Módulo 1 - Introducción - Teoría

Contenido:

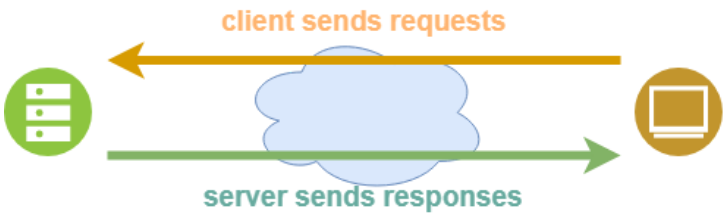
1. Conceptos servidor y cliente
2. Navegadores
3. Qué es un lenguaje de programación
4. JavaScript
5. Editores de código
6. Servidor web
7. Sandboxes

1. Servidor y cliente

Los servidores son los equipos que almacenan las webs y las aplicaciones, mientras que los clientes son los dispositivos conectados a Internet desde los que haremos las peticiones a los servidores, como por ejemplo, abrir la página de Google.

Cuando escribes una dirección web en el navegador, por ejemplo `nodejs.org`, lo que ocurre es lo siguiente:

1. El navegador consultará a una máquina conocida como servidor DNS cuál es la dirección IP real del servidor donde se ubica la ruta hacia `nodejs.org`.
2. Una vez localizada, el navegador envía un mensaje de petición HTTP al servidor, pidiéndole que envíe una copia de la página web. Este mensaje y todos los datos enviados entre el cliente y el servidor se envían a través de tu conexión a Internet (usando TCP/IP, normalmente).
3. Siempre que el servidor apruebe la solicitud del cliente (es decir, que todo sea correcto), el servidor enviará al cliente un mensaje `200 OK` (que significa que la solicitud ha tenido éxito), y comenzará a enviar los archivos de la página web al navegador.
4. El navegador reúne esos ficheros, los interpreta y forma un sitio web completo para mostrarlo.



Servidor DNS

Es importante saber que las direcciones web reales no son como las vemos en la barra de rutas del navegador. Realmente, la dirección real es otra muy distinta.

Si hacemos la prueba con `nodejs.org` (abrimos una consola de comandos y escribimos `ping nodejs.org`), vemos que la url apunta a una dirección IP bien distinta. Pero cuando queremos navegar a esta web, no tenemos que escribir ese chorro de números tan largo y difícil de memorizar.

¿Pero entonces, qué está ocurriendo? Pues que gracias a los servidores DNS (Servidores de Nombres de Dominio), sólo deberemos escribir el nombre de la web y ellos, en su registro interno, encontrarán una dirección real en internet (IP) que cazará con la ruta especificada para `nodejs.org`. Es como una tabla de direcciones web en las que cada url tiene una dirección física a la que apunta. Y esta dirección física es el servidor que la almacena.

Códigos de error

Existen una serie de códigos que nos aportan información sobre lo que ocurre con nuestras peticiones a los servidores.

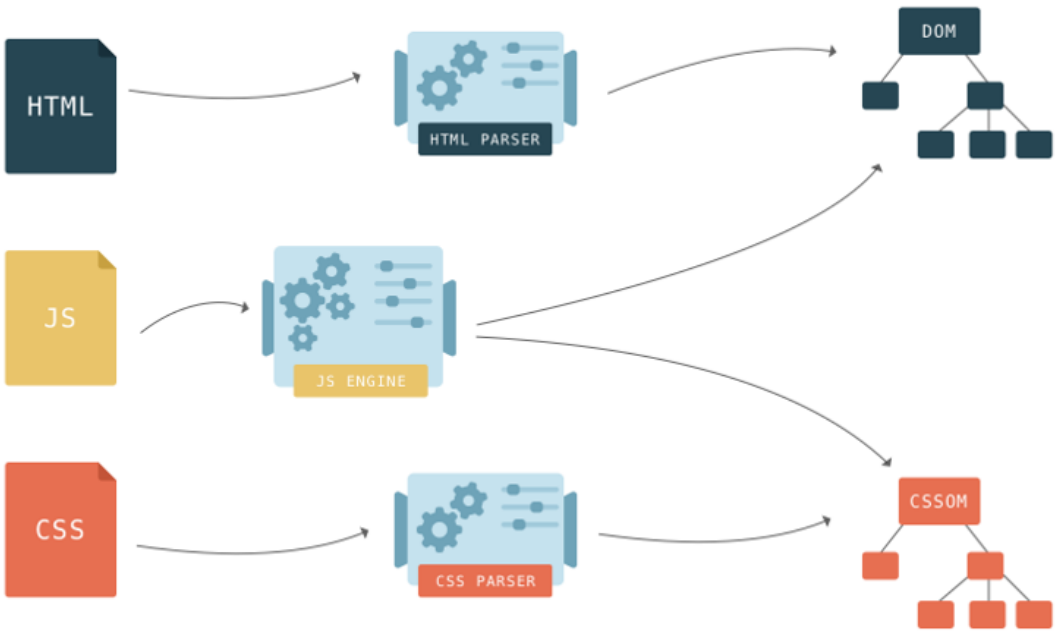
Algunos son muy conocidos como el famoso `404 Not Found`, pero otros son bastante raros y no surgen con mucha frecuencia. Sin embargo, el primer dígito es la indicación de la familia de códigos a los que pertenece y una forma rápida de saber qué información nos está dando es recordar la siguiente lista:

Código	Tipo	Descripción
1xx	Respuestas informativas	Solicitud recibida, continuando el proceso
2xx	Peticiones correctas	La acción se recibió, entendió y aceptó con éxito
3xx	Redirecciones	El cliente tiene que tomar una acción adicional para completar la petición
4xx	Errores del cliente	La solicitud contiene sintaxis incorrecta o no puede procesarse
5xx	Errores de servidor	El servidor falló al completar una solicitud aparentemente válida

2. Navegadores

Los navegadores son aplicaciones que instalamos en nuestro pc y nos permiten visualizar el contenido de páginas web las cuales estarán alojadas en servidores locales o en un servidor en internet.

El navegador, a grandes rasgos, se encarga de unir el código de nuestro HTML, los archivos CSS y JavaScript, interpretarlos y mostrarlo al usuario. ¿Fácil no?



Conceptos:

El Modelo de Objetos CSS (CSS Object Model) es una API (es decir, una interfaz de programación de aplicaciones) que permite manipular CSS desde JavaScript.

Así como el `DOM` (Document Object Model) es para HTML, el `CSSOM` (CSS Object Model) es para el CSS. Permite leer y modificar el estilo de CSS de forma dinámica.

A lo largo de este BootCamp, nosotros utilizaremos el navegador Chrome.

Pero tenemos que tener en cuenta otros navegadores muy conocidos como Firefox, Egde, Safari, Internet Explorer y Opera.

Si consultamos una página de estadísticas de uso como Global StatCounter podemos ver que la mayoría de las personas que acceden a internet lo hacen a través de Chrome.

Esto para nosotros es una buena noticia porque Chrome es un navegador tipo `evergreen` lo que significa que aplica las actualizaciones de forma silenciosa, lo cual nos permite tener un navegador siempre actualizado a la última versión.

Pero entonces, ¿cuál es el problema de tener tantos navegadores? Pues que debemos tenerlo en cuenta cuando hagamos nuestros desarrollos para saber a qué tipo de cliente debemos apuntar.

Es decir, no todos los navegadores implementan las nuevas características que dicta la W3C a la vez. De hecho, ni siquiera las características que no son nuevas y de hecho, tampoco tienen por qué coincidir en su implementación (issues conocidos y demás).

Por suerte, los principales navegadores tienen documentación donde podemos consultar el estado de implementación de una característica, de forma que si todavía no está implementada, sabemos que tenemos que adaptar nuestro código.

- Chrome
- Firefox
- Edge

Además de la propia documentación de los navegadores, contamos con herramientas como CanIUse en la que se nos proporciona una comparativa con los distintos navegadores sobre la característica que queremos consultar.

3. Qué es un lenguaje de programación

Un lenguaje de programación es un programa destinado a la construcción de otros programas informáticos.

Su nombre se debe a que comprende un lenguaje formal que está diseñado para organizar algoritmos y procesos lógicos que serán luego llevados a cabo por un ordenador o sistema informático, permitiendo controlar así su comportamiento físico, lógico y su comunicación con el usuario.

Como ya vimos en el módulo previo, HTML y CSS son lenguajes de programación al igual que JavaScript.

El HTML sería para dar la forma o estructura de nuestra aplicación, el CSS se encargaría del estilado de la misma, y el JavaScript es para la interacción del usuario con la página así como para manipular el DOM y el CSSOM.

En este módulo presentaremos JavaScript cubriendo algunos de sus conceptos fundamentales.

4. JavaScript

JavaScript es un lenguaje de programación ligero, interpretado y orientado a objetos, que provee de interactividad a las páginas web, permitiéndote crear contenido nuevo y dinámico, controlar archivos multimedia, crear imágenes animadas y multitud de posibilidades más que iremos descubriendo.

Podemos escribir código JavaScript en línea, aunque desde luego no será lo más recomendable, y en las etiquetas `script`, reservadas para este lenguaje.

Práctica de JavaScript - Mostrar un mensaje por pantalla

- Abrimos un notepad y creamos un fichero con el siguiente contenido:

```
<html>
  <head>
    <title>Hola mundo</title>
  </head>
  <body>
    <h1>Hola mundo</h1>
    <button type="button" onclick="window.alert('Hola desde JS')">
      Mostrar mensaje
    </button>
  </body>
</html>
```

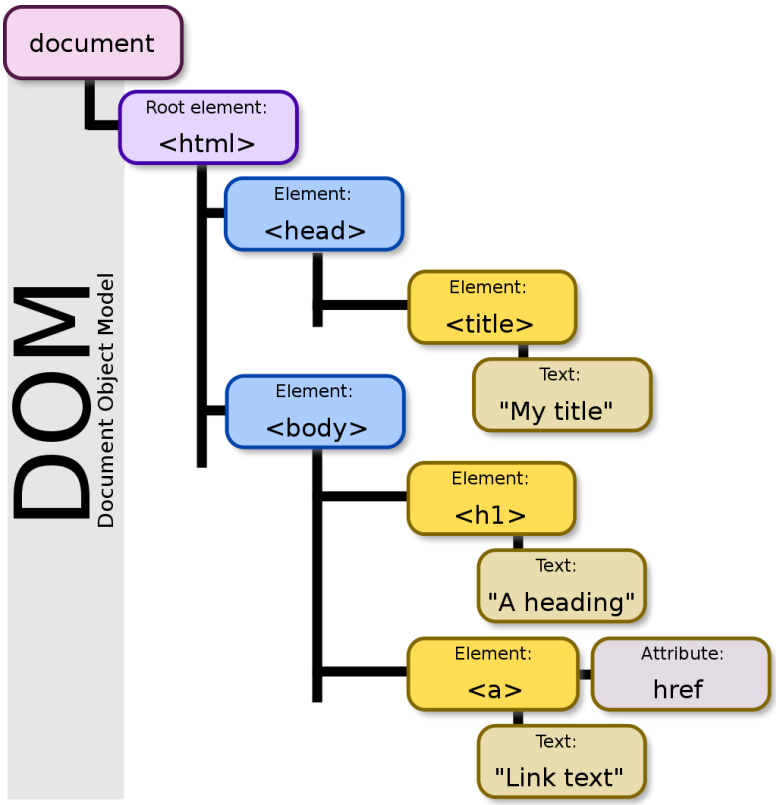
- Lo guardamos como `index.html`.
- Lo abrimos con nuestro navegador y probamos nuestro código.

Si nos fijamos, lo que hemos hecho es agregar código JavaScript en línea. Normalmente, esto no será la forma de trabajar en nuestros desarrollos pero nos sirve como primera aproximación.

Conceptos:

El objeto `window` representa la ventana que contiene el Modelo de Objetos de Documento o `documento DOM`.

El `DOM` es una API que define la estructura lógica de los documentos y el modo en que se accede y manipula.



Agregar una sección de JavaScript

La forma de usar JavaScript en un archivo HTML es mediante la inserción de etiquetas `script`.

Estas etiquetas nos permiten agregar código desde el propio fichero así como desde ficheros externos.

Esta etiqueta se puede poner en el `head` y en el `body`, o en ambos. Además, puedes poner tantas etiquetas `script` como necesites.

Práctica de JavaScript - Agregar una sección de JavaScript al HTML

Desde el propio fichero

- Abrimos el anterior documento `index.html`.
- Agregamos antes de finalizar la etiqueta `body` una nueva etiqueta `script`.

./index.html

```
<html>
  <head>
    <title>Hola mundo</title>
  </head>
  <body>
    <h1>Hola mundo</h1>
    <button type="button" onclick="window.alert('Hola desde JS')">
      Mostrar mensaje
    </button>
+    <script>
+      window.alert('Hola desde una etiqueta script');
+    </script>
  </body>
</html>
```

- Guardamos los cambios y lo abrimos con el navegador.

Desde un fichero externo

- Abrimos un nuevo notepad y creamos nuestro código.

```
console.log("Mostrando mensaje por consola desde un fichero externo");
```

- Guardamos como fichero `demo.js`.
- Abrimos nuestro `index.html` y editamos.

./index.html

```
<html>
  <head>
    <title>Hola mundo</title>
  </head>
  <body>
    <h1>Hola mundo</h1>
    <button
      type="button"
      onclick="window.alert('Hola desde JS')"
    >
      Mostrar mensaje
    </button>
-    <script>
-      window.alert('Hola desde una etiqueta script');
-    </script>
+    <script src="demo.js"></script>
  </body>
</html>
```

- Guardamos los cambios y comprobamos con el navegador.

Colocar scripts en archivos externos tiene algunas ventajas:

- Separa HTML y código.
- Hace que el HTML y el JavaScript sean más fáciles de leer y mantener.
- Los archivos JavaScript en caché pueden acelerar las cargas de página.

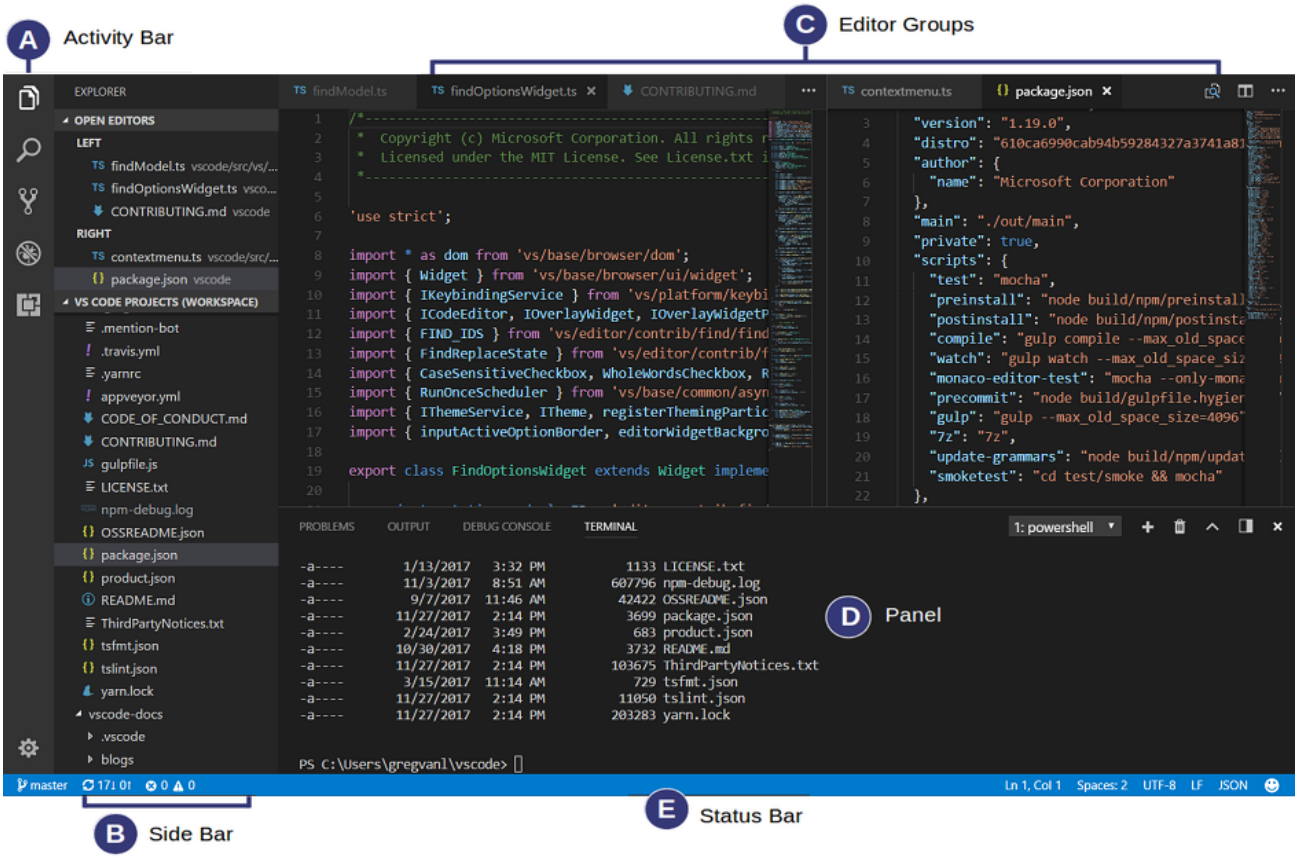
5. Editores de código

Dentro del mundo de los editores de código tenemos infinidad de ellos, unos más conocidos y otros no tanto. Sin embargo, nosotros recomendamos por su facilidad de uso y variedad de plugins el editor Visual Studio Code.

Este editor de código te ofrece:

- una terminal de comandos.
- un panel de código donde tendrás un resaltado de sintaxis según el tipo de fichero que elijas, lo cual te ayudará a desarrollar con más comodidad.
- explorador de carpetas.
- disposición de ventanas de múltiples formas.
- soporte de distintos lenguajes de programación.
- soporte de extensiones muy útiles que te facilitarán el proceso de desarrollo, depuración, testeo, etc...

Estructura de VSCode



- (A) Barra de actividad: que nos permite cambiar de vista y nos aporta indicadores adicionales específicos del contexto en el que estemos, como la cantidad de cambios pendientes cuando Git está habilitado.
- (B) Barra lateral: contiene diferentes vistas (también según el contexto), como el Explorador, para ayudarte mientras trabajas en tu proyecto.
- (C) Editor: el área principal para editar tus archivos. Puedes abrir tantos editores como desees, uno al lado del otro, vertical y horizontalmente.
- (D) Paneles: en estos paneles obtienes información de salida o depuración, errores y advertencias, incluso tenemos un terminal integrado.
- (E) Barra de estado: información sobre el proyecto abierto y los archivos que edita.

6. Servidor WEB

Y si antes dijimos que los navegadores se encargan de interpretar los ficheros HTML, CSS y JavaScript para componer nuestras páginas alojadas en los servidores web, ¿qué son los servidores web?

Pues estos servidores no son ni más ni menos que máquinas conectadas unas a otras donde se realizan peticiones de información, la buscan en sus ficheros y las sirven para que los navegadores puedan interpretarlas y mostrarlas.

Realmente, existe el término servidor que es para referirnos a la máquina y servidor web, que es el software que se encarga de despachar el contenido de un sitio web al usuario.

Para poder instalar un servidor web ligero vamos a necesitar dos herramientas que debemos instalar: `Node.js` y `lite-server`.

Node.js

Node.js es una librería que proporciona un entorno JavaScript en el lado del servidor. Es de código abierto y se ejecuta sobre el motor de JavaScript V8 creado por Google.

Práctica de JavaScript - Descargamos e instalamos `Node.js` desde la página oficial: https://nodejs.org/en/download/

A lo largo del BootCamp utilizaremos `Node.js` para la instalación de los paquetes `npm` o para ejecutar código JavaScript en nuestros servidores locales.

Uno de los puntos fuertes de `Node.js` y que enseguida vamos a utilizar, es su administrador de paquetes Node o `Node Package Manager` (de ahí las siglas `npm`). Es una especie de gestor que da acceso a un conjunto de librerías muy amplia, que además son gratuitas, y generadas a partir de la colaboración de los usuarios de su comunidad.

Más adelante veremos las características que nos aporta `Node.js` y `npm` pero por ahora, nos quedaremos con que con la herramienta `npm` podemos instalar librerías tanto en nuestra máquina de forma global (es decir, se registrará una ruta en nuestra máquina que apuntará un comando concreto a la ejecución de un paquete), como en nuestros proyectos de forma local.

- Instalar un paquete de forma global

```
npm install --global nombre_paquete
```

Servidor WEB local

Para poder hacer nuestras pruebas con el código que iremos generando, necesitamos tener un servidor web que ejecute nuestro contenido web.

Para ello haremos uso de una herramienta muy conocida que se llama LITE-SERVER.

`lite-server` es un servidor web ligero que te permite cargar aplicaciones web, simulando un servidor web. Lo interesante de esta herramienta es que, a parte de servirte las páginas web que generes, es capaz de detectar los cambios cuando guardas tus ficheros y te actualiza el contenido automáticamente.

Práctica de JavaScript - Instalación de un servidor web ligero mediante `npm`

7. Sandboxes

Cuando empecemos a desarrollar nuestras aplicaciones habrá momentos en los que queramos hacer una prueba de concepto, compartir código en vivo o simplemente, no tendremos un equipo con un IDE instalado.

Para esas ocasiones tenemos una serie de recursos online que nos facilitan muchísimo nuestro trabajo. Son los conocidos `sandboxes`.

Algunos son tan populares que webs como StackOverflow admiten la incorporación de enlaces directamente en sus plataformas.

Muchos de ellos suelen ser de uso gratuito, admiten funciones de colaboración de código (ideal para construir conceptos juntos, entrevistas, etc...), múltiples diseños, tamaños de fuente, temas claros / oscuros, etc..., formateo de código, soporte para linters (CSS y JS), implementación de TypeScript, npm, etc...

Tenemos distintos editores de código online pero los más conocidos son las siguientes:

- CodeSandbox
- StackBlitz
- JSFiddle
- CodeAnyWhere