The Java Date and Time API specification requires that Java uses a time scale that

• Has 86,400 seconds per day

• Exactly matches the official time at noon each day

• Closely matches it elsewhere, in a precisely defined way

nanos, millis, seconds, minutes, hours, days

There are two kinds of human time in the Java API, **local date/time** and **zoned time.**

**Local date/time** has a date and/or time of day, but no associated time zone information.

* An example of a local date is June 14, 1903 (the day on which Alonzo Church, inventor of the lambda calculus, was born)

July 16, 1969, 09:32:00 EDT (the launch of Apollo 11) is a **zoned date/time**, representing a precise instant on the timeline

the API designers recommend that you do not use **zoned time** unless you really want to represent absolute time instances.

Birthdays, holidays, schedule times, and so on are usually best represented as **local dates or times**.

**LocalDate** is a date with a year, month, and day of the month

difference between two time instats is a **Duration**.

The equivalent for local dates is a **Period**

**until** method yields the difference between two local dates. For example,

independenceDay.until(christmas) // 5 months 21 days

independenceDay.until(christmas, ChronoUnit.DAYS) // 174 days

.getDayOfWeek()

For scheduling applications, you often need to compute dates such as “the first Tuesday of every month.” The **TemporalAdjusters** class provides a number of static methods for common adjustments

A **LocalTime** represents a time of day, such as 15:30:00

You also need to pay attention when adjusting a date across daylight savings time boundaries.

For example, if you set a meeting for next week, don’t add a duration of seven days:

ZonedDateTime nextMeeting = meeting.plus(Duration.ofDays(7));

// Caution! Won't work with daylight savings time Instead, use the Period

class.

ZonedDateTime nextMeeting = meeting.plus(Period.ofDays(7)); // OK

Types of date time formatter

* Predefined standard formatters (see Table)
* Locale-specific formatters
* Formatters with custom patterns



