Tema 12

Fitxers

1 Introducció

Tots els llenguatges de programació tenen alguna forma d'interatuar amb els fitxers. Els algorismes que treballen amb fitxers solen tindre esta forma:

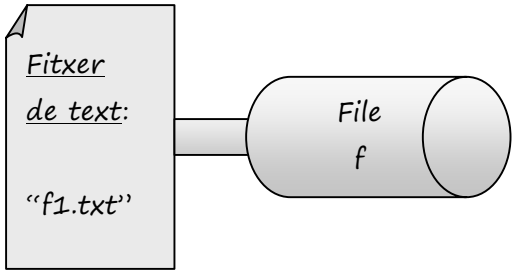
PER A LLEGIR LES DADES D'UN FITXER
Obrir el fitxer per a llegir Mentre (queden dades) fer Llegir dades Fi mentre Tancar el fitxer

PER A ESCRIURE A UN FITXER
Obrir el fitxer per a escriure Mentre (queden dades) fer Escriure dades Fi mentre Tancar el fitxer

2 Fitxers

Abans de vore les operacions de lectura o escriptura sobre fitxers, anem a vore com podem saber les **característiques d'un fitxer existent**, com per exemple, el directori pare, la grandària del fitxer, si té permisos de lectura, etc.

Per a fer això existeix la classe **File** (ja creada). Ens hem de crear un objecte d'eixa classe construint-lo a partir del nom del fitxer. Després, accedirem a les propietats del fitxer amb els mètodes (ja creats) d'eixa classe.



1r pas: associar-li un objecte de la classe File

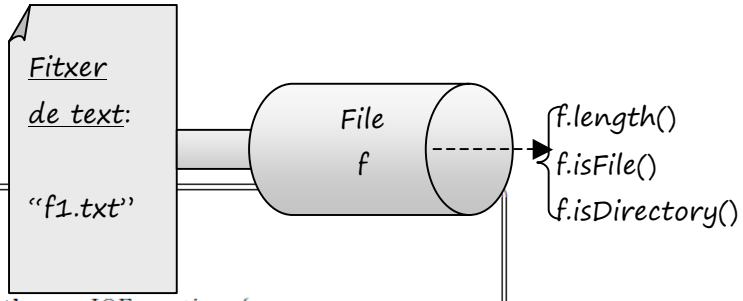
Es pot fer amb 3 constructors diferents:

```
File f = new File( "c:\kk\f1.txt");
```

```
File f = new File("c:\kk", "f1.txt");
```

```
File d = new File("c:\kk");

File f = new File(d, "f1.txt");
```

2n pas: accedir a les propietats del fitxer usant els mètodes de la classe.




```
import java.io.*;

class InfoFichero {

public static void main( String args[] ) throws IOException {
if( args.length > 0 )
{
for( int i=0; i < args.length; i++ )
  {
  File f = new File( args[i] );
  System.out.println( "Nom: "+f.getName() );
  System.out.println( "Camí: "+f.getPath() );
  if( f.exists() )
  {
    System.out.print("Fitxer existent ");
    System.out.print((f.canRead() ? "i es pot Llegir" : "" ) );
    System.out.print((f.canWrite() ? " i es pot escriure" : "" ) );
    System.out.println( "." );
    System.out.println( "La longitud del fitxer són "+
    f.length()+" bytes" );
  }
  else
    System.out.println( "El fitxer no existix." );
  }
} //if
else
  System.out.println( "Ha d'indicar un fitxer." );
}
} // main
}//class
```

Exercicis

1. Fes un programa que demane un fitxer o directori. En cas de ser un fitxer imprimirà els seus atributs: grandària (en Kb), data de modificació, etc (esbrina diferents mètodes existents). I, si es un directori, mostrarà el seu contingut, de manera simple (només els fitxers i directoris d'eixe directori) i de manera recursiva (mostrarà el contingut de tot l'arbre de directoris que conté). Per a fer-ho, implementa estes 3 funcions:

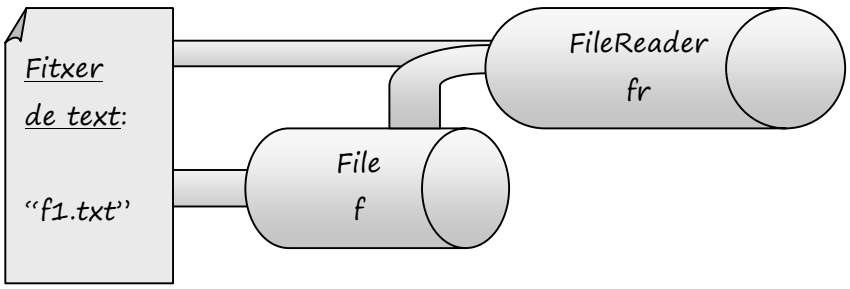
```
void atributs (File f);
void llistar(File d);
void llistarR(File d)
```

3 Fitxers de text. Streams d'entrada

En un fitxer de text pot haver des de dades fins a complexes configuracions com puguen ser els típics fitxers de configuració de Linux.

Per a poder llegir les dades d'un fitxer de text, hem de fer 2 coses:

1r pas: associar-li un objecte de la classe FileReader



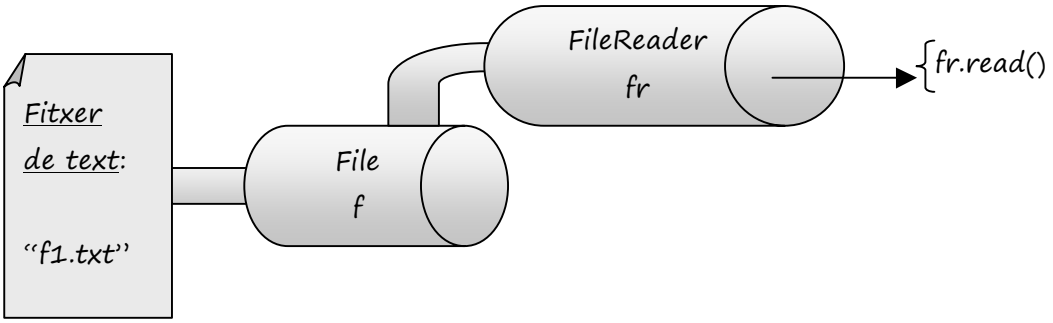
El podem crear amb 2 constructors diferents:

```
FileReader fr = new FileReader(f);
```

```
FileReader fr = new FileReader("c:\kk\kk1.txt");
```

2n pas: Llegir del fitxer (ara ho vorem)

3.1 Lectura caràcter a caràcter



Executarem el mètode ***read( )*** de FileReader tantes voltes com caràcters vulgam llegir. Este mètode retorna un enter, que és el codi del caràcter llegit. O bé, retorna -1 si no hem pogut llegir del fitxer.

Després, el que farem serà promocionar eixe ***int*** a ***char*** per a treballar amb ell.

Exemple: llegim d'un fitxer de text i ho mostrem per pantalla:

```
FileReader fr = new FileReader("proves.txt");
int c;
while ( (c=fr.read()) != -1 ) {
        System.out.print( (char) c );
}
```

Nota: la majoria de mètodes de les classes de lectura/escriptura de fitxers (inclosos els constructors) poden donar errors (anomenats "excepcions"). Per exemple, si no existeix el fitxer, si no té permisos, etc. Per tant, per a evitar possibles finalitzacions incorrectes del programa, tenim 2 solucions:

a) Capturar les possibles excepcions posant les crides a estos mètodes dins d'un bloc ***try-catch***:

```
public static void main(String[] args) {
        try {
                // Ús de mètodes de L/E de fitxers
        }
        catch(IOException e){
                System.out.println( e.getMessage());
}
```

b) Passar el "marró" a un altre:

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

        // Ús de mètodes de L/E de fitxers

}
```

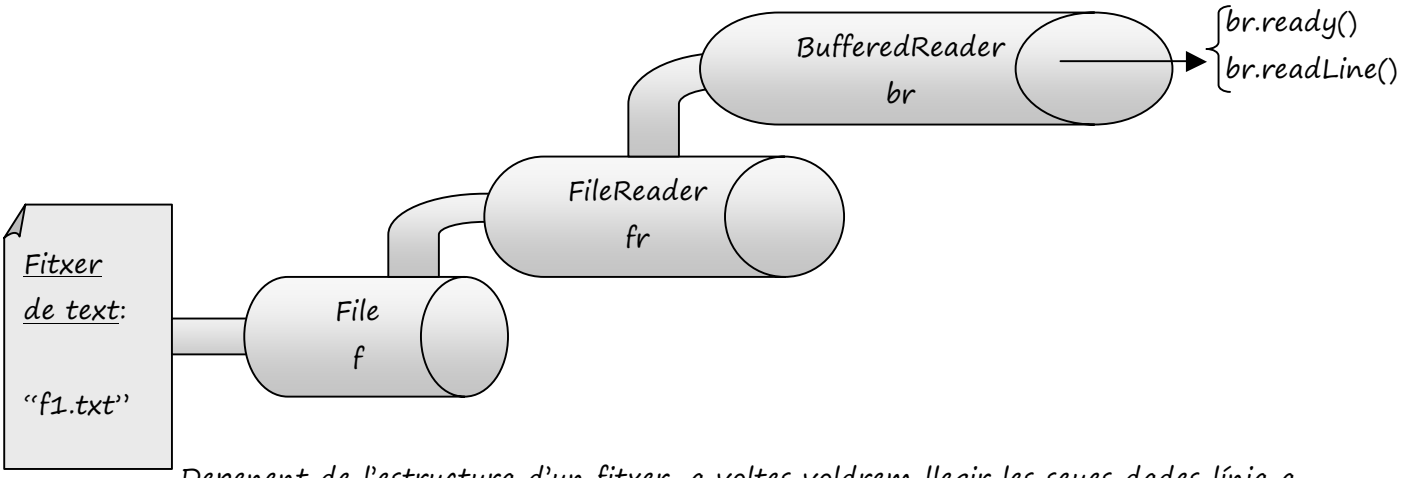
Exercicis

2. Donat un fitxer de text fes un programa que indique:

    a. La quantitat total de vocals que hi ha al fitxer
        i. Quantitat de as
        ii. Quantitat de es
        iii. Quantitat de is
        iv. Quantitat de os
        v. Quantitat de us
    b. Quantitat d'espais en blanc
    c. Quantitat de lletres majúscules
    d. Quantitat de lletres minúscules
    e. Quantitat de paraules. Es consideraran separadors: l'espai en blanc, el punt, la coma, els dos punts, el punt i coma i els signes d'interrogació i exclamació tancats.

3. En un examen de tipus test, cada alumne deixa en un fitxer amb el seu nom (Pep.txt) les seues 20 respostes (que poden ser A, B, C, D o bé un guió si no vol contestar-se una pregunta) en la primera línia del fitxer, per exemple ACBD-CDD-ABBACDD-CCB. En altre fitxer de text (solucio.txt) tindrem la solució, que serà: ABCDDCBAACCDBAABCDDA. Fes un programa tal que demane el nom del fitxer de l'alumne. El programa haurà de mostrar-nos quantes respostes ha encertat i quantes ha fallat. També la nota que ha tret l'alumne, tenint en compte que cada pregunta correcta suma 0.5 punts i cada pregunta incorrecta resta 0.125 punts.

3.2 Lectura línia a línia



Depenent de l'estructura d'un fitxer, a voltes voldrem llegir les seues dades línia a línia (i no caràcter a caràcter). Per a això, hem de crear un objecte de la classe BufferedReader associat a un objecte de la classe FileReader.

Exemple: llegim d'un fitxer de text, línia a línia, i ho mostrem per pantalla:

```
FileReader fr = new FileReader("proves.txt");
BufferedReader br = new BufferedReader(fr);
String s="";
while (  br.ready()  ) {
        s = br.readLine();
        System.out.println(  s );
}
```

Notes:

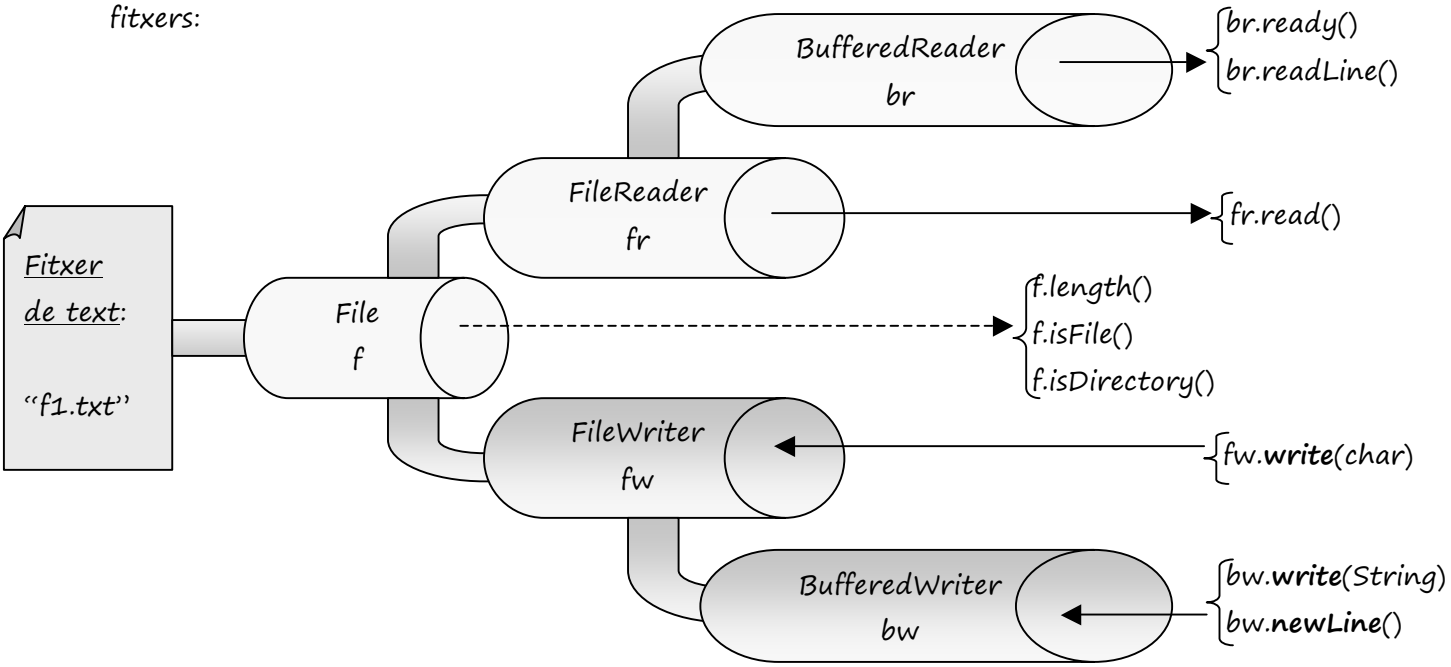
- ***ready()*** ens informa que encara queden dades per llegir al fitxer.
- També cal utilitzar el ***try-catch*** o el ***throws***.

Exercicis

4. Escriu un programa que diga quants paràgrafs té un fitxer de text. Que mostre també una llista amb el número de paràgraf i la quantitat de lletres que té cadascun.

4 Fitxers de text. Streams d'eixida

Igual que per a la lectura de fitxers, tenim classes anàlogues per a l'escriptura a fitxers:



4.1 Escriptura caràcter a caràcter

Exemple: còpia d'un fitxer origen a un fitxer destí:

```
FileReader fr = null;
FileWriter fw = null;
try {
        fr = new FileReader("entrada.txt");
        fw = new FileWriter("sortida.txt");
        int c;
        while ( (c=fr.read() )  !=  -1 ) {
                fw.write( (char)c );
        }
}
```

Problema: esta forma d'obrir el fitxer és destructiva. És a dir: si el fitxer "sortida.txt" ja existia abans, l'esborrarà i sobreescriurà damunt.

Solucions:

a) Abans d'obrir-lo per a escriptura, comprovar l'existència del fitxer.

```
...
if ( f.exists() == false ) {
        ...
}
```

b) Afegir la nova informació al final del fitxer. Per a fer això cal cridar al constructor del FileWriter d'una altra forma:

```
fw = new FileWriter("sortida.txt", true);
```

El segon argument indica que, si està a *true*, afegirà dades al fitxer. I si està a *false*, el fitxer se sobreescriurà.

Exercicis

5. Donat un fitxer de text, realitza un programa que cree una còpia del fitxer invertint les lletres majúscules en minúscules i viceversa.

4.2 Escriptura línia a línia

Exemple: escriure una poesia a un fitxer de text:

```
FileWriter fw = null;
BufferedWriter bw = null;
try {
        fw = new FileWriter("poesia.txt");
        bw = new BufferedWriter(fw);
        bw.write("Un pollet se'n va a la mar");
        bw.newLine();
        bw.write("Es rebolca per l'arena");
        bw.write("Xim, pum, fora!");
        bw.newLine();
}
```

Nota: amb el **newLine()** afegim un salt de línia al fitxer.

Exercicis

6. Fes un programa que vaja demanant frases per teclat (separades per <Intro>) i que les vaja escrivint en un fitxer de text. Es guardarà cada frase en una línia. El programa acabarà quan s'introduïsca la paraula Fi.

7. Escriu un programa que vaja demanant nom i telèfon dels teus amics i que els guarde en un fitxer.

```
PEP
961237652
MARTA
965436754
...
```

5 Fitxers binaris. Accés aleatori

5.1 Format dels fitxers

Els fitxers, a més de textos, poden tindre guardats **números** enters, números amb decimals, etc. Esta informació es guarda als fitxers en **format binari**. És a dir, abans de guardar-se una dada, es transforma a la seua representació binària.

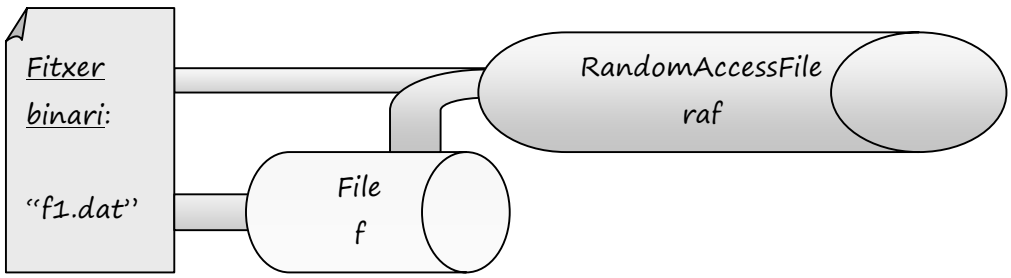
5.2 Accés als fitxers

En les formes vistes fins ara de llegir/escriure fitxers de text , la manera d'accedir a estos és **seqüencial**. És a dir: quan obrim un fitxer, el cursor (punt de lectura o escriptura) se situa al principi del fitxer i, conforme llegim o escrivim, anem avançant.

Doncs bé: la forma d'accedir als fitxers binaris s'anomena **accés aleatori**. Consistix en què podem accedir a una posició determinada del fitxer sense la necessitat de passar per les anteriors (els vectors també tenen accés aleatori).

Exemple: una cinta VHS té accés seqüencial, mentre que un CD té accés aleatori.

La classe RandomAccessFile ens permetrà eixe tipus d'accés:



És a dir, tenim 2 possibles constructors:

```
File f = new File("c:\kk\kk1.txt");
RandomAccessFile raf = new RandomAccessFile(f, mode);
```

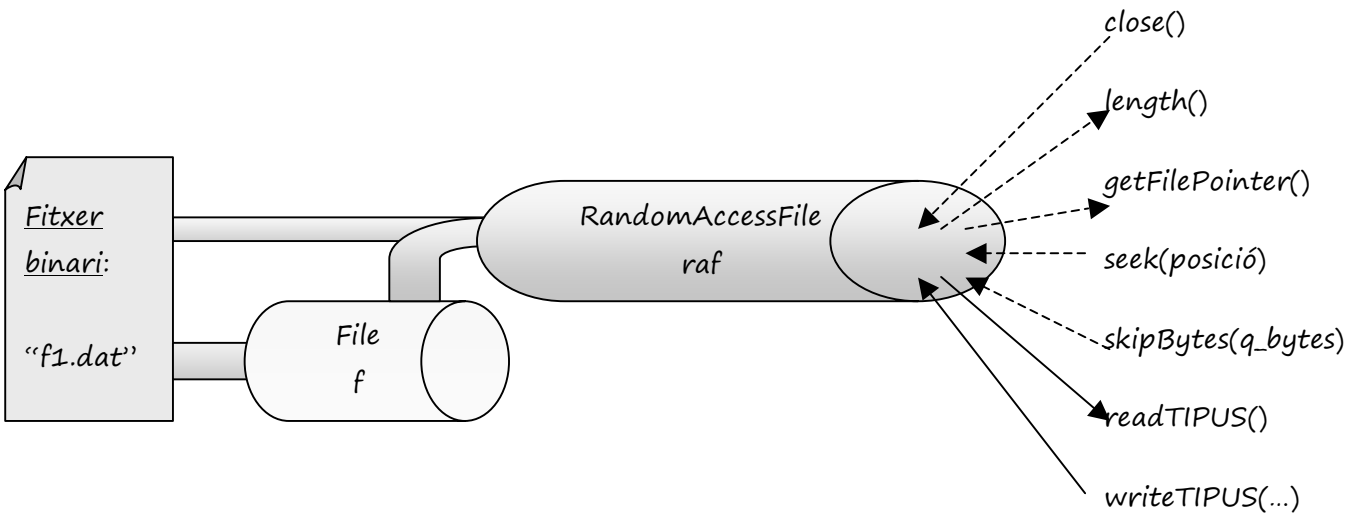
```
RandomAccessFile raf = newRandomAccessFile("c:\kk\kk1.txt", mode);
```

El mode és un String on indiquem si obrim el fitxer només per a lectura o també per a escriptura:

"r" → Només lectura

"rw" → Lectura i escriptura

Una volta ja tenim el constructor, ara vorem quins mètodes podem utilitzar en eixa classe:



MÈTODE	DESCRIPCIÓ
void close()	Tanca el fitxer (si estava obert).
long length()	Torna la grandària del fitxer.
long getFilePointer()	Torna la posició del cursor del fitxer.
void seek(long pos)	Posiciona el cursor a la posició pos (des de l'inici del fitxer).
int skipBytes(int n)	Bota els següents n bytes del fitxer.
readTIPUS	Llig del fitxer el tipus TIPUS.
writeTIPUS	Escriu al fitxer el tipus TIPUS.

Exemple:

```
public static void main(String[] args) {
RandomAccessFile fitxer=null;
try{
    fitxer=new RandomAccessFile("provesbinari.dat","rw");
    fitxer.writeChar('v');
    fitxer.writeDouble(2.4);
    fitxer.writeUTF("Hola a tots");
    fitxer.writeInt(2);
    // ens situem a l'inici
    fitxer.seek(0);

    System.out.println(fitxer.readChar());
    System.out.println(fitxer.readDouble());
    System.out.println(fitxer.readUTF());
    System.out.println(fitxer.readInt());

}
catch (IOException e){
    System.out.println(e.getMessage());
}
finally{
    if (fitxer!=null)
        try {
            fitxer.close();
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
}
}//main
```

Notes:

- El writeUTF i readUTF són per a Strings.
- La lectura de dades s'haurà de fer amb el mateix ordre que es va fer l'escriptura. Si no, es poden produir inconsistències.

Exercicis

8. Fes un programa que demane per teclat el teu nom, any de naixement, el número del NIF i la lletra del NIF. Caldrà vore si la lletra correspon amb el número (busca l'algorisme corresponent en Internet i implementa-ho). A continuació, guarda-ho en un fitxer.

9. Fes un programa que llija les dades del fitxer de l'exercici anterior i les mostre per pantalla.

10. Fes un programa que mostre el contingut d'un fitxer d'enters, que li afegisca més enters i que torne a mostrar el fitxer. Per a això:

    a. Fes el mètode mostrarFitxerEnters al qual se li passa com a paràmetre un fitxer de tipus RandomAccesFile (que suposem que ja està obert per a lectura). El mètode ha de mostrar tots els números del fitxer (sabem que són enters) per pantalla, separats per un tabulador.

    b. El programa principal demanarà el nom d'un fitxer binari on hi ha guardats números, l'obrirà i mostrarà el contingut cridant al mètode anterior. Després demanarà més números per teclat fins que posem el 0 i anirà guardant-los al fitxer després de l'últim número guardat. Finalment, es tornarà a mostrar el contingut cridant al mètode anterior.

11. Programa per a modificar un enter dins d'un fitxer d'enters existent. Per a això, després de mostrar els enters es demanarà la posició que ocupa l'enter a modificar, es mostrarà el valor a modificar, es demanarà el nou valor i s'escriurà el nou valor en la posició indicada. Pistes:

    a. La posició a demanar haurà de ser entre 1 i la quantitat d'enters del fitxer, que, com 1 enter ocupa 4 bytes, es calcula així: fitxer.length() / 4.

    b. Caldrà posar el punter en la posició on està el número, llegir el número, tornar a posar el punter en la posició on està el número i escriure el nou número.

12. Programa per a convertir una paraula a majúscules en un fitxer de text. Es demana una paraula, la busca en un fitxer de text i la modifica escrivint-la en majúsucules cada vegada que apareix en el fitxer. Pistes:

    a. Llegir el fitxer per línies

    b. Per a cada línia llegida, caldrà comprovar si té la paraula buscada (mètode indexOf de la classe String). En cas afirmatiu, es modifica (mètodes replace, length i toUpperCase de la classe String).

    c. Se sobreescriu la línia completa modificada.

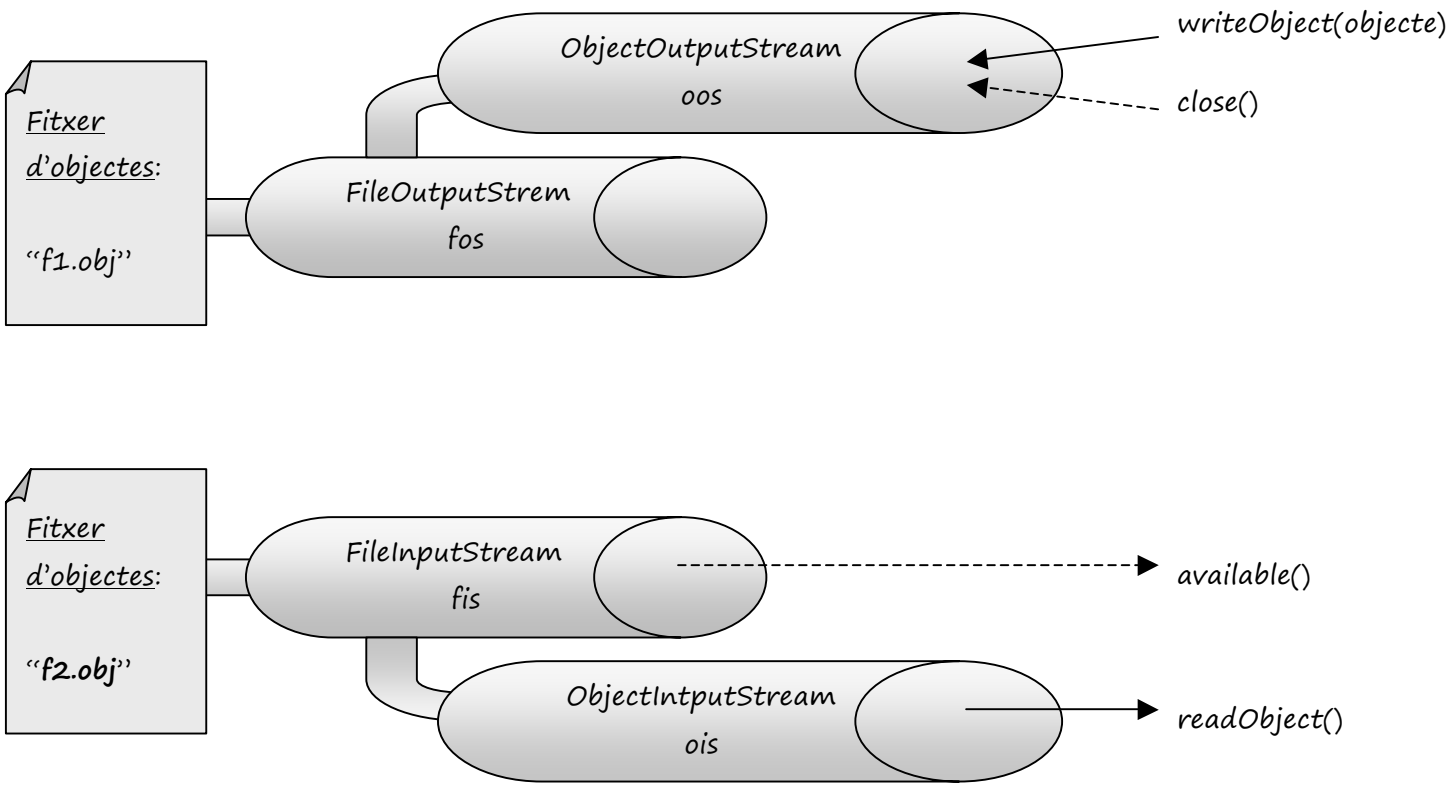
6 Fitxers d'objectes. Lectura i escriptura

L'inconvenient de guardar la informació en mode binari com hem fet abans és que cal llegir en el mateix ordre en què hem escrit les dades. Si, per exemple, volem guardar en un fitxer moltes "fitxes" d'alumnes, escriurem al fitxer els distints objectes i després llegirem objectes.

Per a fer això, **els objectes necessiten ser *serialitzats*** (que l'objecte es puga convertir en una seqüència de bytes). Simplement, hem d'indicar-ho en la definició de la classe. Així:

```
class NomClasse implements java.io.Serializable {
        ...
}
```

I en l'esquema següent tenim les classes que cal utilitzar per a utilitzar els mètodes de lectura i escriptura sobre fitxers d'objectes:



Per a escriure objectes al fitxer cridarem al mètode **writeObject()**, passant-li com a paràmetre l'objecte que volem escriure. L'escriptura és seqüencial I destructiva (cada volta es crearà un nou fitxer).

Per a llegir del fitxer d'objectes cridarem al mètode **readObject()**, que ens retorna un objecte però de la classe Object. Per tant, haurem de fer-li un casting per a convertir-lo al tipus d'objecte que estem llegint. Ara bé, abans de llegir objectes, caldrà assegurar-se que en queden per llegir. Per a això, cridarem al mètode **available()** (de la classe FileInputStream), el qual retorna la quantitat de bytes que falten per llegir al fitxer.

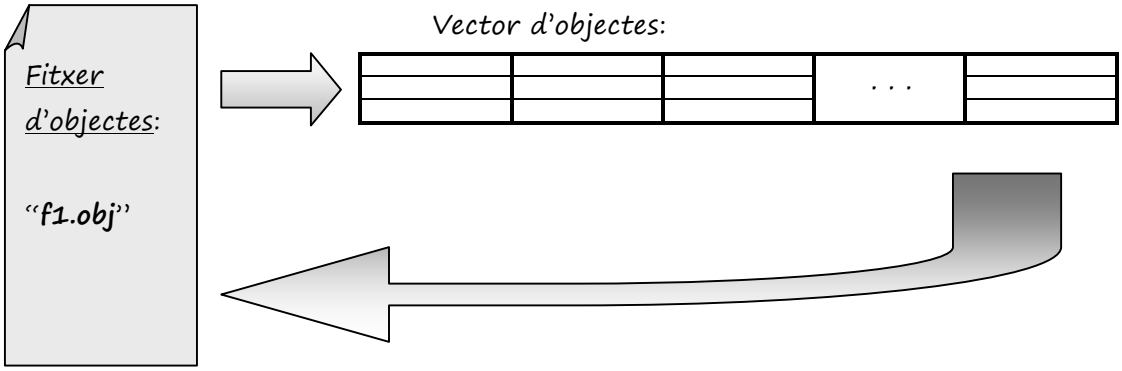
Exemple: Anem a escriure punts (parells de posicions (x,y)) i llegir-los:

```
import java.io.*;

class Punt implements java.io.Serializable{

    private int x;
    private int y;

    Punt(){
        x=y=0;
    }
    Punt (Punt p){
        x=p.x;
        y=p.y;
    }
    Punt (int xx,int yy){
        x=xx;
        y=yy;
    }
    int getX(){
        return x;
    }
    int getY(){
        return y;
    }
    void imprimir(){
        System.out.println("("+x+","+y+")");
    }
} //class Punt

public class FitxersObjectes {
    public static void main(String[] args)  {
        Punt p1=new Punt(0,0);
        Punt p2=new Punt(2,-1);
        p1.imprimir();
        p2.imprimir();
        ObjectOutputStream OSortida=null;
        FileOutputStream FSortida=null;
        ObjectInputStream OEntrada=null;
```

```
            FileInputStream FEntrada=null;
            try {
                FSortida= new FileOutputStream("Punts.obj");
                OSortida=new ObjectOutputStream(FSortida);
                OSortida.writeObject(p1);
                OSortida.writeObject(p2);
                OSortida.writeObject(new Punt(3,3));
                OSortida.close();
                System.out.println("Fitxer tancat. Ara llegirem d'ell");
                FEntrada= new FileInputStream("Punts.obj");
                OEntrada=new ObjectInputStream(FEntrada);
                Punt p;
                //mentre queden bytes per llegir
                while (FEntrada.available()>0){
                    p=(Punt)OEntrada.readObject();
                    p.imprimir();
                }
                System.out.println("Adeu!!");
            }//try
            catch (EOFException ex) {
                System.out.println("Fi de Fitxer.");
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            } catch (ClassNotFoundException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        } //main
    }//class
```

Què fem si volem afegir objectes a un fitxer existent (sense destruir-lo)?

Amb programes amb gran volum d'informació, les dades inicialment estaran al disc dur dins d'un fitxer. El nostre programa, per a utilitzar i manipular eixes dades, les portarà a memòria principal volcant els objectes del fitxer en un vector d'objectes. A continuació, les dades podran ser consultades, esborrades, introduir nous objectes, etc. I, quan volem acabar, volcarem el vector al disc dur, escrivint tots els objectes del vector en el fitxer que teníem.



Exercicis

13. Volem tindre una agenda on pugam guardar el nom, telèfon i adreça de cada amic. Fes un programa amb un bucle i un menú amb les següents opcions:

    a. Donar d'alta un amic
    b. Consultar un amic pel seu nom
    c. Saber la quantitat d'amics enregistrats
    d. Mostrar tota l'agenda per pantalla
    e. Esborrar un amic
    f. Modificar les dades d'un amic
    g. Importar dades
    h. Exportar dades

Les opcions més importants d'este exercici pel que fa a l'ús de fitxers són les d'importar i exportar dades. L'opció d'importar dades demanarà un nom de fitxer on estan els contactes i ho passarà a l'ArrayList. I l'opció d'exportar dades demanarà un nom de fitxer i guardarà allí les dades de l'ArrayList.