élogos

# MANIPULACIÓN Y FORMATO DE FECHAS

élogos

# ÍNDICE

## MANIPULACIÓN Y FORMATO DE FECHAS

## 1. La clase Calendar

Antes de comenzar a hablar de Calendar, debemos comentar brevemente las características de la clase java.util.Date, que es la primera clase aparecida en Java para el tratamiento de fechas.

La mayoría de los constructores y métodos de Date están obsoletos, de ahí que deba utilizarse java.util.Calendar para manipular fechas. Sin embargo, Date sigue siendo utilizada en muchos contexto para representar una fecha y hora concretas.

Uno de los pocos constructores no obsoletos de Date es el constructor sin parámetros, con el que podemos crear un objeto Date asociado a la fecha y hora actuales:

```
Date fecha = new Date();
```

**Fecha y hora absoluta**.

Por convenio, la fecha absoluta o Timestamp viene representada por el número de milisegundos transcurridos desde la 0 horas del 1 de enero de 1970 hasta la fecha indicada. Dado un objeto Date que represente una determinada fecha y hora, la fecha y hora absoluta puede obtenerse a través del método getTme().

**Utilización de la clase Calendar**

La clase Calendar es una clase abstracta que se utiliza para conversiones entre un objeto de tipo Date y un conjunto de campos enteros del tipo YEAR (año), MONTH (mes), DAY (día), HOUR (hora), etc.

Un objeto de Calendar representa una fecha de acuerdo a las reglas de un calendario determinado. El API de java proporciona una subclase específica de Calendar: GregorianCalendar. Cualquier subclase de Calendar podrían representar varios tipos de calendarios lunares utilizados en cualquier lugar del mundo.

El método getInstance() de la clase nos devuelve una subclase de Calendar con el tiempo ajustado a la hora actual. A través del método set de la clase, podemos modificar individualmente los valores de año, mes, día, hora, minutos y segundos.

**Método get**

La clase Calendar tiene solamente un método get para obtener todos sus datos, pero se apoya en una serie de constantes que permiten obtener o ajustar una propiedad determinada de la fecha.

Las principales son:

- YEAR: año.
- MONTH: mes.
- DATE, DAY_OF_MONTH: día del mes.

- DAY_OF_WEEK: día de la semana entre 1 (MONDAY) y 7 (SATURDAY).
- HOUR: hora antes o después del medio día (en intervalos de 12 horas).
- HOUR_OF_DAY: la hora absoluta del día (en intervalos de 24 horas).
- MINUTE: el minuto dentro de la hora.
- SECOND: el segundo dentro del minuto.

**Ejemplo**

Para obtener el año, mes, día y hora, usaríamos el siguiente código:

```
System.out.println(cal.getTime());

System.out.println("AÑO: "+cal.get(Calendar.YEAR));

System.out.println("MES: "+(cal.get(Calendar.MONTH)+1));

System.out.println("DIA: "+cal.get(Calendar.DATE));

System.out.println("HORA:"+cal.get(Calendar.HOUR));

If(cal.get(Calendar.MONTH)==Calendar.OCTOBER){
        System.out.println("ES OCTUBRE");
}else{
        System.out.println("NO ES OCTUBRE");
}
```

```
AÑO: 2008

MES: 10

DIA: 1

HORA: 3

NO ES OCTUBRE
```

**Método set**

Modificar una propiedad de Calendar no es complicado, sólo es necesario usar el método set (int atributo, int valor). Siendo atributo una de las constante vistas anteriormente y valor la cantidad que se le quiere asignar.

**Ejemplo**

cal.set(Calendar.MONTH,Calendar.JUNE)

cal.set(Calendar.YEAR, 1999)

Pondría la fecha almacenada en el objeto "cal" a junio o al año 1999 sin modificar ninguna de las otras propiedades.

**El método add()**

El método add (CONSTATE, valor) suma algebraicamente valor a una fecha.

Por ejemplo para sumar 3 días y 2 meses a la fecha actual.

```
Calendar hoy = Calendar.getInstance();

hoy.add(Calendar.DATE,3);

hoy.add(Calendar.MONTH,2);

System.out.println(hoy.getTime());
```

Para el mismo ejemplo de fecha anterior (15:30 p.m. del 1 de noviembre de 2008 tendríamos la siguiente salida:

Sun Jan 04 15:37:34 CET 2009

## 2. La clase DateFormat

DateFormat es una clase que permite formatear fechas de acuerdo a un determinado estilo y a las características de un idioma o región geográfica.

DateFormat es una clase abstracta que se encuentra definida en el paquete java.text. Dispone del método getDateInstance() para obtener una instancia de la misma y asociarla un determinado estilo.

Una vez obtenido el objeto, podemos utilizar el método format() para aplicar formato a fechas definidas como objetos Date.

**Importar la clase DateFormat**

Dateformat ofrece algunos métodos para obtener la fecha y la hora formateada por defecto o formateada a un determinado valor local y ofrece también un número de estilos de formato.

Si queremos utilizar el formato y localización por defecto crearíamos un DateFormat de la siguiente manera:

```
DateFormat df = DateFormat.getDateInstance();
```

El formateado de la fecha actual sería:

```
Date d = new Date()

System.out.println(df.format(d));
```

**DateFormat: Formato de estilos**

Dateformat analiza el formato y fechas para cualquier localización. Su código es absolutamente independiente de la localización de los convenios de meses, días de la semana, o incluso el formato de calendario, solar o lunar

El formato de los estilos debe incluir todos los detalles, largo, mediano y corto

Si queremos utilizar el formato corto y formato americano deberíamos crear el DateFormat de la siguiente manera:

```
DateFormat df = DateFormat.getDateInstance(DateFormat.SHORT, Locale.US);
```

El formateado de la fecha actual sería igual que antes:

```
Date d = new Date()

System.out.println(df.format(d));
```