**使用LocalDate、LocalTime、LocalDateTime：**

1. 实例为不可变的对象，分别使用ISO-8601日历系统的日期、时间、日期+时间
2. 不包含当前时间的信息，时区相关信息
3. 常用方法：
   1. now() 静态方法，根据当前时间创建对象
      1. LocalDate = LocalDate.now();
      2. LocalTime = LocalTime.now();
      3. LocalDateTime localDateTime = LocalDateTime.now();
   2. of() 静态方法，根据指定日期/时间创建对象
      1. LocalDate localDate = LocalDate.of(2016, 10, 26);
      2. LocalTime localTime = LocalTime.of(02, 22, 56);
      3. LocalDateTime localDateTime = LocalDateTime.of(2016, 10, 26, 12, 10, 55);
   3. plusDays(), plusWeeks(), plusMonths(), plusYears()向当前 LocalDate 对象添加几天几周、几个月、几年
   4. minusDays(), minusWeeks(), minusMonths(), minusYears()从当前 LocalDate 对象减去几天、几周、几个月、几年
   5. plus(), minus() 添加或减少一个 Duration 或 Period
   6. withDayOfMonth(), withDayOfYear(), withMonth(), withYear()将月份天数、年份天数、月份、年份 修 改 为 指 定 的 值 并 返 回 新 的LocalDate 对象
   7. getDayOfMonth() 获得月份天数(1-31)
   8. getDayOfYear() 获得年份天数(1-366)
   9. getDayOfWeek() 获得星期几(返回一个 DayOfWeek枚举值)
   10. getMonth() 获得月份, 返回一个 Month 枚举值
   11. getMonthValue() 获得月份(1-12)
   12. getYear() 获得年份
   13. until() 获得两个日期之间的 Period 对象，或者指定 ChronoUnits 的数字
   14. isBefore(), isAfter() 比较两个 LocalDate
   15. isLeapYear() 判断是否是闰年

**Instant 时间戳，以Unix元年（UTC时区1970）开始**

Instant ins = Instant.now(); //默认使用 UTC 时区

OffsetDateTime odt = ins.atOffset(ZoneOffset.ofHours(8)); //偏移8h

Instant ins2 = Instant.ofEpochSecond(5); //偏移5s

**Duration和Period**

1. Duration：计算两个时间的间隔

Duration.between(Instant1, Instant2);

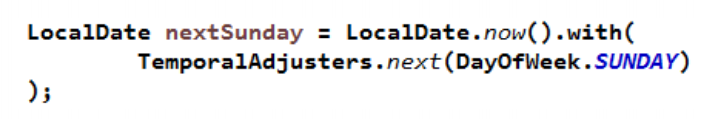
1. Period：计算两个日期间隔

Period pe = Period.between(LocalDate2, LocalDate1);

System.out.println(pe.getYears());

**日期操纵**

1. TemporalAdjuster : 时间校正器。有时我们可能需要获取例如：将日期调整到“下个周日”等操作。
2. TemporalAdjusters : 该类通过静态方法提供了大量的常用 TemporalAdjuster 的实现。



**解析与格式化**

1. java.time.format.DateTimeFormatter 类，提供了三种格式化方法
   1. 预定义的标准格式
   2. 语言环境相关的格式
   3. 自定义的格式

DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyyMMdd");

LocalDateTime ldt = LocalDateTime.now();

String strDate = ldt.format(dtf);

LocalDateTime newLdt = ldt.parse(strDate, dtf);

**时区处理**

1. 带时区的时间类：ZonedDate、ZonedTime、ZonedDateTime
2. 每个时区都对应着ID，地区ID都为“区域/城市”的格式
3. ZoneId：该类中包含了所有的时区信息
   1. getAvailableZoneIds() : 可以获取所有时区信息
   2. of(id) : 用指定的时区信息获取 ZoneId 对象

LocalDateTime ldt = LocalDateTime.now(ZoneId.of("Asia/Shanghai"));