Metodología de la Programación

Apuntes hechos por Marina Sancho, Lía de Miguel, Lorena Ballesteros, y Brianda García

APUNTES: LOG4J, BIG DATA Y FRAMEWORKS DE INTERFAZ

🔹 Apache Log4j y el uso profesional de logs en Java

- Es una buena práctica en programación registrar información interna del programa mediante logs, en lugar de imprimir en consola (System.out.println), que puede generar problemas en entornos reales.
- **Log4j es una librería de Java que permite:**
    - Generar mensajes de depuración clasificados por niveles (INFO, WARN, ERROR, etc.).
    - Guardar los registros en diferentes destinos: archivos, consolas, servidores remotos.
    - Configurar el formato de salida de los mensajes (tiempo, clase, tipo de mensaje…).
- **Para usar Log4j en Java:**
    - Añadir la librería al proyecto.
    - Crear un Logger mediante LogManager.
    - Usar métodos como logger.info(), logger.error() para registrar eventos.
- Se puede configurar que cada clase escriba sus logs en ficheros separados, establecer un tamaño máximo para los archivos y activar la rotación automática.
- **Los logs no deben eliminarse nunca, por motivos:**
    - Técnicos (depuración y control del sistema).
    - Legales (auditorías).
    - Económicos (análisis de uso).

- Actualmente, existen herramientas como OpenTelemetry, que permiten:
    - Obtener métricas, logs y trazas en tiempo real.
    - Medir el rendimiento interno de cada componente o método.
    - Detectar cuellos de botella (tiempos superiores a 100 ms pueden ser críticos).

🔹 Big Data: evolución y conceptos clave

- Big Data es la evolución del concepto de Data Warehouse, que a su vez viene de los sistemas de análisis de datos (analítica) y de los procesos operacionales.
- **Problema inicial:** las bases de datos operacionales no podían soportar la carga de trabajo analítico → se crea el Data Warehouse, donde se trasladan los datos para analizarlos sin afectar el sistema principal.
- Entre 2000 y 2010, empresas como Google y Amazon necesitaban gestionar:
    - Millones de usuarios.
    - Grandes cantidades de datos no estructurados (logs, publicaciones, imágenes, etc.).
- Big Data surge para manejar esta información:
    - Integra datos estructurados, semiestructurados y no estructurados.
    - Usa arquitecturas específicas de almacenamiento y procesamiento masivo.
- Aparecen nuevos conceptos:
    - Data Lake: repositorio donde se almacena todo tipo de datos sin necesidad de estructura previa.
    - Data Lakehouse: combinación entre la potencia analítica del Data Warehouse y la flexibilidad del Data Lake.

🔹 Interfaces, Frameworks y automatización del trabajo

💻 Evolución de las interfaces de usuario

- La interfaz de usuario es el punto de contacto entre el sistema y el usuario. Un diseño adecuado facilita su uso y reduce errores.
- La automatización de tareas ha sustituido a muchas profesiones, especialmente aquellas repetitivas. La IA y los sistemas inteligentes ya están empezando a afectar incluso a trabajos de cuello blanco (médicos, abogados, programadores...).
- Las interfaces evolucionan constantemente para adaptarse a los cambios tecnológicos:
    - En automoción, por ejemplo, un coche cambia de diseño visual cada 5–10 años.
    - En informática, se actualizan partes visuales más frecuentemente para mantener el interés del usuario.

🧩 Tipos de interfaces

Tipo	Características
Nativas	Dependen del sistema operativo (ej. apps Android, Windows). El usuario espera que se comporten como parte del sistema.
Web	Se ejecutan en el navegador, accesibles desde cualquier dispositivo. Definidas con HTML, CSS y JS.
Híbridas	Combinan elementos web con funciones nativas. Buscan lo mejor de ambos mundos.

- Las interfaces web han cambiado mucho desde los años 2000, gracias a la disciplina de la usabilidad.
    - Se definieron patrones de diseño efectivos.
    - Apareció Bootstrap (creado por Twitter), que permite reutilizar componentes visuales estandarizados.
- El usuario suele preferir las interfaces web porque le resultan más agradables, aunque sean menos eficientes.

🔹 Frameworks modernos: Vaadin y Copilot

- Vaadin es un framework Java para crear interfaces web modernas sin necesidad de escribir código en HTML/JS.
    - Usa Java puro para programar la interfaz.
    - Tiene un transpilador que convierte el código Java en HTML ejecutable en el navegador.
    - Ideal para crear aplicaciones híbridas que parecen web pero se comportan como apps.
- Copilot (Microsoft) es un asistente de código basado en IA. Existen versiones adaptadas para frameworks como Vaadin:
    - Sugiere código automáticamente.
    - Acelera el desarrollo.
    - Aprende del contexto del proyecto.

Resumen visual de tecnologías mencionadas

- **Apache Log4j** → Logging profesional en Java.
- **OpenTelemetry** → Telemetría y trazabilidad del rendimiento.
- **Big Data** → Gestión masiva de datos en todas sus formas.
- **Data Lake / Lakehouse** → Nuevas arquitecturas de almacenamiento.
- **Frameworks UI** → Bootstrap, Vaadin.
- **Copilot** → Asistente inteligente de desarrollo.
- **Tipos de UI** → Nativas, Web, Híbridas.