Modificació o transformació d'un fitxer de text

Fins ara, hem vist com podem extreure informació d'un fitxer de text, i fins i tot com deixar aquesta en un altre o altres fitxers. En aquest document veurem com realitzar una de les accions més típiques sobre fitxers de text, que és la seva modificació o transformació. Imaginem que tenim un fitxer de text que volem passar a majúscules, o al volem eliminar tots els seus accents, dièresis i circumflexes. Això té com a resultat un altre fitxer, semblant a l'original, però que ha patit una transformació. Aquesta operació, la podríem realitzar a partir del que sabem fins ara, però ara anem a tractar-ho més formalment, com a cas particular.

Partirem del supòsit que, en la transformació d'una línia, no intervé informació d'altres línies del fitxer. Sota aquest supòsit, que és el més habitual, la transformació del fitxer la farem línia a línia. Si no es complís el supòsit, la forma de procedir podria ser carregar tot el fitxer a la memòria, transformar-lo allà, i escriure el resultat al fitxer de destí (que com ja hem dit no serà el que nosaltres farem). La transformació línia a línia, consumeix menys recursos de memòria, ja que en un moment donat només tenim emmagatzemada al nostre programa una línia del fitxer.

Sempre que transformem un fitxer, hem de preservar la informació d'aquest, per evitar que es pugui produir un problema i perdem el fitxer original. Una forma estàndard de fer-ho és la creació d'un fitxer amb extensió **.bak**, que és un fitxer de **backup** (còpia de seguretat). Nosaltres optarem per fer-ho d'aquesta forma.

La forma de procedir serà doncs:

- Comprovació de si al directori on tenim el fitxer original de partida hi ha un altre fitxer amb el mateix nom i extensió **.bak**. En aquest cas, avortarem l'execució del programa (sempre informant a l'usuari), per evitar perdre la informació valuosa que pogués existir en aquest fitxer. Haurà de ser l'usuari qui mogui o esborri aquest fitxer abans de procedir.
- Preservació del contingut actual del fitxer original en un fitxer de **backup**. Canviarem de nom el fitxer original, afegint-li l'extensió **.bak**, i aquest serà el fitxer que farem servir per realitzar la transformació sobre un fitxer nou que tindrà el mateix nom que l'original. Es a dir, el nostre fitxer de partida d'on llegirem la informació, passa a ser el fitxer **.bak**.
- Llegirem línia a línia del fitxer **.bak**, transformant cada línia llegida independentment, i escrivint-la al fitxer de destí creat, al que hem donat el mateix nom que l'original.
- En acabar sense errors, l'usuari disposa del fitxer transformat, amb el mateix nom que l'original, i del fitxer original sense transformar a la mateixa carpeta que el fitxer transformat, i extensió **.bak**.
- Si es produís algun error en la transformació del fitxer, hi ha diferents possibilitats. Una d'elles seria tornar a canviar el nom al fitxer original (que ara té extensió **.bak**, i que no s'ha pogut transformar per algun motiu). Aquesta alternativa, donat que ha hagut un error, podria conduir a una sèrie d'errors en cascada, i caldria valorar abans que és el que ha passat realment (fitxer original obert per una altra aplicació en mode exclusiu, falta d'espai al disc, usuari que ha retirat la unitat on estaven els fitxers...). El que nosaltres farem és, un cop arribats en aquest punt, informar a l'usuari de que no s'ha pogut realitzar la transformació (el fitxer de destí pot haver quedat escrit parcialment), i que disposa d'una còpia de seguretat de la informació original al fitxer **.bak**. Ell decidirà que fer amb aquesta còpia, i així, ens traiem responsabilitats de sobre.

Sempre que treballem amb fitxers, hem de tenir presents tots els escenaris. En el **99,999%** de les ocasions, no tindrem cap problema i el fitxer original serà transformat correctament, però hem de considerar també el cas poc provable que es produeixi qualsevol tipus d'error en el procés. La informació és el més valuós amb el que treballem els informàtics, i preservar-la ha de ser el nostre principal objectiu. Tot el que fem en aquest sentit, mai es temps perdut, i ens adonarem quan passi alguna cosa que podria haver estat irreparable si no haguéssim fet el que calia.

Procés operatiu per a la transformació del fitxer

El mètode **main** d'un programa per transformar fitxers de text podria ser el següent, on hem optat per simplificar el procés de comprovació dels errors (simplement informem a l'usuari, i ni tan sols comprovem si el fitxer **.bak** se ha generat correctament). Fixeu-vos que hem decidit seleccionar el fitxer mitjançant **JFileChooser**.

```
public static void main(String[] args) {
    // Escollirem el fitxer a transformar amb JFileChooser
    JFileChooser selector = new JFileChooser();
    selector.showOpenDialog(null);
    selector.setFileSelectionMode(JFileChooser.FILES_ONLY);
    // Aquí tenim el nom del fitxer seleccionat
    if (selector.getSelectedFile() != null) {
        String nomFitxer = "" + selector.getSelectedFile();
        System.out.println("El nom del fitxer seleccionat és: " + nomFitxer);
        if (trasformaFitxer(nomFitxer)){
            System.out.println("   Procés completat.  Disposa d'una còpia del fitxer original amb");
            System.out.println("Extensió .BAK al mateix directori.  Esbórri-la si no la necessita");
        }
        else{
            System.out.println("El procés de transformació del fitxer no s'ha pogut realitzar correctament");
            // Al cas més senzill, ni tan sols comprovariem si el fitxer .bak existeix, que ho faci l'usuari
            System.out.println("El fitxer original pot haver quedat al mateix directori amb extensió  .bak");
        }
    }
}
```

Hem deixat la feina principal a la funció **transformaFitxer**, a la que passem el nom del fitxer a transformar, i retorna **true** si el procés s'ha realitzat correctament, i **false** si no. Aquesta funció s'encarrega també de realitzar el control de totes les excepcions, i de informar a l'usuari de qualsevol error que es produeixi, però no propaga aquestes cap al programa principal (que podria ser una alternativa també vàlida, enlloc de retornar **true** o **false**).

Fixeu-vos que el objecte de la classe **File** retornat per **JFileChooser** no és una cadena, i no es pot assignar a una cadena directament, però si que es converteix automàticament en cadena si el concatenem per exemple a la cadena buida.

La funció **transformaFitxer** comprova si el fitxer d'origen existeix, i si no ho troba (seria estrany, ja que l'hem seleccionat amb el diàleg presentat per **JFileChooser**, però l'usuari podria haver tret el pendrive just després d'indicar el fitxer desitjat), surt al sistema operatiu. Comprova també que el fitxer de destí **.bak** no existeix prèviament, i si fos el cas, informa a l'usuari i també finalitza l'execució del programa. En aquests dos supòsits es surt al S.O.

La funció crearà dos objectes **File**, un per al fitxer d'origen (que al nostre cas serà el fitxer amb extensió **.bak**) i un altre per al fitxer de destí (amb el nom de l'original, que serà on escriurem les línies modificades del fitxer).

Per canviar de nom el fitxer (li afegim l'extensió **.bak**), podem fer servir el mètode **move** de la classe **Files** de la forma que s'indica:

```
// Movem el fitxer original al fitxer .bak,  ds és un objecte File que apunta al fitxer original i or al .bak
Files.move(Paths.get(ds.getAbsolutePath()),Paths.get(or.getAbsolutePath()),StandardCopyOption.ATOMIC_MOVE);
```

getAbsolutePath és un mètode que, a partir d'un objecte **File** retorna una cadena amb el **path absolut** (des de l'origen), incloent-hi el nom del fitxer. Per convertir això amb un objecte **Path**, que son els objectes amb els que treballa el mètode **move** de la classe **File**, fem servir el mètode **get** de la classe **Paths**. El canvi de nom el fem de forma atòmica (últim paràmetre), i així ens assegurem que cap procés executant-se a la nostra màquina el pot interrompre (pot executar-se al mig).

Una altra possibilitat seria fer-nos una funció (o una classe) per moure el fitxer amb el que saben de tractament de fitxers (llegint caràcters del fitxer original i escrivint-los al fitxer de destí per exemple), però això és més complicat i susceptible a errors que hauríem de controlar. Podeu trobar un exemple d'això a: http://codigoxules.org/java-copiar-ficheros-java-io/

Per evitar que aquesta funció **transformaFitxer**, que és la que realitza la major part de la feina, creixi molt i es faci difícil de gestionar, podem crear-nos una funció **transformaLinies**, a la que passem els objectes de la classe **BufferedReader** i **BufferedWriter** d'on llegirem i escriurem respectivament les línies del fitxer. Fixeu-vos que a part dels tipus bàsics del llenguatge, podem passar com paràmetres a les funcions qualsevol objecte, fins i tot els que creem a partir de les nostres classes.

```
try (FileReader fr = new FileReader(or);
     BufferedReader br = new BufferedReader(fr)){
    try (FileWriter fw = new FileWriter(ds);
        BufferedWriter bw = new BufferedWriter(fw)){
        transformaLinies(br,bw);
        return true;     // Fitxer transformat correctament
    }catch (Exception e){
        System.out.println("Error en intentar transformar les línies del fitxer");
        System.out.println("L'error reportat pel sistema ha estat:");
        System.out.println(e.getMessage());
        return false;       // no s'ha pogut transformar el fitxer
    }
}catch (Exception e){
```

Si la funció **transformaLinies** que és l'encarregada de realitzar la transformació de les línies que llegim del fitxer d'origen i escriure-les al fitxer de destí, funciona sense errors, la funció **transformaFitxer** retornarà **true**.

La funció **transformaLinies** podria ser tan senzilla com:

```
static void transformaLinies(BufferedReader br, BufferedWriter bw) throws Exception {
    String linia;
    while ((linia = br.readLine()) != null){
        // Escriu la línia transformada al fitxer de destí
        bw.write(transformaLinia(linia));
        bw.newLine();       // Afegeix el retorn de carro al fitxer de destí
    }
}
```

Fixeu-vos que qualsevol excepció que es produeixi en aquesta funció, serà propagada cap enrere, a la funció que l'ha cridat, i és allà on tractem l'error. També podríem haver capturat l'error en aquesta funció, i propagar-lo explícitament, però no és necessari:

```
static void transformaLinies(BufferedReader br, BufferedWriter bw) throws Exception {
    String linia;
    try {
        while ((linia = br.readLine()) != null){
            // Escriu la línia transformada al fitxer de destí
            bw.write(transformaLinia(linia));
            bw.newLine();       // Afegeix el retorn de carro al fitxer de destí
        }
    } catch (Exception e){
        // Propaguem l'excepció si aquesta es produeix
        throw e;
    }
}
```

Per últim, la funció **transformaLinia**, rep com a paràmetre la cadena amb la línia del fitxer que volem transformar, i retorna aquesta cadena modificada. La transformació pot ser passar a majúscules o minúscules aquesta cadena, treure els espais, o els accents, o qualsevol altra que se'ns acudeixi.