Estándares de Implementación y Buenas Prácticas

Proyecto: Establecimiento de una plataforma de investigación traslacional biomédica para el almacenamiento, administración, análisis e integración de la información clínica, ómica, fitoquímica, agronómica y farmacológica derivada de las actividades de investigación del programa científico.

|  |  |
| --- | --- |
| **Elaborado por**: Andrés M. Castillo | **Creado:** 02/02/2018 |
| **Archivo:** Guía de estándares y buenas prácticas.pdf | **Última Modificación**: |
| **Versión:** 01 | **Fecha Impresión:** |

**Historia de Cambios**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha (mm/dd/aa)** | **Modificado Por** | **Versión** | **Comentarios Revisión** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

INTRODUCCIÓN

El grupo de Aplicaciones de Resonancia Magnética Nuclear lleva varios años realizando desarrollos informáticos para resolver problemas de las química por medio de herramientas computacionales. Durante su historia, se han utilizado varios lenguajes de programación como Scilab, Matlab, Java, JavaScript, PHP, python y bash.

Con el ánimo de mantener vigente el código fuente que se produce y facilitar su mantenimiento se han adoptado varios estándares durante el tiempo. Este documento tiene como propósito describir el último estándar definido para los desarrollos en JavaScript y HTML5, dado que son los lenguajes en los cuales se produce la mayor parte del código fuente actualmente.

# 1. GENERALIDADES DEL ESTÁNDAR

## 1.2 Esencia de los estándares

Los estándares son lineamientos, directrices y protocolos que se establecen con el propósito de normalizar la escritura del código que conforma un desarrollo de software, procurando la consistencia y reusabilidad del mismo. Emplear un lenguaje claro al definir cada uno de los elementos de software a través de convenciones permite mantener el código de manera eficaz y eficiente, así mismo agiliza el proceso de detección de errores toda vez que los estándares sean claros y se apliquen cabalmente. Al considerar la selección del estándar de desarrollo se deben tener en cuenta aspectos técnicos como son la definición de variables por medio de una nomenclatura adecuada, la documentación requerida, las herramientas de verificación entre otros conceptos que serán objeto de estudio en este documento. Además de la selección del estándar se requiere el compromiso por parte de los desarrolladores de aplicar los lineamientos mencionados y cumplir con todas las especificaciones que sean solicitadas por la Dirección de Planeación y Sistemas. Es necesario mencionar que cada desarrollo de software que se ha llevado a cabo sin aplicar ningún estándar en su construcción, requiere una dedicación especial de recurso humano y de tiempo debido a que se cuenta solamente con los conceptos aplicados arbitrariamente por el desarrollador a cargo; este comportamiento constituye una mala práctica en cuanto a construcción de software debido a que se limita la escalabilidad del sistema y su reusabilidad que son conceptos ampliamente utilizados al considerar aspectos como calidad y desempeño del software. 1.3. Selección del sistema de notación Uno de los aspectos más relevantes en la definición del estándar de programación es la selección del sistema de notación, debido a que este lineamiento rige directamente la escritura del código fuente del aplicativo. A través de sistema de notación se define la nomenclatura y las convenciones para cada uno de los elementos de software como son los tipos de datos, funciones, formularios y demás componentes que conforman un aplicativo. Notaciones como Reddick, Pascal Casing y Camel Casing son las más usadas actualmente al construir soluciones de software; cada una de ellas tiene aspectos particulares pero a través del análisis y estudio realizado a cada una de ellas se logró concluir que el mejor para nosotros es el Camel Casing, ya que es el que hemos usado durante más tiempo y todos nuestros proyectos anteriores usan esta codificación.

Como gran parte de nuestras aplicaciones están pensadas para ser usadas en aplicaciones web, hemos adoptado los estándares de desarrollo del sitio W3Schools (<https://www.w3schools.com/js/js_conventions.asp>) que mantiene en línea un conjunto de directrices no sólo sobre estándares de codificación, sino todo un conjunto de recomendaciones sobre buenas prácticas, errores, desempeño y versionamiento. Cómo estas prácticas cambian con el tiempo, y nuestra intención es seguir el estándar definido por la organización, este documento es un compendio de los enlaces hacía las definiciones de la página web.

# 2. Convenciones de codificación en JavaScript

<https://www.w3schools.com/js/js_conventions.asp>

En esta sección se describen los aspectos más básicos de la codificación para proyectos en JavaScript. En esta sección se encuentra la codificación para el nombramiento de variables, constantes, funciones, identación y espaciado del código fuente.

# 3. Buenas prácticas

<https://www.w3schools.com/js/js_best_practices.asp>

En esta sección se dan varias recomendaciones sobre las declaraciones de variables globales, locales, uso de objetos versus variables nativas. También se dan recomendaciones sobre el efecto de comparar usando == o ===, el uso de los parámetros por defecto y el uso de eval. Muchas de estas recomendaciones están orientadas a hacer un código menos susceptible a errores inesperados en tiempo de ejecución, así como a evitar problemas de seguridad.

# 4. Errores comunes en JavaScript

<https://www.w3schools.com/js/js_mistakes.asp>

JavaScript es un lenguaje de programación débilmente tipado. Aunque esto es visto por muchos programadores como una ventaja, ya permite olvidarse de las conversiones y el casting en la mayor parte de las operaciones, también genera varios tipos de errores lógicos que son muy difíciles de detectar en códigos grandes. Dos de los errores más comunes debido al débil tipado de los datos en JavaScript son los que encontramos al realizar comparaciones y asignaciones. También se describe el uso de los punto y coma “;” al final de cada instrucción.

# 5. Desempeño de JavaScript

<https://www.w3schools.com/js/js_performance.asp>

En esta sección se dan algunas recomendaciones para el mejor desempeño del código que se escribe en JavaScript, como por ejemplo el uso de variables en lugar de constantes en los bucles de repetición “for”. Si bien varias de estas recomendaciones no garantizan que el código funciones más rápido, debido a que muchas veces el código de JavaScript es compilado y optimizado antes de la ejecución, si es una buena práctica que puede tener un impacto importante en otros lenguajes o entornos de ejecución.

# 6. Versiones de JavaScript

<https://www.w3schools.com/js/js_versions.asp>

JavaScript es uno de los lenguajes más utilizados en la actualidad y también uno de los más rápido cambio. Actualmente o menos cada año se libera una nueva versión del estándar, y por tanto es importante mantenerse actualizado. En este link se describen las diferentes versiones del lenguaje a través de los años y que cambios importantes ha tenido desde que era un lenguaje únicamente para el navegador.