Instituto Tecnológico de Chihuahua II

Ingeniería en Sistemas Computacionales

Programación Front End

El Lenguaje de Programación Javascript

Docente: Carlos Humberto Rubio Rascón

Brandon David Vargas Núñez - 1955072

Erick Fernando Siqueiros Zuñiga - 21550138

Brandon Isaac Leyva Gonzalez – 20550436

10/09/23

Índice

Javascript

Javascript es un lenguaje de programación interpretado e imperativo, orientado a objetos, débilmente tipado y dinámico, basado en prototipos.

Se utiliza principalmente para el desarrollo web del lado del cliente a través de su compatibilidad con otros lenguajes, así como librerías y frameworks.

Sin embargo también puede ser utilizado del lado del servidor así como para crear aplicaciones de escritorio en los sistemas operativos más populares o desarrollo móvil.

De esta forma Javascript se descubre como un lenguaje muy simple pero a la vez poderoso, cuya versatilidad es enorme, lo que lo ha llevado a convertirse en el estandarte y el más popular lenguaje web, aunque sus funciones no terminan ahí.

Origen y Esencia

JavaScript es un poderoso lenguaje de programación construido para el navegador Netscape en 1995. Todos los navegadores modernos lo adoptaron desde entonces para añadir funciones a los sitios web y, más recientemente, a aplicaciones web.

Anteriormente, las páginas web eran estáticas, similares a las páginas de un libro. Una página estática mostraba principalmente información en un diseño fijo y no todo aquello que esperamos de un sitio web moderno. JavaScript surgió como una tecnología del lado del navegador para hacer que las aplicaciones web fueran más dinámicas. Por medio de JavaScript, los navegadores eran capaces de responder a la interacción de los usuarios y cambiar la distribución del contenido en la página web.

JavaScript se clasifica principalmente como un lenguaje de programación interpretado. Es decir, directamente traducido a código de lenguaje de máquina subyacente mediante un motor de JavaScript. El motor analiza y ejecuta cada línea de código a medida que se encuentra durante el proceso de carga de la página. No es necesario compilar el código antes de probarlo, lo que permite un desarrollo rápido y una iteración ágil.

En Javascript no se define el tipo de una variable a la hora de instanciarla. El tipo de la variable se asigna atendiendo al valor que le asignemos a la variable. Además si vamos cambiando el valor asignado a la variable, esta podrá cambiar de tipo de datos. Es por esto que al tipado de Javascript, además de débil se le considera como tipado dinámico.

Cliente y Servidor

Javascript es un lenguaje que puede ser utilizado tanto para el desarrollo de aplicaciones de cliente, dónde han aparecido una gran cantidad de librerías y frameworks como jQuery, AngularJS, EmberJS, VueJS,… como para construir aplicaciones de servidor dónde el máximo exponente es NodeJS. Pero es que también ha ocupado su sitio en áreas para el desarrollo de APIs de bases de datos como sucede con MongoDB.

Javascript es un lenguaje orientado a objetos, aunque podríamos decir que no es un lenguaje puro de orientación a objetos como lo puede ser C++ o Java ya que carece de algunas características como herencia, ocultación,… Javascript utiliza prototipos para poder definir los objetos. Es decir, define un objeto como prototipo el cual se utiliza como base para poder definir nuevos objetos.

Al ser un lenguaje interpretado y que se ejecuta tanto en cliente, como en servidor, le permite a Javascript tener múltiples intérpretes en diferentes navegadores (Google Chrome, Microsoft

Edge, Safari, Opera,…) que se ejecutan en diferentes sistemas operativos (Windows, Mac, Linux,...).

Su popularidad y amplio uso ha permitido una prolífica comunidad y documentación, así como una base de código y librerías existentes que permiten adentrarse en él de forma muy fácil y accesible, sumado a su propia naturaleza de sencillo desarrollo. Es un lenguaje fácil en comparación a otros, pero muy versátil.

Funcionamiento de Javascript

Objetos: en JavaScript, los objetos son colecciones de propiedades y métodos que representan entidades del mundo real. Las propiedades son variables que almacenan valores, mientras que los métodos son funciones que realizan acciones relacionadas con el objeto. Los objetos se crean utilizando la sintaxis de llaves {} y pueden ser personalizados para contener propiedades y métodos específicos.

- **Prototipos:** utiliza un modelo de herencia basado en prototipos en lugar de clases tradicionales. Cada objeto tiene un prototipo, que es otro objeto del cual hereda propiedades y métodos. Cuando se accede a una propiedad o método en un objeto, si no se encuentra directamente en ese objeto, JavaScript busca en su prototipo y continúa escalando por la cadena de prototipos hasta que se encuentra o se alcanza el objeto raíz.
- **Funciones constructoras:** permite la creación de objetos utilizando funciones constructoras. Una función constructora es una función regular que se invoca utilizando la palabra clave «new». Dentro de la función constructora se definen las propiedades y métodos del objeto que se está creando. Cada vez que se crea un objeto utilizando una función constructora, se crea una instancia independiente con sus propias propiedades y métodos.
- **Herencia:** aunque JavaScript no ofrece una sintaxis de clase tradicional para la herencia, se puede conseguir la herencia utilizando prototipos. Un objeto puede heredar propiedades y métodos de otro objeto estableciendo su prototipo en ese objeto padre. Esto permite la reutilización de código y la creación de jerarquías de objetos.
- **Encapsulamiento:** JavaScript no tiene un mecanismo nativo de encapsulamiento como las clases en otros lenguajes, pero se puede lograr usando funciones para crear

ámbitos locales. Al definir variables y funciones dentro de una función, se pueden ocultar y proteger del acceso externo, creando así un nivel básico de encapsulamiento.

- **Polimorfismo:** permite la implementación de polimorfismo, que es la capacidad de objetos de diferentes tipos para responder al mismo mensaje o llamada de método. Dado que JavaScript es un lenguaje dinámico, las variables pueden contener diferentes tipos de objetos en distintos momentos, y se puede acceder a sus métodos y propiedades de manera flexible.

Variables

JavaScript es un lenguaje de programación con el ámbito global como ámbito, visibilidad o *scope* predeterminado y en el que todo se pasa por referencia también de forma predeterminada. Esto significa que una variable declarada fuera de una función es una variable global y es pasada por referencia a scopes descendientes o herederos.

La declaración con `var` define una variable en el ámbito local actual (léase función) y se hereda a scopes descendientes por referencia. Si la variable es declarada fuera de una función, la variable será una variable global.

`Let` y `const` son dos formas de declarar variables en JavaScript introducidas en ES6 que reducen el ámbito de la variable a bloques.

Un bloque en JavaScript se puede entender como «lo que queda entre dos corchetes», ya sean definiciones de funciones o bloques `if`, `while`, `for` y loops similares. Si una variable es declarada con `let` en el ámbito global o en el de una función, la variable pertenece al ámbito global o al ámbito de la función respectivamente, de forma similar a como ocurría con `var`.

El ámbito o scope para una variable declarada con `const` es, al igual que con `let`, el bloque, pero si la declaración con `let` previene la sobreescritura de variables, `const` directamente prohíbe la reasignación de valores (`const` viene de *constant*).

Funciones

Las funciones son uno de los bloques de construcción fundamentales en JavaScript. Es un conjunto de instrucciones que realiza una tarea o calcula un valor, pero para que un procedimiento califique como función, debe tomar alguna entrada y devolver una salida donde hay alguna relación obvia entre la entrada y la salida. Para usar una función, debes definirla en algún lugar del ámbito desde el que deseas llamarla.

Una definición de función (también denominada declaración de función o expresión de función) consta de la palabra clave function, seguida de:

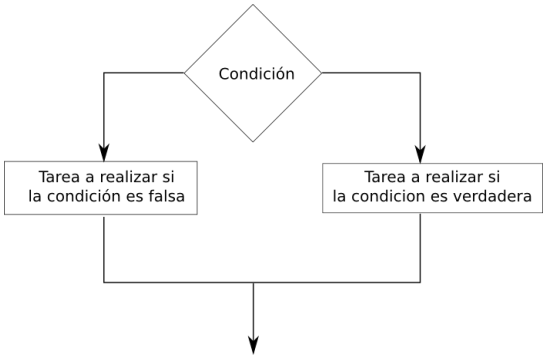
- El nombre de la función.
- Una lista de parámetros de la función, entre paréntesis y separados por comas.
- Las declaraciones de JavaScript que definen la función, encerradas entre llaves, { ... }.
- Por último la función debe llamarse en el código para ser usada actualmente.

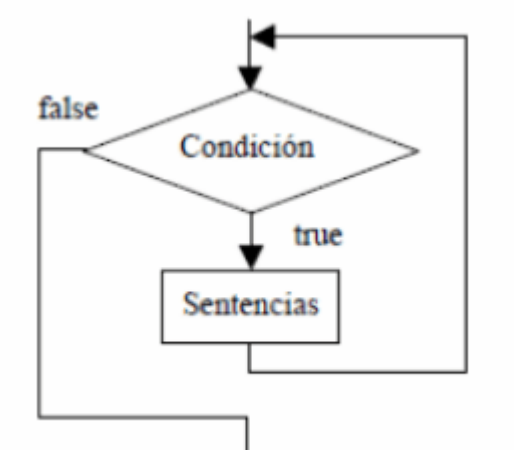
```
const square = function (number) {
  return number * number;
};
var x = square(4); // x obtiene el valor 16
```

Estructuras de control

En Javascript disponemos de 2 tipos de estructuras de control:

- **Estructuras condicionales**. Este tipo de estructura de control tiene como objetivo realizar una bifurcación del flujo de instrucciones. Cuando el programa llega a un punto, nosotros establecemos una condición en función de la misma el programa seguirá ejecutando unas instrucciones u otras.





Estructuras de repetición. Este tipo de estructuras de control también conocidas como bucles se utilizan para realizar de forma repetida varias acciones.
Estructuras de control de errores. Son aquellas que permiten controlar los errores que el usuario final comete de forma fortuita o intencionada y poder seguir trabajando de forma normal.

Conclusiones

Se puede entonces entender que Javascript es un lenguaje de programación interpretado, no compilado, cuya principal diferencia es que se traduce y ejecuta en tiempo real, lo cual permite una versatilidad y velocidad para las fases de prueba y depuración en el desarrollo, así como una tolerancia de errores más alta.

Esta característica a su vez le permite ser compatible con multitud de entornos, al solo requerir dicho motor de intérprete (elemento presente en prácticamente todo), por lo que es ampliamente utilizado en navegadores, donde se encuentra su principal uso: el desarrollo web del lado del cliente.

Pero su funcionamiento sigue siendo muy parecido a otros lenguajes, al ser Orientado a Objetos, mantiene muchas de sus prácticas y estructuras, pero de una forma un poco mas simple (fomentando de nuevo su accesibilidad) como el uso de las Funciones, o las Variables, que en este caso se utilizan solamente para definir la accesibilidad (Global, Local) encargándose el mismo lenguaje de su declaración automáticamente (débil tipado).

Considerando entonces todas sus características, Javascript es un lenguaje sumamente accesible y enormemente utilizado, no sólo ya para su propósito principal (Web) si no que dicha versatilidad le ha permitido evolucionar para ser utilizado en otros fines

(servidor/escritorio) posicionándose así como uno de los lenguajes más prominentes de la actualidad.

References

Características Javascript. (n.d.). Manual Web. Retrieved September 10, 2023, from https://www.manualweb.net/javascript/caracteristicas-javascript/

Coppola, M. (2023, July 10). *Qué es JavaScript, para qué sirve y cómo funciona*. Blog de HubSpot. Retrieved September 10, 2023, from https://blog.hubspot.es/website/que-es-javascript

Estructuras de control en Javascript | CurosHacker.es. (2021, March 23). CursoHacker.es. Retrieved September 10, 2023, from https://cursohacker.es/estructuras-de-control-en-javascript/

Funciones - JavaScript | MDN. (2023, August 3). MDN Web Docs. Retrieved September 10, 2023, from https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Functions

Padial, J. (2018, March 7). *Diferencias clave entre var, let y const en JavaScript*. CybMeta. Retrieved September 10, 2023, from https://cybmeta.com/var-let-y-const-en-javascript