



Universidad Cenfotec

Técnico en Desarrollo y Diseño Web

Fundamentos de Programación Web

Cuestionario JavaScript

Estudiante:

Diana Ponce Faerron

Profesores:

Francisco Jiménez Bonilla.

Mayo 2024

Tabla de contenido

Instrucciones	3
Cuestionario Java Script	4
1. Escriba la historia del lenguaje Java Script	4
2. ¿Por qué se debe aprender Java Script?	5
3. ¿Cuál es la relación entre HTML y Java Script?	6
4. ¿En qué beneficia usar Bootstrap para sitios y aplicaciones web en JS?	7
5. ¿Qué semejanza y diferencia tienen los lenguajes web PHP y Java Script?	7
6. Cite 3 formas en que se puede agregar código JS en una página web.	8
7. ¿Cuál es la función principal de la consola en JS?	9
8. ¿Cuál es la diferencia que existe en las declaraciones var, let y const en JS?	10
9. Explique los 2 tipos de comentarios que se pueden aplicar en JS.	11
10. ¿Qué es ECMAScript6? Explique claramente.	12
Conclusión	13
Referencias	14

Instrucciones

Todos los trabajos del curso deben ser presentados en un repositorio personal GITHUB.

El enlace se debe enviar al docente. Respalidar los trabajos es una responsabilidad total del estudiante.

La carpeta del repositorio debe tener el nombre tarea1.

El trabajo se hace de manera individual.

Las respuestas deben estar completas, lo mejor posible respondidas.

La conclusión debe tener 5 renglones.

Guardar el archivo en formato PDF con el nombre tarea1

Contestar las siguientes preguntas planteadas acerca del Lenguaje de Programación Web Java Script (JS).

Cuestionario Java Script

1. Escriba la historia del lenguaje Java Script.

JavaScript fue originalmente creado en 1995 por Brendan Eich quien desarrollo lo que sería la primera versión de JavaScript para el navegador NetScape actualmente ya no existe. En aquel momento se llamó *Mocha* y después fue renombrado a *LiveScript* [1].

En 1997, se crea un comité para estandarizar JavaScript y es a partir de entonces cuando los estándares de JavaScript se rigen por ECMAScript. A lo largo de los años, JavaScript ha evolucionado considerablemente, incorporando nuevas características y funcionalidades. En Julio del 2015 se estandarizo la versión 6 de JavaScript [1].

JavaScript fue diseñado para añadir efectos y animaciones a los sitios web, pero ha ido evolucionando mucho a lo largo de los años, convirtiéndose en un lenguaje multipropósito [1]. Hoy en día, es uno de los lenguajes de programación más populares del mundo, utilizado para una amplia gama de aplicaciones, desde sitios web y aplicaciones web hasta juegos y software del lado del servidor.

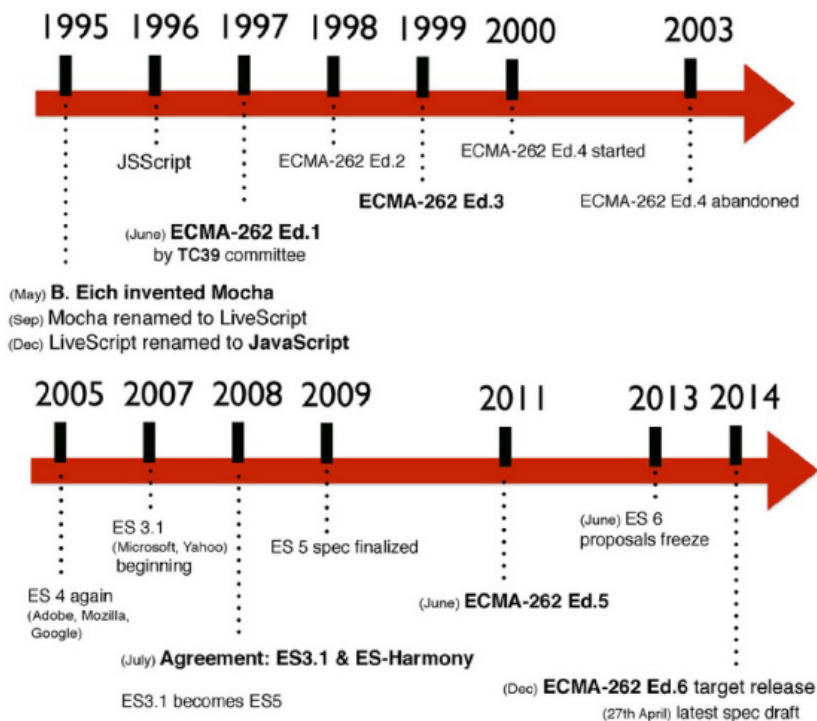


Figura 1. Línea de tiempo JavaScript [1].

2. ¿Por qué se debe aprender Java Script?

En mi opinión, si se busca una carrera en desarrollo de software o bien como ingeniero front-end, es sumamente valioso aprender JavaScript porque es esencial en el desarrollo web moderno. Se puede decir que JavaScript es uno de los lenguajes de programación más populares del mundo, con una alta demanda.

JavaScript es un lenguaje versátil que permite crear interfaces de usuario interactivas y dinámicas. Se utiliza en conjunto con HTML y CSS, y también se puede usar en el servidor con Node.js, lo que permite crear una gama muy amplia de aplicaciones, desde sitios web simples hasta complejos.

A mi parecer, existen muchos recursos disponibles para aprender JavaScript en diferentes sitios web, tutoriales y cursos en línea. También es importante mencionar que hay muchas oportunidades de trabajo en esta área de desarrollo web, por lo que los desarrolladores de JavaScript suelen tener buenos salarios.

3. ¿Cuál es la relación entre HTML y Java Script?

Ambos lenguajes, junto con CSS, tienen una relación complementaria esencial para el desarrollo de sitios web. Los tres lenguajes sirven para propósitos y funciones diferentes dentro del sitio web en su conjunto, por lo que los desarrolladores utilizan diferentes archivos para cada uno, y los tres archivos deben estar en el mismo directorio.

HTML proporciona la base estructural de una página web, definiendo elementos como encabezados, párrafos, imágenes y enlaces a través de un sistema de etiquetas. En esencia, determina qué contenido aparece en la página y cómo está organizado [2]. Por otro lado, JavaScript añade interactividad y comportamiento dinámico a los elementos HTML. También permite la creación de elementos interactivos como formularios, botones y actualizaciones en tiempo real [2]. Finalmente, CSS mejora el estilo de HTML en términos de estética y presentación, dándole una mejor apariencia. Las nuevas funciones de CSS incluso permiten animaciones, disposición en flexbox y un diseño mucho más responsivo [3].

Se puede concluir que los tres lenguajes de programación juntos permiten la creación de aplicaciones web modernas, responsivas y atractivas. Por lo tanto, se puede confirmar la siguiente analogía: JavaScript da vida al HTML, CSS hace que el HTML sea hermoso y HTML le da a JavaScript y CSS su estructura en una página web [3]. Es importante saber que, aunque se pueda escribir todo el código en un archivo HTML, esto eventualmente puede provocar código repetitivo, por lo que la recomendación siempre será crear archivos separados para cada lenguaje de programación.

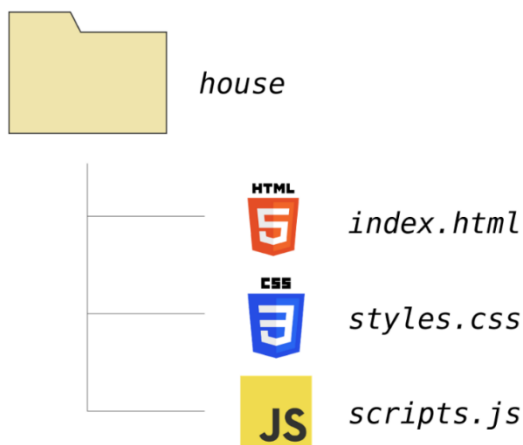


Figura 2. Relación entre los tres lenguajes [4].

4. ¿En qué beneficia usar Bootstrap para sitios y aplicaciones web en JS?

Bootstrap es un framework CSS utilizado en aplicaciones front-end que facilita el diseño de sitios web responsivos y móviles. Proporciona un conjunto de herramientas predefinidas que ayudan a los desarrolladores a crear interfaces coherentes, compatibles con diferentes navegadores y que se adaptan a cualquier dispositivo.

Hay muchos beneficios de usar Bootstrap para sitios y aplicaciones web en JavaScript, ya que nos ahorra tiempo al proporcionar una amplia gama de componentes prediseñados, como botones, formularios, menús y barras de navegación. Esto reduce el esfuerzo necesario para escribir código HTML y CSS. Es un recurso potente y popular para los desarrolladores web front-end, pues les proporciona un framework estructurado y una gran biblioteca de códigos útiles para crear páginas web flexibles y adaptables [\[5\]](#).

5. ¿Qué semejanza y diferencia tienen los lenguajes web PHP y JavaScript?

Ambos son lenguajes de scripting; una de sus principales diferencias radica en que PHP se utiliza principalmente para crear contenido dinámico del lado del servidor. Sin embargo, JavaScript se utiliza principalmente para agregar interactividad a las páginas web del lado del cliente, como manipular el DOM (Document Object Model), responder a eventos del usuario y crear animaciones [\[6\]](#).

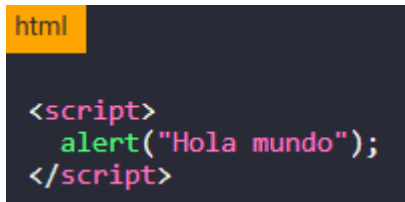
PHP es un lenguaje de programación procedimental, mientras que JavaScript es un lenguaje de programación multiparadigma que admite programación procedimental, orientada a objetos y funcional. Se asemejan en que ambos son lenguajes de programación interpretados y dinámicamente tipados y se pueden utilizar para crear sitios web y aplicaciones web interactivas [\[6\]](#).

Se concluye que JavaScript ofrece más flexibilidad y potencia expresiva que PHP.

6. Cite 3 formas en que se puede agregar código JS en una página web.

Las 3 formas en que se puede agregar código JS en una página web son las siguientes:

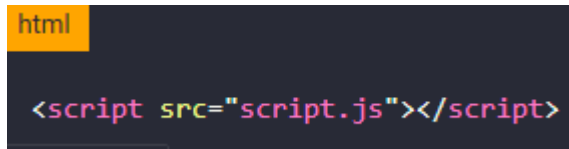
1. **Incrustado en el HTML:** El código JavaScript puede ser escrito directamente en la parte superior de la página web en la zona `<head>` dentro de las etiquetas `<script>` en el HTML de la página lo que permite una integración rápida y sencilla de funcionalidades. Este método es útil para scripts pequeños o específicos de una página [7].



```
html
<script>
  alert("Hola mundo");
</script>
```

Figura 3. Ejemplo de cómo incluir JavaScript en la etiqueta script [7].

2. **Referenciar un archivo externo:** Este método es útil para proyectos más grandes. El código JS puede ser colocado en un archivo (extensión .js) separado y referenciarlo en el código HTML usando la etiqueta `<script>` con el atributo `src`. Lo que permite mantener una mejor organización del código, facilita el mantenimiento y la reutilización del código [7].



```
html
<script src="script.js"></script>
```

Figura 4. Ejemplo de cómo usar JavaScript en HTML desde un archivo externo [7].

3. **Usar atributos de eventos HTML:** JavaScript también puede ser agregado directamente a los atributos de eventos HTML, como `onclick`, `onload`, etc. Estos permiten ejecutar código específico en respuesta a interacciones del usuario. Esta funcionalidad facilita una experiencia interactiva directamente desde las etiquetas HTML, sin alterar el código principal de la página, y activa el código JavaScript cuando se producen eventos específicos, como clics del ratón [7].

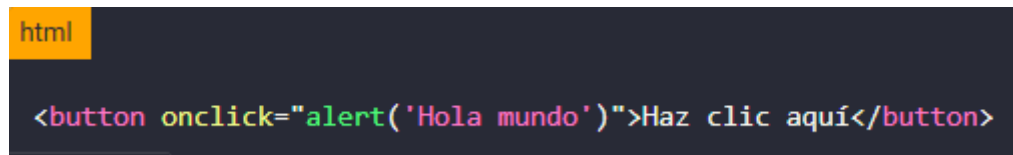
A screenshot of a code editor with a dark background. In the top left corner, there is a small orange tab labeled 'html'. The main area of the editor displays the following HTML code: `<button onclick="alert('Hola mundo')">Haz clic aquí</button>`. The code is color-coded: the opening and closing tags are in pink, the attribute name 'onclick' is in green, the string value 'alert('Hola mundo')' is in green, and the button text 'Haz clic aquí' is in pink.

Figura 4. Ejemplo de cómo incluir JavaScript en un atributo de evento [7].

7. ¿Cuál es la función principal de la consola en JS?

La función principal de la consola en JavaScript es proporcionar una herramienta para probar y depurar código. Además de ejecutar código, la consola proporciona métodos como `console.log()`, `console.error()`, y `console.warn()` para visualizar diferentes tipos de mensajes, errores y variables durante el desarrollo de aplicaciones web [8].

Por lo tanto, se puede concluir que la consola en JavaScript es fundamental para el proceso de desarrollo, ya que permite la ejecución interactiva de código y la visualización inmediata de los resultados, facilitando así la depuración y el testeado de las funcionalidades implementadas [9].

8. ¿Cuál es la diferencia que existe en las declaraciones `var`, `let` y `const` en JS?

Las palabras clave `var`, `let` y `const` se utilizan para declarar variables en JavaScript, pero tienen diferentes características y comportamientos en cuanto a su alcance (scope), elevación (hoisting) y reasignación:

`var` es la manera más antigua de JS. En JavaScript, las variables declaradas con `var` pueden tener un alcance global o de función, dependiendo de si se declaran fuera o dentro de una función, respectivamente. Estas variables están sujetas a hoisting, lo que permite referenciarlas antes de su declaración **efectiva en el código**. Además, **las variables `var` pueden ser reasignadas y re-declaradas** en su alcance, lo que proporciona flexibilidad, pero también puede llevar a errores si no se manejan adecuadamente [\[10\]](#).

`let` las variables declaradas con `let` en JavaScript tienen un alcance de bloque (`{...}`), limitado al bloque donde se declaran. Aunque también están sujetas a hoisting, no se puede acceder a ellas antes de su declaración, evitando errores. Además, no pueden ser re-declaradas, pero sí pueden ser reasignadas dentro de su alcance. Esto ofrece mayor control y seguridad en la gestión de variables en JavaScript [\[10\]](#).

`const` en JavaScript, `const` se utiliza para declarar variables con valores que no cambiarán. Tiene un alcance de bloque y no puede ser reasignado ni re-declarado después de su inicialización. Aunque se eleva, no se puede acceder a su valor antes de su declaración. Esto asegura que una vez asignado, el valor de una variable `const` no pueda ser modificado en ningún otro lugar del contexto donde existe [\[10\]](#).

En resumen, se puede afirmar que `var` permite re-declaraciones y actualizaciones, pero puede causar confusión por su alcance global y hoisting. `let` restringe el alcance a bloques y permite actualizaciones sin re-declaraciones, evitando así conflictos. `const` es el más restrictivo, prohibiendo tanto re-declaraciones como actualizaciones, asegurando que los valores asignados permanezcan constantes. Elegir entre estas opciones depende de la necesidad de mutabilidad y el alcance deseado para las variables.

9. Explique los 2 tipos de comentarios que se pueden aplicar en JS.

En JavaScript, se utilizan dos tipos de comentarios para añadir notas explicativas en el código que no serán ejecutadas durante la ejecución del programa. Estos son:

1. **De una línea:** Se utilizan para comentarios breves y se crean con dos barras inclinadas `//`.

```
<script>// Este un comentario de una única línea
alert("Escribiendo comentarios en javascript!");
//Aquí puedo poner una nota de lo que hace esta línea
// alert("Esto no se ejecuta");
</script>
```

Ejemplo 1. Comentario de una línea [\[11\]](#).

2. **De varias líneas (multilínea):** Se utilizan para comentarios más extensos o para comentar bloques de código. Se crean con una barra inclinada seguida de un asterisco `/*` para comenzar el comentario, y se cierran con un asterisco seguido de una barra inclinada `*/`.

```
<script>
    alert("Escribiendo comentarios multi-línea en javascript");
    /*
    alert("Esto no se ejecuta");
    alert("Esto no se ejecuta");
    alert("Esto no se ejecuta");
    alert("Esto no se ejecuta");
    */
</script>
```

Ejemplo 2. Comentario multilínea [\[11\]](#).

10. ¿Qué es ECMAScript6? Explique claramente.

ECMAScript 6 (ES6), también conocido como ECMAScript 2015, es la sexta edición del estándar ECMAScript. ES6 es una actualización que introduce importantes mejoras y nuevas características al lenguaje JavaScript. Algunas de estas mejoras de ES6 incluyen [\[12\]](#):

- **Sintaxis:** para escribir código más limpio y legible, como las palabras clave `let` y `const` para declarar variables con alcance de bloque.
- **Funciones de flecha:** introdujo las funciones de flecha `(=>)`, una forma más concisa y elegante de escribir funciones anónimas.
- **Clases:** soporte para clases, una forma más estructurada y orientada a objetos, así como nuevas estructuras de datos como `Map` y `Set`.
- **Promesas:** soporte para promesas `(Promise)`, una forma de manejar operaciones asíncronas de manera más eficiente y organizada.

Conclusión

En conclusión, tras investigar y responder las preguntas del cuestionario sobre JavaScript, lo más enriquecedor que aprendí fue la evolución y la importancia fundamental de JavaScript en el desarrollo web moderno. Desde su creación en 1995 y la adopción de ECMAScript en 2015, JavaScript se ha hecho indispensable en la web. La interacción entre JavaScript, HTML y CSS es vital para generar sitios web dinámicos y visualmente atractivos. Además, entender las diferencias entre `var`, `let` y `const`, y cómo comentar adecuadamente el código, son prácticas cruciales para mantener el código organizado y fácil de actualizar. La llegada de ES6 fue un avance importante, introduciendo nuevas formas y funcionalidades que hacen el código JavaScript más eficiente y comprensible. Este aprendizaje me ha enseñado la importancia de JavaScript y su rol esencial para los desarrolladores web.

Referencias

- [1] Azaustre, C. (2016). *Aprendiendo JavaScript: Desde cero hasta ECMAScript 6*. Recuperado de https://books.google.es/books?hl=es&lr=&id=cnjhCwAAQBAJ&oi=fnd&pg=PA4&dq=Historia+de+Javascript&ots=Lefok4xzD8&sig=aVvDaXvp8Bars4FMVA2SOioFX_0#v=onepage&q=Historia%20de%20Javascript&f=false
- [2] Instructive Tech. (2023). *Difference between HTML, CSS, and JavaScript?* Recuperado de <https://techbootcamps.utexas.edu/blog/html-css-javascript/#:~:text=If%20you're%20trying%20to,interactive%20and%20more%20functionally%20complex.>
- [3] Boot.dev. (2023). *HTML vs CSS vs JavaScript Explained*. Recuperado de <https://blog.boot.dev/javascript/html-css-javascript/#:~:text=HTML%20provides%20the%20structure%2C%20CSS,dynamic%20and%20controls%20their%20behavior.>
- [4] FreeCodeCamp. (2023). *The relationship between HTML, CSS, and JavaScript explained by building a city*. Recuperado de <https://www.freecodecamp.org/news/the-relationship-between-html-css-and-javascript-explained-by-building-a-city-a73a69c6343/>
- [5] Sebcreativos. (14 de noviembre de 2024). *Qué es bootstrap y cuáles son sus beneficios en el desarrollo web*. Recuperado de <https://sebcreativos.es/que-es-bootstrap/>
- [6] IONOS. (16 de marzo de 2023). *PHP vs. JavaScript: diferencias y características comunes*. Recuperado de <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/php-vs-javascript/>
- [7] Juneikerc. (2023). *Cómo insertar JavaScript en HTML*. Recuperado de <https://juneikerc.com/es/code-blog/javascript/insertar-en-html/>
- [8] Lenguaje JS. (2023). *La consola Javascript*. Recuperado de <https://lenguajejs.com/javascript/introduccion/consola-de-javascript/>
- [9] Microsoft Edge Developer. (2024). *Ejecución de JavaScript en la consola*. Recuperado de <https://learn.microsoft.com/es-es/microsoft-edge/devtools-guide-chromium/console/console-javascript>
- [10] FreeCodeCamp. (2022). *var, let y const: ¿Cuál es la diferencia?*. Recuperado de <https://www.freecodecamp.org/espanol/news/var-let-y-const-cual-es-la-diferencia/>
- [11] Francisco Jiménez Bonilla. (2024). *Técnicas #1 JS Desarrollo Web - Comentarios*. Recuperado de <https://moodle.ucenfotec.ac.cr/mod/resource/view.php?id=344211>
- [12] Arsys. (2017). *ECMAScript 6: ¿Qué es?* Recuperado de <https://www.arsys.es/blog/ecmascript-javascript>