Tarea #1

Gabriel Suarez Castañedas
UCENFOTEC
Curso : Fundamentos de Programación

Nombre del profesor: Francisco Jose Jimenez Bonilla

Martes, 28 de Mayo 2024

1. ¿Escriba la historia del lenguaje Java Script?

Brendan Eich desarrollo JavaScript para Netscape, un navegador que actualmente no existe. En 1995 Netscape se dio cuenta que, cuando estaba en un archivo Html, no podía interactuar con el usuario. Por esa razón Brandon crea JS, para poder tener un lenguaje de programación con más posibilidades y que facilitaran las interacciones: usuario y aplicación. La idea era desarrollar un lenguaje de scripting ligero y dinámico que pudiera ser utilizado por los diseñadores de páginas web para agregar interactividad a sus sitios.

Los Primeros Días

En solo 10 días, Eich desarrolló una primera versión del lenguaje, inicialmente llamada Mocha, luego renombrada a LiveScript y finalmente a JavaScript, en parte para aprovechar la popularidad de Java en ese momento. A pesar de las diferencias fundamentales entre Java y JavaScript, el nombre ayudó a captar la atención de los desarrolladores y del público.

Estándar ECMAScript

A medida que JavaScript se popularizaba, surgió la necesidad de estandarizar el lenguaje para asegurar la compatibilidad entre diferentes navegadores. En 1997, la primera versión del estándar ECMAScript (ECMA-262) fue publicada por Ecma International, una organización de estándares. Desde entonces, ECMAScript se ha convertido en la base sobre la cual se desarrollan las versiones sucesivas de JavaScript.

Guerra de los Navegadores

Durante finales de los 90 y principios de los 2000, Netscape y Microsoft se enzarzaron en la "guerra de los navegadores", cada uno desarrollando su propia versión de JavaScript (Netscape con JavaScript y Microsoft con JScript). Esta competencia llevó a una fragmentación del lenguaje, haciendo que los desarrolladores tuvieran que escribir

código específico para cada navegador.

Ajax y la Web 2.0

A principios de los 2000, la tecnología Ajax (Asynchronous JavaScript and XML) revolucionó la forma en que las aplicaciones web interactuaban con los servidores. Con Ajax, las páginas web podían actualizarse de manera asíncrona sin tener que recargarse completamente, lo que mejoró significativamente la experiencia del usuario y allanó el camino para la Web 2.0, con aplicaciones más dinámicas y ricas como Google Maps y Gmail.

Node.js y la Popularidad de JavaScript

En 2009, Ryan Dahl lanzó Node.js, un entorno de ejecución para JavaScript basado en el motor V8 de Google Chrome. Node.js permitió a los desarrolladores usar JavaScript del lado del servidor, ampliando enormemente las capacidades y el alcance del lenguaje. Esto, junto con el auge de los frameworks y bibliotecas como AngularJS, React, y Vue.js, catapultó a JavaScript a la cima de la programación web moderna.

ECMAScript 6 (ES6) y Más Allá

En 2015, se lanzó ECMAScript 6 (ES6), también conocido como ECMAScript 2015. Esta versión introdujo una serie de mejoras y características, como clases, módulos, promesas, y let/const, que modernizaron y simplificaron la escritura de JavaScript. Desde entonces, nuevas versiones del estándar se han lanzado anualmente, añadiendo más funcionalidades y manteniendo a JavaScript actualizado con las necesidades de los desarrolladores.

JavaScript Hoy

Hoy en día, JavaScript es uno de los lenguajes de programación más populares del mundo. Su versatilidad y capacidad para funcionar tanto en el lado del cliente como en el servidor lo han convertido en una herramienta indispensable para el desarrollo web moderno. La comunidad de JavaScript es vibrante y en constante evolución, impulsando el lenguaje hacia nuevas fronteras y asegurando que permanezca en el corazón de la innovación tecnológica.

2. ¿Por qué se debe aprender Java Script?

Una razón importante para aprender JavaScript es su versatilidad. JavaScript es el único lenguaje de programación que permite desarrollar tanto la parte del cliente (frontend) como la parte del servidor (backend) de una aplicación web. Con JavaScript, puedes crear interfaces de usuario interactivas, gestionar la lógica del servidor, y manejar bases de datos, todo usando un solo lenguaje. Esta capacidad para abarcar todo el stack de desarrollo web, conocido como desarrollo full-stack.

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

Cuando hablamos de la vinculación entre HTML y JavaScript, nos referimos a la forma en que estos dos lenguajes trabajan juntos para crear experiencias web más ricas y funcionales. A continuación, veremos algunos puntos clave sobre esta relación:

1. **Integración:** JavaScript se integra en las páginas HTML a través de etiquetas <*script*>. Estas etiquetas se utilizan para incluir código JavaScript directamente en el cuerpo o la cabecera del documento HTML.
2. **Manipulación del DOM:** JavaScript permite manipular el Document Object Model (DOM) generado por HTML. Esto significa que se pueden agregar, eliminar o modificar elementos HTML y sus estilos dinámicamente mediante JavaScript.
3. **Eventos:** HTML proporciona la capacidad de definir eventos como clics de ratón, pulsaciones de teclas, entre otros. JavaScript se utiliza para manejar estos eventos y realizar acciones específicas en respuesta a ellos.

La combinación de HTML y JavaScript abre un amplio abanico de posibilidades en el desarrollo web. Desde formularios interactivos hasta animaciones dinámicas, la relación entre estos dos lenguajes es clave para crear sitios web modernos y atractivos para los usuarios.

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

1. Diseño Responsivo Rápido: Bootstrap proporciona una colección de componentes y utilidades preestilizadas que facilitan la creación de sitios web y aplicaciones responsivas, ahorrando tiempo en la escritura de CSS desde cero.

2. Componentes Interactivos Listos para Usar: Bootstrap incluye una variedad de componentes JavaScript listos para usar, como modales, carruseles y menús desplegables, que mejoran la funcionalidad y la experiencia del usuario sin necesidad de desarrollar estos elementos desde cero.

3. Consistencia y Compatibilidad: Usar Bootstrap garantiza un diseño consistente y una experiencia de usuario uniforme en todos los navegadores modernos, simplificando la compatibilidad y reduciendo problemas de diseño cross-browser.

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

- PHP es un lenguaje de programación del lado del servidor, mientras que Javascript es un lenguaje de programación del lado del cliente.

- PHP no se ejecuta dentro del navegador, mientras que Javascript se ejecuta dentro del navegador.

- PHP es ideal para la generación de contenido dinámico del lado del servidor,

como gestionar formularios, interactuar con bases de datos y mantener sesiones de usuario.

- JavaScript es fundamental para añadir interactividad a las páginas web, manipular el DOM, y hacer peticiones asincrónicas al servidor (AJAX). Con Node.js, también se utiliza para construir servidores web y APIs.

En resumen, PHP y JavaScript son lenguajes esenciales en el desarrollo web, cada uno con su rol específico y ventajas en el lado del servidor y del cliente, respectivamente. Con el avance de tecnologías como Node.js, JavaScript ha ampliado su ámbito, haciendo que la línea entre sus usos tradicionales se difumine un poco más.

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

1)Código JavaScript en línea (Inline JavaScript):

- Este método implica agregar JavaScript directamente dentro de los elementos HTML usando atributos de evento, como onclick, onmouseover, etc. Es útil para pequeñas cantidades de código.

2)Código JavaScript Interno:

- Este método utiliza la etiqueta <script> dentro del archivo HTML para incluir código JavaScript. Es adecuado para scripts específicos de la página.

3)Código JavaScript Externo:

- Este método implica escribir el JavaScript en un archivo separado con la extensión .js y luego enlazarlo en el archivo HTML usando la etiqueta <script> con el atributo src. Este enfoque es ideal para mantener el código organizado y reutilizable.

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

La consola de JavaScript es una herramienta esencial para desarrolladores web, ya que facilita la depuración y prueba del código. Su función principal es proporcionar una interfaz para registrar mensajes y valores de variables mediante console.log(), permitiendo el seguimiento del flujo del código. Además, permite registrar errores y advertencias con console.error() y console.warn(), ayudando a identificar problemas. También podemos interactuar dinámicamente con el código directamente en la consola del navegador, probando soluciones sin modificar el archivo fuente. La consola permite inspeccionar objetos y estructuras de datos complejas usando console.dir() y console.table(), y medir el rendimiento del código con console.time() y console.timeEnd(). Estas funcionalidades hacen de la consola una herramienta

imprescindible para el desarrollo y depuración eficientes de aplicaciones web.

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

var

- Ámbito de función: Las variables declaradas con var tienen un ámbito de función. Si se declara una variable con var dentro de una función, solo es accesible dentro de esa función.

- No tiene ámbito de bloque: Las variables var no respetan el ámbito de bloque (un bloque es cualquier código dentro de llaves {}). Esto significa que las variables var son accesibles fuera del bloque en el que se declararon si están dentro de la misma función.

- Hoisting: Las declaraciones var son "hoisted" (elevadas) al principio de su ámbito, lo que significa que se pueden utilizar antes de su declaración en el código. Sin embargo, su inicialización no se eleva, por lo que se inicializan como undefined.

let

- Ámbito de bloque: Las variables declaradas con let tienen un ámbito de bloque. Esto significa que solo son accesibles dentro del bloque en el que se declararon.

- No hay hoisting (en el mismo sentido que var): Aunque las declaraciones let son "hoisted", no se pueden usar antes de ser declaradas en el código debido a la "zona temporal muerta" (TDZ), lo que evita errores de referencia accidental.

- No permite redeclaración en el mismo ámbito: No se puede declarar una variable let con el mismo nombre en el mismo ámbito.

const

- Ámbito de bloque: Igual que let, las variables const tienen un ámbito de bloque.

- Debe ser inicializada al declararse: A diferencia de var y let, las variables const deben ser inicializadas con un valor cuando se declaran.

- Valor constante: Las variables const no pueden ser reasignadas después de su inicialización. Sin embargo, si la variable const contiene un objeto o un array, sus propiedades o elementos sí pueden ser modificados.

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

Comentarios de una sola línea:

- Se utilizan para añadir comentarios breves que ocupan solo una línea.
- Se indican utilizando dos barras diagonales (//).

Comentarios de múltiples líneas:

- Se utilizan para comentarios que abarcan varias líneas.
- Se indican con una combinación de barra diagonal y asterisco (/*) al inicio del comentario y (*/) al final del comentario.

10. ¿Qué es ECMAScript6? Explique claramente.

ECMAScript 6, también conocido como ES6 o ECMAScript 2015, es una versión importante del estándar ECMAScript que define el lenguaje JavaScript. Publicada en junio de 2015, ES6 introdujo una serie de mejoras y nuevas características que modernizaron el lenguaje y facilitaron el desarrollo de aplicaciones web más complejas y eficientes. Entre las principales novedades de ES6 se incluyen:

1 Sintaxis y Funcionalidades Modernas:

   - let y const: Nuevas formas de declarar variables con ámbito de bloque, mejorando el manejo del alcance de las variables.
   - Funciones Flecha: Sintaxis más corta para funciones anónimas y manejo automático del contexto this.
   - Plantillas de Cadenas: Uso de comillas invertidas (`) para crear cadenas multilínea y permitir interpolación de expresiones.

2 Mejoras en la Programación Orientada a Objetos:

   - Clases: Sintaxis simplificada para la creación y herencia de clases.
   - Métodos y Propiedades: Definición más clara y concisa de métodos dentro de las clases.

3 Estructuras de Datos Avanzadas:

   - Mapas y Sets: Nuevos tipos de colecciones para almacenar pares clave-

valor (Map) y valores únicos (Set).

- Destructuración: Asignación más sencilla de valores de arrays u objetos a variables individuales.

4 Funciones y Módulos:

- Funciones Predeterminadas y Rest/Spread: Parámetros predeterminados, rest (...args) para agrupar argumentos, y spread (...array) para expandir arrays.
- Módulos: Importación y exportación de código entre archivos utilizando las palabras clave import y export.

5 Promesas:

- Manejo más sencillo y legible de operaciones asíncronas, como llamadas a APIs, mediante la creación de Promises.

ES6 ha sido fundamental en la evolución de JavaScript, proporcionando herramientas más poderosas y expresivas a los desarrolladores, y estableciendo una base para futuras actualizaciones del lenguaje.

Imagenes:

Conclusion:

En resumen, JavaScript es un lenguaje esencial en el desarrollo web, con una historia

rica desde su creación en 1995 hasta su modernización con ECMAScript 6, que introdujo características avanzadas como let, const, y clases. Su versatilidad permite su uso tanto en el cliente como en el servidor, ampliada por entornos como Node.js. Además, la integración de JavaScript en páginas web puede hacerse de manera en línea, interna o externa, facilitando diversas necesidades de desarrollo. Comparado con PHP, JavaScript ofrece un alcance y contexto de uso más amplio, adaptándose a múltiples paradigmas de programación. La consola de JavaScript es una herramienta vital para la depuración, permitiendo el registro de mensajes, la identificación de errores, la interacción dinámica con el código, la inspección de objetos y la medición de rendimiento. Estos conocimientos consolidan la comprensión de JavaScript como una tecnología fundamental y multifacética en el desarrollo web moderno.