Introdución a JavaScript

HTML y CSS incluyen instrucciones para indicar al navegador cómo debe organizar y visualizar un documento y su contenido, pero la interacción de estos lenguajes con el usuario y el sistema se limita solo a un grupo pequeño de respuestas predefinidas. Podemos crear un formulario con campos de entrada, controles y botones, pero HTML solo provee la funcionalidad necesaria para enviar la información introducida por el usuario al servidor o para limpiar el formulario. Algo similar pasa con CSS; podemos construir instrucciones (reglas) con pseudoclases como **:hover** para aplicar un grupo diferente de propiedades cuando el usuario mueve el ratón sobre un elemento, pero si queremos realizar tareas personalizadas, como modificar los estilos de varios elementos al mismo tiempo, debemos cargar una nueva hoja de estilo que ya presente estos cambios.

Con el propósito de alterar elementos de forma dinámica, realizar operaciones personalizadas, o responder al usuario y a cambios que ocurren en el documento, los navegadores incluyen un tercer lenguaje llamado JavaScript.

JavaScript apareció en 1995 con el principal objetivo de encargarse de la validación de entradas que previamente se habían dejado en manos de lenguajes del lado del servidor, como Perl. Así, se necesitaba un viaje de ida y vuelta al servidor para determinar si un campo obligatorio se había dejado en blanco o si el valor introducido en dicho campo era válido. JavaScript ha ido evolucionando hasta convertirse en una característica básica de todos los Navegadores Web.

Breve historia de JavaScript

A principios de los años 90, la mayoría de usuarios que se conectaban a Internet lo hacían con módems a una velocidad máxima de 28.8 kbps. En esa época, empezaban a desarrollarse las primeras aplicaciones web y por tanto, las páginas web comenzaban a incluir formularios complejos.

Con unas aplicaciones web cada vez más complejas y una velocidad de navegación tan lenta, surgió la necesidad de un lenguaje de programación que se ejecutara en el navegador del usuario. De esta forma, si el usuario no rellenaba correctamente un formulario, no se le hacía esperar mucho tiempo hasta que el servidor volviera a mostrar el formulario indicando los errores existentes.

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.

Posteriormente, Netscape firmó una alianza con Sun Microsystems para el desarrollo del nuevo lenguaje de programación. Además, justo antes del lanzamiento Netscape decidió cambiar el nombre por el de JavaScript. La razón del cambio de nombre fue exclusivamente por marketing, ya que Java era la palabra de moda en el mundo informático y de Internet de la época.

La primera versión de JavaScript fue un completo éxito y Netscape Navigator 3.0 ya incorporaba la siguiente versión del lenguaje, la versión 1.1. Al mismo tiempo, Microsoft lanzó JScript con su navegador Internet Explorer 3. JScript era una copia de JavaScript al que le cambiaron el nombre para evitar problemas legales.

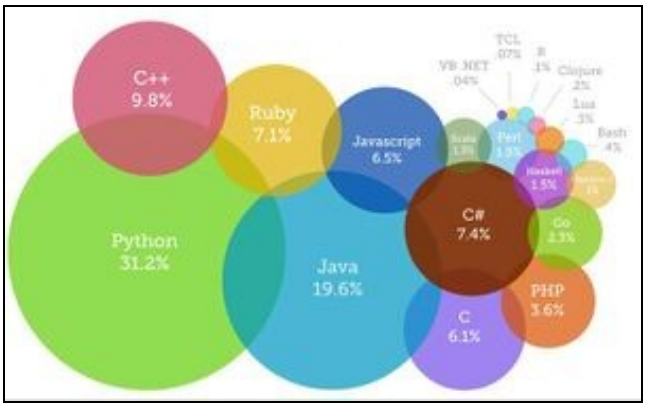
Para evitar una guerra de tecnologías, Netscape decidió que lo mejor sería estandarizar el lenguaje JavaScript. De esta forma, en 1997 se envió la especificación JavaScript 1.1 al organismo ECMA (*European Computer Manufacturers Association*).

ECMA creó el comité TC39 con el objetivo de "estandarizar 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.

Por este motivo, algunos programadores prefieren la denominación ECMAScript para referirse al lenguaje JavaScript. De hecho, JavaScript no es más que la implementación que realizó la empresa Netscape del estándar ECMAScript. La organización internacional para la estandarización (ISO) adoptó el estándar ECMA-262 a través de su comisión IEC, dando lugar al estándar **ISO/IEC-16262**.

¿Por qué JavaScript?

En el mundo de la informática nos podemos encontrar con un gran número de lenguajes de programación diferentes, un número tan elevado que es prácticamente imposible que un programador sea capaz de controlar todos ellos a la perfección. Lo que sí se puede conseguir es llegar a dominar aquellos lenguajes más importantes y que más opciones ofrecen para encontrar un buen trabajo.



Programming Languages 2020

Mientras que en el desarrollo de aplicaciones y servicios web del lado del servidor son varios los lenguajes de programación que compiten entre sí, en el lado del cliente, es decir, en el navegador web, *JavaScript* se ha convertido en la tecnología ganadora, desplazando completamente a otras tecnologías como *Flash* o los *applets* de *Java* que también intentaron liderar la programación de aplicaciones en el navegador web. Hoy en día, cualquier aplicación *JavaScript* se ejecuta en cualquier navegador web.

Partiendo de *JavaScript*, también nos centraremos en **AJAX, acrónimo de *Asynchronous JavaScript And XML* (JavaScript asíncrono y XML),** que es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (*Rich Internet Applications*). Estas aplicaciones se ejecutan en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. Así, es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

En los últimos años han aparecido multitud de *frameworks* para desarrollar aplicaciones web del lado del cliente con *JavaScript*. Estos *frameworks* son herramientas que agilizan, reducen los tiempos de desarrollo y mejoran la calidad y mantenibilidad de las aplicaciones. Los ejemplos más sobresalientes son *Angular*, *Vue.js*, *React.js* o *Ember.js*.

Por todo esto, existe un interés creciente por parte de los desarrolladores web en usar estos potentes *frameworks*.

Volver