2.5.- Ficheros de registro (Logs) con `java.util.logging`

Los ficheros de registro permiten almacenar información sobre las distintas acciones que realiza la aplicación, facilitando el seguimiento detallado de los eventos del sistema. Java incorpora de forma nativa la clase **Logger** para este propósito.

Pasos para implementar el registro de eventos

Para trabajar con ficheros de registro en Java, debemos seguir este orden de ejecución:

1. **Importar la librería:** `import java.util.logging.*;`
2. **Crear el objeto Logger:** `Logger logger = Logger.getLogger("MyLog");`
3. **Configurar el destino (FileHandler):** Creamos un manejador de archivo para definir dónde se guardará el log. `FileHandler fh = new FileHandler("C:\\temp\\MyLogFile.log", true);` *El valor `true` indica que los nuevos registros se añaden al final del fichero (append mode).*
4. **Definir el formato:**
    a. **SimpleFormatter:** Para texto plano legible.
    b. **XMLFormatter:** Para estructura de datos XML. `fh.setFormatter(new SimpleFormatter());`
5. **Asignar el manejador al Logger:** `logger.addHandler(fh);`
6. **Configurar la visibilidad y nivel:** Para evitar que los mensajes salgan duplicados por la consola del sistema: `logger.setUseParentHandlers(false);`

Jerarquía de niveles de registro

Es fundamental definir el filtro de importancia mediante `logger.setLevel()`. El sistema solo registrará los eventos que tengan un nivel **igual o superior** al configurado:

Nivel	Importancia	Descripción
SEVERE	7 (Máxima)	Errores críticos del sistema.
WARNING	6	Advertencias de posibles problemas.
INFO	5	Mensajes informativos del flujo.
CONFIG	4	Detalles de configuración técnica.

FINE/FINER/FINEST	3 - 1	Niveles de depuración detallada.

Exportar a Hojas de cálculo

Ejemplo de implementación completa

Este código muestra cómo integrar todos los pasos anteriores en una clase funcional:

Java

```
import java.util.logging.*;
import java.io.IOException;

public class GestionLogs {

    public static void main(String[] args) {
        Logger logger = Logger.getLogger("PruebaLog");
        FileHandler fh;

        try {
            // 1. Configuración del archivo y modo 'append'
            fh = new FileHandler("RegistroActividad.log", true);
            logger.addHandler(fh);

            // 2. Formato de texto simple
            SimpleFormatter formatter = new SimpleFormatter();
            fh.setFormatter(formatter);

            // 3. Desactivar salida por consola estándar
            logger.setUseParentHandlers(false);

            // 4. Configurar nivel de filtrado (Registrar todo)
            logger.setLevel(Level.ALL);

            // 5. Ejemplos de registros de diferentes niveles
            logger.log(Level.INFO, "La aplicación se ha iniciado
correctamente.");
            logger.log(Level.WARNING, "Intento de acceso fallido con
usuario: admin");

            // Ejemplo de registro de error en un bloque catch
            try {
                int division = 10 / 0;
```

```
            } catch (ArithmeticException e) {
                logger.log(Level.SEVERE, "Error matemático
detectado", e);
            }

        } catch (SecurityException | IOException e) {
            e.printStackTrace();
        }
    }
}
```

¿Te gustaría que te ayude a crear una clase "Wrapper" o auxiliar para que puedas llamar al logger desde cualquier parte de tu aplicación sin repetir toda la configuración?