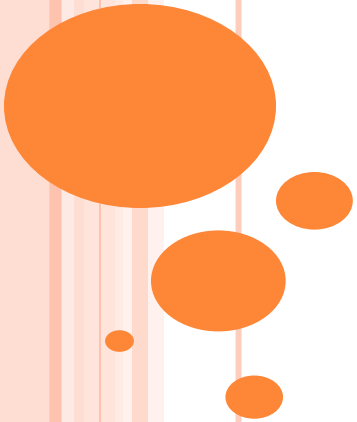
Fechas en Java



Dos tipos de clases

- Clásicas
- Nuevas

Clases clásicas para fecha

- **Tradicionalmente, Java contaba con las siguientes clases para manejo de fechas**
  - **java.util.Date. Representa una fecha y hora concretas. La mayor parte de métodos y constructores están deprecated, pero sigue en uso**
  - **java.util.Calendar. Permite trabajar con los componentes de la fecha(años, días, minutos,..)**
  - **java.sql.Date y java.sql.Timestamp. Para trabajar con fechas y fechas-hora, respectivamente, en bases de datos**
  - **java.text.SimpleDateFormat. Utilizada para formatear y parsear fechas**

Creación de un objeto fecha

- **Fecha y hora actuales:**

```
Date date=new Date();
```

- **Fecha y hora concreta:**

```
Calendar cal=Calendar.getInstance();
//15:20:11 del 3/12/2022
cal.set(2022,11,03,15,20,11);
Date date=cal.getTime();
```

- **Desde una cadena de caracteres:**

```
SimpleDateFormat format=new SimpleDateFormat("dd/MM/yyyy");
String fecha="11/07/2020";
Date date=format.parse(fecha);
```

Comparar y formatear fechas

- La clase Date proporciona los siguientes métodos para realizar comparaciones entre fechas:
  - boolean before(Date other)
  - boolean after(Date other)
  - int compareTo(Date other)
- Mediante SimpleDateFormat se puede dar formato a una fecha:

```
SimpleDateFormat format=new SimpleDateFormat("dd/MM/yyyy");
Date date=new Date();;
System.out.println(format.format(date));
```

Fechas en bases de datos

- Para manejar fechas en JDBC se utiliza java.sql.Date que es subclase de java.util.Date

- En persistencia con JPA hibernate no se utiliza dicha clase: los campos de fecha son mapeados directamente a java.util.Date o java.time.LocalDate

Nuevas clases para fechas

- **Desde Java 8, se incluye el paquete java.time con las siguientes clases**
  - **LocalDate. Representa una fecha concreta:**

```
LocalDate f1=LocalDate.now();
LocalDate f2=LocalDate.of(2021, 7,22); //2021-07-22
```

  - **LocalTime. Representa una hora:**

```
LocalTime t1=LocalTime.of(10,23,50); //10:23:50
```

  - **LocalDateTime. Para respresentar una combinación de fecha más hora:**

```
LocalDateTime dt=LocalDateTime.of(2010,11,1,10,23,50); //2010-11-01T10:23:50
```

Formateado y parseado

- **Para parsear y formatear fechas, utilizamos la clase java.time.format.DateTimeFormatter:**

```
DateTimeFormatter format=DateTimeFormatter.ofPattern("dd/MM/yyyy");
String fecha="20/09/2019";
LocalDate date=LocalDate.parse(fecha, format);
```

Fecha a partir de una cadena

```
DateTimeFormatter format=DateTimeFormatter.ofPattern("dd/MM/yyyy");
LocalDate date=LocalDate.of(2022,10,20);
System.out.println(date.format(format));  //20/10/2022
```

Formateado de fecha

Comparar fechas

- Las nuevas clases de fecha también proporcionan métodos para comparar fechas:
  - boolean isBefore(TipoFecha other)
  - boolean isAfter(TipoFecha other)
  - int compareTo(TipoFecha other)

Manipulación de fechas

- Las nuevas clases de fecha disponen de métodos para manipular fechas/horas.

- En el caso de LocalDate:
  - LocalDate minus/plusYears(long cantidad)
  - LocalDate minus/plusMonths(long cantidad)
  - LocalDate minus/plusDays(long cantidad)

Conversiones

➢Para convertir objeto Date en LocalDate:

```
Date date = new Date();
// Primero, se convierte java.util.Date a Instant
Instant instant = date.toInstant();
// Finalmente, convertimos Instant a LocalDate
LocalDate localDate = instant.atZone(ZoneId.systemDefault()).toLocalDate();
```

➢Para convertir objeto LocalDate en Date:

```
// Crear un objeto LocalDate
LocalDate localDate = LocalDate.now();
// Convertir LocalDate a Date
Date date = Date.from(localDate.atStartOfDay(ZoneId.systemDefault()).toInstant());
```