Manipulación de fechas con java.time

Clases fecha/hora

- Desde Java 8, se incluyen las siguientes clases en java.time para manipulación de fechas y horas:

LocalDate	• Manipulación de fechas
LocalTime	• Manipulación de horas
LocalDateTime	• Manipulación de fechas más horas
ZonedDateTime	• Manipulación de fechas y horas en zona horaria
Instant	• Instante de tiempo concreto

Creación de los objetos

- No disponen de constructores públicos
- Se crean a través de métodos estáticos (of):

```
var f1=LocalDate.of(2020,11,30); //yyyy,MM,dd
var f2=LocalDate.of(2010,Month.May,15); //utiliza enumeración para el mes
var h1=LocalTime.of(15,20,30); //HH, mm, ss
var lt1=LocalDateTime.of(2022,9,30,20,11,15);
var lt2=LocalDateTime.of(f1,h1); //a partir de objetos fecha y hora
var zi=ZoneId.of("Europe/Amsterdam");
var zdt1=ZonedDateTime.of(f2,h1,zi); //datos de fecha y hora más zona horaria
```

- Valores erroneos provocarían una excepción:

```
var e1=LocalDate.of(2009,13,25); //DateTimeException
```

Número de mes incorrecto

Parseado

- Se pueden crear objetos de fecha-hora, a partir de una cadena de caracteres con el formato estándar de fecha/hora:

```
var f1=LocalDate.parse("2021-10-20");
var h1=LocalTime.parse("15:23:45");
var lt1=LocalDateTime.parse("2019-06-03T16:15:30");
var zdt1=ZonedDateTime.parse("2019-06-03T16:15:30+01:00[Europe/Madrid]");
```

- Si la cadena tiene un formato incorrecto se produce una excepción:

```
var e1=LocalDate.parse("11/10/2022"); //DateTimeParseException
```

- Aunque se puede indicar que la cadena tiene un formato específico:

```
var e1=LocalDate.parse("11/10/2022", DateTimeFormatter.ofPattern("dd/MM/yyyy"));
```

Manipulación de fechas/horas

- Las clases anteriores disponen de métodos para obtener una nueva fecha resultante de la suma/extracción de una cantidad.

- Estos métodos no modifican el valor original, dado que los objetos son inmutables:

```
var f1=LocalDate.of(2020,12,1);
var lt1=LocalDateTime.of(2022,9,30,20,11,15);
var fn=f1.minusDays(1);
var ltn=lt1.plusSeconds(50);
System.out.println(f1); //se mantiene sin cambios  2020-12-01
System.out.println(fn); //2020-11-30
System.out.println(ltn); //2022-09-30T20:12:05
```

- Cuidado con llamadas a métodos no existentes en esa clase:

```
var f1=LocalDate.of(2020,11,30);
f1.minusHours(5);//error de compilación
```

Revisión conceptos

Indica qué se mostrará al ejecutar el siguiente bloque de código:

```
var lt1=LocalDateTime.of(2022,9,30,20,11,15);
lt1.plusHours(5);
lt1.plus(1,ChronoUnit.DAYS);
System.out.println(lt1);
```

a. 2022-10-02T01:11:15
b. 2022-10-01T25:11:15
c. 2022-09-30T20:11:15
d. Se producirá una excepción

Respuesta

La respuesta es la **c**. Los objetos de fecha son inmutables, por lo que las llamadas a los métodos plus, minus, etc, devuelven un nuevo objeto pero no afectan a aquel sobre el que se aplican. No hay excepción porque al ser fecha/hora, admite que se puedan sumar tanto días como horas.

Instante de tiempo

- Un objeto Instant representa un momento de tiempo concreto en la zona GMT

```
var i1=Instant.now(); //instante de tiempo actual
var i2=Instant.ofEpochSecond(2_000_000); //momento de tiempo dos millones de segundos
                                          //después de 1970-01-01T00:00:00
System.out.println(i2); // 1970-01-24T03:33:20Z
```

- Se puede crear a partir de un ZonedDateTime:

```
var zi=ZoneId.of("America/New _York");
var f1=LocalDate.of(2020,11,30);
var h1=LocalTime.of(15,20,30);
var zdt1=ZonedDateTime.of(f1,h1,zi);
var i3=zdt1.toInstant();
System.out.println(zdt1); //2020-11-30T15:20:30-05:00[America/New_York]
System.out.println(i3);  //2020-11-30T20:20:30Z
```

Periodos de tiempo

- La clase Period representa un periodo de tiempo, medido en años, meses y días

```
Period p1=Period.of(2,5,11); //P2Y5M11D
Period p2=Period.ofYears(3); //P3Y
Period p3=Period.ofDays(200); //P200D
Period d4=p2.plus(p3); //P3Y200D
```

- Se pueden sumar/restar periodos de tiempo a objetos de fecha, pero no a LocalTime:

```
var f1=LocalDate.of(2020,11,30);
Period p1=Period.ofWeeks(3);
System.out.println(f1.plus(p1)); //2020-12-21
var h1=LocalTime.of(15,20,30);
System.out.println(h1.minus(p1)); //excepcion UnsupportedTemporalTypeException
```

Intervalo entre fechas

- Mediante el método estático *between()* de Period se puede obtener el intervalo de tiempo entre dos fechas en formato Period
- Únicamente puede utilizarse con objetos LocalDate:

static Period between(LocalDate startDateInclusive, LocalDate endDateExclusive)

```
var f1=LocalDate.of(2020,11,11);
var f2=LocalDate.of(2020,11,30);
var ldt1=LocalDateTime.of(2020,11,20,22,30,15);
var ldt2=LocalDateTime.of(2020,11,21,01,30,15);
System.out.println(Period.between(f1, f2)); //P19D
System.out.println(Period.between(ldt1.toLocalDate(), ldt2.toLocalDate())); //P1D
```

Duraciones

- Una duración, representada por el objeto Duration, representa un intervalo de tiempo medido en horas, minutos, segundos.

```
Duration d1=Duration.ofDays(2); //se almacena el número de horas: PT48H
Duration d2=Duration.ofSeconds(150); //PT150S
Duration d3=Duration.ofHours(4); //PT4H
Duration d4=d3.plus(d2);//PT4H2M30S
```

- Se pueden sumar/restar duraciones a objetos de hora, pero no a objetos LocalDate

```
Duration d1=Duration.ofHours(4);
var ldt1=LocalDateTime.of(2003,8,11,15,20,30);
var ldt2=ldt1.minus(d1);
System.out.println(ldt2);   //2003-08-11T11:20:30
```

Intervalo entre horas

- Mediante el método estático *between()* de Duration se puede obtener el intervalo de tiempo entre dos fechas en formato Duration

- Puede utilizarse con objetos que contengan datos de hora:

static Duration between(Temporal startInclusive, Temporal endExclusive)

```
var t1=LocalTime.of(2,11,30);
var t2=LocalTime.of(15,11,40);
var ldt1=LocalDateTime.of(2020,11,20,22,30,15);
var ldt2=LocalDateTime.of(2020,11,21,1,30,15);
System.out.println(Duration.between(t1, t2)); //PT13H10S
System.out.println(Duration.between(ldt1, ldt2)); //PT3H
var zdt1=ZonedDateTime.of(ldt1,ZoneId.of("Europe/Amsterdam")); //GMT+1
var zdt2=ZonedDateTime.of(ldt2,ZoneId.of("America/New_York")); //GMT-5
System.out.println(Duration.between(zdt1, zdt2)); //PT9H
```

se tiene en cuenta la diferencia horaria entre ambas zonas

Revisión conceptos

Teniendo en cuenta que la diferencia horaria entre Madrid y Nueva York es de 6 horas, ¿qué se mostrará al ejecutar el siguiente código?:

```
Instant i1=Instant.now();
Instant i2=i1.plus(2, ChronoUnit.HOURS);
ZonedDateTime z1=i1.atZone(ZoneId.of("America/New_York"));
ZonedDateTime z2=i2.atZone(ZoneId.of("Europe/Madrid"));
System.out.println(Duration.between(z1, z2));
```

a. PT2H
b. PT6H
c. PT4H
d. PT8H

Respuesta

La respuesta es la **a.** Al calcular el intervalo de tiempo entre dos ZonedDateTime, se debe tener en cuenta la diferencia horaria, pero al haberse creado los dos a partir de objetos Instant, la diferencia será la que haya entre ambos Instant.
Por ejemplo, si i1 son las 14h, i2 serán las 16h. Tomando como referencia Madrid, i1 en Nueva York serán las 8h. Así pues 16-8 hacen 8, menos 6 que es la diferencia horaria resultaría en 2, que es la diferencia entre los Instant.

Cambios de hora

- Java tiene en cuenta los cambios de hora en la manipulación de objetos ZonedDateTime:

```
LocalDateTime ld1=LocalDateTime.of(2022, 10, 30,2,30,30);
ZonedDateTime zd1=ZonedDateTime.of(ld1, ZoneId.of("Europe/Madrid")); //media hora antes del atraso de otoño
ZonedDateTime zd2=zd1.plusHours(1);
System.out.println(zd1); //2022-10-30T02:30:30+02:00[Europe/Madrid]
System.out.println(zd2); //2022-10-30T02:30:30+01:00[Europe/Madrid]
System.out.println(Duration.between(zd1, zd2)); //PT1H
System.out.println(zd1.getOffset()); //+02:00
System.out.println(zd2.getOffset()); //+01:00
```

Misma fecha/hora

Diferencias con GMT

Revisión conceptos

Teniendo en cuenta que en Nueva York, el cambio de hora de primavera se produjo el 13 de marzo a las 2:00 horas, ¿qué mostrará el siguiente código?:

```
LocalDateTime ld1=LocalDateTime.of(2022, 3, 13,1,30,30);
ZonedDateTime zd1=ZonedDateTime.of(ld1, ZoneId.of("America/New_York"));
ZonedDateTime zd2=zd1.plusDays(1);
System.out.println(zd2.toLocalDateTime());
```

a. 2022-03-14T02:30:30
b. 2022-03-14T01:30:30
c. 2022-03-14T00:00:00
d. Se producirá una excepción

Respuesta

La respuesta es la **b**. Al sumar un día entero, se mantiene la hora original. Si la suma hubiera sido de 24 horas, entonces si tendríamos como nueva hora las 2:30:30

Revisión conceptos

Teniendo en cuenta que en Nueva York, el cambio de hora de primavera se produjo el 13 de marzo a las 2:00 horas, ¿qué mostrará el siguiente código?:

```
LocalDateTime ld1=LocalDateTime.of(2022, 3, 13,1,30,30);
ZonedDateTime zd1=ZonedDateTime.of(ld1, ZoneId.of("America/New_York"));
ZonedDateTime zd2=zd1.plusHours(1);
ZonedDateTime zd3=zd1.minusHours(1);
System.out.println(Duration.between(zd3,zd2));
```

a. PT0H
b. PT1H
c. PT2H
d. PT3H

Respuesta

La respuesta es la **c**. Se debe tener en cuenta la diferencia horaria, ya que una es GMT-5 y la otra será GMT-4