FUNDAMENTOS DE PROGRAMACIÓN WEB

Katherine Maria Moreira Solano

TAREA #1 CUESTIONARIO JAVA SCRIPT

(10 puntos)

INSTRUCCIONES

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

Desarrollo

1.¿Escriba la historia del lenguaje Java Script?

JavaScript, creado por Brendan Eich en 1995 para Netscape, originalmente se llamaba "LiveScript" pero cambió su nombre a "JavaScript" para capitalizar la popularidad de Java. Rápidamente se convirtió en un componente esencial de la web, permitiendo la interactividad en el navegador. Fue estandarizado como ECMAScript en 1997. A lo largo de los años, JavaScript ha evolucionado para adaptarse a las demandas de la web moderna, introduciendo nuevas funcionalidades y siendo impulsado por la competencia con tecnologías rivales como JScript de Microsoft. Hoy en día, JavaScript es uno de los lenguajes de programación más utilizados y sigue siendo fundamental para el desarrollo web, móvil y de aplicaciones.

2.¿Por qué se debe aprender Java Script?

JavaScript es el lenguaje principal para crear interactividad en páginas web. Con JavaScript, puedo hacer que los sitios web sean más dinámicos y atractivos para los usuarios al agregar funcionalidades como animaciones, validación de formularios, interacciones en tiempo real y mucho más**.**

Además comparte muchas similitudes con otros lenguajes de programación. Aprender JavaScript me facilita el aprendizaje de otros

lenguajes en el futuro, ya que muchos conceptos fundamentales de programación son universales.

En resumen, aprender JavaScript es una habilidad valiosa tanto para el desarrollo web como para el desarrollo de aplicaciones en general, y puede abrirme muchas puertas en el mundo de la tecnología.

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

La relación entre HTML y JavaScript radica en que JavaScript se utiliza dentro de las páginas HTML para manipular y controlar los elementos HTML y para agregar funcionalidades interactivas. Puedo incrustar código JavaScript directamente en el HTML utilizando etiquetas de script o cargar archivos externos de JavaScript en una página HTML utilizando la etiqueta <script>.

En resumen, HTML proporciona la estructura y el contenido de una página web, mientras que JavaScript proporciona la interactividad y el dinamismo necesarios para crear experiencias web ricas y atractivas. Juntos, HTML y JavaScript son fundamentales en el desarrollo web moderno.

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

Usar Bootstrap en sitios y aplicaciones web en JavaScript ofrece beneficios como diseño responsivo automático, desarrollo rápido con componentes pre estilizados, consistencia visual, compatibilidad con navegadores, amplia documentación y personalización fácil. En resumen, Bootstrap acelera el desarrollo, mejora la apariencia y simplifica la creación de interfaces de usuario atractivas y adaptables**.**

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

PHP y JavaScript ambos son dos lenguajes interpretados ampliamente utilizados en el desarrollo web, lo que significa que su código se ejecuta línea por línea por un intérprete en lugar de compilarlo a un lenguaje de máquina.

Sin embargo, difieren en su uso principal: PHP es principalmente un lenguaje del lado del servidor, mientras que JavaScript se usa tanto en el lado del cliente como en el servidor. Además, tienen diferencias en su sintaxis y paradigma de programación. A pesar de estas diferencias, ambos son fundamentales en el desarrollo web moderno**.**

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

Una de las formas es Incrustado en el HTML: Podemos agregar código JavaScript directamente en la sección `<script>` dentro del archivo HTML. Por ejemplo:

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Mi Página Web</title>
</head>
<body>
    <h1>Mi Página Web</h1>
    <script>
        // Tu código JavaScript aquí
        alert("¡Hola, mundo!");
    </script>
</body>
</html>
```

Enlazado externamente: Podemos crear un archivo separado con extensión .js y luego enlazarlo a un archivo HTML usando la etiqueta <script> con el atributo src. Por ejemplo:

```html
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Mi Página Web</title>
    <script src="ruta/a/tu/archivo.js"></script>
</head>
<body>
    <h1>Mi Página Web</h1>
</body>
</html>
```

Cargado dinámicamente: Podemos cargar el JavaScript de forma dinámica utilizando métodos como `document.createElement()` y `appendChild()`.

Este enfoque es útil cuando necesitamos cargar JavaScript basado en ciertas condiciones o eventos. Por ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Mi Página Web</title>
</head>
<body>
    <h1>Mi Página Web</h1>
    <button onclick="cargarScript()">Cargar Script</button>

    <script>
        function cargarScript() {
            var script = document.createElement('script');
            script.src = 'ruta/a/tu/archivo.js';
            document.body.appendChild(script);
        }
    </script>
</body>
</html>
```

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

Su función principal es ayudar a depurar código, registrar mensajes de información, advertencias o errores, probar rápidamente fragmentos de código y ejecutar comandos JavaScript directamente en el navegador. Es esencial para comprender el flujo de ejecución del código, detectar

errores y mejorar la eficiencia del desarrollo.Básicamente, la Consola le brinda la capacidad de escribir, administrar y monitorear JavaScript a pedido.

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

Las diferencias entre **var**, **let** y **const** en JavaScript son:

var: Alcance de función o global, permite redeclaración y hoisting.

let: Alcance de bloque, no permite redeclaración pero sí reasignación.

const: Alcance de bloque, no permite reasignación ni redeclaración después de la inicialización.

En resumen, **var** tiene un alcance de función, **let** y **const** tienen alcance de bloque, **var** puede ser redeclarado y reasignado, mientras que **let** puede ser reasignado pero no redeclarado, y **const** no puede ser ni reasignado ni redeclarado después de su inicialización.

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

Comentarios de una línea: Estos comentarios son útiles para agregar explicaciones breves o notas dentro del código. Se inician con // y todo lo que sigue después de estos caracteres en la misma línea se considera un comentario y es ignorado por el intérprete de JavaScript. Por ejemplo:

```
// Este es un comentario de una línea
var x = 5; // Asigna el valor 5 a la variable x
```

Comentarios de múltiples líneas: Estos comentarios permiten escribir comentarios más extensos de varias líneas en el código. Se inician con /* y se cierran con */. Todo lo que se encuentra entre estos delimitadores se considera un comentario. Por ejemplo:

```
/* Este es un comentario
que abarca varias líneas.
Se utiliza para explicar secciones largas
o bloques de código. */
var y = 10; // Asigna el valor 10 a la variable y
```

10. ¿Qué es ECMAScript6? Explique claramente.

ECMAScript 6, también conocido como ES6 o ES2015, es una versión importante y ampliamente adoptada del estándar ECMAScript, que es la especificación subyacente de JavaScript. ES6 introduce numerosas características nuevas y mejoras significativas al lenguaje JavaScript, proporcionando a los desarrolladores herramientas más poderosas y expresivas para escribir código más limpio, modular y eficiente.

Algunas de las características clave de ECMAScript 6 incluyen:

1.Declaración de variables mejoradas: Introduce let y const para reemplazar el uso de var, ofreciendo un manejo más predecible del alcance de las variables.

2.Arrow functions (funciones de flecha): Proporciona una sintaxis más concisa y clara para definir funciones, especialmente útil para funciones anónimas y funciones de devolución de llamada.

3.Plantillas de cadenas (template strings): Permite la interpolación de variables dentro de cadenas de texto utilizando el símbolo de backtick () y ${}`. Esto hace que la construcción de cadenas complejas sea más legible y mantenible.

4.Parámetros con valores por defecto: Permite definir valores predeterminados para los parámetros de una función, simplificando la escritura de funciones con argumentos opcionales.

5.Desestructuración (destructuring): Facilita la extracción de datos de arrays y objetos en variables individuales, lo que simplifica la manipulación de estructuras de datos complejas.

6.Spread operator y rest parameters: Introduce el operador de propagación (...) para expandir arrays y objetos en lugares donde se esperan múltiples argumentos o elementos. También permite la definición de parámetros de función indefinidos mediante el uso de los "rest parameters".

7.Clases: Introduce una sintaxis más clara y orientada a objetos para definir clases y crear objetos en JavaScript, facilitando la creación y herencia de objetos.

8.Módulos (modules): Proporciona un sistema de módulos nativo en JavaScript para organizar y reutilizar código de manera más efectiva, ofreciendo un mejor encapsulamiento y separación de preocupaciones.

Estas son solo algunas de las características destacadas de ECMAScript 6. En general, ES6 ha mejorado significativamente la capacidad de JavaScript para manejar tareas más complejas y ofrece a los desarrolladores una sintaxis más moderna y poderosa para escribir código más legible y mantenible.

Conclusión

En conclusión, JavaScript es un lenguaje de programación fundamental en el desarrollo web moderno. A través de su evolución, especialmente con la introducción de ECMAScript 6, ha mejorado significativamente en términos de funcionalidad y expresividad. Desde la declaración de variables con let y const, hasta las funciones de flecha y los módulos, ES6 ha ampliado las capacidades del lenguaje y ha hecho que la codificación sea más eficiente y legible. Comprender estas características es esencial para aprovechar al máximo el potencial de JavaScript y crear aplicaciones web dinámicas y robustas.