Capítulo 1

Introducción a JavaScript

1.1. Historia de JavaScript

JavaScript, comunmente abreviado como JS, es un lenguaje de programación interpretado. En los comienzos, el lenguaje se utilizaba para agregar dinamismo del lado del cliente a las páginas web. Sin embargo, hoy en día se pueden crear aplicaciones de escritorio o del lado del servidor.

El lenguaje fue creado por **Brendan Eich** en 1995, quien en ese entonces trabajaba para Netscape. Eich denominó a su lenguaje `LiveScript`, y el objetivo inicial del lenguaje era solucionar problemas de validación de formularios complejos en el lado del cliente para el navegador Netscape Navigator, tratando de adaptarlo a tecnologías ya existentes.

La empresa Netscape junto con Sun Microsystems desarrollaron en conjunto este lenguaje de programación. Pero por cuestiones de mercado antes del lanzamiento, Netscape decidió cambiar el nombre del lenguaje a JavaScript (ya que en ese entonces Java estaba de moda en el mundo informático).

Al poco tiempo, la empresa Microsoft lanzó `JScript` para Internet Explorer. Para no entrar en una guerra informática, Netscape decidió que lo mejor sería estandarizar el lenguaje. Para ello, enviaron la especificación de JavaScript 1.1 al organismo ECMA (European Computer Manufacturers Association).

ECMA creó el comité TC39 con el objetivo de "*estandarizar de un lenguaje de script multiplataforma e independiente de cualquier empresa*". El primer estándar que creó el comité TC39 se denominó ECMA-262, en el que se definió por primera vez el lenguaje ECMAScript (abreviado comunmente como `ES`).

Es así entonces, que cuando hablamos de JavaScript, estamos haciendo referencia a una implementación de lo que se conoce como `ECMAScript`. El estándar ha ido evolucionando con el paso del tiempo. En la actualidad, la mayoría de los navegadores corren algún intérprete que soporta la mayoría de las características de las versiones 5.1 y 6.

Aunque la última versión sea la de ECMAScript 8 (lanzada en Junio de 2017), la versión 6 es más popular, ya que en ésta se han agregado muchos cambios significativos para el lenguaje. En este documento se hará énfasis en las versiones 5.1 y 6.

1.2. Características del lenguaje

JavaScript es un lenguaje de alto nivel, interpretado y multiparadigma. Es dinámica y débilmente tipado. Posee herencia basada en prototipos.

Se dice que es multiparadigma porque soporta los paradigmas imperativo, funcional, orientado a objetos (prototipado) y dirigido por eventos.

1.2.1. Influencias

JavaScript tiene fuertes influencias de varios lenguajes. Sus características más sobresalientes surgen de los siguientes lenguajes:

Java y C – No solo tiene la influencia sobre el nombre, sino que además tiene influencia sobre la sintaxis del lenguaje. Tanto Java como JavaScript sintácticamente emergen del lenguaje C. Sin embargo, Java y JavaScript tienen semánticas y propósitos diferentes.

Perl y Python – Tanto Perl como Python han influido en el manejo de strings, arreglos y expresiones regulares en JavaScript.

Scheme – De la familia del paradigma funcional. Adopta las funciones de primera clase y *closures*, los cuales se tratarán más adelante.

Self – Un lenguaje desarrollado por Sun Microsystems. Es de los pocos lenguajes que tienen herencia prototipada. Además de ésta característica, también se adopta la inusual notación de objetos.

1.2.2. Intérpretes

(To Do)

1.3. Nociones básicas

1.3.1. Tipos primitivos

Undefined – El tipo indefinido tiene un único valor, `undefined`. A toda variable que aún no se le haya asignado valor, tendrá el valor `undefined`.

Null – El tipo nulo tiene un único valor, `null`.

Boolean – El tipo booleano representa una entidad lógica con dos posibles valores, `true` ó `false`.

String – ...

Number – ...

Symbol – Fue agregado en la versión de ES6. Abarca el conjunto de todos los valores no String que pueden ser usados como clave en la propiedad de un Object. Cada valor posible de Symbol es único e inmutable. Se los puede pensar como tokens que sirven como identificadores únicos.

Object – ...

1.3.2. Palabras reservadas

Las palabras reservadas del lenguaje se dividen en cuatro conjuntos:

- Palabras claves (*keywords*)
- Palabras reservadas a futuro
- Literal nulo (`null`)
- Literales booleanos (`true` y `false`)

Las siguientes son palabras claves, a excepción de `null`, `true` y `false`, que son literales.

CUADRO 1.1: Lista de palabras claves del lenguaje.

break	do	import	throw
case	else	in	true
catch	export	instanceof	try
class	extends	new	typeof
const	false	null	var
continue	finally	return	void
debugger	for	super	while
default	function	switch	with
delete	if	this	yield

Por otro lado, existe un conjunto de palabras reservadas a futuro. En un principio son solamente dos: `await` y `enum`. Pero si se especifica la directiva de *strict mode*, aparecen otras más: `implements`, `interface`, `package`, `private`, `protected` y `public`.

En resumen, las palabras reservadas a futuro (en modo estricto) son:

CUADRO 1.2: Lista de palabras reservadas a futuro.

await	implements	package	protected
enum	interface	private	public

```
1 var x;
```