java

- `int compareTo(LocalTime other)` - Compares this time to another time.
- `boolean isAfter(LocalTime other)` - Checks if this time is after the specified time.
- `boolean isBefore(LocalTime other)` - Checks if this time is before the specified time.

5. Methods to convert String to LocalTime in java

- `static LocalTime parse(CharSequence text)` - Obtains an instance of LocalTime from a text string such as 10:15.
- `static LocalTime parse(CharSequence text, DateTimeFormatter formatter)` - Obtains an instance of LocalTime from a text string using a specific formatter.

6. Method to convert LocalTime to String in java

- `String format(DateTimeFormatter formatter)` - Formats this time using the specified formatter.

- Reference: all the methods description available on LocalTime JavaDoc

Let's discuss above each method with examples.

1. LocalTime Methods to Create the Current Time and Specific Time

LocalTime class provides below APIs to a current time and specific time object respectively.

- *static LocalTime now()* - Obtains the current time from the system clock in the default time-zone.
- *static LocalTime now(Clock clock)* - Obtains the current time from the specified clock.
- *static LocalTime now(ZoneId zone)* - Obtains the current time from the system clock in the specified time-zone.

```
import java.time.Clock;
import java.time.LocalTime;
import java.time.ZoneId;

/**
 * Program to demonstrate LocalTime Class APIs.
 * @author javaguides.net
 *
 */
public class LocalTimeExample {

    public static void main(String[] args) {
         createLocalTime();
    }
    private static void createLocalTime() {
        // Current Time
        LocalTime localTime = LocalTime.now();
        System.out.println(localTime);

        // Specific Time
        LocalTime localTime2 = LocalTime.of(4, 30, 45);
        System.out.println(localTime2);

        LocalTime localTime3 = LocalTime.now(Clock.systemDefaultZone());
        System.out.println(localTime3);

        LocalTime localTime4 =
LocalTime.now(Clock.system(ZoneId.of("Indian/Cocos")));
```

```
        System.out.println(localTime4);
    }
}
```

Output:

```
17:38:35.349
04:30:45
17:38:35.350
18:38:35.350
```

2. LocalTime Methods get Hour, Minute, Second from LocalTime

LocalTime class provides below APIs to get Hour, Minute, Second from LocalTime.

- *int getHour()* - Gets the hour-of-day field.
- *int getMinute()* - Gets the minute-of-hour field.
- *int getNano()* - Gets the nano-of-second field.
- *int getSecond()* - Gets the second-of-minute field.

```
import java.time.LocalTime;

/**
 * Program to demonstrate LocalTime Class APIs.
 * @author javaguides.net
 *
 */
public class LocalTimeExample {

    public static void main(String[] args) {
        getHourMinuteSecondfromLocalTime();
    }
    private static void getHourMinuteSecondfromLocalTime() {
        LocalTime localTime = LocalTime.now();
        System.out.println("Gets the hour-of-day field : " + localTime.getHour());
        System.out.println("Gets the minute-of-hour field : " +
localTime.getMinute());
        System.out.println("Gets the second-of-minute field : " +
localTime.getSecond());
       System.out.println("Gets the nano-of-second field : " + localTime.getNano());
    }
}
```

Output:

```
Gets the hour-of-day field : 17
Gets the minute-of-hour field : 40
Gets the second-of-minute field : 30
Gets the nano-of-second field : 182000000
```

3. LocalTime Methods to add or subtract hours, minutes and seconds to LocalTime

LocalTime class provides below APIs to add or subtract hours, minutes and seconds to LocalTime.

- *LocalTime plusHours(long hoursToAdd)* - Returns a copy of this LocalTime with the specified number of hours added.
- *LocalTime plusMinutes(long minutesToAdd)* - Returns a copy of this LocalTime with the specified number of minutes added.
- *LocalTime plusNanos(long nanosToAdd)* - Returns a copy of this LocalTime with the specified number of nanoseconds added.
- *LocalTime plusSeconds(long secondstoAdd)* - Returns a copy of this LocalTime with the specified number of seconds added.
- *LocalTime minusHours(long hoursToSubtract)* - Returns a copy of this LocalTime with the specified number of hours subtracted.
- *LocalTime minusMinutes(long minutesToSubtract)* - Returns a copy of this LocalTime with the specified number of minutes subtracted.
- *LocalTime minusNanos(long nanosToSubtract)* - Returns a copy of this LocalTime with the specified number of nanoseconds subtracted.
- *LocalTime minusSeconds(long secondsToSubtract)* - Returns a copy of this LocalTime with the specified number of seconds subtracted.

```
import java.time.LocalTime;

/**
 * Program to demonstrate LocalTime Class APIs.
 * @author javaguides.net
 *
 */
public class LocalTimeExample {

    public static void main(String[] args) {
        addorSubtractHoursMinutesAndSecondstoLocalTime();
    }

    private static void addorSubtractHoursMinutesAndSecondstoLocalTime() {
         LocalTime localTime = LocalTime.now();
```

```
        System.out.println("Current Time : " + localTime);

        // LocalTime's plus methods
        System.out.println("Addition of Hours : " + localTime.plusHours(2));
        System.out.println("Addition of Minutes : " + localTime.plusMinutes(30));
        System.out.println("Addition of Seconds : " + localTime.plusSeconds(20));
        System.out.println("Addition of Nanos : " + localTime.plusNanos(60000));

        // LocalTime's minus methods
        System.out.println("Subtraction  of Hours : " + localTime.minusHours(2));
        System.out.println("Subtraction  of Minutes : " +
localTime.minusMinutes(30));
        System.out.println("Subtraction  of Seconds : " +
localTime.minusSeconds(20));
        System.out.println("Subtraction  of Nanos : " +
localTime.minusNanos(60000));
    }
}
```

Output:

```
Current Time : 17:41:55.127
Addition of Hours : 19:41:55.127
Addition of Minutes : 18:11:55.127
Addition of Seconds : 17:42:15.127
Addition of Nanos : 17:41:55.127060
Subtraction  of Hours : 15:41:55.127
Subtraction  of Minutes : 17:11:55.127
Subtraction  of Seconds : 17:41:35.127
Subtraction  of Nanos : 17:41:55.126940
```

4. LocalTime Methods to compare LocalTime objects in Java

LocalTime class provides below APIs to compare LocalTime objects in Java.

- *int compareTo(LocalTime other)* - Compares this time to another time.
- *boolean isAfter(LocalTime other)* - Checks if this time is after the specified time.
- *boolean isBefore(LocalTime other)* - Checks if this time is before the specified time.

```
import java.time.LocalTime;

/**
 * Program to demonstrate LocalTime Class APIs.
 * @author javaguides.net
```

```
 *
 */
public class LocalTimeExample {

    public static void main(String[] args) {
        compareLocalTimeObjects();
    }

    private static void compareLocalTimeObjects() {
        LocalTime localTime1 = LocalTime.of(9, 10, 50);
        LocalTime localTime2 = LocalTime.of(9, 10, 50);
        LocalTime localTime3 = LocalTime.of(11, 45, 20);

        // compareTo() example
        if (localTime1.compareTo(localTime2) == 0) {
            System.out.println("localTime1 and localTime2 are equal");
        } else {
            System.out.println("localTime1 and localTime2 are not equal");
        }

        // isBefore() example
        if (localTime2.isBefore(localTime3)) {
             System.out.println("localTime2 comes before localTime3");
        }

        // isAfter() example
        if (localTime3.isAfter(localTime1)) {
            System.out.println("localTime3 comes after localTime1");
        }
    }
}
```

Output:

```
localTime1 and localTime2 are equal
localTime2 comes before localTime3
localTime3 comes after localTime1
```

5. LocalTime Methods to convert String to LocalTime in java

LocalTime class provides below APIs to convert String to LocalTime in java.

- *`static LocalTime parse(CharSequence text)`* - Obtains an instance of LocalTime from a text string such as 10:15.

- *static LocalTime parse(CharSequence text, DateTimeFormatter formatter)* - Obtains an instance of LocalTime from a text string using a specific formatter.

```
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

/**
 * Program to demonstrate LocalTime Class APIs.
 * @author javaguides.net
 *
 */
public class LocalTimeExample {
    public static void main(String[] args) {
        convertStringToLocalTime();
    }
    private static void convertStringToLocalTime() {
        LocalTime isoTime = LocalTime.parse("10:15:30",
DateTimeFormatter.ISO_LOCAL_TIME);
        System.out.println(isoTime);

        // hour-of-day (0-23)
        LocalTime localTime = LocalTime.parse("22:45:30",
DateTimeFormatter.ofPattern("HH:mm:ss"));
        System.out.println(localTime);

        // clock-hour-of-am-pm (1-24)
        LocalTime localTime2 = LocalTime.parse("22:45:30",
DateTimeFormatter.ofPattern("kk:mm:ss"));
        System.out.println(localTime2);

        // clock-hour-of-am-pm (1-12)
        LocalTime localTime3 = LocalTime.parse("10:45:30 PM",
DateTimeFormatter.ofPattern("hh:mm:ss a"));
        System.out.println(localTime3);

        // hour-of-am-pm (0-11)
        LocalTime localTime4 = LocalTime.parse("10:45:30 AM",
DateTimeFormatter.ofPattern("KK:mm:ss a"));
        System.out.println(localTime4);
    }
}
```

Output:

```
10:15:30
22:45:30
22:45:30
22:45:30
10:45:30
```

6. LocalTime Methods to convert LocalTime to String in java

LocalTime class provides below API to convert LocalTime to String in java.

- *String format(DateTimeFormatter formatter)* - Formats this time using the specified formatter.

```
import java.time.LocalTime;
import java.time.format.DateTimeFormatter;

/**
 * Program to demonstrate LocalTime Class APIs.
 * @author javaguides.net
 *
 */
public class LocalTimeExample {

    public static void main(String[] args) {
        convertLocalTimeToString();
    }

    private static void convertLocalTimeToString(){
        LocalTime localTime = LocalTime.now();

       // ISO Format
       DateTimeFormatter timeFormatter = DateTimeFormatter.ISO_LOCAL_TIME;
       System.out.println(localTime.format(timeFormatter));

       // hour-of-day (0-23)
       DateTimeFormatter timeFormatter1 = DateTimeFormatter
              .ofPattern("HH:mm:ss");
       System.out.println(localTime.format(timeFormatter1));

       // clock-hour-of-am-pm (1-24)
       DateTimeFormatter timeFormatter2 = DateTimeFormatter
              .ofPattern("kk:mm:ss");
```

```
        System.out.println(localTime.format(timeFormatter2));

        // clock-hour-of-am-pm (1-12)
        DateTimeFormatter timeFormatter3 = DateTimeFormatter
                .ofPattern("hh:mm:ss a");
        System.out.println(localTime.format(timeFormatter3));

        // hour-of-am-pm (0-11)
        DateTimeFormatter timeFormatter4 = DateTimeFormatter
                .ofPattern("KK:mm:ss a");
        System.out.println(localTime.format(timeFormatter4));
    }
}
```

Output:

```
17:47:10.932
17:47:10
17:47:10
05:47:10 PM
05:47:10 PM
```

References

https://docs.oracle.com/javase/8/docs/api/java/time/LocalTime.html

Related Java 8 Date and Time Posts

- Date and Time API Guide
- Java 8 - LocalTime Class API Guide
- Java 8 - LocalDate Class API Guide
- Java 8 - LocalDateTime Class API Guide
- Java 8 - ZonedDateTime Class API Guide
- Java 8 - Duration Class API Guide
- Java 8 - Instant Class API Guide