Instituto Politécnico Profesor
Juan Bosch

JavaScript
Programación Orientada a Objeto

DOCENTE : Yisel Garo

ASIGNATURA : Desarrollo de Aplicaciones

GRADO : 5to de Aplicaciones

GRUPO #3

Participantes:

- Leanne caba #7
- Adrián Cordero #12
- Elianna Jiménez #22
- Zoilibel Novas #27
- Isaac Déhuel #33
- Keyver Reyes #34
- Omar Rosado #35

Introducción

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, aplicaciones de escritorio (mayoritariamente widgets) es también significativo.

JavaScript se diseñó con una sintaxis similar a C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

Tradicionalmente se venía utilizando en páginas web HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. Actualmente es ampliamente utilizado para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX. JavaScript se interpreta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

JavaScript aprovecha su naturaleza de prototipo para dar la bienvenida a los desarrolladores de POO a su ecosistema. También proporciona formas sencillas de crear prototipos y organizar datos relacionados.

Historia de JavaScript

○ ¿Qué es JavaScript? Un poco de historia

Se trata de un lenguaje de programación que permite realizar acciones de diversos grados de complejidad en sitios web sin necesidad de compilación. El hecho de que los mismos navegadores lean y asimilen el código para efectuar las instrucciones indicadas por éste, ha convertido a JavaScript en un lenguaje de programación muy utilizado y apreciado por los desarrolladores.

A día de hoy, es el lenguaje en el que se desarrolla una página web cuando los mecanismos que se desean implantar encierran un mayor nivel de dificultad de los que se suelen plantear a menudo. Pero, antes de continuar hablando de la actualidad de JavaScript, vamos a hacer un breve regreso al pasado para entender qué circunstancias dieron origen a este lenguaje de programación.

Esa época en la que usábamos los módems que se conectaban a la línea telefónica, fue también el momento en el que a las aplicaciones para web se le empezaba a añadir complejidad. Esto se trasladó a las páginas web, que incluían formularios cada vez más completos y complicados, lo que afectó a la calidad de la navegación online.

Con una velocidad de navegación tan lenta y unas aplicaciones que avanzaban sin parar, resultó evidente que HTML se estaba quedando corto, así que los programadores tuvieron que pararse a pensar cómo solucionar este desfase. La respuesta llegó en forma de un lenguaje de programación que podía ejecutarse directamente en el navegador del lado del usuario.

Adaptando tecnologías de programación ya existentes, el navegador Netscape desarrolló LiveScript, un lenguaje que permitía crear programas pequeños en las páginas y que fuese más sencillo que Java, creado por Sun Microsystems. Ambas compañías, Netscape y Sun Microsystems, unieron fuerzas y conocimiento para desarrollar juntas la programación JavaScript.

Y así, en líneas muy generales, nació un lenguaje de desarrollo web fácil de utilizar y accesible a personas con nociones básicas de programación o, incluso, con ningún conocimiento previo en la materia, lo cual explica porque la popularidad de JavaScript gozó de

tan buena salud desde sus comienzos y también porque lo sigue haciendo a día de hoy.

Origen de JavaScript

JavaScript fue desarrollado originalmente por Brendan Eich de Netscape con el nombre de Mocha, el cual fue renombrado posteriormente a LiveScript, para finalmente quedar como JavaScript. El cambio de nombre coincidió aproximadamente con el momento en que Netscape agregó soporte para la tecnología Java en su navegador web Netscape Navigator en la versión 2.002 en diciembre de 1995. La denominación produjo confusión, dando la impresión de que el lenguaje es una prolongación de Java, y se ha caracterizado por muchos como una estrategia de mercadotecnia de Netscape para obtener prestigio e innovar en lo que eran los nuevos lenguajes de programación web.5 6.

Brendan Eich, un programador que trabajaba en Netscape, pensó que podría solucionar este problema adaptando otras tecnologías existentes (como *ScriptEase*) al navegador Netscape Navigator 2.0, que iba a lanzarse en 1995. Inicialmente, Eich denominó a su lenguaje *LiveScript*.

JavaScript se diseñó con una sintaxis similar a C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

«JavaScript» es una marca registrada de Oracle Corporation. Es Usada con licencia por los productos creados por Netscape Communications y entidades actuales como la Fundación Mozilla.

Microsoft dio como nombre a su dialecto de JavaScript «JScript», para evitar problemas relacionados con la marca. JScript fue adoptado en la versión 3.0 de Internet Explorer, liberado en agosto de 1996, e incluyó compatibilidad con el Efecto 2000 con las funciones de fecha, una diferencia de los que se basaban en ese

momento. Los dialectos pueden parecer tan similares que los términos «JavaScript» y «JScript» a menudo se utilizan indistintamente, pero la especificación de JScript es incompatible con la de ECMA en muchos aspectos.

Tradicionalmente se venía utilizando en páginas web HTML para realizar operaciones y únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. Actualmente es ampliamente utilizado para enviar y recibir información del servidor junto con ayuda de otras tecnologías como AJAX. JavaScript se interpreta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

En 1997 los autores propusieron 9 JavaScript para que fuera adoptado como estándar de la European Computer Manufacturers 'Association ECMA, que a pesar de su nombre no es europeo sino internacional, con sede en Ginebra. En junio de 1997 fue adoptado como un estándar ECMA, con el nombre de ECMAScript. Poco después también como un estándar ISO.

Elementos de JavaScript

Posee algunos objetos predefinidos u objetos intrínsecos como son: Array, Boolean, Date, Function, Global, Math, Number, Object, RegExp, y String. Además, es posible crear objetos nuevos, con sus propios métodos y propiedades, adaptados a las necesidades concretas de cada aplicación.

○ Array

El objeto Array de JavaScript es un objeto global que es usado en la construcción de arrays, que son objetos tipo lista de alto nivel.

Los *arrays* son objetos similares a una lista cuyo prototipo proporciona métodos para efectuar operaciones de recorrido y de mutación. Tanto la longitud como el tipo de los elementos de un *array* son variables. Dado que la longitud de un *array* puede cambiar en cualquier momento, y los datos se pueden almacenar en ubicaciones no contiguas, no hay garantía de que los *arrays* de JavaScript sean densos; esto depende de cómo el programador elija

usarlos. En general estas características son cómodas, pero si, en su caso particular, no resultan deseables, puede considerar el uso de *arrays* con tipo.

- Crear un Array

```
let frutas = [ “Manzana”, “Banana” ];
console.log(frutas.length)
```

○ Date

Permite trabajar con fechas y horas.

El constructor Date:

```
new Date();
new Date(milisegundos);
new Date(cadenaFecha);
```

○ Math

Math es un objeto incorporado que tiene propiedades y métodos para constantes y funciones matemáticas. No es un objeto de función. Math funciona con el tipo Number.

- Descripción

A diferencia de los demás objetos globales, el objeto Math no se puede editar. Todas las propiedades y métodos de Math son estáticos. Usted se puede referir a la constante *pi* como Math.PI y puede llamar a la función *seno* como Math.sin(x), donde x es el argumento del método. Las constantes se definen con la precisión completa de los números reales en JavaScript.

```
Math.gcd = function() {
  if (arguments.length == 2) {
    if (arguments[1] == 0)
      return arguments[0];
    else
      return Math.gcd(arguments[1], arguments[0] % arguments[1]);
  } else if (arguments.length > 2) {
    var result = Math.gcd(arguments[0], arguments[1]);
    for (var i = 2; i < arguments.length; i++)
      result = Math.gcd(result, arguments[i]);
    return result;
  }
```

- ## Number

Number es un objeto primitivo envolvente que permite representar y manipular valores numéricos cómo 37 o -9.25. El constructor Number contiene constantes y métodos para trabajar con números. Valores de otro tipo pueden ser convertidos a números usando la función Number().

- ## Object

La clase Object representa uno de los tipos de datos de JavaScript. Es usado para guardar una colección de datos definidos y entidades más complejas. Los objetos pueden ser creados utilizando el constructor Object() o la sintaxis literal de objeto. El constructor Object crea una envoltura al objeto.

```
let person = {
  firstName: "John",
  lastName: "Doe",
  age: 50,
  eyeColor: "blue"
};
```

- ## String — Cadena de caracteres

El objeto String se utiliza para representar y manipular una secuencia de caracteres.

- Descripción

Las cadenas son útiles para almacenar datos que se pueden representar en forma de texto. Algunas de las operaciones más utilizadas en cadenas son verificar su length, para construirlas y concatenarlas usando operadores de cadena + y +=, verificando la existencia o ubicación de subcadenas con indexOf() o extraer subcadenas con el método substring().

- Crear cadenas

Las cadenas se pueden crear como primitivas, a partir de cadena literales o como objetos, usando el constructor String():

Aquí podemos ver 3 ejemplos de cadenas String().

```
var string1 = "Una Cadena primitiva";
let string2 = "También una cadena primitiva";
const string3 = "Otra cadena primitiva más";
```

Función de JavaScript (POO)

Un estilo de programación orientada a objetos (POO), en el que la herencia se produce mediante la definición de clases de objetos, en lugar de que la herencia se produzca únicamente a través de los objetos.

El modelo más popular de POO está basado en clases, pero como mencioné, JavaScript no es un lenguaje basado en clases, es un lenguaje basado en prototipos.

Según la documentación de Mozilla:

Un lenguaje basado en prototipos toma el concepto de *objeto prototípico*, un objeto que se utiliza como una plantilla a partir de la cual se obtiene el conjunto inicial de propiedades de un nuevo objeto.

Demos un vistazo a este código:

```
let nombres = {
    nombre: “Juan”,
    apellido: “Pérez”
};
console.log(nombres.nombre);
console.log(nombres.hasOwnProperty(“segNombre”));
```

El objeto de la variable nombres solo tiene dos propiedades: nombre y apellido. Ningún método en absoluto.

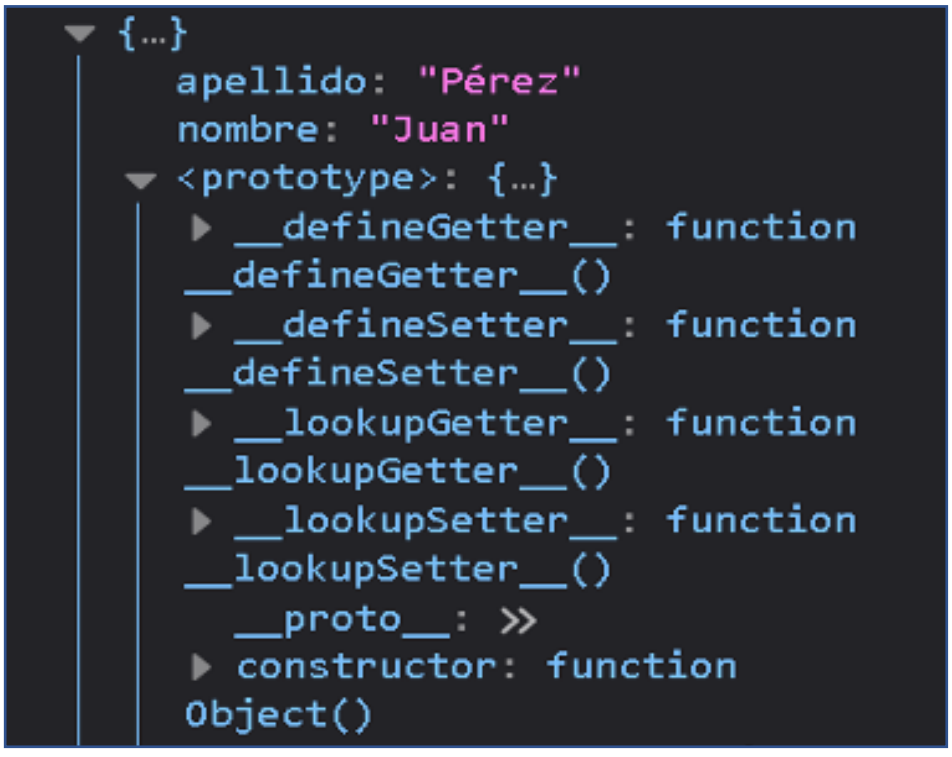
Entonces, ¿De donde viene hasOwnProperty?

Bueno, viene del prototipo Object.
Intenta registrar el contenido de la variable en la consola:

```
console.log(nombres);
```

Cuando expandas los resultados en la consola, obtendrás esto:

¿Notas la última propiedad: **<prototype>**? Intenta expandirlo:



Verás un conjunto de propiedades en el constructor Object. Todas estas propiedades provienen del *prototipo* Object global. Si observas de cerca, también notarás nuestra propiedad hasOwnProperty oculta.

En otras palabras, todos los objetos tienen acceso al prototipo de Object. No poseen estas propiedades, pero se les concede acceso a las propiedades del prototipo.

Cada objeto en JavaScript tiene acceso al prototipo de Object por defecto. Si está configurado para usar otro prototipo, digamos prototype2, entonces prototype2 también tendría acceso al prototipo de Object por defecto, y así sucesivamente.

Características de la Programación Orientada a Objeto

○ La abstracción

Abstracción denota las características esenciales de un objeto, donde se capturan sus comportamientos. Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar cómo se implementan estas características. Los procesos, las funciones o los métodos pueden también ser abstraídos, y, cuando lo están, una variedad de técnicas es requeridas para ampliar una abstracción. El proceso de abstracción permite seleccionar las características relevantes dentro de un conjunto e identificar comportamientos comunes para definir nuevos tipos de entidades en el mundo real. La abstracción es clave en el proceso de análisis y diseño orientado a objetos, ya que mediante ella podemos llegar a armar un conjunto de clases que permitan modelar la realidad o el problema que se quiere atacar.

○ Encapsulación

La encapsulación significa reunir todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción. Esto permite aumentar la cohesión de los componentes del sistema. Algunos autores confunden este

concepto con el principio de ocultación, principalmente porque se suelen emplear conjuntamente.

○ Herencia

Herencia, las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. La herencia organiza y facilita el polimorfismo y el encapsulamiento, permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que volver a implementarlo. Esto suele hacerse habitualmente agrupando los objetos en clases y estas en árboles o enrejados que reflejan un comportamiento común. Cuando un objeto hereda de más de una clase se dice que hay herencia múltiple.

○ Modularidad

Modularidad se denomina modularidad a la propiedad que permite subdividir una aplicación en partes más pequeñas (llamadas módulos), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las restantes partes. Estos módulos se pueden compilar por separado, pero tienen conexiones con otros módulos. Al igual que la encapsulación, los lenguajes están soportan modularidad de diversas formas.

○ Polimorfismo

Polimorfismo comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre; al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. O, dicho de otro modo, las referencias y las colecciones de objetos pueden contener objetos de diferentes tipos, y la invocación de un comportamiento en una referencia producirá el comportamiento correcto para el tipo real del objeto referenciado.

Ventajas y desventajas

Ventajas de usar JavaScript

○ Es de fácil aprendizaje

JavaScript es un lenguaje muy sencillo de dominar, ya que su curva de aprendizaje es baja. Asimismo, si eres un usuario con conocimiento en programación, te vas a dar cuenta que la sintaxis es muy similar a otros programas de desarrollo como C y Java. Además, cuenta con comunidades consolidadas, donde puedes conseguir información como manuales, tutoriales, incluso hasta códigos ya listo para implementar.

○ Es muy versátil

JavaScript es un lenguaje estándar en la industria web, por tanto, se puede integrar con otras tecnologías. Por esta razón, el código se puede insertar en cualquier página independientemente de la extensión del archivo, es decir, añadir scripts en archivos JSP, PHP, Perl, por mencionar algunos.

○ Carga del servidor

Como el programa se ejecuta del lado del cliente se reduce la carga en el servidor de la página web. En consecuencia, tu sitio va a responder de manera más rápida y los usuarios lo van a percibir.

○ Crea interfaces dinámicas

Es otra ventaja notable, con JavaScript puedes desarrollar elementos como menús desplegables, botones, formularios de registros, encuestas, agregar efectos al texto, cambiar el color de la fuente, etc.

○ Es multiplataforma

JavaScript es un lenguaje de programación web multiplataforma, es decir, se ejecuta en distintos sistemas operativos, como Mac, Linux y Windows. Por ello, puedes desarrollar cualquier aplicación, y va a funcionar sin ningún problema. En este sentido, el lenguaje se ejecuta en cualquier navegador, con el plus que es compatible con los dispositivos más modernos de la actualidad, incluyendo iPhone, SmartPhone y PS3.

○ Es compatible con los distintos CMS

JavaScript es un lenguaje muy sencillo de dominar, ya que su curva de aprendizaje es baja. Si tienes un portal web desarrollado en Joomla, WordPress o Drupal, tranquilamente puedes agregar scripts JavaScript para ofrecer mayor interactividad a tu sitio web.

○ Se actualiza con frecuencia

Debido a la flexibilidad del lenguaje, empresas como Google, Microsoft han desarrollado frameworks agregando el modelo vista controlador (MVC), esto facilita construir aplicaciones web en poco tiempo. Esta evolución permite ejecutar programas tanto en el lado del servidor como en la computadora, por tanto, puedes programar con libertad.

A continuación, te indico los frameworks JavaScript más populares:

- Angular.
- React
- React Native
- Vue
- ElectronJS.

Desventajas de usar JavaScript

A continuación, te indico las desventajas de usar JavaScript en la programación web.

○ Sus scripts lo hacen vulnerable

Como es un programa que se ejecuta en el lado del cliente, sus códigos pueden ser leídos por otros usuarios.

○ Es posible desactivar el JavaScript

A veces los usuarios por desconocimiento pueden desactivar la funcionalidad de JavaScript en el navegador, esto genera que no se ejecuten los códigos dinámicos en la página web.

○ Requiere de otras aplicaciones

Para poder diseñar un sitio web completo se debe usar con otros lenguajes de programación.

Sintaxis

○ Clases

Las clases de JavaScript, introducidas en ECMAScript 2015, son una mejora sintáctica sobre la herencia basada en prototipos de JavaScript. Las clases de JavaScript proveen una sintaxis mucho más clara y simple para crear objetos y lidiar con la herencia.

○ Constructor

El método constructor es un método especial para crear e inicializar un objeto creado con una clase. Solo puede haber un método especial con el nombre "constructor" en una clase. Si esta contiene mas de una ocurrencia del método **constructor,** se arrojará un *Error* SyntaxError

○ this.

La palabra clave this de una función se comporta un poco diferente en Javascript en comparación con otros lenguajes.

En general, el valor de this está determinado por cómo se invoca a la función. No puede ser establecida mediante una asignación en tiempo de ejecución, y puede ser diferente cada vez que la función es invocada.

○ Subclases con extends

La palabra clave extends es usada en *declaraciones de clase* o *expresiones de clase* para crear una clase hija.

Conclusión

JavaScript es un lenguaje de programación sumamente amplio ya que cada versión o cada actualización le van mejorando y agregando mas cosas, JavaScript vino con el propósito de darle dinamismo a los sitios webs.

El lenguaje JavaScript mediante los framework, podemos desarrollar distintas áreas de la programación. Ejemplo, framework para tener un desarrollo mas rápido de los sitios web pueden ser React, Angular, Vue, Svelte, entre muchos mas.

Podemos utilizar framework para desarrollar aplicaciones móviles, hibridas, multiplataforma, mediante los distintos framework del desarrollo móvil. Estos framework pueden ser Ionic, React Native, Cordova, entre otros mas.

También podemos desarrollar Juegos con los framework Phaser, Playground.js, Panda.js, Quintus, entre otros mas. Otra área de desarrollo puede ser la creación de software para computadoras, podemos utilizar los framework como EletronJs, React Desktop.

Podemos desarrollar en muchísimas áreas de programación con este lenguaje ya que tienen diferentes framework que lo ayudar a poder adaptarse.

Referencias Bibliografías

https://hackaboss.com/blog/historia-javascript

https://es.wikipedia.org/wiki/JavaScript

https://www.jairogarciarincon.com/clase/elementos-del-lenguaje-javascript

https://developer.mozilla.org/es/docs/Web/JavaScript

https://www.freecodecamp.org/programacion-orientada-a-objectos /

https://es.wikibooks.org/wiki/Caracteristica_de_la_POO

https://blogueropro.com/blog/ventajas-y-desventajas-de-usar-javascript