Taller 1 – Análisis de la Incorporación de JavaScript en el Proyecto del Hospital

Javier Lagos

1. Informe de Investigación: Generalidades del Lenguaje JavaScript.

JavaScript es un lenguaje de programación ligero e interpretado que permite implementar funciones complejas en las páginas web y aplicaciones móviles. Este se fue desarrollado a partir de otro lenguaje inicial llamado “Mocha”, por Brendan Eich en 1995, como una forma de hacer que las páginas web del navegador Netscape Navigator fueran más dinámicas; su nombre original era LiveScript.

Este se encarga de manejar la interactividad y la manipulación del DOM. Entre muchas de funciones se encuentra el cargar contenido en una página sin tener que recargarla, animar elementos de la página, modificar el contenido en tiempo real, validar valores de entrada de un formulario, responder a eventos de usuario, entre otros usos.

El entorno virtual de JavaScript más utilizado y conocido es Node.js, este se utiliza para ejecutar aplicaciones web en el lado del servidor ya que permite ejecutar JavaScript de manera asíncrona. En el frontend se utilizan muchas de las herramientas de NodeJS para optimizar las aplicaciones web y mejorar los flujos de trabajo; y para el backend se suele utilizar para crear servicios web o APIs REST.

Sus principales diferencias con otros lenguajes de programación:

- JS y Java: Java crea aplicaciones ejecutables en máquina o explorador y su código requiere de compilación, mientras que JavaScript sólo se ejecuta en un explorador y el código está en todo el texto.
- JS y Python: Python es un lenguaje multipropósito ya que tiene distintas aplicaciones, como desarrollo web o ciencia de datos, eso si en el desarrollo web se suele usar solo para el backend. JavaScript es un lenguaje orientado a eventos e interpretado que se usa principalmente para el desarrollo web, pero este es full-stack, se puede usar tanto para el frontend como para el backend.
- JS y C++: C++ es un lenguaje de nivel medio (más similar al código binario que al lenguaje humano) y este debe compilarse antes de ejecutarse. JavaScript es un lenguaje de alto nivel (más similar al lenguaje humano) y es un interpretado, no compilado.
- JS y Ruby: Ruby se usa principalmente en backend, es más lento que JS, tiene una sintaxis más sencilla; es mejor para desarrollar programas que requieren de mucho procesamiento de la CPU, pero no tan buena herramienta para sitios de gran tamaño.

Algunas de las ventajas de este lenguaje de programación son que es un lenguaje sencillo, es rápido, cuenta con muchas opciones para efectos visuales, es soportado por una gran cantidad de navegadores, compatible con muchos de los dispositivos modernos (iPhone y Android), muy versátil en sus usos y una de sus mejores características es que permite el trabajo FullStack.

Pero de igual manera tiene algunas debilidades como que, aunque es compatible con muchos navegadores, cada uno interpreta su código de manera diferente por lo que es necesario probarlo en cada uno de los que se va a implementar, este podría llegar a tener un rendimiento más lento en comparación con otros lenguajes y a medida que las aplicaciones se vuelven más complejas, su código puede volverse más difícil de entender y mantener.

Como último punto es necesario mencionar que JavaScript puede trabajas de manera asíncrona en caso de necesitarse, esto se debe a que muchas funciones de los navegadores, como todo lo que son las peticiones HTTP pueden tardar un buen tiempo en ejecutarse, es por esto que se requiere que se procesen las interacciones del usuario y las peticiones sin que se bloquee el hilo principal de ejecución. Para poder implementar estas asincronías se usan los siguientes elementos:

- Callbacks: Funciones que se pasan como argumento de otra función
- Promesas: Objetos que representan el resultado de una operación asíncrona.
- Async/Await: Función que admite el uso de procesos asíncronos.

2. Evolución del Lenguaje JavaScript y el Estándar ECMAScript.

Como se mencionó anteriormente, JavaScript es un lenguaje interpretado, esto quiere decir que se va ejecutando línea a línea por el navegador o servidor. En cambio, los lenguajes de programación que son compilados, significa que se compilan a código máquina antes de que se ejecuten.

- ECMAScript

Este lenguaje de programación está basado en el estándar ECMAScript, implementa este estándar que siguen los navegadores y entornos. ECMAScript en sí define cómo se interpreta y funciona este lenguaje, garantizando la interoperabilidad de páginas web en diferentes navegadores. Cuando JS recién iniciaba no tenía un estándar formal por lo que se encontraba en riesgo de fragmentación debido a las diferentes implementaciones que le daba cada navegador, es gracias a esto que surge ECMAScript en 1997.

- ECMAScript 2 (1998): Introdujo mejoras en la sintaxis y la manipulación de cadenas, incorporó nuevas características como el bloque try/catch y las excepciones.
- ECMAScript 3 (1999): Estandarizó JavaScript y su funcionalidad esencial, introdujo conceptos como los objetos literales y la notación de matriz.
- ECMAScript 4 (Abandonada): Nunca se completó esta versión debido a desacuerdos sobre la dirección que debía tomar el lenguaje, lo que conllevo una pausa en el proceso de estandarización.
- ECMAScript 5 (2009): Importantes mejoras, como incluir el soporte de JSON, strict mode, propiedades getters/setters, métodos de manipulación de matrices y cadenas y mejoras en la manipulación de objetos.
- ECMAScript 6 (2015): Marca un punto de inflexión en la evolución del lenguaje, introduce características fundamentales como let, const, clases, funciones flecha, módulos, promesas, entre otros.

- ECMAScript 7 (2016), 8 (2017), 9 (2018), 10 (2019) y más: Siguen introduciendo nuevas funcionalidades como async/await, mejoras en la gestión asíncrona, operadores spread/rest, entre otras mejoras.

- TypeScript

Este es un lenguaje de programación de código abierto que fue desarrollado por Microsoft y está basado en JavaScript. Es un superconjunto de JS, lo que significa que puede usar las mismas características de que este, además de las que viene en TypeScript. Tiene un tipado estático lo que permite especificar tipos de datos de variables, parámetros de funciones y propiedades de objetos durante la fase de desarrollo, lo que facilita el desarrollo de grandes aplicaciones al detectar errores durante esta misma fase. Se pensó para utilizarlo en grandes proyectos permitiendo trabajar de manera estructurada.

Sus principales ventajas tienen que ver con puntos ya mencionados como que es de tipado estático, su escalabilidad, compatibilidad con estándares de JS, herramientas mejoradas, amplia comunidad y ecosistema y su compatibilidad con entornos de desarrollo integrado (IDE).

Sus desventajas en cambio son que requiere de un tiempo adicional para definir tipos, involucra una curva de aprendizaje, tiempo para que los desarrolladores de acostumbren a este tipo de tipado. La necesidad que tiene de compilación antes de que el código pueda ejecutarse en un navegador o entorno. Y el hecho de que la documentación disponible que hay puede llegar a ser mucho menor que la que JS tiene.

3. **Análisis de la Pertinencia de Integrar JavaScript Avanzado o TypeScript en el Proyecto.**

Al momento de comparar las opciones de utilizar JS o TS para el desarrollo del proyecto para el hospital, una de sus ventajas es el hecho que TS ayuda mucho más a prevenir errores antes de poner a prueba el proyecto final y facilita el orden dentro de todo el proyecto. Una desventaja o el hecho de que no sea tan útil utilizar TS tiene que ver con el hecho de que este lenguaje está orientado a proyectos de gran tamaño y por ahora, el proyecto del hospital sigue siendo pequeño, por lo que quizás no sería necesario implementar TS y es mejor seguir con JS.

En el caso particular de este desarrollador web, entre utilizar un lenguaje de programación o el otro, conviene aplicar JavaScript ya que es el que conoce más y ya ha utilizado antes por lo que se maneja muy bien con este, en cambio si se tomara la decisión de utilizar TypeScript se enfrentaría con el hecho de que necesita un tiempo antes de seguir desarrollando el proyecto solo para entender como es que funciona este lenguaje, cómo redactar el código y cómo se implementa en general, ya que nunca lo ha utilizado antes.

A modo de conclusión no se ve que sea necesario el implementar TypeScript por el momento debido a dos de los puntos mencionados anteriormente, el hecho de que este lenguaje está enfocado a proyectos de gran escala y al hecho de que, si se tiene la opción, es mejor escoger el lenguaje de programación que se adapte de mejor manera a los desarrolladores del proyecto y en este caso, ese no es TS.