Alumno: Jilai Xu
Grado: Desarrollo de Aplicaciones Multiplataforma

Actividad 1.1 Investigación sobre Ficheros y tratamiento I

- **File**: Representa la ruta de un archivo o directorio en el sistema de archivos. Permite consultar metadatos (tamaño, permisos, si existe), crear o borrar archivos y carpetas pero no tiene permisos de lectura ni escritura.

- **FileReader**: Flujo de caracteres diseñado para leer datos de texto desde un archivo de manera secuencial (normalmente carácter a carácter o en pequeños arrays).

- **BufferedReader**: Permite leer líneas completas de texto de forma muy eficiente mediante el método readLine().

- **FileWriter**: Flujo de caracteres diseñado para escribir texto directamente en un archivo.

- **BufferedWriter**: Acumula los datos en memoria antes de escribirlos físicamente en el disco, lo que optimiza el rendimiento.

- **PrintWriter**: Clase de conveniencia que permite escribir datos con formato (con métodos intuitivos como print(), println() y printf()). Facilita la impresión de tipos de datos primitivos y cadenas.

Alumno: Jilai Xu
Grado: Desarrollo de Aplicaciones Multiplataforma

¿Qué diferencia hay entre FileReader y BufferedReader? ¿Cuál es más eficiente en el tratamiento de información? ¿Por qué?

Característica	FileReader	BufferedReader
Funcionamiento	Lee los datos directamente del archivo físico carácter por carácter o en pequeños bloques.	Lee bloques grandes de datos hacia una memoria intermedia (búfer) en la RAM.
Eficiencia	Menor eficiencia, ya que realiza accesos frecuentes al disco duro por cada carácter o bloque pequeño.	Alta eficiencia, ya que reduce drásticamente el número de accesos físicos al disco.
Lectura por líneas	No cuenta con un método integrado y eficiente para leer líneas completas de texto.	Cuenta con el método readLine(), lo que facilita enormemente la lectura línea por línea.
Uso recomendado	Ideal únicamente para ficheros muy pequeños donde el rendimiento no sea crítico.	La opción estándar y recomendada para la lectura eficiente de archivos de texto.

¿Qué diferencia hay entre FileWriter, BufferedWriter y PrintWriter?

Aspecto	**FileWriter**
Diferencia principal	• **FileReader**: Lee los datos directamente del archivo físico (carácter por carácter o bloque pequeño). • **BufferedReader**: Lee bloques grandes de datos hacia una memoria intermedia (búfer) en RAM
¿Cuál es más eficiente?	BufferedReader
¿Por qué?	Al utilizar un búfer, se reduce drásticamente el número de accesos físicos al disco (operaciones de E/S), que son costosos en términos de rendimiento. Además, facilita la lectura por líneas completas en lugar de tener que procesar el texto carácter por carácter.

Alumno: Jilai Xu
Grado: Desarrollo de Aplicaciones Multiplataforma

Entre FileWriter y PrintWriter ¿Cuál de las dos clases lanza excepciones de tipo IOException?¿Cómo gestionará los errores la otra clase? ¿Cuál de las dos clases usarías para añadir/escribir texto al final de un fichero?

¿Cuál lanza IOException?:

FileWriter sí lanza excepciones de tipo IOException de forma explícita, obligando a gestionarlas mediante bloques try-catch o declarándolas con throws.

¿Cómo gestiona los errores la otra?:

PrintWriter no lanza excepciones de E/S directamente en sus métodos de impresión. Si ocurre un error, la clase lo silencia internamente y activa un indicador de fallo que se puede verificar mediante el método checkError().

¿Cuál usar para añadir texto al final (*append*)?:

Ambas clases permiten añadir texto al final del archivo sin sobrescribirlo si se pasa un true como segundo parámetro en su constructor (por ejemplo: new FileWriter("fichero.txt", true)). PrintWriter combinado con un FileWriter en modo *append* es una opción muy utilizada y cómoda por su facilidad de uso.