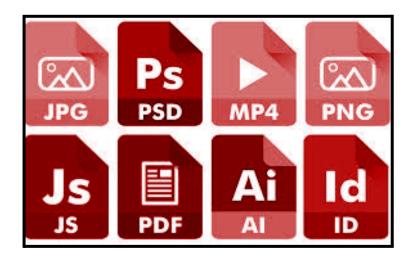
Manual técnico conversor



-Por Daniel Álvarez Morales

ÍNDICE

1 Menú y muestras de directorio	. 2
2 Excepciones	. 2
3 Extensiones	. 3
4 Escritura del fichero	. 4
5 Conversiones	.4

1.- Menú y muestras de directorio

-El menú se compone por un bucle while y un booleano como salida y la información de las carpetas se controlan con "if" si es nulo se exige un directorio sino se lee dicho directorio.

```
public boolean verRutaDirectorio(File directorio) {
   boolean esNull = true;
   if (directorio != null) {
        // Path ruta = directorio.toPath();

        System.out.println("Carpeta seleccionada: " + directorio.getAbsolutePath());
        return esNull = false;

} else {
        System.out.println("Selecciona primero un directorio para ver su ruta.");
        return esNull = true;
    }
}
```

2.- Excepciones

-Algunas excepciones existen en el main para controlar que el usuario introduzca letras o números raros en los scanners tipo int (InputMismatchException).

```
try {
    System.out.print("Seleccione una opción: ");
    opcion = sc.nextInt();
    sc.nextLine();
    volverIntroducir = true;
} catch (InputMismatchException e) {
    System.out.println("Error, entrada incorrecta, asegurate de introducir un numero del menú.");
    sc.nextLine();
    e.printStackTrace();
}
```

-Con esta práctica el programa no se interrumpirá de golpe y será más robusto a fallos.

-La mayoría de excepciones que sirven para detectar errores se lanzan, se detectan y tratan en el main. Simplificando así el control de excepciones.

3.- Extensiones

-La extensión se obtiene con el método de la imagen "detectoExtension" que trabaja con un string pasado por parámetro.

```
public String detectorExtension(String nombreArchivo) {
    String[] partes = nombreArchivo.split("\\.");

String extension = partes[partes.length - 1].toLowerCase();
    return extension;
}
```

-Y con un switch case se llevará a su correspondiente conversion según su extensión y su opción asignada por el usuario que solo pueden ser dos opciones en contraparte de su extensión.

```
case "xml":
    System.out.println("Convirtiendo '" + extension + "' a '" + extensionAconvertir + "' ");
    Files.createFile(rutaYSalida);
    List<String> textoConvertido = conversionCsvAxml(texto);
    escribirFichero(rutaYSalida, textoConvertido);
    break;
```

4.- Escritura del fichero

-La escritura se desarrolla en la función "escribirFichero".

```
public void escribirFichero(Path ruta, List<String> contenidoPorLinea) throws IOException {
    Files.write(ruta, contenidoPorLinea, StandardOpenOption.APPEND);
}
```

-Los parámetros son ruta (el path del archivo convertido) y contenidoPorLinea (el texto convertido).

5.- Conversiones

-Las funciones de las conversiones son diversas y densas.

Se basan sobre todo en un texto **con una forma que tiene ser concreta** por parámetro que será procesado , reconstruido y devuelto en una estructura dinámica del mismo tipo llamado "textoWork".

```
public List<String> conversionCsvAxml(List<String> texto) {
    List<String> textoWork = new ArrayList<>();
```

-Formas concretas que tienen que tener los archivos para convertir con éxito:

a) Forma archivo vcs:

```
id,nombre,edad
1,Juan,25
2,Ana,30
3,Luis,22
```

b) Forma archivo xml:

c) Forma archivo json:

-Al convertirse, si se vuelven a convertir tienen que tener dicha forma mostrada anteriormente, sino las líneas no se procesarán como es previsto y no se imprimirán en el fichero correctamente. Quizás en una futura actualización se pueda mejorar.