Nivelación

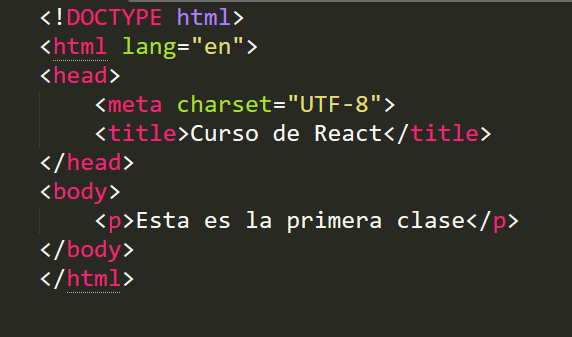
# Nivelación: HTML, CSS y JS

**React Js**

React JS es una biblioteca para desarrollo web, por lo cual debemos contar con conocimientos mínimos sobre los lenguajes que el navegador web interpreta.

# HTML

HTML es un lenguaje de etiquetas, el cual dará la estructura para nuestras páginas web.



**HTML 5**

HTML 5 es una nueva versión de diversas especificaciones, entre las que se encuentran:

* HTML 4
* XHTML 1
* CSS Nivel 2
* DOM Nivel 2 (Document Object Model)

**¿Cuáles son las novedades de HTML 5?**

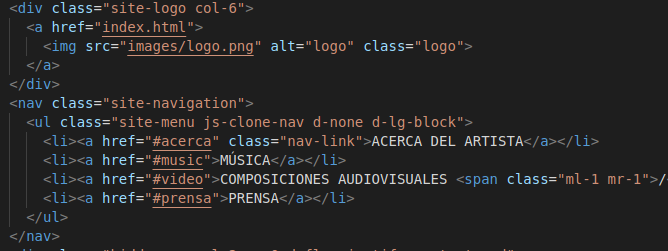
HTML 5 incluye novedades significativas en diversos ámbitos. Este nuevo estándar supone mejoras en áreas que hasta ahora quedaban fuera del lenguaje y para las que se necesitaba utilizar otras tecnologías:

* Estructura del <body>
* Etiquetas para contenido específico
* Canvas
* Bases de datos locales
* Web Workers
* Aplicaciones web Offline
* Geolocalización

**Un poco más de HTML**

Ahora que ya entendemos qué son HTM y HTML 5, comencemos a ver un poco

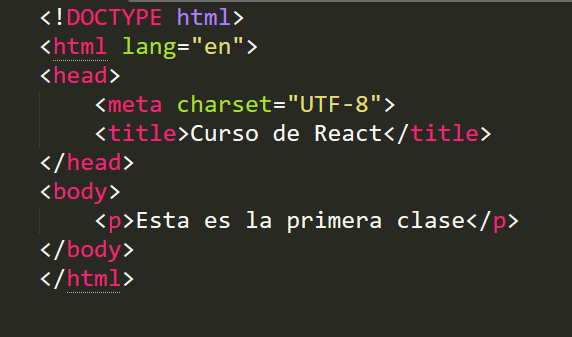
más de estructuración en HTML:



**Doctype**

DOCTYPE no es una etiqueta, sino una instrucción para indicar al navegador qué versión de HTML vamos a utilizar.

El DOCTYPE mostrado en nuestro ejemplo anterior es del estándar HTML 5.



**Metadatos del documento**

* **<title>:** define el título del documento, el cual se muestra en la barra de título del navegador o en las pestañas de página.
* **<base>**: define la URL base para las URLs relativas en la página.
* **<link>**: utilizada para enlazar JavaScript y CSS externos.
* **<meta>**: sirve para aportar información sobre el documento. Entre otras cosas con esta etiqueta definimos la codificación que tendrá nuestro archivo, los mismos pueden ser:

-UTF-8

-ANSI: es el formato estándar de codificación de archivos utilizados en el Bloc de notas.

**Scripting**

**<script>**: define tanto un script interno como un enlace hacia un script externo. El lenguaje de programación es JavaScript

**<noscript>:** define un contenido alternativo a mostrar cuando el navegador no soporta scripting.

**Noscript**

El lenguaje HTML define la etiqueta <noscript> para mostrar un mensaje al usuario cuando su navegador no puede ejecutar JavaScript. El siguiente código muestra un ejemplo del uso de la etiqueta <noscript>:

**<head> ... </head>**

**<body>**

**<noscript>**

**<p>Bienvenido a Mi Sitio</p>**

**<p>La página que estás viendo requiere para su funcionamiento el uso de JavaScript.**

**Si lo has deshabilitado intencionadamente, por favor vuelve a activarlo.</p>**

**</noscript>**

**</body>**

La etiqueta <noscript> se debe incluir en el interior de la etiqueta <body> (normalmente se incluye al principio de <body>). El mensaje que muestra <noscript> puede incluir cualquier elemento o etiqueta XHTML.

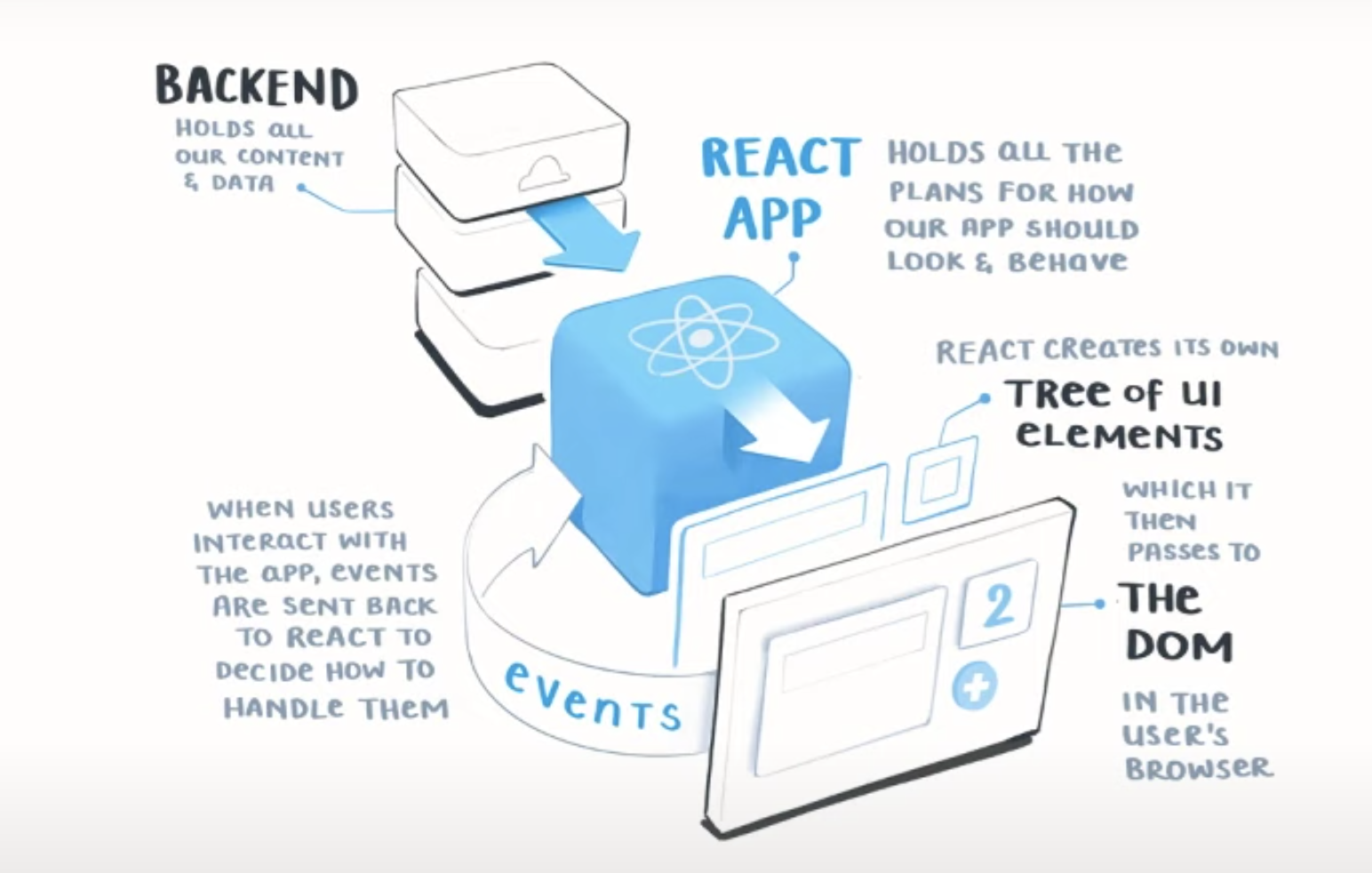
Fuente: <https://uniwebsidad.com/libros/javascript/capitulo-1/etiqueta-noscript>

# DOM

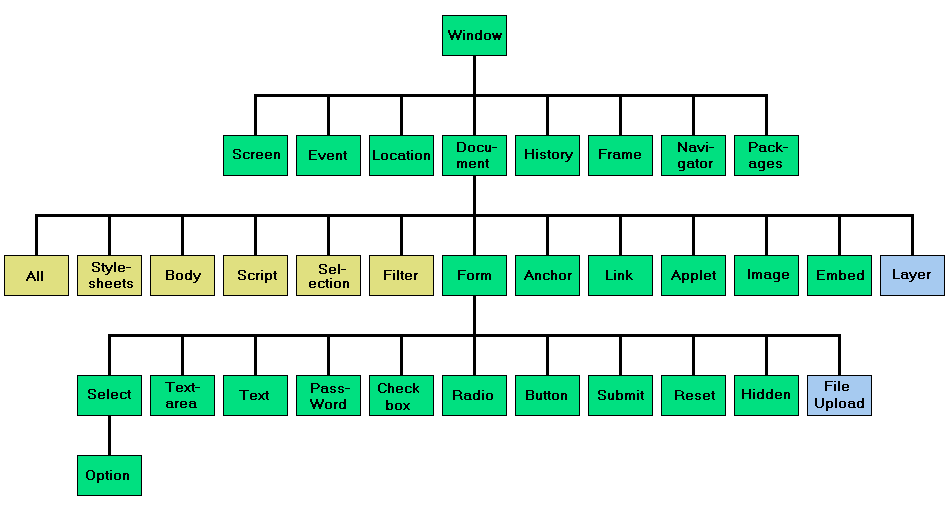
DOM (Document Object Model o modelo de objetos del documento) nos sirve para acceder a cualquiera de los componentes que hay dentro de una página.

Por medio del DOM podremos controlar al navegador en general y a los distintos elementos que se encuentran en la página.

**DOM como metáfora**

****

**Jerarquía del DOM**

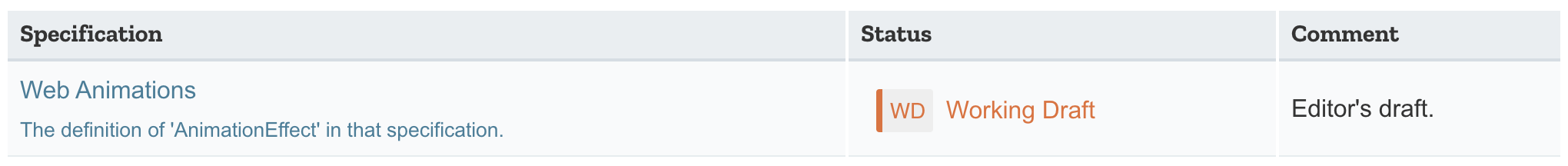
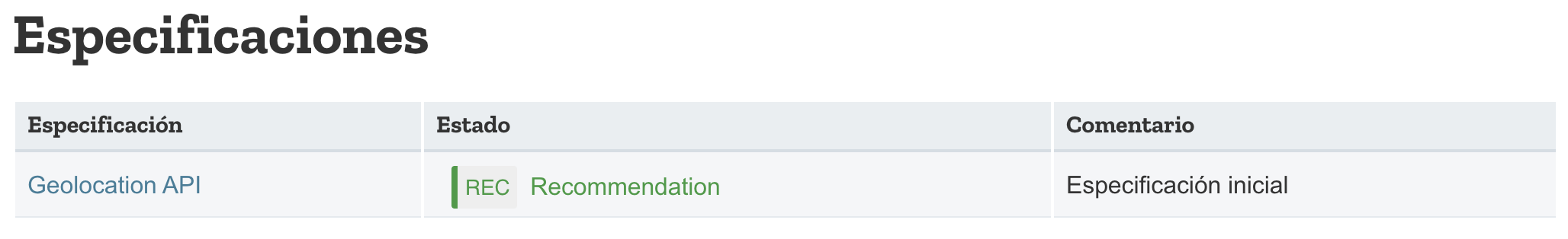
****

**Un mundo de API’S**

Siempre es recomendable avanzar con criterio y ver qué funcionalidades

usamos del Navegador, ya que hay centenas de API’s, pero:

* No todas están listas para producción.
* No todas tienen soporte en todos los navegadores.

****

**Fuente:** [**https://developer.mozilla.org/es/docs/Web/API**](https://developer.mozilla.org/es/docs/Web/API)

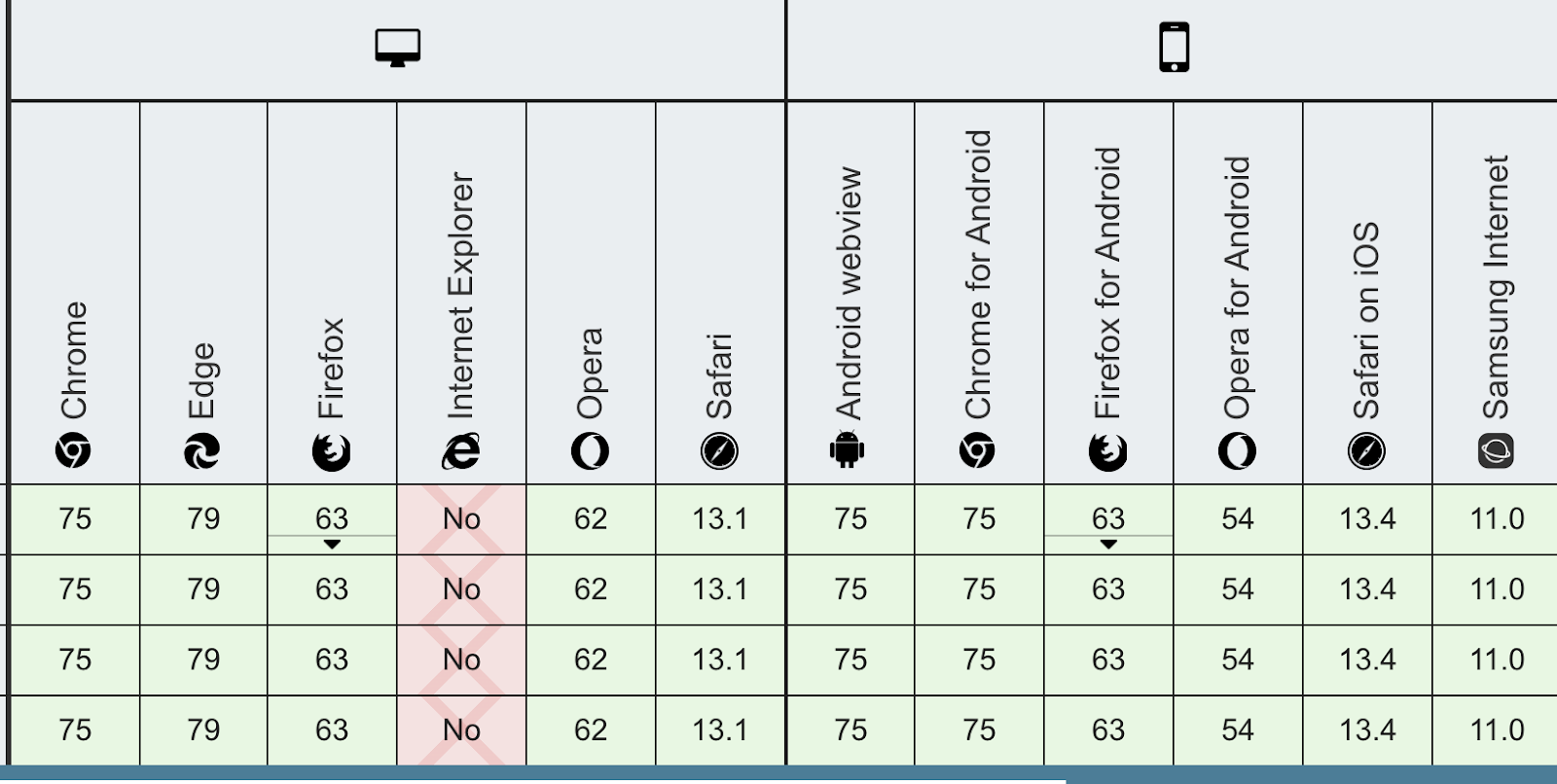
**Un mundo de API’S**

Todo depende del contexto del trabajo y el target de usuarios / plataformas.

Puedo emocionarme para luego enterarme que algunos usuarios no pueden utilizar la experiencia.

Sugerencia: https://caniuse.com

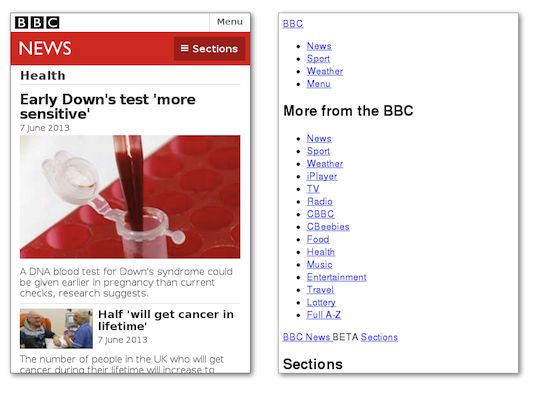
Intenta chequear si puedes utilizar ES6 en cualquier navegador



Fuente: <https://developer.mozilla.org/es/docs/Web/API>

# CSS

CSS (cascading style sheets - hojas de estilo en cascada) es un lenguaje de diseño gráfico con el cual podremos dar estilos (diseño, colores, márgenes) a nuestras webs desarrolladas con HTML. Veremos la sintaxis básica de CSS



**Con CSS. Sin CSS.**

**Incluir hoja de estilo en HTML**

Para incluir una hoja de estilo en nuestro documento

HTML podemos utilizar el elemento <link>.

Dentro del archivo CSS externo incluiremos las sentencias correspondientes, como lo hacemos con un archivo css en línea.

**Por ejemplo:**

****

**Clases CSS**

Podemos seleccionar uno o más elementos html a través de su clase o su atributo “id” en CSS. O todos los elementos de un tipo Modos básicos de “seleccionar/capturar” elementos por su:

| Tipo de elemento (div, ul, li, nav, input) | div { background: lightgray; } |
| --- | --- |
| ID (usando # adelante) | #nombre { background: coral; } |
| Clase (usando punto inicial) | .apellido |

**Clases CSS**

****

**Error clásico:**

Los # y los puntos “.” de los selectores se usan en la hoja de estilos, no en el elemento

****

# Javascript

**¿Qué es?**

JavaScript es el lenguaje de programación web por excelencia.

Decimos que se trata de un lenguaje de programación interpretado.

Su uso más conocido es del lado del cliente (client-side), corriendo en el navegador web, permite mejoras en la interfaz de usuario.

React está desarrollado en JS, por eso lo llamamos reactjs.

**¡Pregunta rápida!**

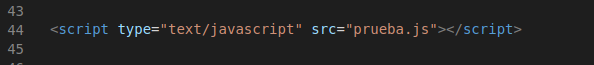
“En una propuesta laboral decía que necesitaba saber Vanilla JS, ¿Dónde lo puedo aprender?”

¡Tal como el helado más común, el saborizante más conocido, o el clásico de los clásicos!

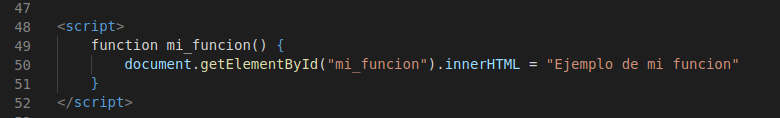
¡JS sin aditivos ni conservantes!

(mejor dicho, sin librerías externas)

**Incluir el archivo JS en nuestro HTML**

****

**Esto debe ser incluido en el head del html. También podemos introducir nuestro código javascript dentro de la etiqueta <script></script>. Por ejemplo:**

****

**Tipos de ejecución**

**Ejecución directa**

Es el método de ejecución de scripts más básico. En este caso se incluyen las instrucciones dentro de la etiqueta <script>, y cuando el navegador lee la página y encuentra un script va interpretando las líneas de código y las va ejecutando una después de otra.

**Tipos de ejecución**

**Respuesta a un evento**

Los eventos son acciones que realiza el usuario. Los programas como Javascript están preparados para atrapar determinadas acciones realizadas, en este caso sobre la página, y realizar acciones como respuesta.

Por ejemplo, la pulsación de un botón, el movimiento del ratón o la selección de texto de la página.

**Variables**

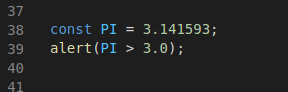
La manera estándar de JS de almacenar valores en memoria es a través del uso de variables.

****

**Definir una constante**

Desde ES6 (ECMA 2015) podemos definir una constante con la palabra reservada const

Las const no pueden cambiar la referencia con la que fueron inicializadas.



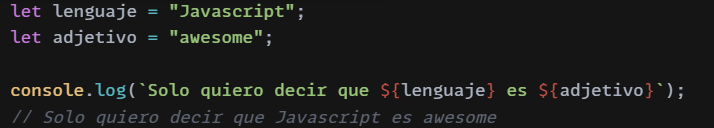
**DOM**

DOM (Document Object Model o modelo de objetos del documento) nos sirve para acceder a cualquiera de los componentes que hay dentro de una página.

Por medio del DOM podremos controlar al navegador en general y a los distintos elementos que se encuentran en la página.

**LET**

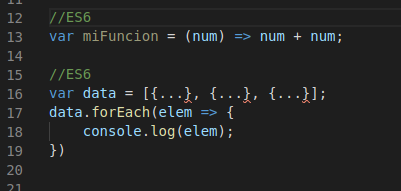
Podemos utilizar let en lugar de var a la hora de declarar una variable, cuando queremos que ésta sólo sea accedida de manera local en determinado ámbito. Por ejemplo:

****

**Función ARROW**

En ambos ejemplos se sustituye el uso de la palabra reservada function, dando simplicidad al código.

**(parametro1,parametro2,…,parámetro n) => {Definición de la función}**

****

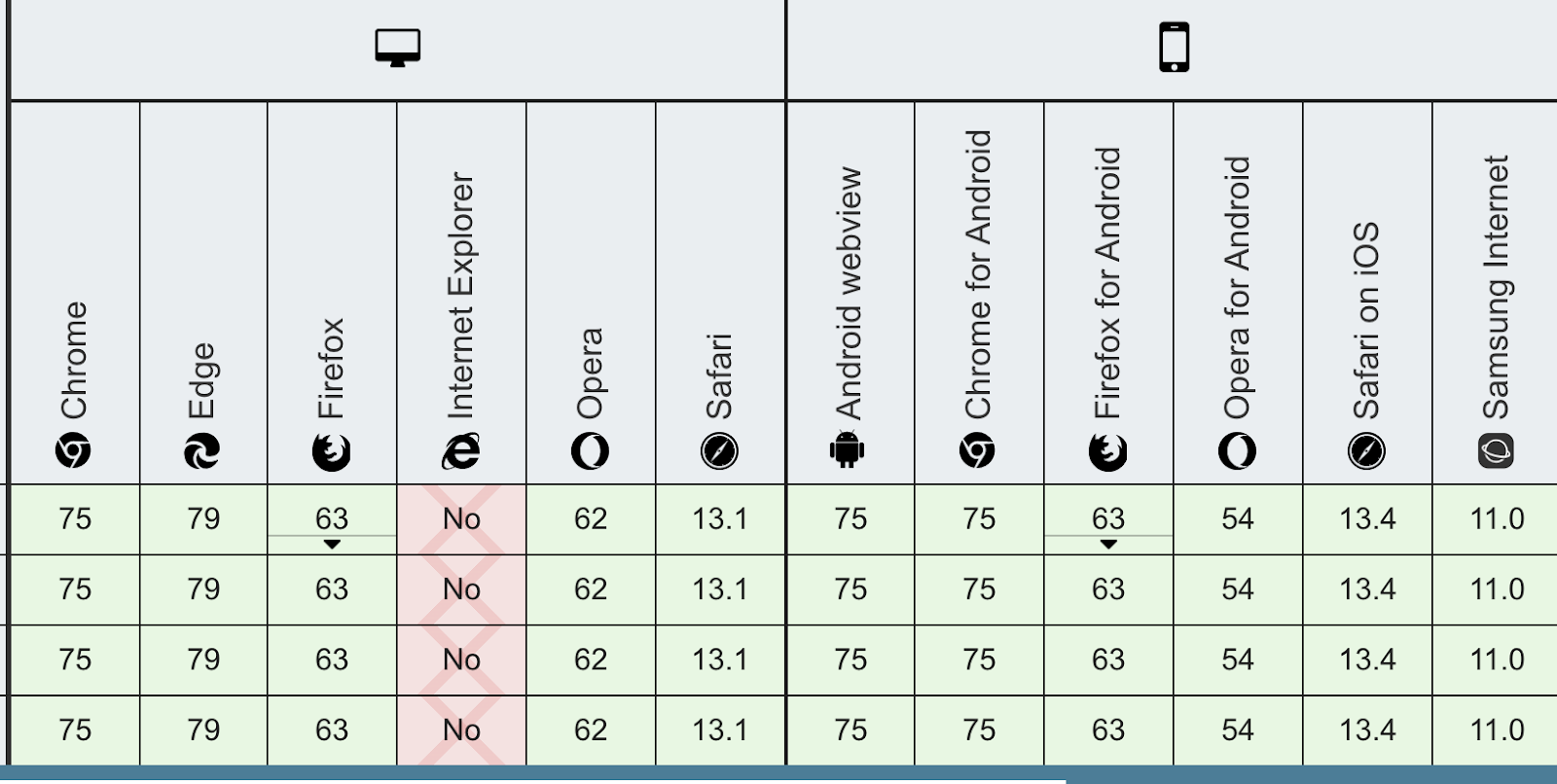
**Un mundo de API’S**

Todo depende del contexto del trabajo y el target de usuarios / plataformas.

Puedo emocionarme para luego enterarme que algunos usuarios no pueden utilizar la experiencia.

Sugerencia: https://caniuse.com

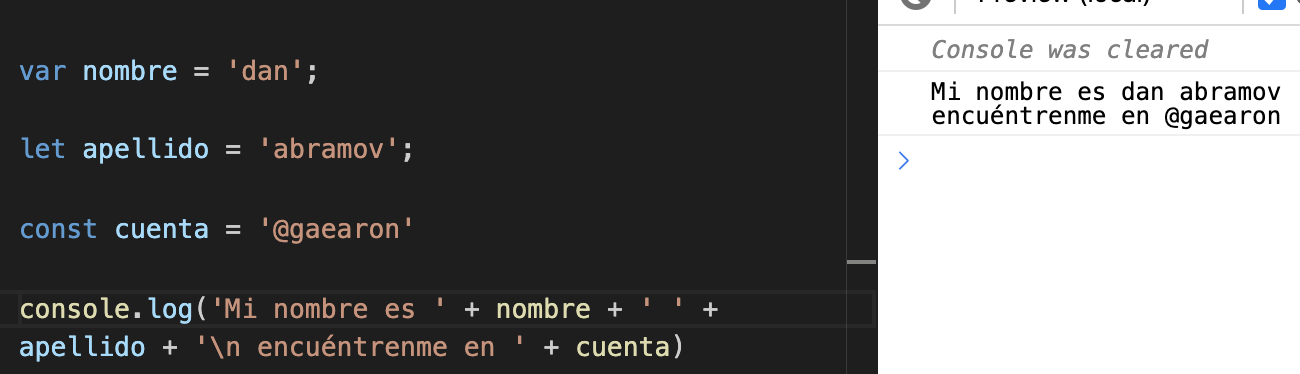
Intenta chequear si puedes utilizar ES6 en cualquier navegador

****

Fuente: <https://developer.mozilla.org/es/docs/Web/API>

**Template Strings**

Antes de ES6, creábamos mensajes compuestos de la siguiente manera:

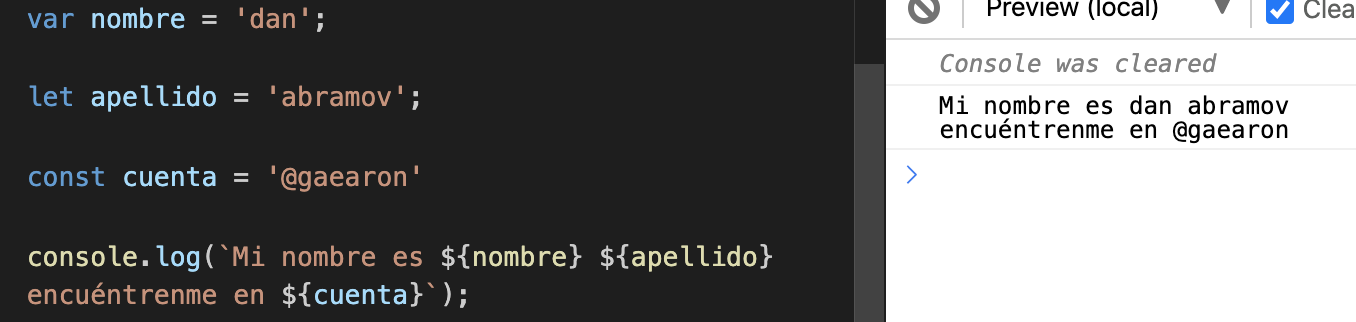


Problemas: ¿cuáles detectan?

Agregar espacios manuales entre concatenaciones ‘+’, dificultad para modificar texto, multiline explícito \n, poca legibilidad.

**Template Strings**

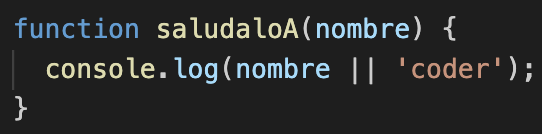
Con ES6 podemos interpolar strings de una forma más sencilla, a como estábamos haciendo hasta ahora. Miremos el ejemplo:



**Valores por defecto: antes y hoy**

Antes teníamos que comprobar si la variable ya tenía un valor, para dar una respuesta por defecto.

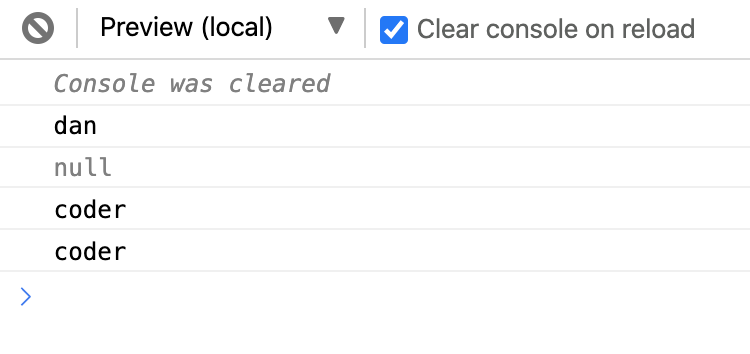
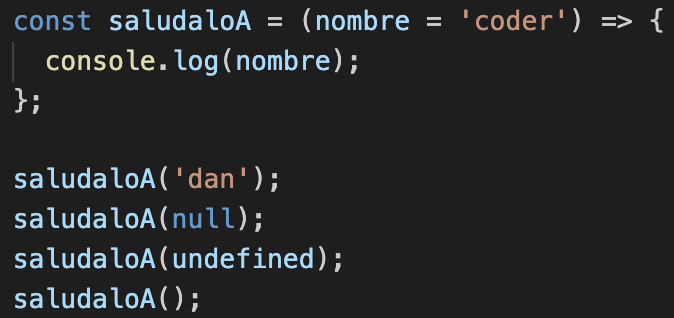
En cambio, hoy podemos usar otra sintaxis, aunque tenga algunas diferencias de comportamiento.



**Valores por defecto: ¡CUIDADO!**

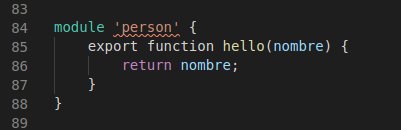
En muchas convenciones null es considerado como algo explícito de un valor que es la nada. Mientras que al undefined se lo considera como un valor que aún no se puede inferir que es, cuando no pasamos un parámetro será undefined (indefinido)

Dicho esto, si le paso null a function, ¡el valor por defecto no se aplicará!

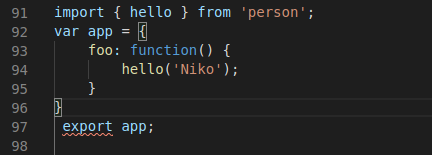


**Módulos**

Ahora JavaScript se empieza a parecer a lenguajes como Python o Ruby. Llamamos a las funciones desde los propios scripts, sin tener que importarlos en el HTML si usamos JavaScript en el navegador.

****

**Y para importar en otro fichero:**

****

# REACT JS: Los inicios

**React Js**

React JS fue creada por Jordan Walke, un ingeniero de software en Facebook, inspirado por los problemas que tenía la compañía con el mantenimiento del código de los anuncios dentro de su plataforma.

React intenta ayudar a los desarrolladores a construir aplicaciones que usan datos que cambian todo el tiempo. Su objetivo es ser sencilla, declarativa y fácil de combinar.

**Sitios basados en REACT JS**

****