



Desarrollo web en entorno cliente





Tema 1: Introducción al desarrollo web en el lado del cliente

1. Página o aplicación web

Tanto una página web como una aplicación web son aplicaciones software alojadas en un servidor de tal manera que puedan ser accedidas a través de Internet, o de una red local, mediante un navegador.

Se trata de aplicaciones que no requieren de instalación previa (más que la del navegador que se vaya a utilizar) y que, de forma general, son multiplataforma y multidispositivo.

La diferencia entre aplicación web y página web radica en que, con la primera, normalmente, se puede interactuar e incluso cambiarla, mientras que el contenido de la página web suele ser, únicamente, información estática. Por lo tanto, una página web es una aplicación web con menor funcionalidad.

SPA

Una SPA (Single Page Application), es una página o sitio web con un solo archivo HTML donde está contenida toda la página.

En una SPA sólo se carga la página de inicio (la única que existe) que se va modificando, y cambiando sus datos, como respuesta a la interacción del usuario con la misma. Para obtener los nuevos datos se realizan peticiones al servidor, siendo la respuesta de éste fragmentos de código HTML a incluir en determinadas partes, o datos (JSON, XML, ...) que modifican la página mostrada.

Una SPA se crea en JavaScript, ya que se trata de una aplicación ejecutada del lado del cliente. Por supuesto será necesario incluir el código HTML y las propiedades CSS necesarias.

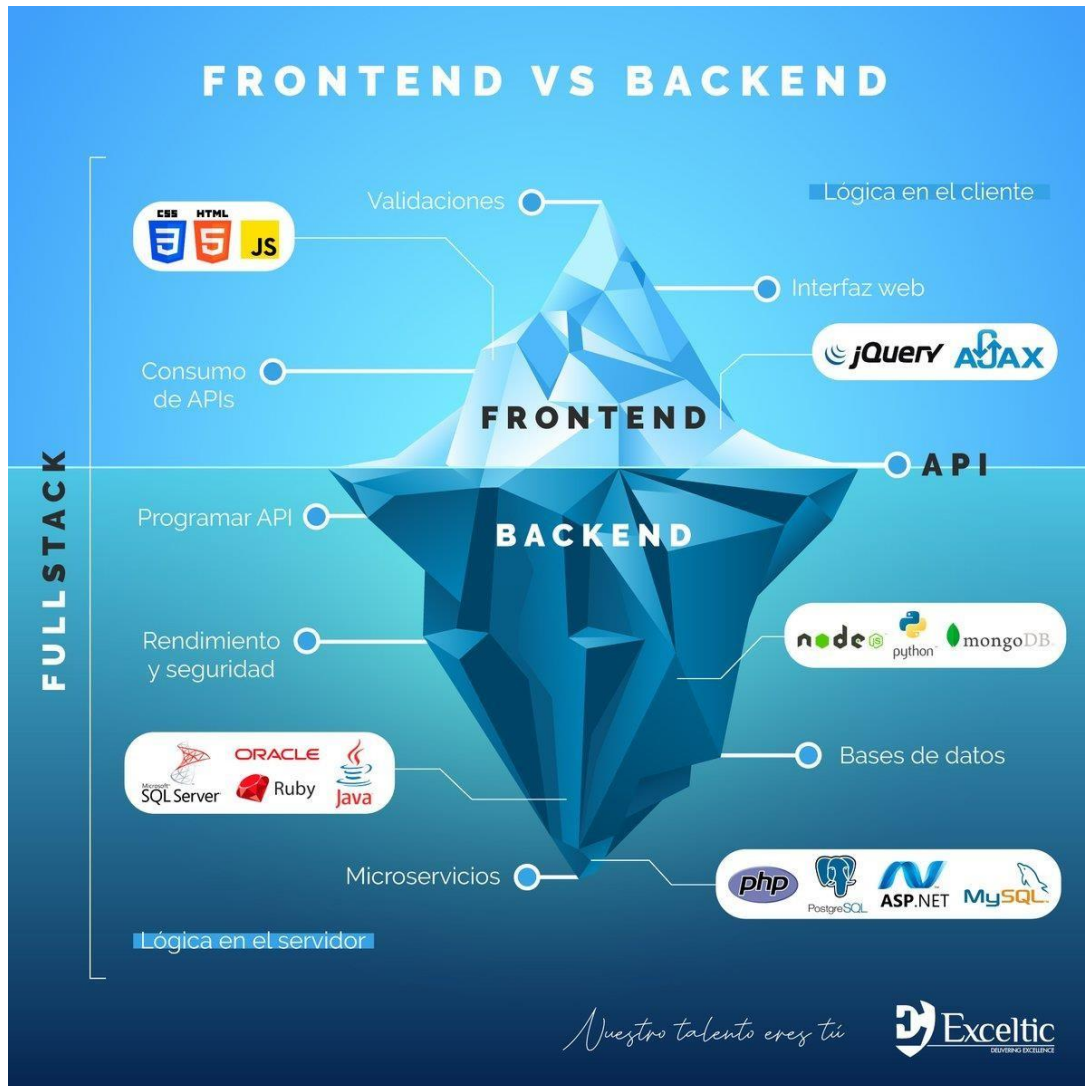
Ajax es la base para construir SPAs, que permiten al usuario interactuar con una aplicación web como si se tratara de una aplicación de escritorio. Los enlaces internos de la página no supondrán la carga de otra página completa, sino que simplemente se actualizará la existente con lo devuelto por el servidor tras una petición. Por ejemplo, GMail es una SPA.

El concepto clave en una SPA es la vista. Solo hay una página, pero se pueden encontrar multitud de vistas.

El usar o no una SPA dependerá de muchos factores, por ejemplo, si se trata de un portal muy extenso puede que no sea lo más conveniente.

Front-end y Back-end

En el desarrollo de una aplicación web, se diferencian claramente dos partes:



El front-end es el conocido como lado del cliente y se centra en el usuario final del sitio web, en toda la interacción que puede hacer con la aplicación y todo lo que va a ver mientras navega. Para ello se utiliza HTML y CSS, lenguajes de script como JavaScript, plugins y APIs (Application Programming Interfaces) del navegador, AJAX, etc. Una vez conocidos los lenguajes de programación, marcas o script necesarios, el uso de frameworks o librerías facilita la creación de aplicaciones web y aporta funcionalidades. Algunos frameworks utilizados del lado del cliente son Angular, React, Vue, Bootstrap o Sass.

Por otro lado, el back-end es el lado del servidor, encargado de que todo lo que está detrás de un sitio web funcione correctamente. Es el encargado de la lógica de la aplicación, que incluye la conexión con el servidor, conexión con base de datos, desarrollo de funcionalidad no accesible por el usuario, etc. Algunos de los lenguajes utilizados en el lado del servidor son PHP, Python, .NET, SQL o incluso JavaScript. El desarrollo del backend debe tener en cuenta, por ejemplo, una buena capacidad de respuesta y una velocidad óptima del sitio. También existen frameworks para el back-end como Django, Laravel o ExpressJS.



2. Tecnologías del lado cliente

Las tecnologías utilizadas del lado del cliente son, básicamente, las siguientes (las 3 primeras son los llamados estándares web):

- **HTML:** Lenguaje de marcas que se utiliza para dar estructura a los contenidos del sitio web. La estructura y elementos creados con HTML serán interpretados por el navegador de forma estática.
- **CSS:** Mediante CSS se establece la presentación visual de los elementos y estructura creados con HTML, permite que se adapten a dispositivos diversos, etc. Separando el contenido HTML y el contenido CSS en dos archivos diferentes se consigue diferenciar contenido de presentación facilitando el desarrollo y el mantenimiento posterior del sitio web.
- **JavaScript:** Lenguaje de programación que aporta interactividad a las aplicaciones web. El uso de JavaScript hace que los sitios web sean más dinámicos, permitiendo que no haya que refrescar las páginas para realizar ciertas acciones y, así, se actualice la aplicación de forma rápida y automática.
- **AJAX:** Utilizado para integrar parte cliente y parte servidora con un proceso asíncrono, es decir, se trata de un mecanismo que posibilita las peticiones asíncronas al servidor desde una aplicación web.

3. Navegadores y herramientas útiles

3.1. Navegador web

Un navegador es un software para el acceso a una red, como Internet o una red local, que permite visitar sitios web e interactuar con ellos. Dichos sitios web contendrán hipervínculos o enlaces que dirigen a otras partes del mismo sitio web o de otro distinto. La posibilidad de acceder desde unas páginas a otras es lo que se denomina navegación.

Los recursos o archivos a los que se accede mediante la navegación tienen un identificador único denominado URL (Uniform Resource Locator). El formato de una URL es:



Si no se especificara el directorio, se toma el directorio raíz. Y en caso de que se omita el fichero, tomará alguno de los nombres por defecto como, por ejemplo, `index.html`.

El protocolo que permite la comunicación entre el servidor web y el navegador será, normalmente, `http` o `https`, aunque la mayoría de los navegadores soportan otros como `FTP`.

Por tanto, la función principal de un navegador web es obtener los documentos HTML solicitados por el usuario junto con el resto de imágenes, vídeo, archivos de sonido, etc. a los que referencia y permitir el acceso a los mismos.

Los navegadores más antiguos soportaban versiones no estandarizadas de HTML lo que provocaba grandes problemas de compatibilidad e interoperabilidad. Hoy en día, la mayoría de los navegadores los estándares HTML y XHTML, siendo cada vez más fácil que las páginas se vean igual en todos ellos. Para eso es muy importante que cuando se desarrolla una aplicación web ésta sea lo más estándar posible.

Alguno de los navegadores que podemos encontrar son:

- **Google Chrome**: En la actualidad es el navegador líder del mercado. Está desarrollado por Google, tiene versiones para diferentes plataformas y sistemas operativos, y está disponible de forma gratuita.
 - **Mozilla Firefox**: Se trata de un navegador multiplataforma, libre y de código abierto desarrollado por la Fundación Mozilla.
 - **Microsoft Edge**: Navegador que sustituye al antiguo Explorer, está disponible tanto para PC como para dispositivos móviles. Dispone de un motor de renderizado de código abierto.
-



- Safari: Se trata del navegador propietario de la compañía Apple, tanto para equipos de escritorio como para dispositivos móviles.
- Opera: Navegador incluido en una suite bastante completa. Está disponible para multitud de plataformas.

EJERCICIO 1

- Busca la diferencia entre el protocolo http y el protocolo https.
- Busca un esquema básico que indique como se lleva a cabo la comunicación web entre cliente y servidor.
- ¿Qué es un motor de renderizado?
- Instala en tu equipo, si no están instalados, los navegadores Chrome, Firefox y Edge.

3.2. Herramientas

Para el desarrollo de aplicaciones web son muy útiles algunas herramientas que los navegadores ponen a disposición de los usuarios, bien ya incorporadas o a través de extensiones o plugins, tanto para el desarrollo como para la depuración de código. De los navegadores nombrados con anterioridad, uno de los que más herramientas ofrece es Firefox y será el que se tome como ejemplo de aquí en adelante.

Para acceder a las herramientas de desarrollo pulsar la tecla F12. En Firefox se puede abrir mediante la combinación Ctrl+Shift+K, mientras que en Chrome y Edge será Ctrl+Shift+i y en Opera Ctrl+Shift+J.



En las distintas pestañas de las herramientas de desarrollo se pueden destacar:

- Inspector de elementos: Permite inspeccionar estilos y árbol DOM, pudiéndose modificar cualquier propiedad para ver el resultado de forma inmediata.
- Red: Registros de peticiones http.
- Consola:



- Información CSS en la consola: Errores y análisis de las hojas de estilo.
- JS: En la consola saldrán también errores y análisis JavaScript, pudiéndose utilizar la línea de comandos de la misma para probar sentencias JS.
- Registros: mensajes enviados al objeto “window.console”.
- Errores, Advertencias: Permite visualizar cualquier error o warning que se produzca en la ejecución.
- Depurador: Permite depurar código javascript, poniendo puntos de interrupción.
- Editor de estilos: muestra las propiedades CSS aplicadas, pudiéndose comprobar cambios en las mismas.

Por otro lado, es muy recomendable validar el código en base a los estándares establecidos:

- <https://jigsaw.w3.org/css-validator/>: Para validar el código CSS.
- https://validator.w3.org/#validate_by_input: Para validar código HTML, XHTML, etc.
- <https://codebeautify.org/jsvalidate>: Para validar código JavaScript.

La validación de código, ajustándolo a los estándares lo máximo posible, asegura en gran medida tanto su correcto funcionamiento como que el sitio web se vea de forma correcta en la mayoría de los navegadores.

Por último, pero no menos importante, aparte de validar el código hay que probar la aplicación abriéndola con diferentes navegadores. Si se quiere validar al máximo, existen herramientas online que permiten comprobar el aspecto del sitio en multitud de navegadores y no solo en los más usados.

EJERCICIO 2:

- Abre la consola en los distintos navegadores que tengas instalados y comprueba las diferencias que existen entre ellos.
 - Escribe, en la consola del navegador que hayas elegido para trabajar, un mensaje de alerta con javascript y comprueba que se ejecuta correctamente.
-

4. Entornos de desarrollo y control de versiones

Existen infinidad de entornos de desarrollo, de entre los cuales en esta asignatura se va a utilizar Visual Studio Code.

VSC, como otros IDEs, permite personalizar múltiples aspectos y ofrece distintos modos de trabajo. Por ejemplo, si se quiere actualizar el tema de color, basta con ir a *Archivo->Preferencias->Tema de Color* y seleccionar el que más guste.

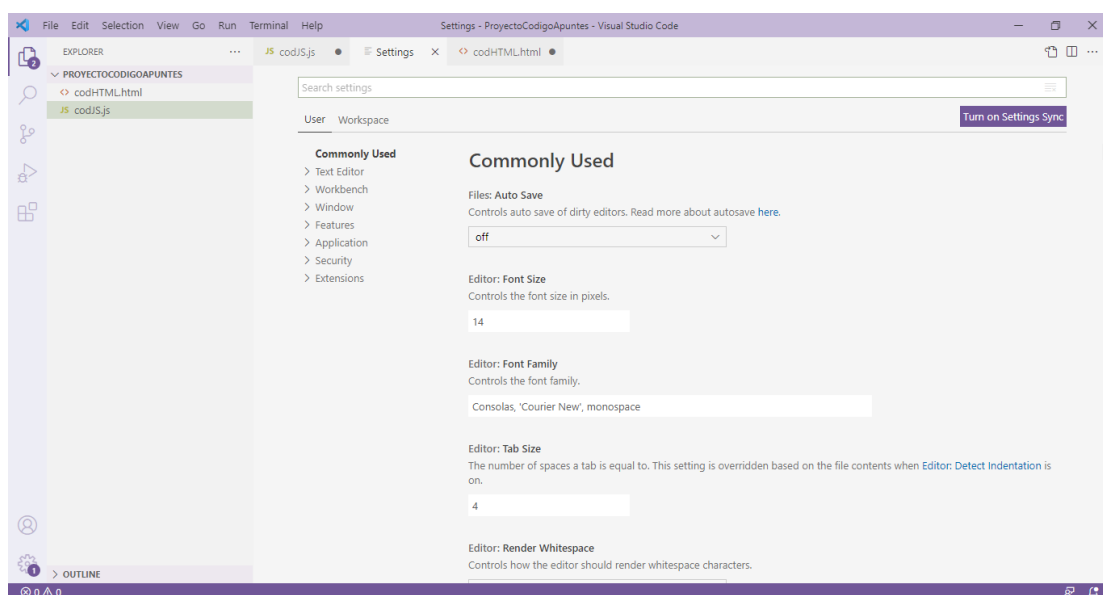
Otra característica de trabajar con VSC son las áreas de trabajo o workspace. VSC proporciona dos tipos:

- **Carpetas:** Se trata de una carpeta, que será la carpeta raíz del área de trabajo (suele ser la carpeta raíz de un proyecto), y todo su contenido. Para abrir una carpeta *File->Open Folder ...* Para cambiar de área de trabajo basta con seleccionar, de la misma manera, una carpeta distinta.
- **Workspace:** Opción que permite tener abiertas varias carpetas(proyectos) distintas a la vez. Una vez abierto un espacio de trabajo, se puede añadir otra carpeta con la opción *File->Add Folder to Workspace*. Es posible guardar el área de trabajo con la opción *File->Save workspace as*.

Es posible establecer una configuración VSC a dos niveles:

- **configuración de usuario:** estas preferencias se aplican a cualquier documento editado, no importando el área de trabajo al que pertenezca.
- **configuración de área de trabajo:** estas preferencias se aplican únicamente a los ficheros contenidos en un área de trabajo o en una determinada carpeta o en sus subcarpetas.

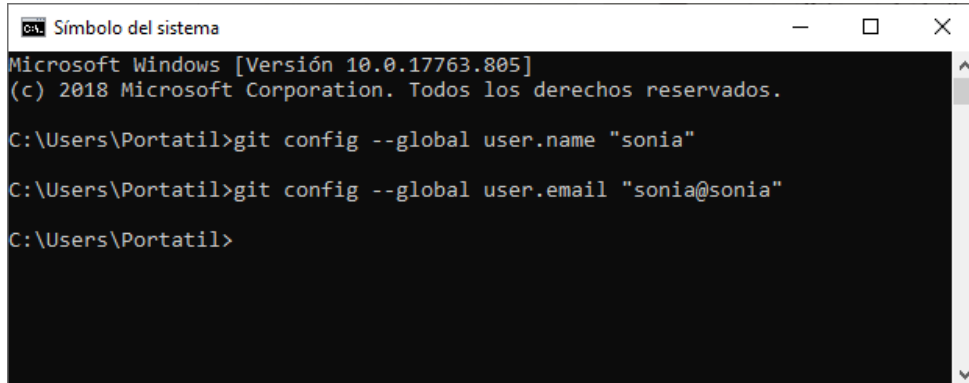
Para actualizar las preferencias de configuración: *File->Preferences->Settings*:



GIT desde VSC

Para hacer uso de un repositorio Git desde VSC, primero hay que instalar Git:

- <https://git-scm.com/downloads>: Descargar la versión que se corresponda con el Sistema Operativo utilizado.
- Indicar usuario y correo electrónico que se van a utilizar para realizar todas las operaciones con Git:



```
Símbolo del sistema
Microsoft Windows [Versión 10.0.17763.805]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Portatil>git config --global user.name "sonia"

C:\Users\Portatil>git config --global user.email "sonia@sonia"

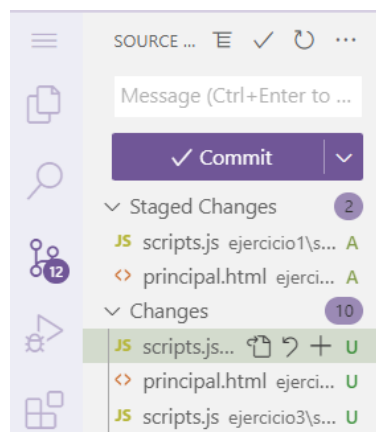
C:\Users\Portatil>
```

Una vez instalado, ya se puede clonar un repositorio y trabajar con él. Para ello:

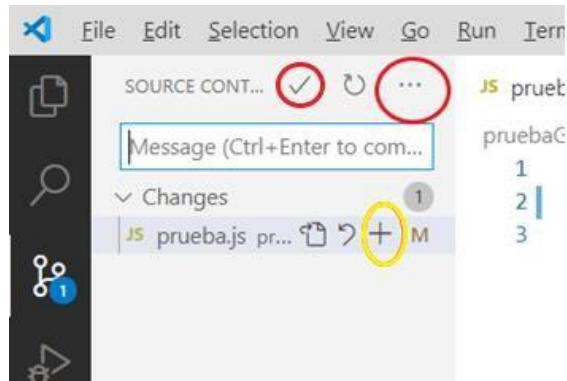
- Abrir la paleta de comandos de VSC y buscar la opción Git clone.
- Introducir la url del repositorio.
- Introducir la carpeta que actuará como repositorio local.

Para hacer commit:

- Acceder a la pestaña de control de código:
- En esa pestaña se pueden ver los cambios realizados. Los ficheros estarán marcados como untracked, modificados, añadidos, etc. Mediante el símbolo + de cada uno de ellos se pueden añadir éstos al índice (En amarillo en la siguiente imagen).

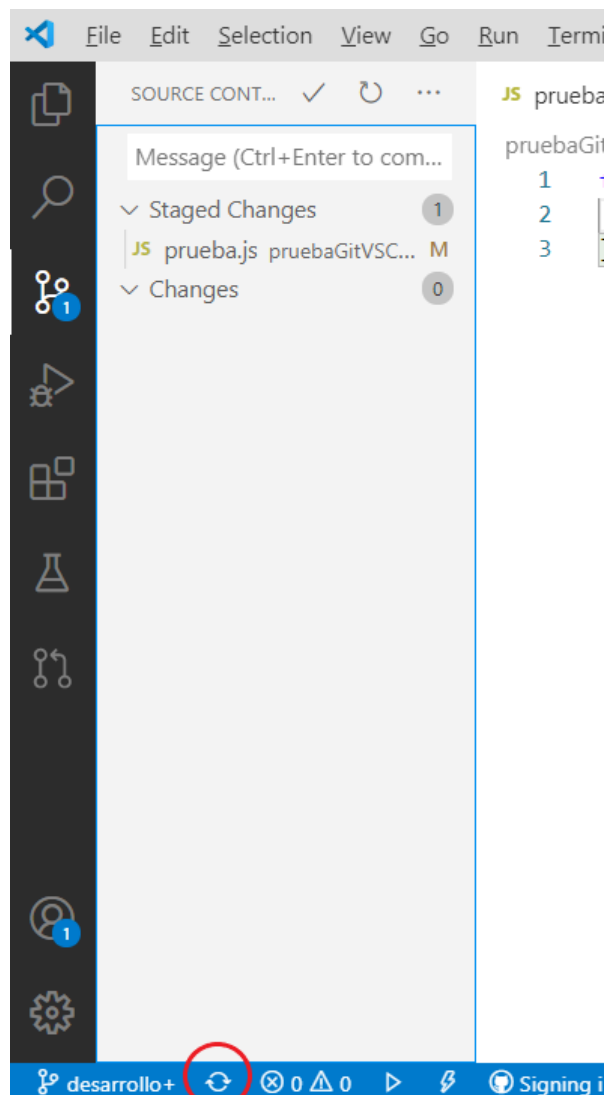


- Una vez añadidos al índice, se pulsa el botón de commit o se va a las opciones de commit del menú superior derecho de la pestaña (Opciones marcadas en rojo). La aplicación solicitará un comentario para el commit que se va a realizar:



Para persistir/descargar cambios en/desde el repositorio remoto:

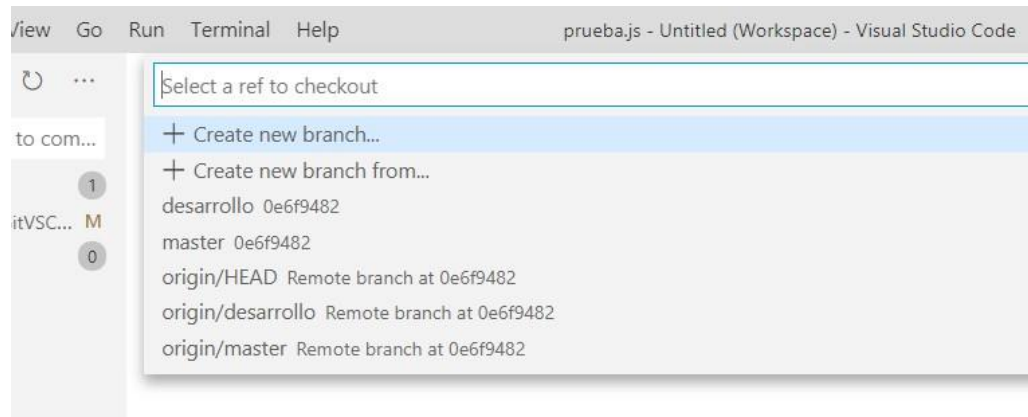
- En el mismo menú superior derecho se pueden encontrar las opciones pull y push.
- También pulsando sobre la opción de Sincronizar Cambios ubicada en la barra inferior del editor. En este caso, se realizará pull y push a la vez:





Para crear una rama o cambiar a una existente:

- En la barra inferior, a la izquierda, aparece la rama sobre la que se está trabajando. Pulsando sobre ella, se desplegará un menú en la parte superior que permitirá cambiar a una rama existente o crear una nueva:



Se puede hacer uso también de alguna de las extensiones existentes, como por ejemplo “GitHub Pull Requests and Issues” que permite la integración de VSC con GitHub.

EJERCICIO 3:

- Instala Visual Studio Code en tu equipo.
 - Instala Git en tu equipo y configura tu usuario. Pon tu nombre y primer apellido como usuario.
 - Crea un repositorio en GitHub llamado “Ejercicios DWEC-TuNombre”. A lo largo del curso, cada proyecto, práctica o ejercicio que vayas a desarrollar tendrá su carpeta en dicho repositorio.
 - Configura VSC para trabajar con tu repositorio de GitHub.
 - Crea un primer proyecto llamado “PruebaVSCConGitHub” en tu repositorio. Incluye un fichero de texto donde ponga “Repositorio configurado: Listo para empezar” y súbelo al repositorio remoto. Comprueba que se ha subido con el usuario que configuraste en tu equipo.
-

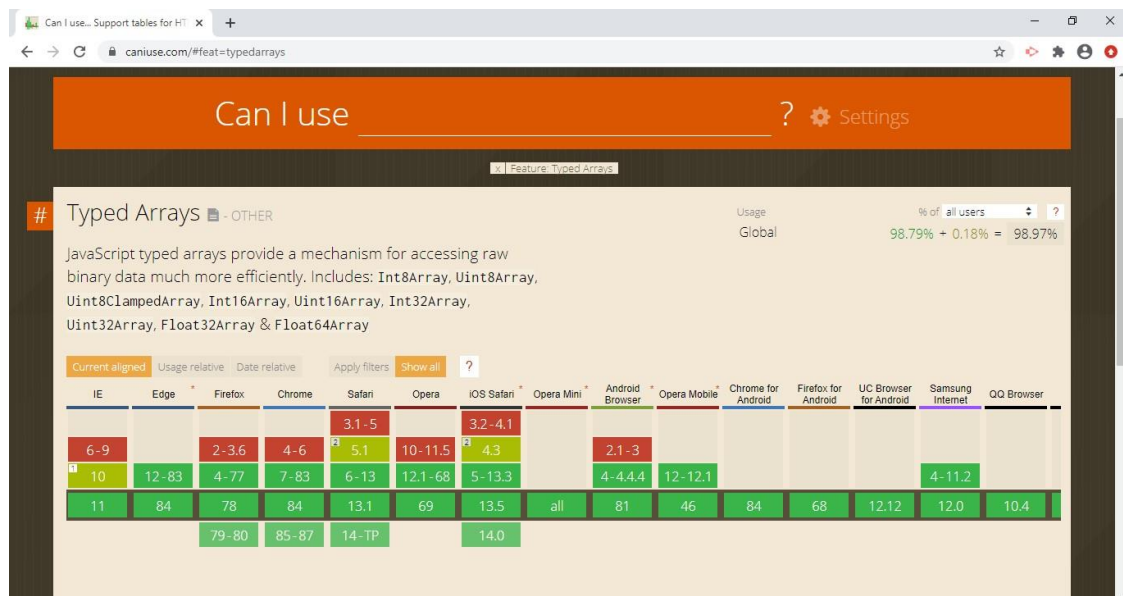
5. JavaScript

Como ya se ha indicado anteriormente, JavaScript permite a un sitio web reaccionar a las acciones del usuario, dotándolo de vida. Se utiliza para cambiar el contenido de una página de forma dinámica, modificar los atributos de un elemento de la misma, actualizar la apariencia de algún elemento o sección, validar datos de formularios antes de ser enviados al servidor, etc.

JavaScript se define como un lenguaje:

- interpretado, no compilado.
- que, generalmente, se ejecuta del lado del cliente.
- orientado a objetos basado en prototipos, pudiendo crearse objetos sin instanciarlos.
- con tipificación dinámica, es decir, con variables declaradas sin tipo de dato específico.

Los navegadores van adaptándose a las nuevas versiones de JavaScript (o HTML o CSS) con cierto retraso desde que éstos son publicados. Por ello, hay que tener cuidado en el momento de desarrollar un sitio web y asegurarse de la compatibilidad de los recursos usados. Para ello, es conveniente probar la aplicación en desarrollo en varios navegadores diferente, pero existen también páginas que facilitan el trabajo indicando la compatibilidad con los distintos navegadores de un recurso concreto. Por ejemplo, <https://www.caniuse.com> permite conocer la compatibilidad de un elemento concreto de Javascript, HTML5 o CSS3:

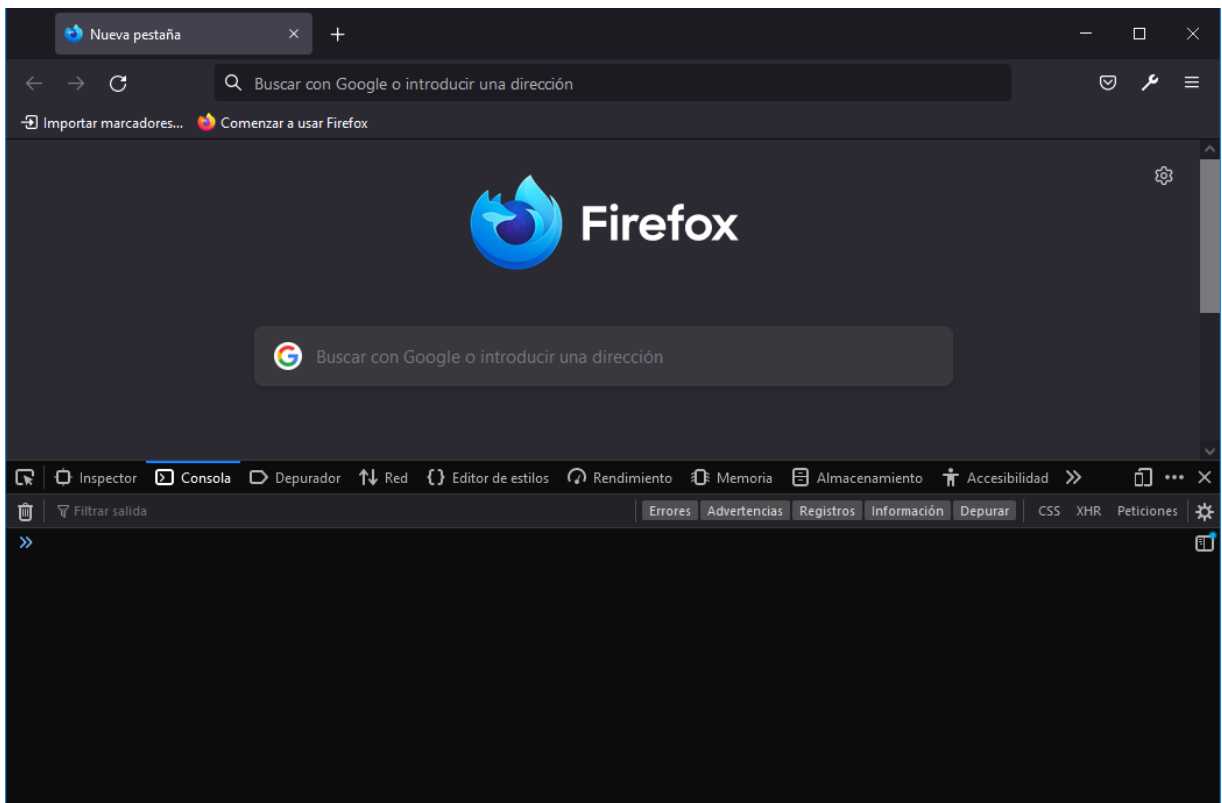




5.1. Consola del navegador

La consola del navegador va a permitir:

- probar instrucciones JavaScript viendo el resultado de la ejecución en el momento.
- establecer puntos de ruptura y así poder depurar el código.
- inspeccionar el valor de una variable concreta.
- Por ejemplo, en el navegador Firefox Developer Edition existe un modo de editor multilínea que permite probar líneas de código de manera más rápida.



5.2. JavaScript en VSC

Algunas de las extensiones que pueden ser útiles para empezar a programar con JavaScript son:

- JavaScript (ES6) code snippets: Permite trabajar con abreviaturas acelerando el desarrollo de código.
 - Prettier: para formatear el código y hacerlo legible.
 - ESLint: permite detectar invalidaciones de reglas JavaScript.
-