### Annex I.- Dates en Java

Per obtenir la data actual i, en general, per a treballar amb dates en Java tenim diverses possibilitats i depenen de què?, com? i on? vulguem representar, haurem de fer ús d'unes classes o altres. Però abans de continuar, ha de quedar clar que el maneig d'estructures de dades que representen dates ha sigut històricament un problema en Java, sobretot, la gestió de les zones horaries. Si la nostra aplicació requereix representar les dates en diferents zones horàries amb exactitud hauriem de considerar fer ús de la llibreria **Joda Time** (https://www.joda.org/joda-time/). Al no ser part del llenguatge no anem a utilitzar-la, però convé que li pegueu una ullada.

## Date

És la classe més simple per a representar Dates en Java. Tenim tres constructors:

- Date(). Permet crear un objecte de tipus Date amb la data i hora actuals del sistema.
- Date(int year, int month, int day). Permet crear un objecte de tipus Date a partir de l'any, el mes i dia indicats com a paràmetre. Aquest mètode està obsolet (Deprecated) i no s'hauria d'utilitzar.
- Date(long date). Permet crear un objecte de tipus Date a partir del número de milisegons indicat com a paràmetre, que han passat des de l'1 de Gener de 1970 a les 00:00:00.

Com a mètodes significatius tenim:

- int getHours(). Deprecated.
- int getMinutes(). Deprecated.
- int getSeconds(). Deprecated.
- void setHours(int i). Deprecated.
- void setMinutes(int i). Deprecated.
- void setSeconds(int i). Deprecated.
- void setTime(long date). Permet establir la data actual indicant un número de milisegons.
- String toString(). Converteix la data a un String en format yyyy-mm-dd.

## GregorianCalendar

Si volem més opcions, com per example sumar i/o restar dates, la classe GregorianCalendar és una bona opció. Es tracta d'una subclasse de la classe Calendar, que ens proporciona un calendari gregorià estàndard.

```
GregorianCalendar data = new GregorianCalendar();
```

Si volem emprar GregorianCalendar hem de fer el següent import:

```
import java.util.GregorianCalendar;
```

Alguns dels seus constructors són:

- GregorianCalendar(). Construix un objecte GregorianCalendar a partir de la data i hora del sistema amb la zona horaria i localització per defecte.
- GregorianCalendar(int year, int month, int dayOfMonth). Construix un objecte GregorianCalendar a

partir de l'any, mes i dia indicats com a paràmetre amb la zona horaria i localització per defecte.

- GregorianCalendar(int year, int month, int dayOfMonth, int hourOfDay, int minute, int second). Construix un objecte GregorianCalendar a partir de l'any, mes, dia, hora, minuts i segons indicats com a paràmetre amb la zona horaria i localització per defecte.
- GregorianCalendar(Locale aLocale). Construix un objecte GregorianCalendar amb la representació corresponent a la localització indicada com a paràmetre a partir de la data i hora del sistema amb la zona horaria per defecte.
- GregorianCalendar(TimeZone zone). Construix un objecte GregorianCalendar amb la zona horaria indicada com a paràmetre a partir de la data i hora del sistema amb la localització per defecte.

Una vegada creat l'objecte que conté la data del sistema, per obtenir un camp concret de la data s'utilitza el mètode get(camp) on camp indica la part de la data que es vol obtenir. Cada una d'estes parts està definida com una constant de la classe Calendar.

Per exemple, per obtenir l'any de la data actual podem escriure les següents instruccions:

```
GregorianCalendar data = new GregorianCalendar();
int any = data.get(Calendar.YEAR);
```

Exemple: programa Java que obté la data i hora actual del sistema i la mostra per pantalla. El programa mostrarà l'hora, el mes, el dia, l'hora, els minuts i els segons.

```
import java.util.*;
public class Dates {
   public static void main(String[] args) {
      //Instanciem l'objecte Calendar
      //en data obtenim la data i hora del sistema
      GregorianCalendar data = new GregorianCalendar();
      //Obtenim el valor de l'any, el mes i el dia
      //hora, minut i segon del sistema
      //emprant el mètode get i el paràmetre corresponent
      int any = data.get(GregorianCalendar.YEAR);
      int mes = data.get(GregorianCalendar.MONTH);
      int dia = data.get(GregorianCalendar.DAY_OF_MONTH);
      int hora = data.get(GregorianCalendar.HOUR_OF_DAY);
      int minut = data.get(GregorianCalendar.MINUTE);
      int segon = data.get(GregorianCalendar.SECOND);
      System.out.println("Data actual: " + dia + "/" + (mes+1) + "/" + any);
      System.out.println("Hora actual: " + hora + ":" + minut + ":" +segon);
   }
}
```

El programa mostra la data i hora actual de la següent forma:

```
Data actual: 10/1/2019
```

```
Hora actual: 10:20:58
```

Hem de tindre en compte que el valor retorna per Calendar.MONTH corresponent al mes comença per 0, és a dir, el valor 0 correspon a Gener i el valor 11 correspon a Desembre, per tant haurem de sumar-li 1 a este valor abans d'operar amb ell.
A banda d'obtenir la data, també podem assignar valors a un camp determinat de la data. Per assignar valors a camps concrets o a dates completes s'utilitza el mètode set:
Exemple:

```
data.set(2019,5,1); //assigna any, mes, dia
data.set(2019,5,1,3,45,12); //assigna valor a tots els camps
data.set(GregorianCalendar.YEAR,2019); //assigna el valor 2019 a l'any
```

També podem sumar o restar valors a la data i hora.
Exemple: sumar 10 dies a la data actual:

```
data.add(GregorianCalendar.DAY_OF_MONTH, 10);
```

Exemple: sumar 10 minuts a l'hora actual:

```
data.add(GregorianCalendar.MINUTE, 10);
```

Exemple: restar 2 anys a la data actual:

```
data.add(GregorianCalendar.YEAR, -2);
```

Altre mètode que ens pot interesar és:

- Date getTime(). Permet obtenir un objecte de tipus Date a partir d' un GregorianCalendar.

## SimpleDateFormat

Per a donar-li format a la sortida d'una data tenim diverses opcions, però una de les més utilitzades és **SimpleDateFormat**. Aquesta classe permet especificar amb quin format volem mostrar la data, ha de quedar clar que la data no és modificada per a res, simplement la seua visualització.
Constructors:

- SimpleDateFormat(String pattern). On pattern és un String (patró de representació) indica com s'haurà de mostrar la data. Veiem alguns dels patrons (compte majúscules i minúscules):
  - y: Any
  - M: Mes
  - w: Setmana de l'any
  - W: Setmana del mes
  - d: dia del mes
  - D: dia de l'any
  - m: minuts
  - s: segons
  - S: milisegons

Un exemple podria ser:

```
SimpleDateFormat data = new SimpleDateFormat("dd/MM/yyyy hh:mm:ss");
```

Com a mètode més destacat tenim:

- `String format(Date data)`. Que reb un objecte de tipus Date i obté la seua representació segons el pattern indicat en el constructor.

Vegem un exemple complet:

---

```
import java.util.*;
import java.text.SimpleDateFormat;
public class calendari {

   public static void main(String[] args) {

      //Quin dia de la semana va ser NADAL en l'any 2005
      Calendar nadal2005 = new GregorianCalendar(2005, Calendar.DECEMBER, 25);
      int diaSetmana = nadal2005.get(Calendar.DAY_OF_WEEK);
      System.out.println("Dia (1=Domingo): " + diaSetmana );

      //Quants dies va tindre febrer de 2005
      Calendar febrer2005 = new GregorianCalendar(2005, Calendar.FEBRUARY, 1);
      int diesFebrer = febrer2005.getActualMaximum(Calendar.DAY_OF_MONTH);
      System.out.println("dias en Feb 2005: " + diesFebrer );

      //Quants dies va tindre febrer de 2000
      Calendar febrer2000 = new GregorianCalendar(2000, Calendar.FEBRUARY, 1);
      diesFebrer = febrer2000.getActualMaximum(Calendar.DAY_OF_MONTH);
      System.out.println("Dies en Feb 2000: " + diesFebrer );

      //Obtindre la diferència en milisegons
      long diffMillis = nadal2005.getTimeInMillis() - febrer2005.getTimeInMillis();

      //Obtindre la diferència en segons
      long diffSecs = diffMillis/1000;

      //Obtindre la diferència en minuts
      long diffMins = diffSecs/60;

      //Obtindre la diferència en horas
      long diffHours = diffMins/60;

      //Obtindre la diferència en dias
      long diffDays = diffHours/24;
      //Obtindre la diferència en meses
      long diffMonths = diffDays/30;

      System.out.println("diffMillis: " + diffMillis );
      System.out.println("diffSecs: " + diffSecs );
      System.out.println("diffMins: " + diffMins );
      System.out.println("diffHours: " + diffHours );
      System.out.println("diffDays: " + diffDays );
      System.out.println("diffMonths: " + diffMonths );

      Calendar data = new GregorianCalendar();
      // Obtenim el valor de l'any, mes i dia.
      int any = data.get(Calendar.YEAR);
```

```
        int mes = data.get(Calendar.MONTH)+1;
        int dia = data.get(Calendar.DATE);
        int hora = data.get(Calendar.HOUR);
        int minut = data.get(Calendar.MINUTE);
        int segon = data.get(Calendar.SECOND);

        System.out.println("ANY ACTUAL: " + any);
        System.out.println("MES ACTUAL: " + mes);
        System.out.println("DÍA ACTUAL: " + dia);


        data.set(Calendar.YEAR, 2020);
        data.set(Calendar.MONTH, 02); //els mesos comencen per 0.
        data.set(Calendar.DATE, 30);
        // Assignem any, mes i dia.
        data.set(2020, 02, 30);
        // Assignem, any, mes, dia, hora i minuts
        data.set(2020, 02, 30, 19, 00);

        // Afegim 30 minutos a la data actual.
        data.add(Calendar.MINUTE, 30);
        // Afegim 100 dies a la data actual.
        data.add(Calendar.DATE, 100);
        // Llevem 10 anys a la data actual.
        data.add(Calendar.YEAR, -10);
        System.out.println("ANY ACTUAL: " + any);

        System.out.println("DATA ACTUAL: " + dia + "/" + (mes) + "/" + any);
        System.out.printf("Hora Actual: %02d:%02d:%02d %n", hora, minut, segon);

        //Imprimir la data i hora actuals FORMA 2
        Date araMateix = new Date();
        SimpleDateFormat dataImprimir = new SimpleDateFormat("dd/MMMMM/yyyy hh:mm:ss");
        String longDataActual = dataImprimir.format(araMateix);
        System.out.println(longDataActual);
    }
}
```