Automatización de Cache Busting dinámico en núcleo estático

Cuadernillo | Vol. 1

mercedev.es — 2026-05-07 | Fase 7

Automatización de Cache Busting dinámico en núcleo estático

El Desafío (Síntoma)

Conocido comúnmente como "La tiranía de la caché". Tras refactorizar una rejilla CSS en SASS de 4 a 5 columnas, el entorno de desarrollo local reflejaba los cambios perfectamente. Sin embargo, al desplegar a producción, los navegadores móviles y el proxy inverso (Varnish) seguían mostrando el diseño antiguo. La solución tradicional era incrementar manualmente la variable de versión (`?v=X`) en las etiquetas `<link>` del archivo `public/index.html`. Esta intervención manual constante generaba fricción operativa severa y un alto riesgo de olvido.

La Maniobra (Lógica)

Se refactorizó el orquestador de publicación (`merci-publish.py`) para que asuma la responsabilidad del versionado durante la fase de Construcción (Build):

1. **Lectura de Metadatos:** El script lee la fecha exacta de modificación física (`st_mtime`) de los archivos compilados `main.css` , `main.js` y `MerciController.js` .
2. **Inyección Quirúrgica:** Mediante expresiones regulares (`re.sub`), el motor busca el patrón `?v=...` directamente en el código fuente de `public/index.html` y reemplaza el número antiguo por el *timestamp* (marca de tiempo) actualizado (ej. `?v=1715086000`).

El Aprendizaje / Deuda Técnica

Fricción Cero y Repercusión en Cadena. Tratar la portada (`public/index.html`) verdaderamente como la Única Fuente de Verdad (SSOT - Single Source of Truth) produce una automatización elegante. Dado que `merci-publish.py` inyecta la versión actualizada primero, cuando el script secundario `merci-sync-pages.py` se ejecuta inmediatamente después, recorta ese bloque HTML ya actualizado y lo propaga a páginas estáticas secundarias (como Contacto). Todo el ecosistema queda blindado contra cachés obsoletas sin que el desarrollador haya tenido que teclear un solo número.