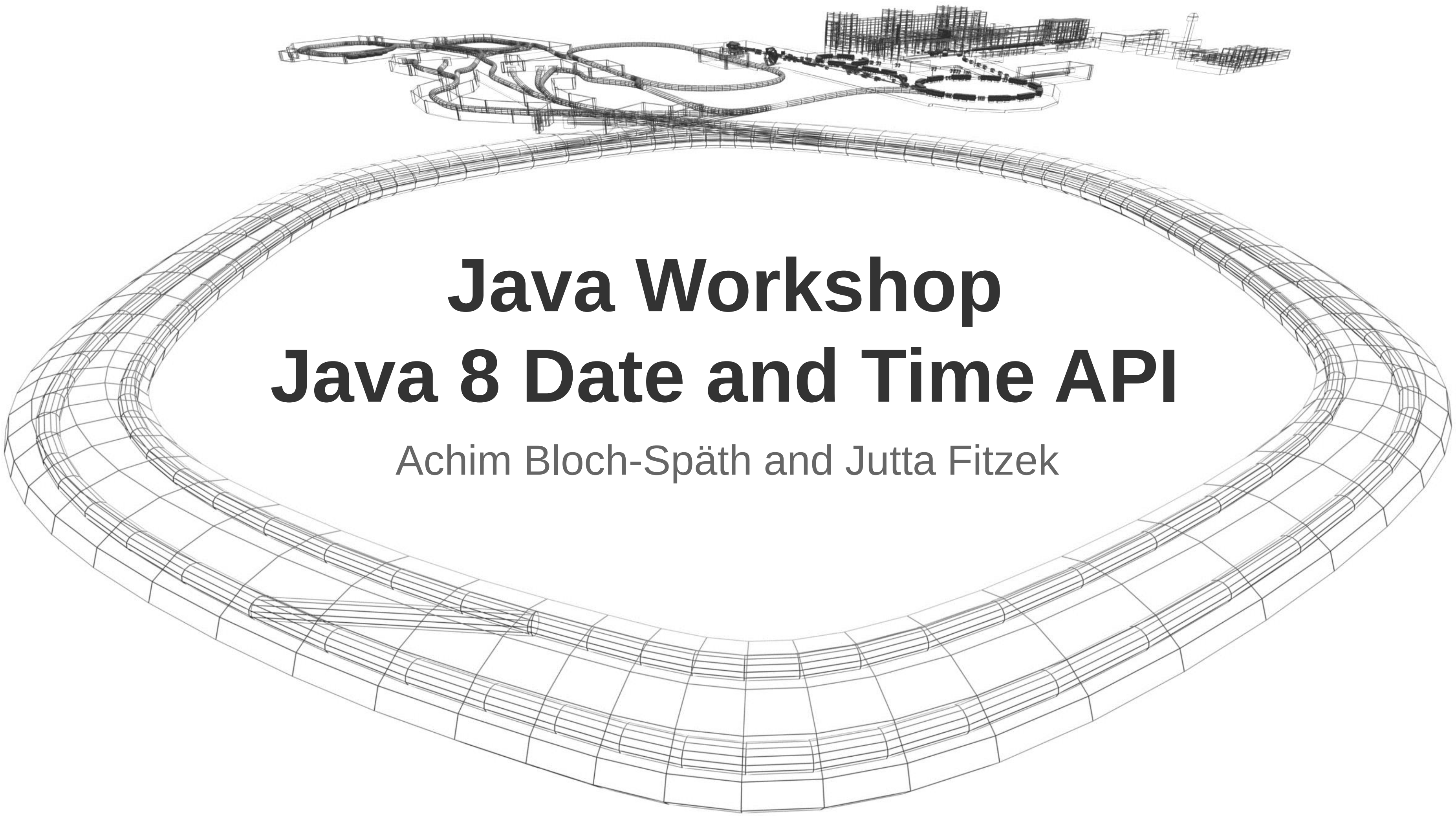
GSI Helmholtzzentrum für Schwerionenforschung GmbH



Java Workshop
Java 8 Date and Time API

Achim Bloch-Späth and Jutta Fitzek

Basics

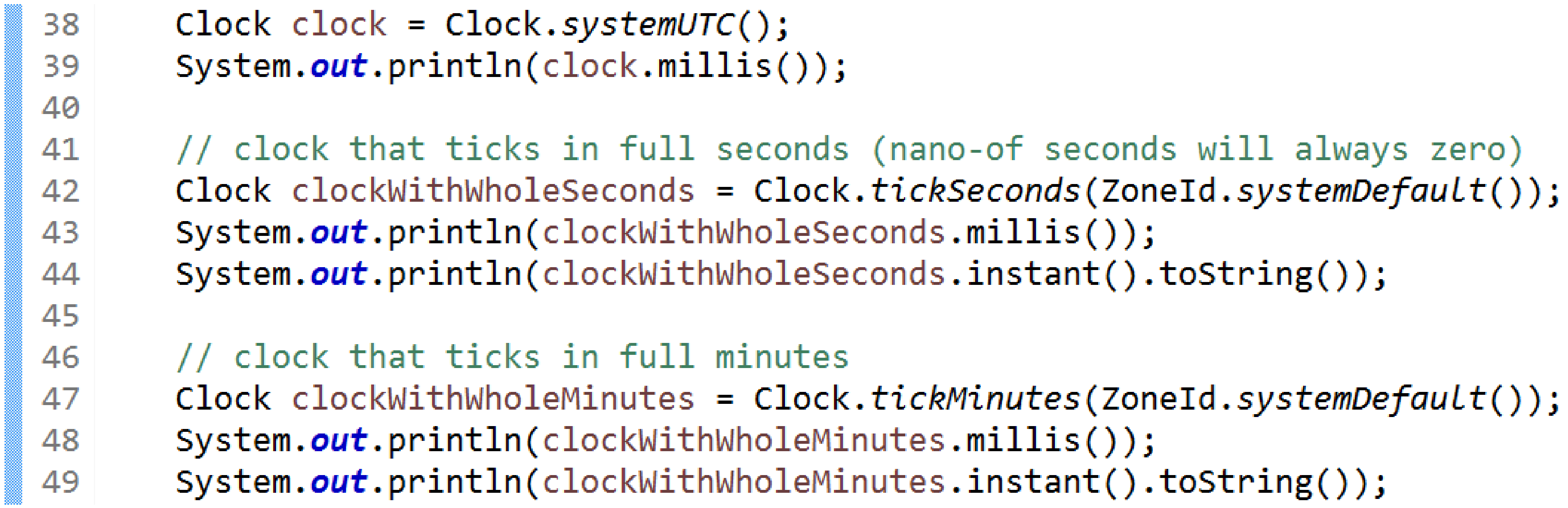
- Bases on Joda-Time Library (introduced with JSR 310)
- Immutable Objects
- Thread Safe
- APIs now through NullPointerExceptions when arguments are null (changed behavior!)
- Month starts with 1“

Package Overview

Package	Description
java.time	The core of the API for representing date and time. It includes classes for date, time, date and time combined, time zones, instants, duration, and clocks. These classes are based on the calendar system defined in ISO-8601, and are immutable and thread-safe.
java.time.chrono	The API for representing calendar systems other than the default ISO-8601. You can also define your own calendar system. This tutorial does not cover this package in any detail.
java.time.format	Classes for formatting and parsing dates and times.
java.time.temporal	Extended API, primarily for framework and library writers, allowing interoperations between the date and time classes, querying, and adjustment. Fields (TemporalField and ChronoField) and units (TemporalUnit and ChronoUnit) are defined in this package.
java.time.zone	Classes that support time zones, offsets from time zones, and time zone rules. If working with time zones, most developers will need to use only ZonedDateTime, and ZoneId or ZoneOffset.

Clock

- new wrapper around System.currentTimeMillis()



```java
    Clock clock = Clock.systemUTC();
    System.out.println(clock.millis());

    // clock that ticks in full seconds (nano-of seconds will always zero)
    Clock clockWithWholeSeconds = Clock.tickSeconds(ZoneId.systemDefault());
    System.out.println(clockWithWholeSeconds.millis());
    System.out.println(clockWithWholeSeconds.instant().toString());

    // clock that ticks in full minutes
    Clock clockWithWholeMinutes = Clock.tickMinutes(ZoneId.systemDefault());
    System.out.println(clockWithWholeMinutes.millis());
    System.out.println(clockWithWholeMinutes.instant().toString());
```

LocalDateTime

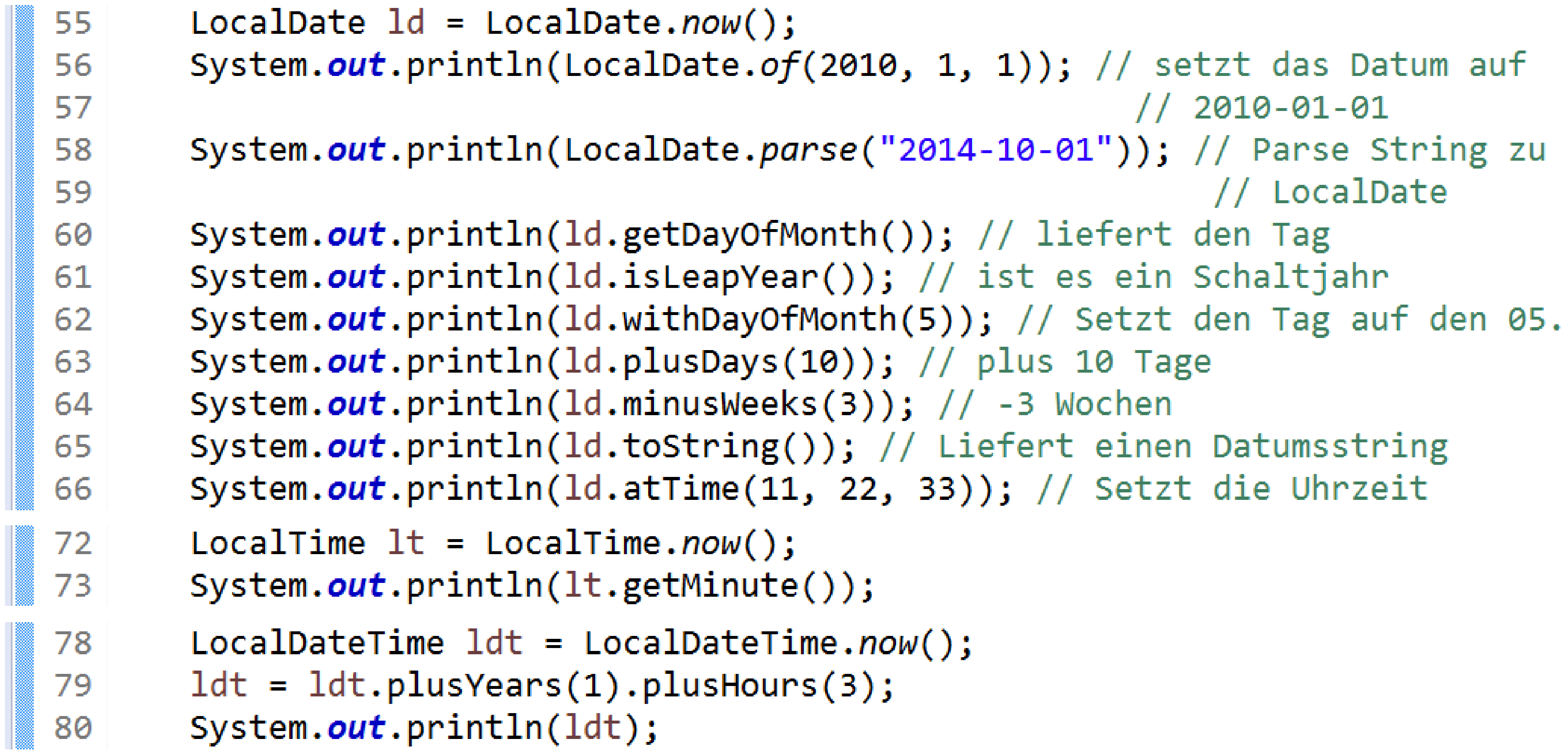
- LocalDateTime contains information without relation to any timezone

LocalDate	a date, without time of day, offset or zone
LocalTime	the time of day, without date, offset or zone
LocalDateTime	the date and time, without offset or zone

Unification of Methods

of	static factory method
parse	static factory method focused on parsing
get	gets the value of something
is	checks if something is true
with	the immutable equivalent of a setter
plus	adds an amount to an object
minus	subtracts an amount from an object
to	converts this object to another type
at	combines this object with another, such as date.atTime(time)

Examples



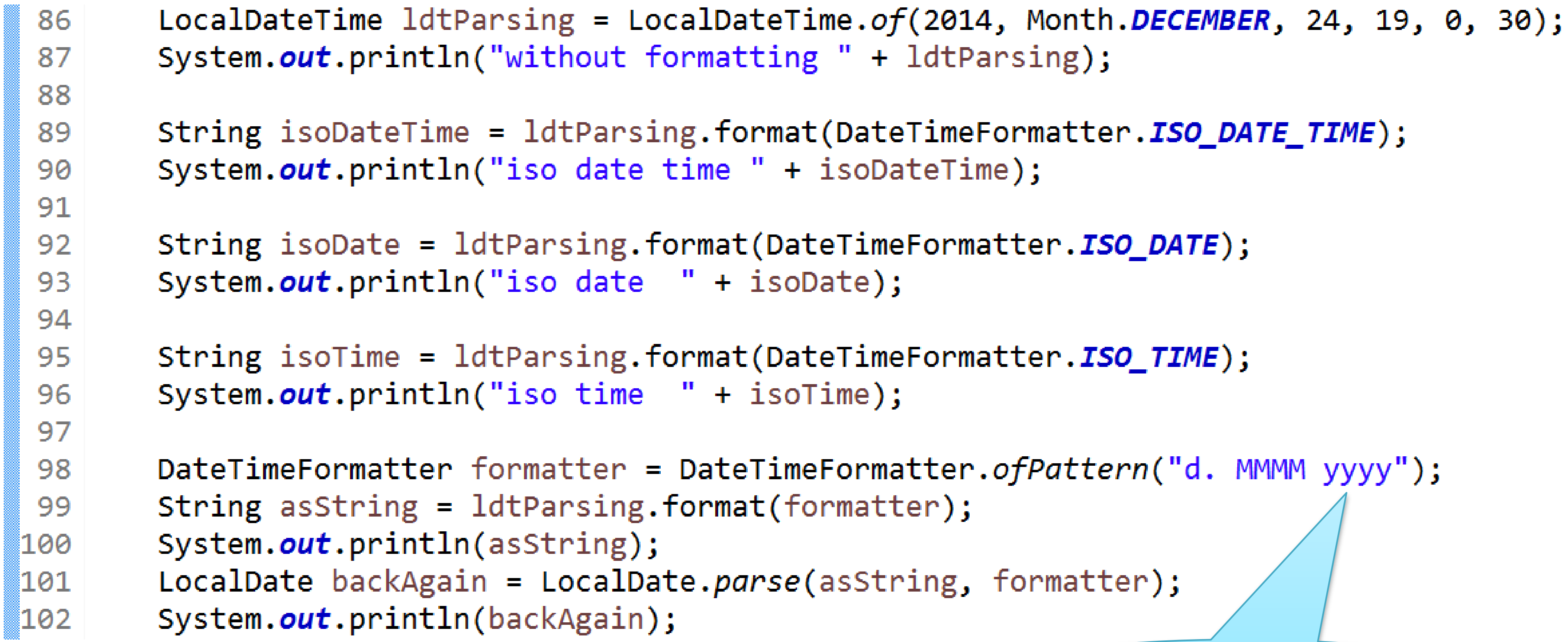
```java
    LocalDate ld = LocalDate.now();
    System.out.println(LocalDate.of(2010, 1, 1)); // setzt das Datum auf
                                                    // 2010-01-01
    System.out.println(LocalDate.parse("2014-10-01")); // Parse String zu
                                                        // LocalDate
    System.out.println(ld.getDayOfMonth()); // liefert den Tag
    System.out.println(ld.isLeapYear()); // ist es ein Schaltjahr
    System.out.println(ld.withDayOfMonth(5)); // Setzt den Tag auf den 05.
    System.out.println(ld.plusDays(10)); // plus 10 Tage
    System.out.println(ld.minusWeeks(3)); // -3 Wochen
    System.out.println(ld.toString()); // Liefert einen Datumsstring
    System.out.println(ld.atTime(11, 22, 33)); // Setzt die Uhrzeit
```

```java
    LocalTime lt = LocalTime.now();
    System.out.println(lt.getMinute());
```

```java
    LocalDateTime ldt = LocalDateTime.now();
    ldt = ldt.plusYears(1).plusHours(3);
    System.out.println(ldt);
```

Parsing and Formatting

- format() is used for formatting, parse() for parsing

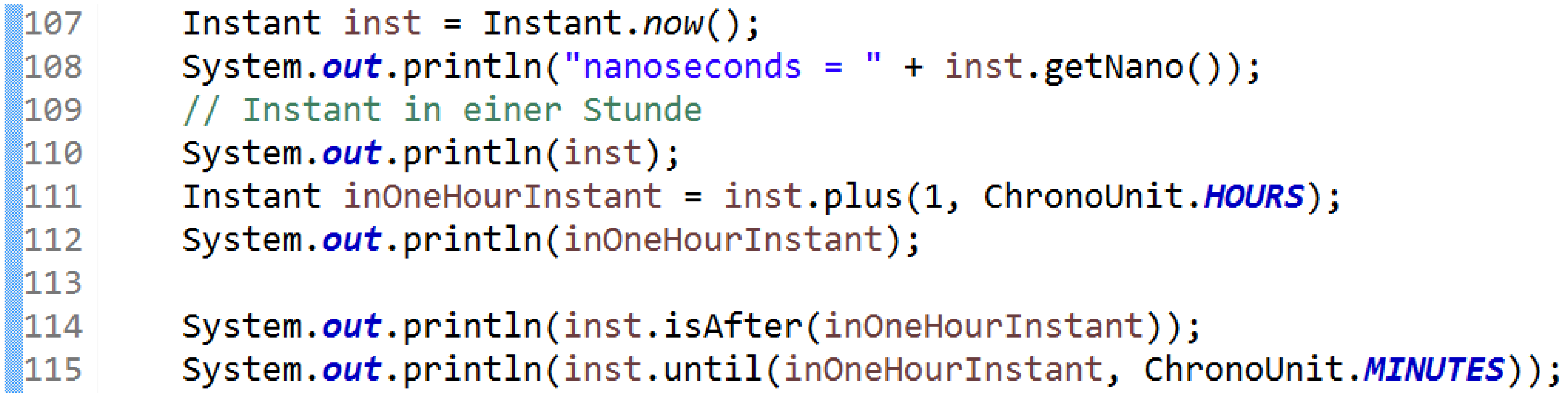


```java
LocalDateTime ldtParsing = LocalDateTime.of(2014, Month.DECEMBER, 24, 19, 0, 30);
System.out.println("without formatting " + ldtParsing);

String isoDateTime = ldtParsing.format(DateTimeFormatter.ISO_DATE_TIME);
System.out.println("iso date time " + isoDateTime);

String isoDate = ldtParsing.format(DateTimeFormatter.ISO_DATE);
System.out.println("iso date  " + isoDate);

String isoTime = ldtParsing.format(DateTimeFormatter.ISO_TIME);
System.out.println("iso time  " + isoTime);

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("d. MMMM yyyy");
String asString = ldtParsing.format(formatter);
System.out.println(asString);
LocalDate backAgain = LocalDate.parse(asString, formatter);
System.out.println(backAgain);
```

http://docs.oracle.com/javase/tutorial/i18n/format/simpleDateFormat.html

Date and Time API

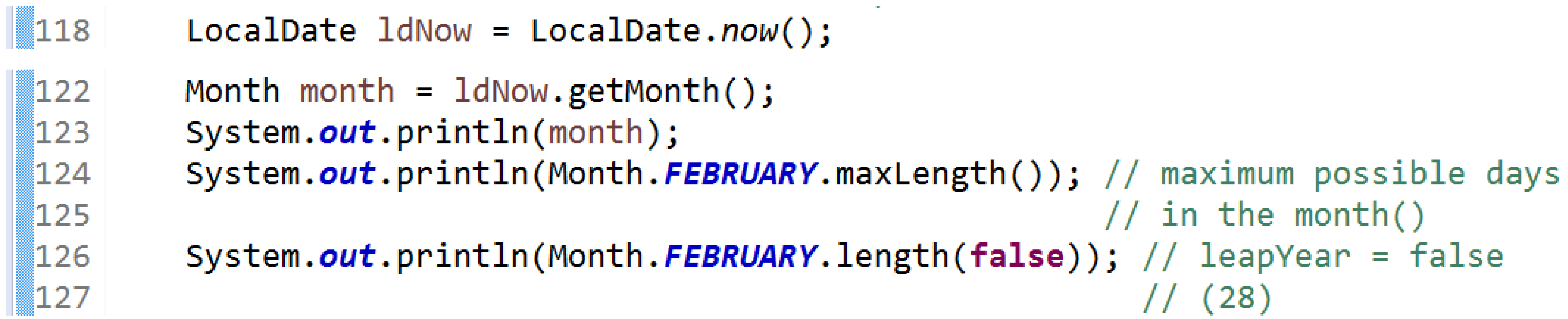
- Instant



```java
        Instant inst = Instant.now();
        System.out.println("nanoseconds = " + inst.getNano());
        // Instant in einer Stunde
        System.out.println(inst);
        Instant inOneHourInstant = inst.plus(1, ChronoUnit.HOURS);
        System.out.println(inOneHourInstant);

        System.out.println(inst.isAfter(inOneHourInstant));
        System.out.println(inst.until(inOneHourInstant, ChronoUnit.MINUTES));
```

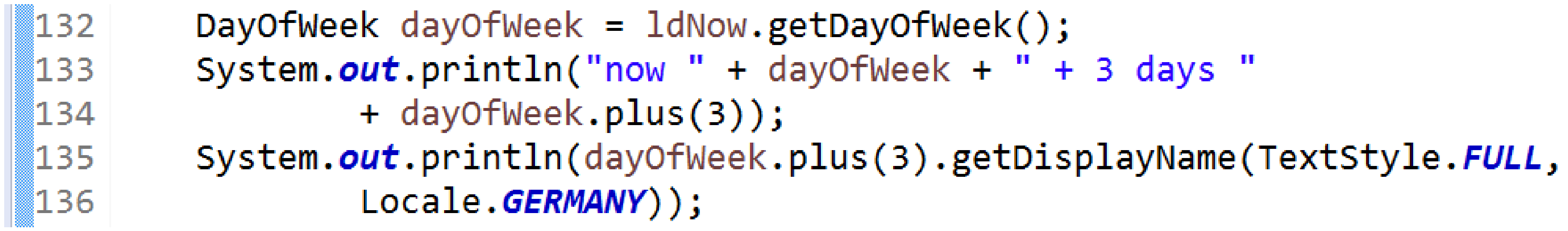
Date and Time API

- Month



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

Month month = ldNow.getMonth();
System.out.println(month);
System.out.println(Month.FEBRUARY.maxLength()); // maximum possible days
                                                // in the month()
System.out.println(Month.FEBRUARY.length(false)); // leapYear = false
                                                  // (28)
```

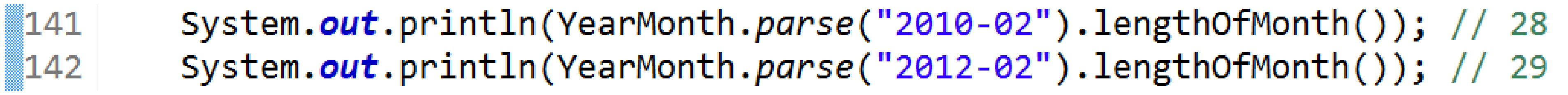
- DayOfWeek



```
DayOfWeek dayOfWeek = ldNow.getDayOfWeek();
System.out.println("now " + dayOfWeek + " + 3 days "
        + dayOfWeek.plus(3));
System.out.println(dayOfWeek.plus(3).getDisplayName(TextStyle.FULL,
        Locale.GERMANY));
```

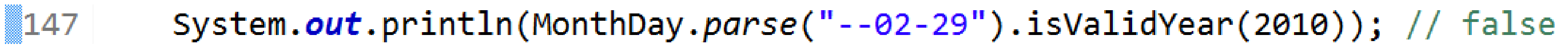
Date and Time API

- YearMonth



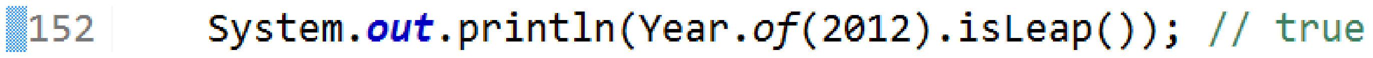
```
    System.out.println(YearMonth.parse("2010-02").lengthOfMonth()); // 28
    System.out.println(YearMonth.parse("2012-02").lengthOfMonth()); // 29
```

- MonthDay



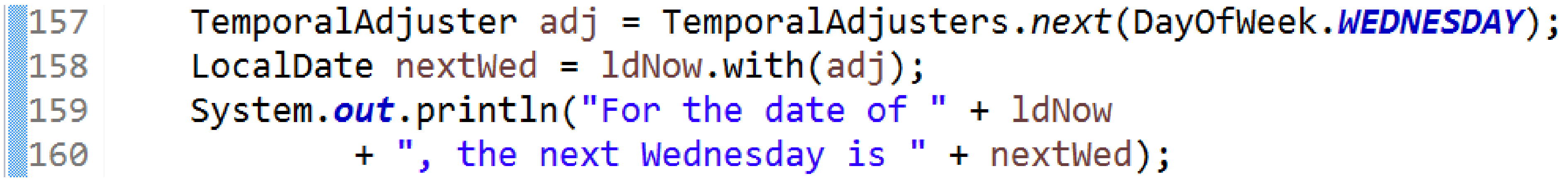
```
    System.out.println(MonthDay.parse("--02-29").isValidYear(2010)); // false
```

- Year:



```
    System.out.println(Year.of(2012).isLeap()); // true
```

Temporal Adjuster

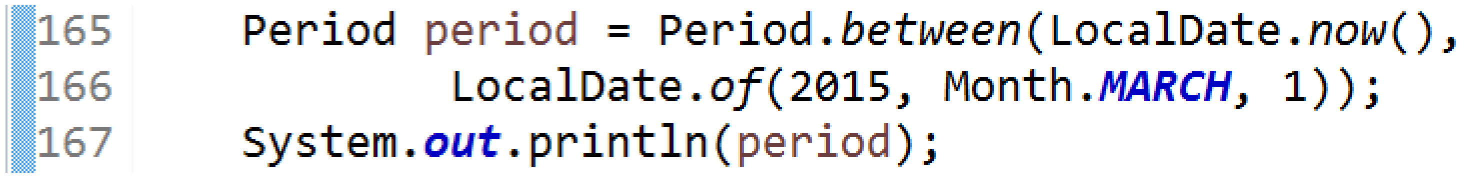


```java
TemporalAdjuster adj = TemporalAdjusters.next(DayOfWeek.WEDNESDAY);
LocalDate nextWed = ldNow.with(adj);
System.out.println("For the date of " + ldNow
        + ", the next Wednesday is " + nextWed);
```

- dayOfWeekInMonth(int ordinal, DayOfWeek dayOfWeek)
- firstDayOfMonth()
- firstDayOfNextMonth()
- firstDayOfNextYear()
- firstDayOfYear()
- firstInMonth(DayOfWeek dayOfWeek)
- lastDayOfMonth()
- lastDayOfYear()
- lastInMonth(DayOfWeek dayOfWeek)
- next(DayOfWeek dayOfWeek)
- nextOrSame(DayOfWeek dayOfWeek)
- ofDateAdjuster(UnaryOperator<LocalDate> dateBasedAdjuster)
- previous(DayOfWeek dayOfWeek)
- previousOrSame(DayOfWeek dayOfWeek)

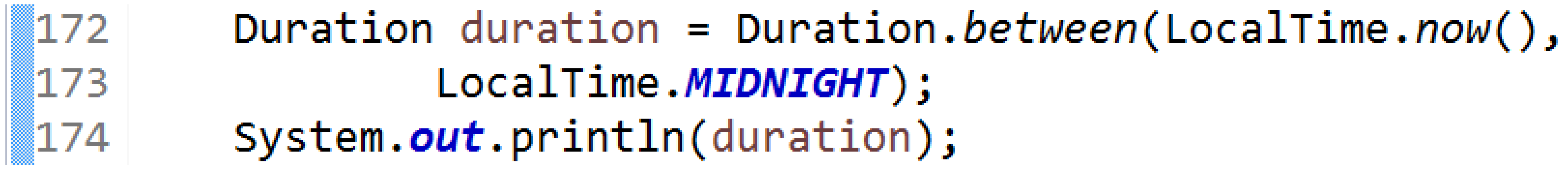
Period and Duration

- Period – distance in the timeline



```java
    Period period = Period.between(LocalDate.now(),
            LocalDate.of(2015, Month.MARCH, 1));
    System.out.println(period);
```

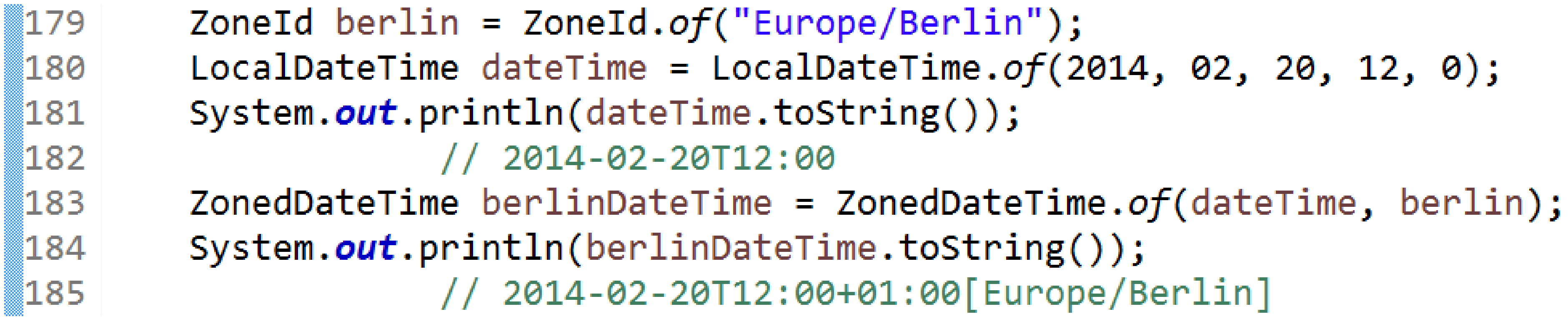
- Duration – distance in the timeline



```java
    Duration duration = Duration.between(LocalTime.now(),
            LocalTime.MIDNIGHT);
    System.out.println(duration);
```

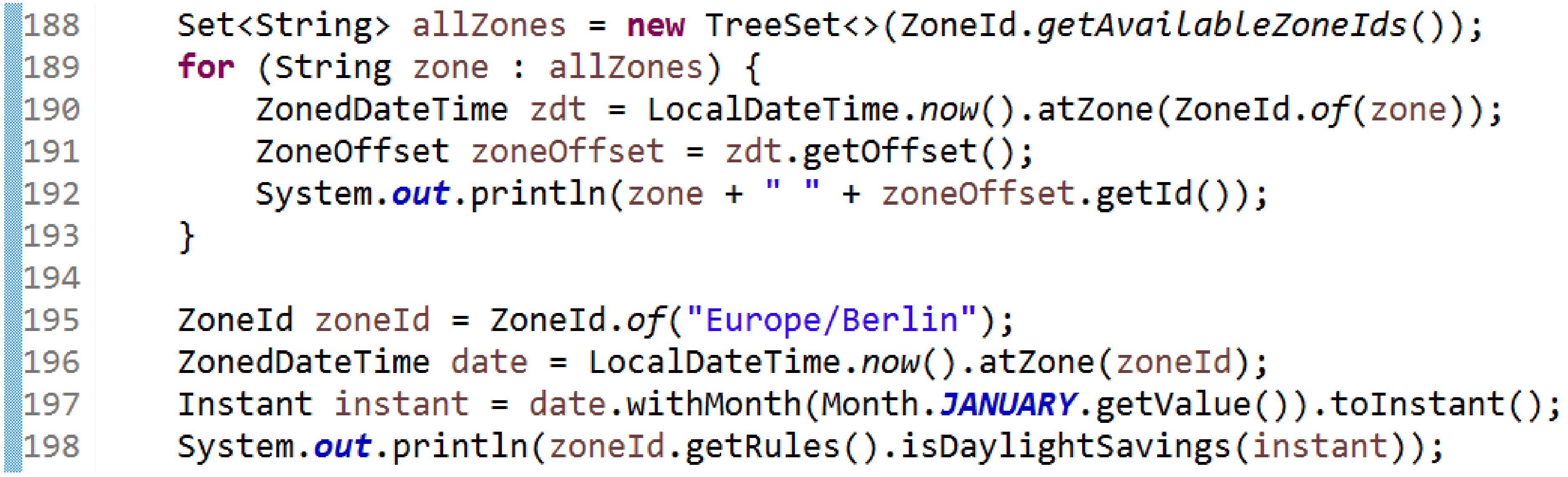
Time Zones

- Classes:
  - ZoneId: Representation of the Timezone
  - ZonedDateTime: DateTime with TimeZone



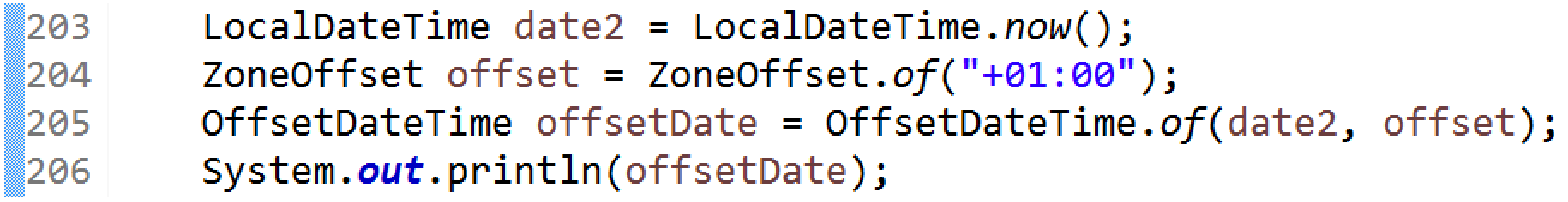
```java
    ZoneId berlin = ZoneId.of("Europe/Berlin");
    LocalDateTime dateTime = LocalDateTime.of(2014, 02, 20, 12, 0);
    System.out.println(dateTime.toString());
                // 2014-02-20T12:00
    ZonedDateTime berlinDateTime = ZonedDateTime.of(dateTime, berlin);
    System.out.println(berlinDateTime.toString());
                // 2014-02-20T12:00+01:00[Europe/Berlin]
```

Time Zones – Examples 1/2



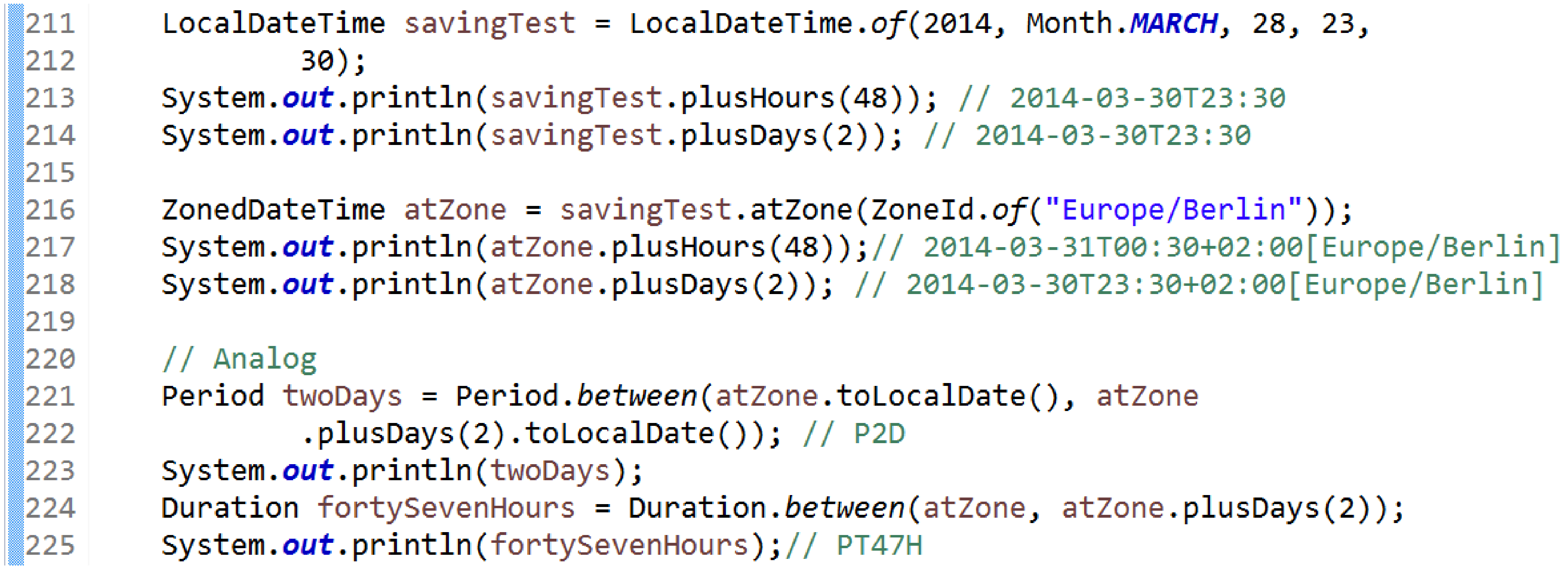
```java
    Set<String> allZones = new TreeSet<>(ZoneId.getAvailableZoneIds());
    for (String zone : allZones) {
        ZonedDateTime zdt = LocalDateTime.now().atZone(ZoneId.of(zone));
        ZoneOffset zoneOffset = zdt.getOffset();
        System.out.println(zone + " " + zoneOffset.getId());
    }

    ZoneId zoneId = ZoneId.of("Europe/Berlin");
    ZonedDateTime date = LocalDateTime.now().atZone(zoneId);
    Instant instant = date.withMonth(Month.JANUARY.getValue()).toInstant();
    System.out.println(zoneId.getRules().isDaylightSavings(instant));
```

- OffsetTime OffsetDate



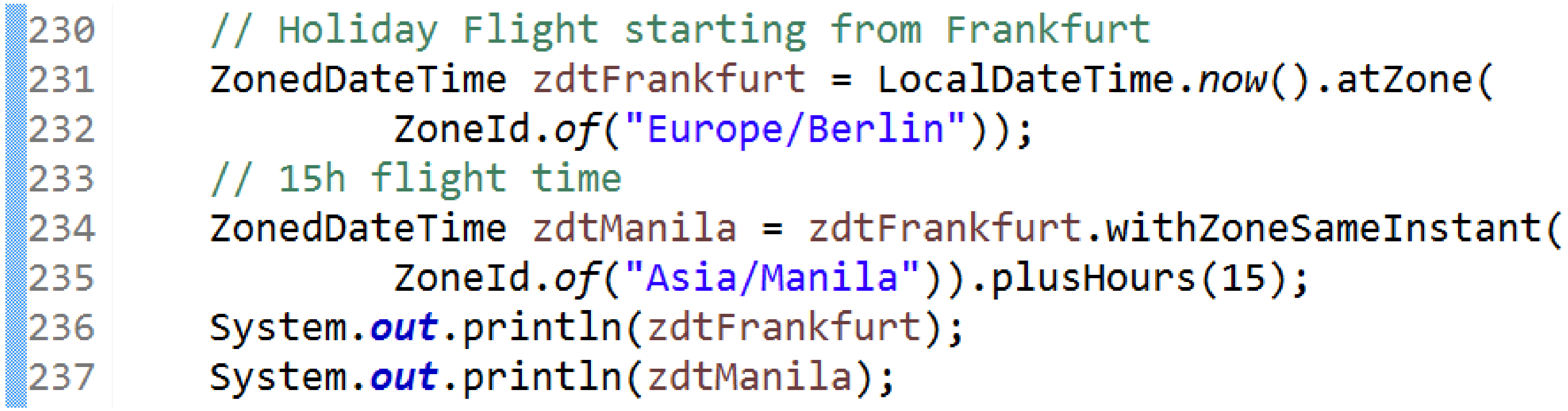
```java
    LocalDateTime date2 = LocalDateTime.now();
    ZoneOffset offset = ZoneOffset.of("+01:00");
    OffsetDateTime offsetDate = OffsetDateTime.of(date2, offset);
    System.out.println(offsetDate);
```

Time Zones – Examples 2/2



```
LocalDateTime savingTest = LocalDateTime.of(2014, Month.MARCH, 28, 23,
        30);
System.out.println(savingTest.plusHours(48)); // 2014-03-30T23:30
System.out.println(savingTest.plusDays(2)); // 2014-03-30T23:30

ZonedDateTime atZone = savingTest.atZone(ZoneId.of("Europe/Berlin"));
System.out.println(atZone.plusHours(48));// 2014-03-31T00:30+02:00[Europe/Berlin]
System.out.println(atZone.plusDays(2)); // 2014-03-30T23:30+02:00[Europe/Berlin]

// Analog
Period twoDays = Period.between(atZone.toLocalDate(), atZone
        .plusDays(2).toLocalDate()); // P2D
System.out.println(twoDays);
Duration fortySevenHours = Duration.between(atZone, atZone.plusDays(2));
System.out.println(fortySevenHours);// PT47H
```

Time Zones – Flight



```java
    // Holiday Flight starting from Frankfurt
    ZonedDateTime zdtFrankfurt = LocalDateTime.now().atZone(
            ZoneId.of("Europe/Berlin"));
    // 15h flight time
    ZonedDateTime zdtManila = zdtFrankfurt.withZoneSameInstant(
            ZoneId.of("Asia/Manila")).plusHours(15);
    System.out.println(zdtFrankfurt);
    System.out.println(zdtManila);
```

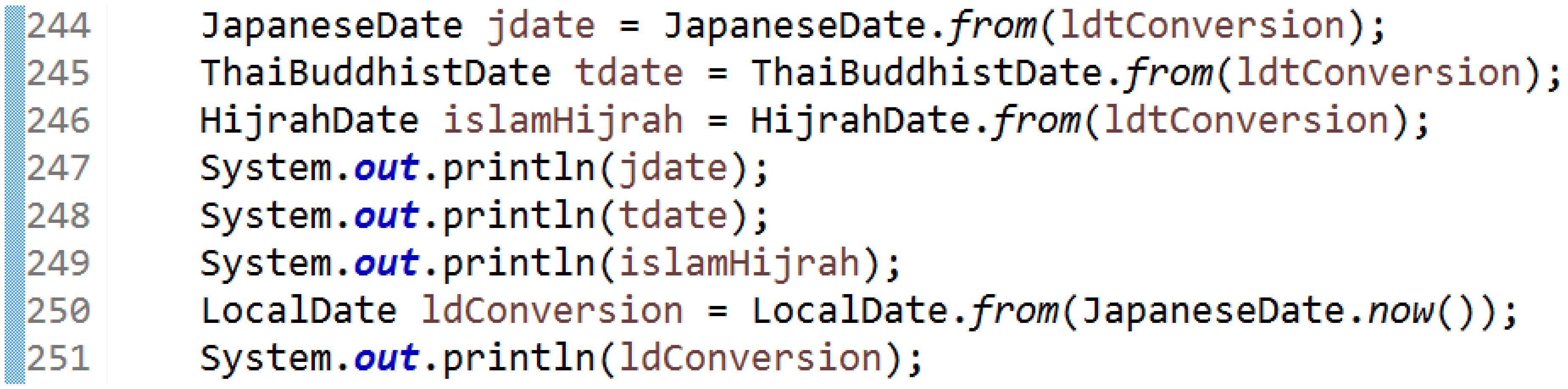
Time Zones – DB Roundtrip

- see Example

Conversion from / to legacy code

Calendar.toInstant()	converts the Calendar object to an Instant
GregorianCalendar. toZonedDateTime()	converts a GregorianCalendar instance to a ZonedDateTime
GregorianCalendar. from(ZonedDateTime)	creates a GregorianCalendar object using the default locale from a ZonedDateTime instance
Date.from(Instant)	creates a Date object from an Instant
Date.toInstant()	converts a Date object to an Instant
TimeZone.toZoneId()	converts a TimeZone object to a ZoneId

Non-ISO Date Conversion



```
        JapaneseDate jdate = JapaneseDate.from(ldtConversion);
        ThaiBuddhistDate tdate = ThaiBuddhistDate.from(ldtConversion);
        HijrahDate islamHijrah = HijrahDate.from(ldtConversion);
        System.out.println(jdate);
        System.out.println(tdate);
        System.out.println(islamHijrah);
        LocalDate ldConversion = LocalDate.from(JapaneseDate.now());
        System.out.println(ldConversion);
```

Infos and Links

Infos:

- http://docs.oracle.com/javase/tutorial/datetime/overview/index.html
- http://www.heise.de/developer/artikel/Die-neue-Date-Time-API-in-Java-8-2198399.html
- http://examples.javacodegeeks.com/core-java/java-8-datetime-api-tutorial/
- http://jaxenter.de/artikel/java-se-8-date-time-api-178388

Examples:

- http://www.mscharhag.com/2014/02/java-8-datetime-api.html