T1 – INTRODUCCIÓN A JAVASCRIPT

https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Introducci%C3%B3n#What_is_JavaScript

¿Qué es JavaScript?

JavaScript es un lenguaje de scripting multiplataforma y orientado a objetos. Es un lenguaje pequeño y liviano. Dentro de un ambiente de host, javaScript puede conectarse a los objetos de su ambiente y proporcionar control programático sobre ellos.
JavaScript contiene:

- una librería estándar de objetos, tales como Array, Date, y Math, y
- un conjunto central de elementos del lenguaje, tales como operadores, estructuras de control, y sentencias.

El núcleo de JavaScript puede extenderse para varios propósitos, complementándolo con objetos adicionales, por ejemplo:

- ***Client-side JavaScript*** extiende el núcleo del lenguaje proporcionando objetos para controlar un navegador y su modelo de objetos (o DOM, por las iniciales de Document Object Model). Por ejemplo, las extensiones del lado del cliente permiten que una aplicación coloque elementos en un formulario HTML y responda a eventos del usuario, tales como clicks del ratón, ingreso de datos al formulario y navegación de páginas.
- ***Server-side JavaScript*** extiende el núcleo del lenguaje proporcionando objetos relevantes a la ejecución de javaScript en un servidor. Por ejemplo, las extensiones del lado del servidor permiten que una aplicación se comunique con una base de datos, proporcionar continuidad de la información de una invocación de la aplicación a otra, o efectuar manipulación de archivos en un servidor.

JavaScript y Java

JavaScript y Java son similares en algunos aspectos, pero fundamentalmente diferentes en otros:

- El lenguaje JavaScript se parece a Java pero no tiene el tipo estático (static) de Java, ni tiene un chequeo de tipos fuerte.
- JavaScript usa la mayoría de la sintaxis de expresiones de Java, convenciones de nombrado, y las construcciones básicas de control de flujo, razón por la cual se le cambió el nombre del original LiveScript a JavaScript.

En contraste con el sistema de clases construidas por declaraciones que se usa en tiempo de compilación de java:

- JavaScript soporta un sistema de tiempo de ejecución basado en un pequeño número de tipos de datos que representan valores numéricos, lógicos, y de cadena de caracteres (string).
- **JavaScript tiene un modelo de objetos basado en prototipos en lugar del modelo de objetos basado en clases**, que es más común. **El modelo basado en prototipo proporciona herencia dinámica; esto es, que lo que se hereda puede variar entre objetos individuales**.
- JavaScript también soporta funciones sin ningún requerimiento declarativo especial. Las funciones pueden ser propiedades de los objetos, ejecutándose como métodos levemente tipados.

Comparado con Java:

- JavaScript es un lenguaje muy libre de forma.
- En JavaScript:
  - No hay que declarar todas las variables, clases, y métodos.
  - No hay que preocuparse de si los métodos son públicos, privados, o protegidos, y no hay que implementar interfaces.
  - Las variables, parámetros, y retornos de funciones no tienen que declararse explícitamente de un tipo dado.

Java es un lenguaje de programación:

- basado en clases, diseñado para lograr **seguridad de tipos**, y ejecución rápida.
  - **Seguridad de tipos significa**, por ejemplo, que no es posible hacer una conversión de tipos a un Integer de java para obtener una referencia a un objeto, o acceder a memoria protegida mediante la alteración del *bytecode* de una clase.

El **modelo basado en clases de Java** implica:

- que los programas consisten exclusivamente de clases y sus métodos.
- La herencia de clases de Java y el tipado fuerte generalmente requiere de jerarquías de objetos fuertemente acopladas.
- Estos requerimientos hacen que la programación en Java sea más compleja que la programación en JavaScript.

En contraposición, JavaScript:

- proviene en espíritu de una línea de lenguajes de programación más pequeños, con **tipado dinámico**, como HyperTalk, y dBASE. Estos lenguajes de *scripting* hacen accesibles las herramientas de programación a audiencias mucho más amplias. Esto se debe a su sintaxis más indulgente, funcionalidad especializada ya incluida, y mínimos requisitos para la creación de objetos.

JavaScript en comparación a Java

JavaScript	Java
Orientado a objetos. Sin distinción entre tipos de objetos. La herencia se modela mediante el mecanismo de prototipado, y es posible añadir propiedades y métodos a cualquier objeto de forma dinámica.	Basado en clases. Los objetos se dividen en clases e instancias. Toda la herencia está sujeta a una jerarquía de clases. Las clases e instancias no pueden obtener nuevas propiedades o métodos de forma dinámica.
No se declaran los tipos de datos de las variables (**tipado dinámico** = comprobación de tipos en el momento de la ejecución, son interpretados).	Los tipos de datos de las variables deben ser declarados (**tipado estático** = los tipos se comprueban en el proceso de compilación, antes de ejecutarse).
No se puede escribir automáticamente en el disco duro.	No se puede escribir automáticamente en el disco duro.

Para más información sobre las diferencias entre JavaScript y Java, vea el capitulo Details of the object model.

JavaScript y la especificación ECMAScript

JavaScript está estandarizado en Ecma International — la asociación europea para la creación de estándares para la comunicación y la información (ECMA originalmente era un acrónimo: European Computer Manufacturers Association) con el fin de ofrecer un lenguaje de programación estandarizado e internacional, basado en Javascript. Esta versión de JavaScript, llamada ECMAScript, se comporta de la misma manera en todas las aplicaciones que implementan el estándar. Cualquier compañía puede usar el lenguaje basado en estándares abiertos, para desarrollar su implementación de JavaScript. El estándar ECMAScript está documentado en la especificación ECMA-262. Vea New in JavaScript para conocer más acerca de las diferentes versiones de JavaScript y ediciones de la especificación ECMAScript.
El estándar ECMA-262 también está aprobado por ISO (Organización Internacional de Normalización), como ISO-16262. También se puede encontrar la especificación en el sitio de Ecma International.

La especificación **ECMAScript no describe el Document Object Model (DOM)**, que está estandarizado por el World Wide Web Consortium (W3C). El DOM define la forma como los objetos de los documentos HTML se exponen a sus scripts. Para tener una mejor idea acerca de las diversas tecnologías que se usan al programar con JavaScript, consulte el artículo JavaScript technologies overview.

Documentación de JavaScript versus especificación de ECMAScript

La especificación ECMAScript es un conjunto de requisitos para la implementación de ECMAScript; es útil si desea implementar las características del lenguaje compatible con los estándares en la implementación ECMAScript o motor (como SpiderMonkey en Firefox, o v8 en Chrome).

La documentación ECMAScript no está destinada a ayudar a los programadores de script; utilizar la documentación JavaScript para la información sobre la escritura de scripts.

La especificación ECMAScript utiliza terminología y sintaxis que puede ser desconocida para un programador de JavaScript. Aunque la descripción del lenguaje puede variar en ECMAScript, el lenguaje sigue siendo el mismo. JavaScript soporta toda la funcionalidad se indica en la especificación ECMAScript.

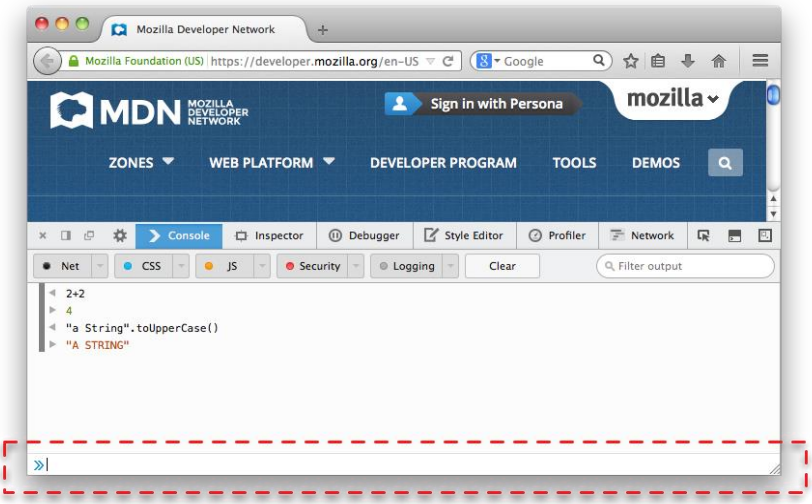
La documentación JavaScript describe los aspectos del lenguaje que son apropiados para el programador JavaScript.

¿Cómo empezar con JavaScript?

Empezar con JavaScript es fácil: todo lo que necesitas es un navegador moderno. Esta guía incluye algunas características que sólo están disponibles en las últimas versiones de Firefox, así que se recomienda usar la versión más reciente de Firefox.
Hay dos herramientas integradas en Firefox que son útiles para experimentar con JavaScript: **la Consola Web y Borrador**.

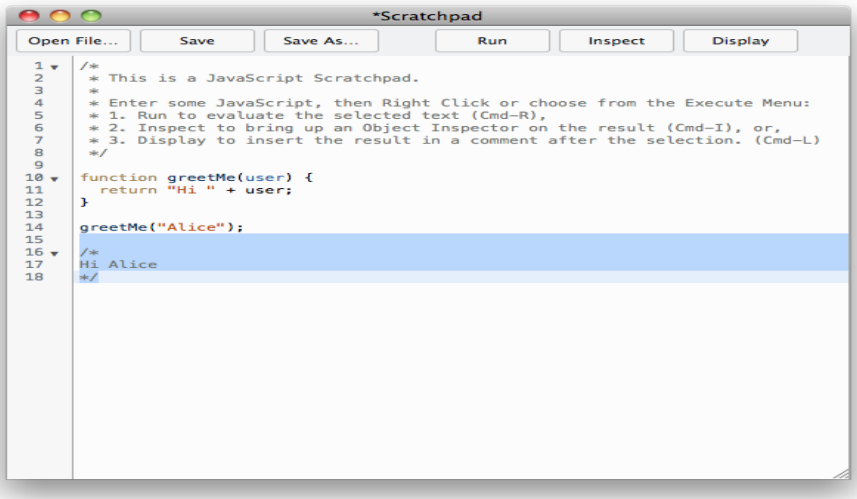
La Consola Web

La Consola Web muestra información sobre la página web que se está ejecutando en ese momento, y también incluye la línea de comando que puedes usar para ejecutar expresiones JavaScript en la página actual.
Para abrir la Consola Web, seleccionar "Consola web" desde el menú "Desarrollador web" que se encuentra en el menú "Herramientas" en Firefox. Aparece en la parte inferior de la ventana del navegador. A lo largo de la parte inferior de la consola esta la línea de comando que puede utilizar para ejecutar JavaScript, y el resultado aparece en el panel de arriba:



Borrador

La Consola Web es genial para ejecutar sencillas líneas en JavaScript, pero aunque se pueden ejecutar múltiples líneas, no es muy conveniente para esto, y no puedes guardar tus ejemplos de código usando la Consola Web. Así que para los ejemplos más complejos Borrador es la mejor herramienta.
Para abrir Borrador, selecciona "Borrador" desde el menú "Desarrollador Web", que se encuentra en el menú de "Herramientas" en Firefox. Abre una nueva ventana y es un editor que puedes usar para escribir y ejecutar JavaScript en el navegador. Puedes también guardar los scripts en el disco y cargarlos desde el disco.
Si eliges "Mostrar", el bloque de su código es ejecutado en el navegador y el resultado es insertado de nuevo dentro del bloque como un comentario:



Hola Mundo

Para empezar a escribir con JavaScript, abre la Consola Web o el Borrador y escribe tu primer código JavaScript "Hola Mundo" To get started with writing JavaScript, open the Web Console or the Scratchpad and write your first "Hello world".

```
//Declaramos una función llamada saludar, que lleva
//'nombre' como argumento.
function saludar(nombre) {
 return "Hola " + nombre; //Cuando se llama, esta función devuelve "Hola " y el nombre que se le ha pasado como argumento.
}

saludar("Usuario"); // Nos devolverá "Hola Usuario".
```