Programación

Fechas en Java

Eladio Blanco

ÍNDICE

ÍNDICE

INTRODUCCIÓN

Java dispone de clases específicas para manejar datos de fechas y horas de forma correcta.

Con estas clases será más sencillo crear fechas y horas, convertirlas, realizar cálculos con ellas, "parsearlas" o mostrarlas con un formato determinado.

ÍNDICE

ÍNDICE

CLASES OBSOLETAS

Dos clases típicas para trabajar con fechas han sido java.util.Date y java.util.Calendar, pero a día de hoy se encuentran obsoletas.

Después, apareció java.sql.Date, pero realmente es una subclase de la primera y tampoco se usa a día de hoy.

ÍNDICE

ÍNDICE

EL NUEVO PAQUETE JAVA.TIME

Con Java 8 (2014) se crea el paquete *java.time* con las clases oficiales.

Se solucionan bastantes problemas de las clases tradicionales.

Incluyen soporte automático para años bisiestos, zonas horarias y cambio automático de zona horaria.

EL NUEVO PAQUETE JAVA.TIME

Clases más importantes:

- **LocalDate**: Representa fechas y facilita su manejo para declararlas, compararlas, sumarlas...
- **LocalTime**: Igual que la anterior, pero para horas.
- **LocalDateTime**: Combinación de las 2 anteriores.

EL NUEVO PAQUETE JAVA.TIME

Clases más importantes:

- **ZonedDateTime**: Como LocalDateTime, pero teniendo en cuenta la zona horaria.
- **Instant**: Parecida a LocalDateTime, pero almacena número de nanosegundos desde epoch de UNIX (01/01/1970).
- **Period**: Obtener diferencias (días, horas, minutos...) entre fechas.
- **Duration**: Similar a la anterior, pero para horas exclusivamente.

ÍNDICE

ÍNDICE

CONSTRUIR FECHAS Y HORAS

Estas clases carecen de constructores públicos.

Se instancian usando métodos de tipo *factoría* que van a construir estos objetos a partir de parámetros que le pasemos.

CONSTRUIR FECHAS Y HORAS

Todas las clases de manejo de fechas y horas disponen de 3 métodos importantes:

- now(): Crean instancias nuevas a partir de la fecha y hora actual.
- of(): Construyen fechas y horas a patir de sus partes.
- with(): Modifican el objeto actual en función del parámetro pasado con alguna cantidad (años, días, horas...) o clase de ajuste (TemporalAdjuster).

CONSTRUIR FECHAS Y HORAS

Ejemplos de fecha actual con *now()*:

```
import java.time.*;

public class Fecha01now {
        public static void main(String[] args) {
                // Creación de distintos objetos con now()
                System.out.println(LocalDate.now());      // 20
                System.out.println(LocalTime.now());      // 13
                System.out.println(LocalDateTime.now()); // 20
                System.out.println(Instant.now());       // 202
                System.out.println(ZonedDateTime.now()); // 20
        }
}
```

CONSTRUIR FECHAS Y HORAS

Ejemplos de fechas específicas con *of()*:

```
import java.time.*;

public class Fecha02of {
        public static void main(String[] args) {
                // Creación de fechas con of()
                System.out.println(LocalDate.of(1983, Month.MA
                System.out.println(LocalDateTime.of(1983, Mont
        }
}
```

CONSTRUIR FECHAS Y HORAS

Ejemplos de fechas a partir de otras con *with()*: y ajustadores temporales (más info).

```
LocalDate ahora = LocalDate.now();                  // 2022-11-14

System.out.println(ahora.withDayOfMonth(1)); // 2022-11-01
System.out.println(ahora.withMonth(6));         // 2022-06-14
System.out.println(ahora.withYear(2050));     // 2050-11-14
System.out.println(ahora.with(TemporalAdjusters.lastDayOfMonth
System.out.println(ahora.with(TemporalAdjusters.firstDayOfNext
```

ÍNDICE

ÍNDICE

INFORMACIÓN DE FECHAS

Mediante los métodos *get* de las fechas se puede extraer mucha información:

```
// Fecha construida el 14/11/2022 a las 13:52:13 horas
LocalDateTime fecha = LocalDateTime.now();
System.out.println(fecha.getYear());          // 2022
System.out.println(fecha.getMonth());         // NOVEMBER
System.out.println(fecha.getMonthValue());    // 11
System.out.println(fecha.getDayOfWeek());     // MONDAY
System.out.println(fecha.getDayOfMonth());    // 14
System.out.println(fecha.getDayOfYear());     // 318
System.out.println(fecha.getHour());          // 13
System.out.println(fecha.getMinute());        // 52
System.out.println(fecha.getSecond());        // 13
System.out.println(fecha.getNano());          // 180998800
```

ÍNDICE

ÍNDICE

TRANSFORMAR FECHAS Y HORAS

Según la clase utilizada dispondremos de métodos para añadir o quitar intervalos.

Por ejemplo, LocalDateTime dispone de *plusDays()*, *minusDays()*, *plusHours()*, *minusHours()*, *isBefore(fecha)*, *isAfter(fecha)*...

Ojo, estas las clases producen instancias inmutables (no pueden ser modificadas una vez que su información ha sido definida).

TRANSFORMAR FECHAS Y HORAS

```
LocalDateTime fecha = LocalDateTime.of(2022, Month.NOVEMBER, 2
System.out.println(fecha.plusYears(10));     // 2032-11-22T14:3
System.out.println(fecha.plusMonths(3));     // 2023-02-22T14:3
System.out.println(fecha.minusDays(10));     // 2022-11-12T14:3
System.out.println(fecha.minusHours(2));     // 2022-11-22T12:3
System.out.println(fecha.minusMinutes(20)); // 2022-11-22T14:1
System.out.println(fecha.plusSeconds(50));   // 2022-11-22T14:3
System.out.println(fecha); // fecha es INMUTABLE: 2022-11-22T1
```

ÍNDICE

ÍNDICE

PARSEAR FECHAS

Es posible crear fechas a partir de una cadena de texto mediante *parse()*.

Opcionalmente, admite un segundo parámetro para especificar el formato.

Patrones para parseo y formato y formateadores predefinidos.

```
LocalDate hoy = LocalDate.parse("2022-11-14");
LocalDate seisNov = LocalDate.parse("6/11/2022", DateTimeForma
System.out.println(hoy);     // 2022-11-14
System.out.println(seisNov);// 2022-11-06
```

ÍNDICE

ÍNDICE

MOSTRAR FECHAS

Podemos convertir una clase temporal en una cadena de texto (proceso inverso a parse) usando el formato que nos interese mediante *format()*.

Patrones para parseo y formato y formateadores predefinidos.

```
LocalDateTime fechaConHora = LocalDateTime.now();
// Formato por defecto  2022-11-14T18:08:36.957851900
System.out.println(fechaConHora);
// Formato ISO 8601     2022-11-14T18:08:36.9578519
System.out.println(fechaConHora.format(DateTimeFormatter.ISO_D
// Formato manual       14/11/2022 06:08:36
DateTimeFormatter esDateFormat = DateTimeFormatter.ofPattern("
System.out.println(fechaConHora.format(esDateFormat));
```

ÍNDICE

ÍNDICE

TIEMPO ENTRE FECHAS

Para obtener la diferencia entre dos instantes de tiempo exite una inferfaz *java.time.temporal.TemporalUnit*, una enumeración *CronoUnit* y una clase *Period*.

Mediante between() obtenemos el tiempo transcurrido entre dos instantes y con until(), el tiempo que falta hasta una fecha u hora.

TIEMPO ENTRE FECHAS

Con LocalDateTime *between* y *until* devuelven un long.

```
LocalDateTime fNacimiento = LocalDateTime.of(1983, Month.MARCH
LocalDateTime hoy = LocalDateTime.now();
// between con LocalDateTime devuelve long
System.out.println(ChronoUnit.YEARS.between(fNacimiento, hoy))
System.out.println(ChronoUnit.MONTHS.between(fNacimiento, hoy)
System.out.println(ChronoUnit.DAYS.between(fNacimiento, hoy));
System.out.println(ChronoUnit.SECONDS.between(fNacimiento, hoy
// until con LocalDateTime devuelve long
System.out.println(fNacimiento.until(hoy, ChronoUnit.YEARS));
System.out.println(fNacimiento.until(hoy, ChronoUnit.MONTHS));
System.out.println(fNacimiento.until(hoy, ChronoUnit.DAYS));
System.out.println(fNacimiento.until(hoy, ChronoUnit.SECONDS))
```

TIEMPO ENTRE FECHAS

Con LocalDate *between* y *until* devuelven un Period.

```
LocalDate finAnio = LocalDate.of(2022, 12, 31);
LocalDate ahora = LocalDate.now(); // 14/11/2022

// until y between con LocalDate devuelven Period
Period hastaFinAnio = ahora.until(finAnio);
Period hastaFinAnio2 = Period.between(ahora, finAnio);

System.out.println(hastaFinAnio.getYears());     // 0
System.out.println(hastaFinAnio.getMonths());    // 1
System.out.println(hastaFinAnio.getDays());      // 17
```

TRABAJAR CON FECHAS Y HORAS

Ejercicios???

REPASAR

- Java Classes - Java Date

TIPS DE LA PRESENTACIÓN

¿Imprimir en PDF?

1. Clic aquí
2. `CTRL` / `CMD` + `P`
3. Guardar como PDF

Navegar por las diapositivas

1. Pulsa `ESC`
2. Clic a la que quieras ir


A TRABAJAR...