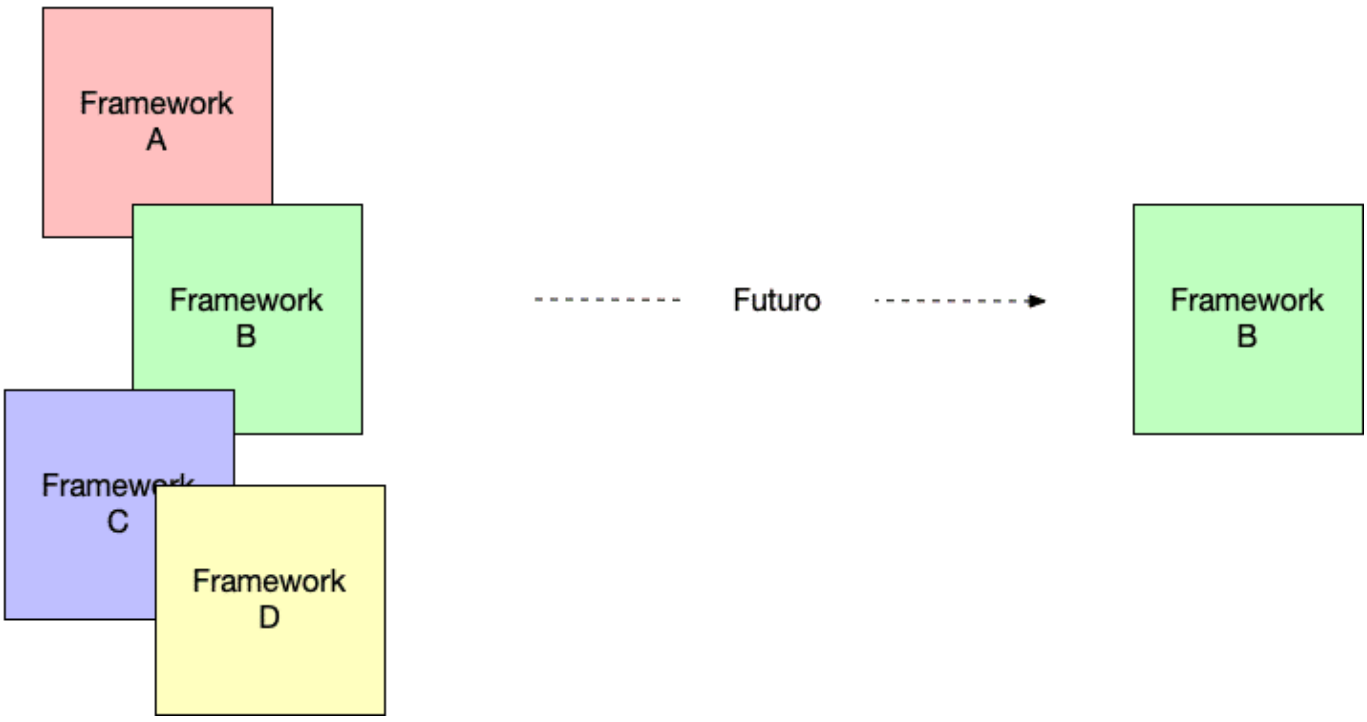
Tabla de Contenidos

- La Fatiga de JavaScript
  - ¿Cómo podemos solventar esto? .
- ¿ Que podemos hacer? .
  - Conclusiones
  - Otros artículos relacionados

La Fatiga de JavaScript hace referencia a como el mundo de JavaScript ha terminado por ser demasiado grande para abordarlo para la mayoría de la gente o para todos . Existen multitud de Frameworks y miles de librerías esto hace que cada día sea mucho más complejo obtener una visión clara de la plataforma . Pero sobre todo el problema más importante es que cada día aparecen más frameworks y más enfoques de realizar las cosas. Esto implica que muchos desarrolladores que en su momento quizás apostaron por un framework A o B se encuentren la circunstancia de que ese framework ha sido descatalogado y ya no existe o no se espera nueva versión de él.



La Fatiga de JavaScript

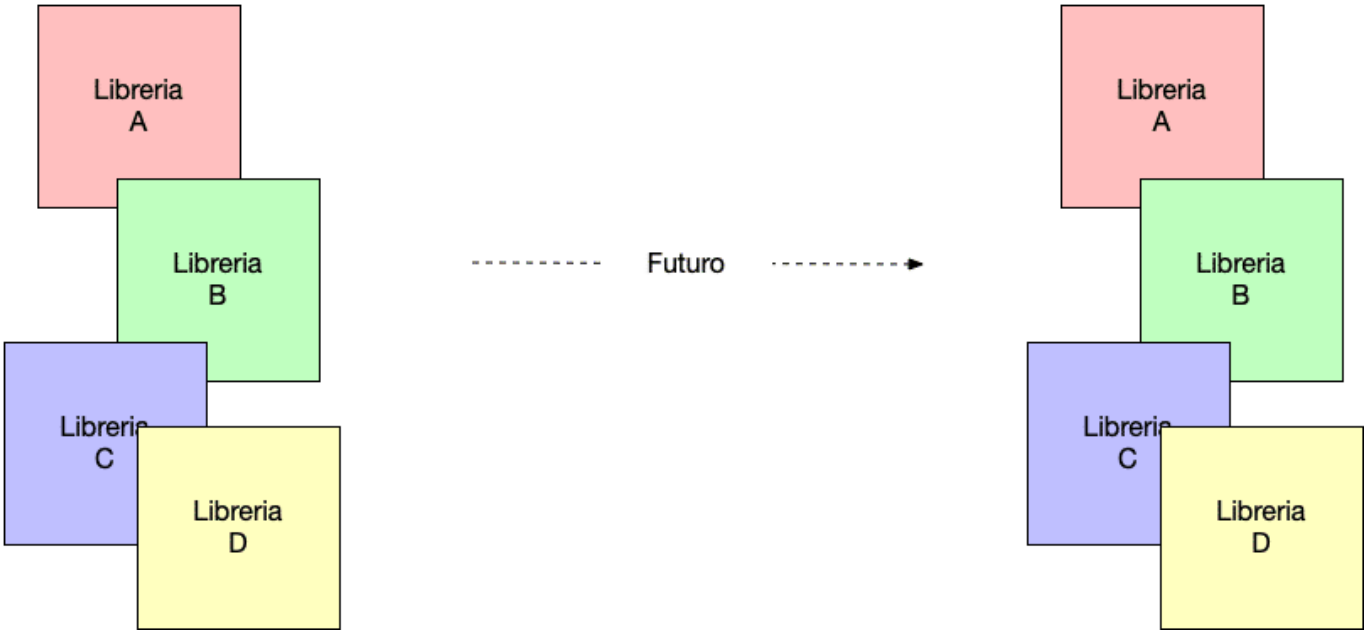
No es posible abordarlo todo, las cosas son así de sencillas . El problema fundamental viene de si cometemos el error de elegir una tecnología que queda obsoleta rápidamente y con ella hemos implementado decenas de aplicaciones . En esta casuística nos encontraremos con el problema de que más pronto que tarde tendremos que volver a construirlas.

¿Cómo podemos solventar esto? .

Bueno ojalá hubiera una solución sencilla o directa , pero desde mi experiencia las cosas no son tan sencillas . La primera pregunta que tenemos que hacernos a nosotros mismos es si usamos un Framework o usamos librerías a la hora de construir la parte cliente . El uso de frameworks siempre tiene la ventaja de aumentar la productividad , pero también tiene la desventaja de acoplarnos mucho a una tecnología muy concreta.

¿ Que podemos hacer? .

Si queremos usar un framework lo mejor es elegir alguno que tenga un gran fabricante detrás y que sepamos que le espera un futuro. Uno de estos ejemplos puede ser Angular el cual sigue teniendo una gran aceptación en el mercado . Sino elegimos un Framework estaremos optando por una Arquitecturas Frameworkless basada en librerias. Las librerías tienen una mayor longevidad que los Frameworks por ejemplo JQuery es de 2006 y sigue estando vigente.



Eso sí probablemente nos exija un mayor conocimiento de Arquitectura y JavaScript a cambio de no depender tanto de un Framework muy concreto que pueda descontinuarse en el futuro.

Algunas de las librerías o tecnologías que podemos usar para diseñar una Arquitectura FrameworkLess son:

1. React y Redux : React como librería de componentes y Redux como librería de gestión de estado
2. Vue y Vuex: Es la alternativa a React y Redux
3. Axios:Para gestionar peticiones Ajax
4. Lodash; para gestión de colecciones de objetos
5. jQuery : Para el manejo de DOM
6. Babel : Para el uso de JavaScript ES6
7. TypeScript :Como lenguaje de programación con mayor proyección en el futuro
8. Jest: para pruebas unitarias

Conclusiones

Javascript está en constante evolución y hay que pararse a pensar las cosas mucho cuando estamos hablando del diseño de la arquitectura de nuestras aplicaciones.

Otros artículos relacionados

1. Introducción a JavaScript ES6
2. Angular ngIf, opciones y componentes
3. ¿Angular o React cuál elegir?