COMPROBACIONES:

exists

isDirectory
isFile

canRead
canWrite

MÉTODOS:

list //solo si es directorio
listFiles //solo si es directorio

mkdir
mkdirs

createNewFile
delete //si es directorio, borrar antes lo de dentro

Importante siempre cerrar el fichero cuando acabemos de trabajar con él
Por ejemplo, si escribimos y no hacemos .flush() o .close(), no guardará los cambios

file.close()

// Excepciones típicas ficheros: | FileNotFoundException | EOFException | IOException
(no se encuentra) (while true) (falta permisos, error al usar los métodos)

LEER (fichero de texto)

```
// Mostrar archivos de un directorio
File f = new File([ruta], “directorio|fichero”);
String[] archivos = f.list();  //si es directorio!
for (int i = 0; i < archivos.length; i++) {
    System.out.println(archivos[i]);
}
```

```
// Leer fichero byte a byte
FileReader -> Leer contenido de un fichero
fr.read() -> lee un byte
while ((i = fr.read()) != -1) {
    System.out.print((char) i);
 }
```

```
// FileReader lee byte a byte, con BufferedReader podremos leer línea a línea
BufferedReader br = new BufferedReader(new FileReader(new File(“./prueba/fichero.txt”)));
while ((linea = br.readLine()) != null) { System.out.println(linea); }
```

ESCRIBIR (ficheros de texto)

```
// FileWriter no tiene saltos de línea
FileWriter fw = new FileWriter(new File("./prueba/fichero.txt")[, true]);
for (String mensaje : mensajes)
   fw.write(mensaje + "\n");
fw.close();
```

```
// BufferedWriter tiene método para saltos de línea
BufferedWriter br = new BufferedWriter(fw));
bw.write(“Prueba1”);
bw.writeLine(); // salto de línea
bw.close();
```

```
// PrintWriter salta línea a la par que escribe
PrintWriter pw = new PrintWriter(new FileWriter("./prueba/fichero.txt"));
pw.println("Prueba1"); // escribe y hace salto de línea a la vez
pw.close();
```

ESCALA PARA LA CREACIÓN DE FICHEROS DE TEXTO (LECTURA | ESCRITURA)

```
File f = new File(“fichero.txt”);
  FileReader fr = new FileReader (f);
    BufferedReader br = new BufferedReader (fr);
    PrintReader pr = new PrintReader(fr);
```

```
File f = new File(“fichero.txt”);
  FileWriter fw = new FileWriter(f);
    BufferedWriter bw = new BufferedWriter(fw);
    PrintWriter pw = new PrintWriter(fw);
```

FICHEROS BINARIOS

```
FileInputStream fileIS = new FileInputStream(file);
FileOutputStream fileOS = new FileOutputStream(file);

fileOS.write("hola".getBytes()) // las tildes dan problemas
while ((i = fileIS.read()) != -1) System.out.println((char) i);
```

DATA INPUT + OUTPUT STREAM

```
FileOutputStream fileOS = new FileOutputStream(f);
DataOutputStream dataOS = new DataOutputStream(fileOS);

FileInputStream fileIS = new FileInputStream(f);
DataInputStream dataIS = new DataInputStream(fileIS);
```

```
dataOS.writeInt(1);                dataIS.readInt(); // 1
dataOS.writeUTF("Túúúú");          dataIS.readUTF(); //Túúúú
                                   //EOFException si hacemos bucle infinito
```

Importante: leer en el mismo orden que se ha escrito

OBJECT INPUT + OUTPUT STREAM

```
FileInputStream fileIS = new FileInputStream(f);
ObjectInputStream objIS = new ObjectInputStream(fileIS);

FileOutputStream fileOS = new FileOutputStream(f);
ObjectOutputStream objOS = new ObjectOutputStream(fileOS);
```

Para poder escribir y leer objetos, la clase del objeto a insertar tendrá que implementar **Serializable**

```
objOS.writeObject(obj);        Objeto obj = (Objeto) objIS.readObject();
                               //EOFException si hacemos bucle infinito
```

Importante: hacer casting del tipo de objeto

Para editar un dato podremos o bien guardar todos los objetos en un ArrayList editando el objeto a editar y luego plasmar todos esos objetos del ArrayList en el fichero reemplazando los datos que tenia antes, **o bien utilizar un fichero auxiliar (por ejemplo, Departamentos.dat -> DepartamentosAux.dat) donde iremos guardando todos los objetos que leamos del fichero original en el auxiliar (modificando el objeto a editar en caso de existir y encontrarlo), y en caso de haber modificado algún valor, llamar a un método que vuelque todos los datos de ese fichero auxiliar en el fichero original.**

RANDOM ACCESS FILE

```java
public static void escribirRandomAccessFile(File fichero) {
    try {
        RandomAccessFile raf = new RandomAccessFile(fichero, mode: "rw");
        raf.setLength(0);
        String[] apellidos = {"MENDEZ", "LOPEZ", "ETXEBERRIA", "CASTILLO", "AGIRRE", "PEREZ", "ALVAREZ"};
        int[] departamentos = {10, 20, 10, 10, 30, 30, 20};
        Double[] salarios = {1000.45, 2400.60, 3000.0, 1500.50, 2200.0, 1400.65, 2000.0};

        StringBuffer buffer; //buffer para almacenar el apellido
        int n = apellidos.length; //número de elementos del array (por ej. elementos del apellido)

        for (int i = 0; i < n; i++) {
            raf.writeInt( v: i + 1); //Para identificar al empl
            buffer = new StringBuffer(apellidos[i]);
            buffer.setLength(10); //Dejamos 10 caracteres para
            raf.writeChars(buffer.toString()); //Insertamos ap
            raf.writeInt(departamentos[i]); //Insertamos depar
            raf.writeDouble(salarios[i]); //Insertamos salario
        }
        raf.close();
    } catch (IOException ignored) { }
}
```

```java
public static void leerRandomAccessFile(File fichero, String cabeceraLista) {
    try {
        RandomAccessFile raf = new RandomAccessFile(fichero, mode: "r");
        int id, departamento;
        double salario;
        char[] apellido = new char[10];

        try {
            System.out.println(cabeceraLista);
            while (true) {
                id = raf.readInt(); //Leemos id

                for (int i = 0; i < apellido.length; i++)
                    apellido[i] = raf.readChar(); //Leemos caracter a caracter el apellido

                String apellidoStr = new String(apellido); //Guardamos en String el apellido
                departamento = raf.readInt(); //Leemos departamento
                salario = raf.readDouble(); //Leemos salario

                if (id > 0) {
                    System.out.printf("\tID: %2d, Apellido: %-12s Departamento: %3d, Salario: %.2f %n",
                            id, apellidoStr.trim(), departamento, salario);
                }

                if (raf.getFilePointer() == raf.length())
                    break;
            }
        } catch (EOFException ignored) { }
        raf.close();
        System.out.println();
    } catch (IOException ignored) {}
}
```

```java
public static void addAlFinalRandomAccessFile(File fichero)
    try {
        RandomAccessFile raf = new RandomAccessFile(fichero,
        long posicionFinal = raf.length();
        raf.seek(posicionFinal);

        StringBuffer bufferUltimoDato;
        bufferUltimoDato = new StringBuffer("TAMARGO");
        bufferUltimoDato.setLength(10);
        int idUltimoDato = (int) (posicionFinal / 36) + 1;

        raf.writeInt(idUltimoDato); //Insertamos id
        raf.writeChars(bufferUltimoDato.toString()); //Inser
        raf.writeInt( v: 10); //Insertamos departamento
        raf.writeDouble( v: 3500.00); //Insertamos salario

        raf.close();
    } catch (IOException ignored) {}
}
```

```java
public static void modificarDatoConcretoRandomAccesFile(File fichero) {
    try {
        RandomAccessFile raf = new RandomAccessFile(fichero, mode: "rw");
        int idDatoAModificar = new Random().nextInt( bound: 8) + 1;
        System.out.println("Modificamos el dato con el id: " + idDatoAModificar);
        long posicionALeer = (idDatoAModificar - 1) * 36;
        raf.seek(posicionALeer);

        StringBuffer bufferUltimoDato;
        bufferUltimoDato = new StringBuffer("ZUGASTI");
        bufferUltimoDato.setLength(10);

        raf.writeInt(idDatoAModificar); //Insertamos id
        raf.writeChars(bufferUltimoDato.toString()); //Insertamos apellido
        raf.writeInt( v: 40); //Insertamos departamento
        raf.writeDouble( v: 700.00); //Insertamos salario
    } catch (IOException ignored) { }
}
```

```java
public static void leerDatoConcretoRandomAccesFile(File fiche
    try {
        RandomAccessFile raf = new RandomAccessFile(fichero,
        int datoALeer = new Random().nextInt( bound: 8) + 1;
        System.out.println("Toca leer el id: " + datoALeer);
        long posicionALeer = (datoALeer - 1) * 36;
        raf.seek(posicionALeer);

        int id, departamento;
        double salario;
        char[] apellido = new char[10];

        id = raf.readInt();

        for (int i = 0; i < apellido.length; i++)
            apellido[i] = raf.readChar();

        String apellidoStr = new String(apellido);
        departamento = raf.readInt();
        salario = raf.readDouble();

        if (id > 0) {
            System.out.printf("\tID: %2d, Apellido: %-12s Departamento: %3d, Salario: %.2f %n",
                    id, apellidoStr.trim(), departamento, salario);
        }

    } catch (IOException ignored) { }
}
```

XSTREAM

//Hace falta importar su librería

```
public class EscribirPersonas {
    public static void main(String[] args) throws IOException , ClassNotFoundException {
        File fichero = new File("FichPersona.dat");
        FileInputStream filein = new FileInputStream(fichero);//crea el flujo de entrada
        //conecta el flujo de bytes al flujo de datos
        ObjectInputStream dataIS = new ObjectInputStream(filein);
        System.out.println("Comienza el proceso de creación del fichero a XML ...");
        //Creamos un objeto Lista de Personas
        ListaPersonas listaper = new ListaPersonas();
        try {
            while (true) { //lectura del fichero
            Persona persona= (Persona) dataIS.readObject(); //leer una Persona
            listaper.add(persona); //añaadir persona a la lista
        }
        }catch (EOFException eo) {}
            dataIS.close(); //cerrar stream de entrada
        try {
            XStream xstream = new XStream();
            //cambiar de nombre a las etiquetas XML
            xstream.alias("ListaPersonasMunicipio", ListaPersonas.class);
            xstream.alias("DatosPersona", Persona.class);
            //quitar etiqueta lista (atributo de la clase ListaPersonas)
            xstream.addImplicitCollection(ListaPersonas.class, "lista");
            //Insrtar los objetos en el XML
            xstream.toXML(listaper, new FileOutputStream("Personas.xml"));
            System.out.println("Creado fichero XML....");
        }catch (Exception e)
            {e.printStackTrace();}
    } // fin main
} //fin EscribirPersonas
```

Gestión fichero: ficheros binarios – objetos serializables

XStream

```
public class LeerPersonas {
    public static void main(String[] args) throws IOException {

        XStream xstream = new XStream();

        xstream.alias("ListaPersonasMunicipio", ListaPersonas.class);
        xstream.alias("DatosPersona", Persona.class);
        xstream.addImplicitCollection(ListaPersonas.class,"lista");

        FileInputStream fichero = new FileInputStream("Personas.xml");

        ListaPersonas listadoTodas = (ListaPersonas) xstream.fromXML(fichero);

        System.out.println("Número de personas: " + listadoTodas.getListaPersonas().size());

        List<Persona> listaPersonas = new ArrayList<Persona>();
        listaPersonas = listadoTodas.getListaPersonas();

        Iterator iterador = listaPersonas.listIterator();

        while(iterador.hasNext()){
            Persona p = (Persona) iterador.next();
            System.out.printf("Nombre: %s, edad: %d %n", p.getNombre(), p.getEdad());
        }
        System.out.println("Fin de listado ....");
    } //fin main
}//fin LeerPersonas
```