



Universidad CENFOTEC

Fundamentos de Programación Web

Tarea #1

Investigación sobre JavaScript

III Cuatrimestre 2024

Autor:

Stuart Piñeiro Conejo

Setiembre 2024



1. Historia del Lenguaje JavaScript

JavaScript fue creado en 1995 por Brendan Eich mientras trabajaba en Netscape Communications. Originalmente llamado Mocha, luego renombrado a LiveScript, y finalmente a JavaScript. La primera versión fue lanzada en diciembre de 1995. JavaScript se diseñó para ser un lenguaje ligero y de scripting para ejecutar en el navegador, permitiendo interactividad en las páginas web. En 1996, Netscape sometió JavaScript a ECMA International para estandarizarlo, lo que resultó en la creación de ECMAScript. Desde entonces, JavaScript ha evolucionado con nuevas versiones y características, convirtiéndose en un lenguaje esencial para el desarrollo web.

2. ¿Por qué se debe aprender JavaScript?

Aprender JavaScript es crucial por varias razones:

- **Interactividad en el Navegador:** JavaScript permite crear páginas web interactivas y dinámicas que responden a las acciones del usuario.
- **Popularidad y Demanda:** Es uno de los lenguajes de programación más populares y demandados en el mercado laboral, utilizado en el desarrollo frontend y backend.
- **Versatilidad:** JavaScript se utiliza en una variedad de entornos, incluidos navegadores, servidores (con Node.js), y aplicaciones móviles.
- **Frameworks y Librerías:** Existen numerosos frameworks y librerías (como React, Angular, y Vue.js) que facilitan el desarrollo y mejora la productividad.

3. Relación entre HTML y JavaScript

HTML (HyperText Markup Language) es el lenguaje de marcado que estructura el contenido de las páginas web. JavaScript, por otro lado, es un lenguaje de programación que permite agregar comportamiento dinámico a las páginas. La relación entre ambos es fundamental: HTML proporciona la estructura y el contenido, mientras que JavaScript añade interactividad y dinámica a esa estructura. JavaScript puede manipular el DOM (Document Object Model) que es la representación estructural de un documento HTML.

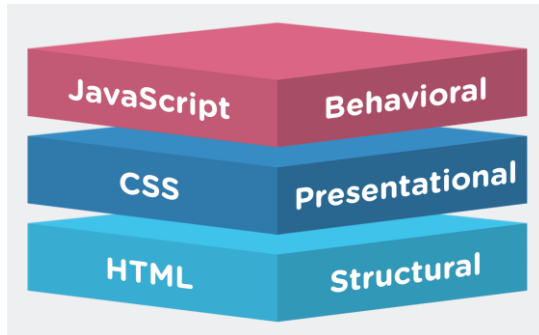
A continuación se puede observar un ejemplo de manipulación al DOM.

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8">
5   <title>Manipulación del DOM</title>
6 </head>
7 <body>
8   <h1 id="titulo">Este es un título en HTML</h1>
9   <button id="cambiarTitulo">Cambiar Título</button>
10
11   <script>
12     document.getElementById("cambiarTitulo").onclick = function() {
13       document.getElementById("titulo").innerText = "¡El título ha cambiado con JavaScript!";
14     };
15   </script>
16 </body>
17 </html>
```

4. Beneficios de Usar Bootstrap para Sitios y Aplicaciones Web en JS

Bootstrap es un framework de diseño web que facilita la creación de sitios web responsivos y estéticamente agradables. Beneficios de usar Bootstrap con JavaScript incluyen:

- **Componentes Predefinidos:** Ofrece una amplia gama de componentes listos para usar, como botones, formularios y navegación, que se integran fácilmente con JavaScript.
- **Consistencia en el Diseño:** Proporciona un sistema de rejilla y estilos prediseñados, asegurando una apariencia consistente en diferentes dispositivos.
- **Facilidad de Personalización:** Los componentes se pueden personalizar mediante CSS y JavaScript para adaptarse a las necesidades del proyecto.



5. Semejanza y Diferencia entre PHP y JavaScript

Semejanza:

- Ambos son lenguajes de scripting utilizados para el desarrollo web y pueden interactuar con bases de datos.

Diferencia:

- **PHP:** Se ejecuta en el servidor y genera HTML dinámico para ser enviado al navegador.
- **JavaScript:** Se ejecuta en el cliente (navegador) y se usa para crear interactividad y modificar el contenido HTML en tiempo real.

6. Formas de Agregar Código JS en una Página Web

1. Dentro de etiquetas <Script> en HTML

```
<script>  
  alert('Hello, World!');  
</script>
```

2. Enlazando un Archivo Externo:

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

3. En el Atributo on... de un Elemento HTML:

```
<button onclick="alert('Hello, World!')">Click me</button>
```

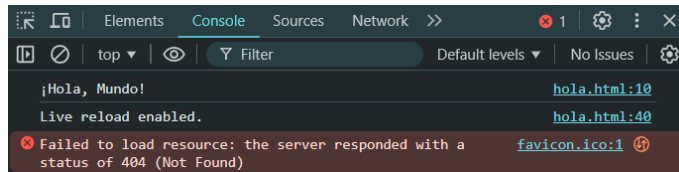
7. Función Principal de la Consola en JS

La consola en JavaScript, accesible a través del navegador, se utiliza principalmente para depuración y prueba de código. Permite a los desarrolladores:

- **Registrar Mensajes:** Usar `console.log()` para imprimir mensajes y variables para ayudar a rastrear el flujo del programa.
- **Ejecutar Código Dinámico:** Probar fragmentos de código en tiempo real.
- **Detectar Errores:** Ver errores y advertencias generados por el código.

Así se mira un código de `console.log` observado en la consola del navegador:

```
1 <!DOCTYPE html>
2 <html lang="es">
3 <head>
4   <meta charset="UTF-8">
5   <title>Uso de la Consola</title>
6 </head>
7 <body>
8   <script>
9     let saludo = "¡Hola, Mundo!";
10    console.log(saludo); // Ver el mensaje en la consola
11  </script>
12 </body>
13 </html>
```



8. Diferencia entre var, let, y const en JS

- **var:** Declaración de variables con ámbito de función. Permite redeclarar y modificar su valor.
- **let:** Introducido en ES6, tiene un ámbito de bloque y no permite redeclaraciones dentro del mismo bloque.
- **const:** También introducido en ES6, define variables cuyo valor no puede ser reasignado. Tiene un ámbito de bloque.

```
// var: alcance de función
function ejemploVar() {
  var mensaje = "Hola desde var";
  if (true) {
    var mensaje = "Cambiado dentro del bloque";
    console.log(mensaje); // "Cambiado dentro del bloque"
  }
  console.log(mensaje); // "Cambiado dentro del bloque"
}
ejemploVar();

// let: alcance de bloque
function ejemploLet() {
  let mensaje = "Hola desde let";
  if (true) {
    let mensaje = "Cambiado dentro del bloque";
    console.log(mensaje); // "Cambiado dentro del bloque"
  }
  console.log(mensaje); // "Hola desde let"
}
ejemploLet();

// const: valor no reasignable
const PI = 3.1416;
console.log(PI); // 3.1416
```

9. Tipos de Comentarios en JS

1. Comentarios de una sola línea:
`// Esto es un comentario de una línea`
2. Comentarios de varias líneas:
`/* Esto es un comentario
de varias líneas */`

10. ECMAScript6

ECMAScript6 (ES6), también conocido como ECMAScript 2015, es una versión importante del estándar ECMAScript que introdujo muchas características nuevas para JavaScript. Algunas de las principales mejoras incluyen:

- **let y const:** Nuevas formas de declarar variables con ámbito de bloque.
- **Funciones de flecha (=>):** Una forma más concisa de escribir funciones.
- **Clases:** Una sintaxis más clara para la programación orientada a objetos.
- **Módulos:** Permite importar y exportar funciones y variables entre archivos.
- **Promesas:** Facilita la gestión de operaciones asíncronas.

Conclusión del Aprendizaje Obtenido

Creo que siempre es importante aprender de la historia del tema que se estudia. En este caso de JavaScript, se evidencia la evolución de la herramienta desde lo simple hasta el lenguaje más amplio con múltiples aplicaciones que es hoy en día.

Además, me emociona profundizar más en HTML y JavaScript en general y también el poder manejar otros lenguajes como PHP que son muy solicitados en el mercado actual. También y muy importante de igual manera me parece el uso de Bootstrap para la creación de páginas web de tipo “responsive”.

Por último, pero no por ello menos importante, está la parte más técnica de la programación, que abarca elementos fundamentales como las palabras clave `var`, `let` y `const`. Estas son esenciales para la declaración de variables. Junto con las nuevas características introducidas en ECMAScript6, como las funciones flecha, los módulos y las promesas, se ha enriquecido significativamente la funcionalidad del lenguaje. Estos avances no solo permiten escribir código más limpio y eficiente, sino que también facilitan el desarrollo de aplicaciones modernas, lo que revela la importancia de dominar estos términos y mantenerse actualizado para cualquier persona que quiera ser desarrollador web.