CLASE FILEINPUTSTREAM y FILEOUTPUTSTREAM

Con esta clase escribimos la información byte a byte. Hereda de la clase InputStream y OutputStream.

a) FILEINPUTSTREAM, abrimos el fichero de lectura. Y se va a leer de forma secuencial byte a byte. Si el fichero no existe nos manda la excepción **FileNotFoundException**.

FileInputStream fp = new FileInputStream(fichero);
FileInputStream fp = new FileInputStream(File);

Los métodos más usuales son:

- ✓ **int read( )**, devuelve en ASCII el byte leído. Nos devuelve **-1** si no tiene más información el fichero.
- ✓ **int read(byte cadByte[ ]),** lee los bytes del array.
- ✓ **int read(byte cadByte[ ], int offset, int n)**, lee los n bytes indicados del array a partir del offset.
- ✓ **int skip(long n)**, salta n bytes.
- ✓ **int available()**, devuelve el número de bytes en el buffer de entrada.

b) FILEOUTPUTSTREAM, abrimos el fichero para escritura. La información se va a guardar byte a byte.

FileOutputStream fp = new FileOutputStream(File);
FileOutputStream fp = new FileOutputStream(fichero[, boolean]);
FileOutputStream fp = new FileOutputStream(fichero, true); añade al final

Los métodos más usuales son:

- ✓ **void write(entero)**, escribe el byte en el fichero.
- ✓ **void write(byte cadByte[ ]),** escribe un array de bytes.
- ✓ **void write(byte cadByte[ ], int offset, int n)**, escribe los n bytes indicados del array, desde el offset.
- ✓ **void flush( ),** se vacía el buffer de salida.

Para cerrar el canal:

void close( ), cierra el fichero.

https://docs.oracle.com/javase/7/docs/api/java/io/FileInputStream.html
https://docs.oracle.com/javase/7/docs/api/java/io/FileOutputStream.html
http://dit.upm.es/~pepe/libros/vademecum/index.html?n=331.html
https://www.discoduroderoer.es/clases-fileinputstream-y-fileoutputstream-para-ficheros-binarios-en-java/

ESCRITURA DEL FICHERO

```
import java.io.*;
import java.lang.*;

class fileoutputstream
{
  public static void main(String [] args) throws IOException
  {
    char letra;
    FileOutputStream fp = null;
    fp = new FileOutputStream("datos.txt");  // abro el canal de lectura
    do{
          letra =(char)System.in.read();
         fp.write((byte)letra);
      }while(letra !='\r');
    fp.close();
  }
}
```

LECTURA DE UN FICHERO DE TEXTO.

```
import java.io.*;
import java.lang.*;

class fileinputstream
{
  public static void main(String [] args) throws IOException
  {
    int letra;
    FileInputStream fp = null;
    fp = new FileInputStream("datos.txt");  // abro el canal de lectura
    letra =  fp.read();
    while(letra !=-1)
    {
       System.out.print((char)letra);
       letra =  fp.read();
    }
    fp.close();
  }
}
```

CLASE DATAINPUTSTREAM y DATAOUTPUTSTREAM

Nos permite leer datos de distintos tipos: enteros, reales, carácter, cadena sin convertirlos en bytes. Y utiliza el formato UTF-8 (formato de 8 bits). Necesita de la clase FileInputStream o FileOutputStream, para la modificación y/o transformación de los datos. Para que no de error en la lectura debemos trabajar con excepciones. La excepción para el fichero es **EOFException**.

a) Declaración del objeto.

FileInputStream fp = new FileInputStream(fichero);
DataInputStream var = new DataInputStream(FILE);

FileOutputStream fp = new FileOutputStream(fichero[, boolean]);
DataOutputStream var = new DataOutputStream(FILE);

b) Métodos de lectura de datos

- ✓ read( ), devuelve el array de bytes leídos.
- ✓ boolean readBoolean( ), para leer un booleano.
- ✓ byte readByte( ), lee un byte.
- ✓ char readChar( ), lee un char.
- ✓ short readShort( ), para entero corto.
- ✓ int readInt( ), para entero.
- ✓ Long readLong( ), lee un número entero largo.
- ✓ float readFloat( ), para número real.
- ✓ double readDouble( ), para número real de doble precisión.
- ✓ String readLine( ), lee una línea de caracteres hasta \n ó \r\n. Devuelve null si llega al final de archivo.
- ✓ String readUTF( ), lee caracteres en formato UTF.
- ✓ skipBytes(int n), salta n bytes que deben existir, sino están espera a tenerlos.
- ✓ readFully(byte array[]), lee hasta llenar el array.

c) Métodos de escritura de datos. Son básicamente los mismos métodos pero cambiando read por write y el argumento del método no va vacío.

- ✓ write(int), escribe un byte.
- ✓ writeBoolean(boolean), escribe un dato booleano.
- ✓ writeByte(byte), escribe un byte.
- ✓ writeChar(char), escribe un char
- ✓ writeShort(short), escribe un entero corto.
- ✓ writeInt(int), escribe un entero.
- ✓ writeLong(entero largo).
- ✓ writeFloat(float), escribe un número real del tipo Float.
- ✓ writeDouble(doublé), escribe un número real del tipo Double.
- ✓ writeBytes(byte array[]), escribe n bytes.
- ✓ writeChars(cadena), para cadenas.
- ✓ writeUTF(cadena), los dos primeros bytes de la cadena guarda el número total de bytes que tiene la cadena.
- ✓ flush( ), vacía el buffer.

https://docs.oracle.com/javase/7/docs/api/java/io/DataInputStream.html
https://docs.oracle.com/javase/7/docs/api/java/io/DataOutputStream.html
https://www.discoduroderoer.es/clases-datainputstream-y-dataoutputstream-para-ficheros-binarios-en-java/
http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/archivos/primitivos.htm

```
import java.io.*;
import java.util.*;

class ejbinario
{
public static void main(String [] args) throws IOException, EOFException
{
    String nombre;
    char sexo;
    int edad;
    /* FileOutputStream fis=new FileOutputStream("personas.dat");
     DataOutputStream dos=new DataOutputStream(fis);
     InputStreamReader entrada = new InputStreamReader(System.in);
     BufferedReader valor = new BufferedReader(entrada);
     int cnt;
     String sex;
     for(cnt=1;cnt<4;cnt++)
     {
          System.out.println("DIME TU NOMBRE: ");
          nombre = valor.readLine();
          System.out.println("SEXO M(masculino)/F(femenino): ");
          sex = valor.readLine();
          sexo = sex.charAt(0);
          System.out.println("EDAD: ");
          edad = Integer.parseInt(valor.readLine());
          dos.writeUTF(nombre);
          dos.writeChar(sexo);
          dos.writeInt(edad);
    }
    dos.close();*/
          // VISUALIZACIÓN DEL FICHERO
     try{
           FileInputStream fis=new FileInputStream("personas.dat");
           DataInputStream dos=new DataInputStream(fis);
           System.out.println("\n\tLISTADO DEL FICHERO\n");
           nombre = dos.readUTF();
           while(nombre != null)
            {
              sexo = dos.readChar();
              edad = dos.readInt();
              System.out.println(nombre+"\t"+sexo+"\t"+edad);
              Nombre = dos.readUTF();
            }
           dos.close();
         }
      catch(FileNotFoundException e){   System.out.println("No se encontró el archivo");  }
      catch(IOException e){  System.out.println("");// para anular el error de fin de fichero  }
}
}
```