Cuaderno de ejercicios

Tema 1 - Ficheros

Sección 1

1.1 Realiza un programa que reciba como parámetro de entrada un directorio y lo muestre per pantalla.
1.2 Ampliar el programa anterior para que muestre todas las características de interés del directorio, tomando como referencia la información que proporciona la clase File.
1.3 Introducir una comprobación en el programa anterior para determinar si el directorio existe.
1.4 Realizar un programa que, dado un directorio, compruebe si existe y devuelva un mensaje de confirmación si existe o una alerta en caso contrario.
1.5 Realiza un programa que reciba como parámetros de entrada un directorio y una extensión de fichero (por ejemplo .txt) y devuelva por pantalla todos los ficheros del directorio que cumplan el criterio.
1.6 Modifica el programa anterior para que tenga en cuenta que, si no se le pasa ninguna extensión como parámetro, muestre todo el contenido del directorio.
1.7 Modifica el programa anterior para que admita como parámetros de entrada un nombre cualquiera de extensiones, devolviendo después por pantalla todos los ficheros del directorio que tengan alguna de las extensiones indicadas.
1.8 Desarrolla un programa que, dado un fichero, realice una copia del mismo (en el mismo directorio y cambiándole el nombre) y lo borre después. Muestra una traza por pantalla de las acciones para ver que se realizan.

1.9 Miniproyecto

En este proyecto se va a trabajar con las funcionalidades de la clase `File` de Java.

Se debe desarrollar una aplicación en Java que tenga como parámetro de entrada un directorio de trabajo en el que haya al menos un subdirectorio y un fichero y que implemente los siguientes métodos:

Método	Funcionalidades
`getInformacion`	• Mostrar el nombre, tipo (si es fichero o directorio), la ubicación (ruta completa), la fecha de la última modificación en formato fecha y si es oculto o no. • Si es un fichero deberá mostrar su tamaño en bytes. • Si es un directorio deberá mostrar el número de elementos que contiene, espacio libre, espacio disponible y espacio total.
`creaCarpeta`	Crear una carpeta en directorio local.
`creaFichero`	Crear un fichero en la carpeta creada.
`elimina`	Eliminar ficheros/carpetas.
`renombra`	Renombrar ficheros/carpetas.

Deberá haber una gestión de errores (como mínimo, controlar que exista y que se tienen permisos).

Tendrás que localizar los métodos de la clase `File` que permiten realizar la funcionalidad deseada a través del API de Java. El resultado de las llamadas a los distintos métodos, se puede

concatenar en forma de un objeto `String`, que se utilizará para mostrar la información desde la interfaz.

Como ampliación se propone implementar un pequeño menú de texto que permita al usuario elegir qué funcionalidad quiere utilizar y solicitarle (si procede) la información necesaria para su ejecución (por ejemplo, nombre del fichero que se quiera crear).

Sección 2

2.1 Escribe un programa que reciba como parámetro de entrada la ruta de un fichero, lea su contenido y lo muestre por pantalla carácter a carácter.
2.2 Introduce una modificación en el programa anterior para que admita otro parámetro de entrada adicional que permita especificar la velocidad a la que se muestren los caracteres.
2.3 Realiza otro programa que muestre un número determinado de caracteres por pantalla (por ejemplo 100), espere a que el usuario presione alguna tecla, muestre otro bloque de caracteres, vuelva a esperar, y así sucesivamente hasta mostrar todo el contenido.
2.4 Crea un programa que, dado un fichero de texto, lea y muestre su contenido línea a línea.
2.5 Modifica el programa anterior para que acepte como parámetros de entrada un número que indique la velocidad a la que se muestran las líneas.
2.6 Crea otro programa a partir del anterior que en vez de mostrar el contenido por consola lo escriba en otro fichero del mismo directorio.
2.7 Realiza un programa que permita recibir por teclado una serie de strings por parte del usuario y los vaya escribiendo en un fichero de texto. Como condición de finalización, el usuario deberá escribir un string que sea “exit”.
2.8 Modifica el programa anterior para que el nombre del fichero incluya la fecha y la hora de creación.

2.9 Miniproyecto

Se pide desarrollar una pequeña aplicación con interfaz gráfica siguiendo un patrón de arquitectura MVC similar al visto en los ejemplos del tema.

La aplicación deberá tener las siguientes funcionalidades:

- Mostrar un texto en la interfaz gráfica obtenido a partir de un fichero TXT que se proporcionará para la actividad.
- Función Buscar: buscar una palabra en el texto y contar cuántas veces aparece. La palabra se proporcionará a través de la interfaz gráfica por el usuario. El número de veces que aparece la palabra se puede mostrar como una ventana tipo popup (popup message) o dentro del mismo campo donde se introduce la palabra a buscar.
- Función Reemplazar: reemplazar en el texto la palabra buscada por otra especificada en otro campo de texto. El texto resultante se mostrará en otro `textArea` de la interfaz y se guardará automáticamente como un fichero nuevo en el mismo directorio donde está el fichero original.

Se pide seguir la siguiente arquitectura:

- Clase `Modelo`: gestiona el acceso a los ficheros. Aquí se implementarán funciones para abrir y cerrar buffers, así como las funciones que hagan uso de estos buffers.
- Clase `Vista`: gestiona la interfaz gráfica. Aquí se implementa el acceso a los componentes gráficos. Se proporciona una clase `Vista.java` con los componentes y métodos necesarios (figura 1), pero también se puede desarrollar una interfaz gráfica desde cero.

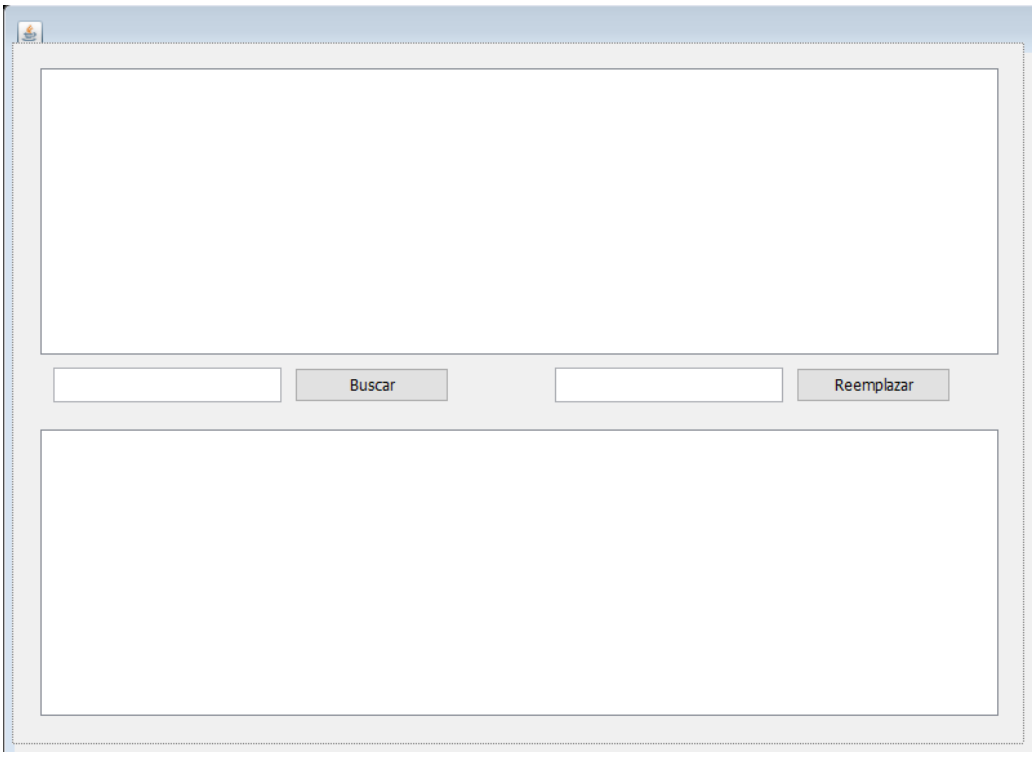

Figura 1. Captura de pantalla de la interfaz proporcionada para la realización de la actividad.

- Clase `Controlador`: gestiona los eventos que se producen según los botones que se pulsan.
- Clase `Principal`: gestiona la creación de los objetos de las clases anteriores.

Aspectos a tener en cuenta:

- Las excepciones que se produzcan en la clase Modelo se deberán mostrar mediante un diálogo de error (popup message).

Sección 3

3.1 Abre un editor de texto (Notepad++ o similar) y crea un fichero XML con una estructura típica que represente un objeto y sus principales características. Asegúrate de incluir varios objetos en el fichero.

3.2 Realiza un programa que dado el fichero creado en el ejercicio anterior lo muestre por pantalla línea a línea.

3.3 Crea un programa que implemente un parser para gestionar el fichero XML y devuelva por pantalla el número de nodos (objetos) que haya en el fichero.

3.4 Amplía el programa anterior para que además recorra los nodos uno a uno y muestre por pantalla sus atributos.

3.5 Introduce en el programa anterior un método que implemente la clase objeto que has elegido para el XML. Puede ser un objeto Java común (POJO, Plain Old Java Object) con constructor, setters y getters.

3.6 Para probar que el objeto funciona correctamente, realizamos otra modificación que implemente que a medida que se lean los nodos del XML se vayan creando objetos y guardándolos en una lista.

3.7 Crea otra funcionalidad que permita a un usuario introducir objetos nuevos en la lista. Para ello se le deben pedir los valores de los atributos, posteriormente crear un objeto con dichos atributos y, finalmente, añadir el objeto a la lista.

3.8 Como última funcionalidad, se pide que se guarde la lista completa de objetos en un nuevo fichero XML. Se debe comprobar que el formato del fichero resultante se corresponda al esperado para un fichero XML (indent o sangría adecuados).

3.9 Miniproyecto

Este proyecto tiene como objetivo gestionar una biblioteca de libros (mínimo 5 libros), para la cual cosa se creará una clase `Biblioteca`.

Se deberá definir un objeto de tipo `Libro` (clase) con los siguientes atributos: identificador, título, autor, año de publicación, editorial y número de páginas.

La información de cada libro se deberá almacenar en formato XML. Puedes guardar todos los libros en un mismo fichero XML o utilizar ficheros independientes XML para cada libro que estén en un mismo directorio.

Los métodos que se pide implementar son los siguientes:

- `crearLibro`: crear un nuevo libro como XML a partir de los datos proporcionados por el usuario, devuelve el identificador del libro.
- `mostrarLibro`: muestra los atributos del libro por pantalla.
- `borrarLibro`: borra un objeto libro a partir de un identificador.
- `actualizaLibro`: actualiza (modifica) la información de un objeto libro a partir de un identificador.
- `recuperarTodos`: devuelve una lista con todos los libros de la biblioteca.
- `main`: deberá mostrar un menú con los siguientes elementos: “1. Mostrar todos los títulos de la biblioteca (solo el título)”, “2. Mostrar información detallada de un libro”, “3. Crear nuevo libro”, “4. Actualizar libro”, “5. Borrar libro” y “6. Cerrar la biblioteca”.