**Curso Java Fullstack**

La web funciona con la arquitectura cliente - servidor. El cliente y el servidor son dos

computadoras: el **cliente** es quien solicita la información y el **servidor** es quien se la

suministra. El proceso cuando el cliente pide información se lo llama **petición**.

**APIs:**

La API es la que se encarga de conectar al frontend y al backend, de esta manera, un frontend puede funcionar con cualquier backend conectándose desde una api.

**Frontend, Backend y Full Stack**

El **Frontend** es el cliente y es el que se encarga de toda la lógica de este cuando

desea realizar alguna petición.

El **Backend** está del lado del servidor y se encarga de la lógica, de todas las

funciones que requiere el cliente.

Un **stack** es un conjunto de tecnologías que se utiliza para desarrollar una aplicación, son todas las tecnologías que usa el Frontend y el Backend, por lo tanto, un desarrollador full stack puede manejar tanto el frontend como el backend de una app.

**Desarrollador Fullstack:**

- Programador con perfil técnico muy completo.  
- Encargado de manejar cada uno de los aspectos relacionados con la creación y el mantenimiento de una aplicación web.

- Es un programador multiusos y dentro del desarrollo del proyecto es

responsable del montaje de los servidores, hasta el diseño con hojas de

estilos.

- Es fundamental que tenga conocimientos en desarrollo Frontend y

Backend.

Frontend: parte de un sitio web que interactúa con los usuarios.

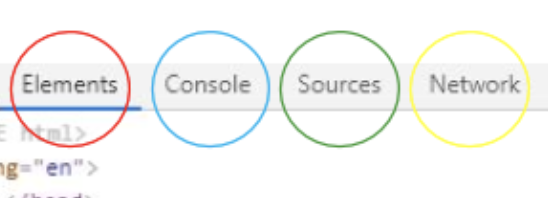
Backend: parte del sitio web que se conecta con la base de datos

**Chrome Developer Tools**

**¿Qué son?**

Son un conjunto de herramientas de creación y depuración de páginas web integradas en Google Chrome.

Las DevTools proporcionan a los desarrolladores acceso a los componentes internos del navegador y su aplicación web. Son útiles para rastrear de forma rápida los errores.



**Elementos** (Elements): Nos permite visualizar y editar el HTML y CSS de nuestra web.

**Consola** (Console): Es muy útil para interactuar y hacer pruebas de JavaScript. Por ejemplo podemos llamar a funciones de nuestra página y ver sus resultados.

**Fuentes (Sources):** nos permite navegar por todos los ficheros HTML, CSS y JavaScript que utiliza la página que estamos viendo.

**Red (Network):** Nos permite ver los elementos que se descargan y sus tiempos, permitiéndonos localizar problemas sobre los que debemos prestar atención. Por ejemplo, podemos ver que imágenes tardan en cargar y reducir su size.

**Introducción a HTML**

**¿Qué es HTML?**

Es un lenguaje utilizado para definir la estructura y semántica de una

página web.

HTML significa lenguaje de marcado de hipertexto, marcado se refiere a

que debemos utilizar determinadas etiquetas para indicarle al navegador lo

que queremos mostrar y hipertexto porque esas mismas etiquetas

contienen metadatos.

Llamamos HTML semántico a un documento que usa correctamente las

etiquetas para que la estructura resultante, quitando la capa de diseño,

tenga sentido por sí sola.

**¿Cómo funciona?**

1 - El navegador (cliente) le pide información al servidor.

2 - El servidor devuelve la información al cliente en un archivo HTML.

3 - El navegador en el cliente lee el archivo de arriba hacia abajo y de

izquierda a derecha para interpretar la información.

4 -Tiene en cuenta las etiquetas que tiene el documento y las va

renderizando en pantalla (lo que se ve en el navegador).

**Estructura básica**

Imagen que contiene pájaro

Descripción generada automáticamente

Una captura de pantalla de un celular con letras

Descripción generada automáticamente

Captura de pantalla de un celular

Descripción generada automáticamente

Imagen que contiene texto

Descripción generada automáticamente

**Enlaces o Hipervínculos**

**Una captura de pantalla de un celular

Descripción generada automáticamente**

**Imágenes**

Para mostrar una imagen en una página tenemos dos formas de hacerlo, una es usando el

elemento <img/> y otras es mediante CSS (que veremos más adelante).

Esta etiqueta sólo requiere de dos atributos obligatorios que son:

src (de source) y alt (de alternative).

Por ejemplo**:**

<img src="https://picsum.photos/200" alt=“Imagen de Picsum">

**Etiquetas básicas**

**Captura de pantalla de un celular con texto

Descripción generada automáticamente**

**Elementos Inline y Block**

HTML clasifica a todos los elementos en dos grupos: inline y block.

De forma predeterminada, los elementos en bloque comienzan en una nueva línea y los

elementos en línea pueden comenzar en cualquier parte de una línea.

**Captura de pantalla de un celular

Descripción generada automáticamente**

**HTML Parte 2**

**Listas**

Una captura de pantalla de un celular con texto

Descripción generada automáticamente

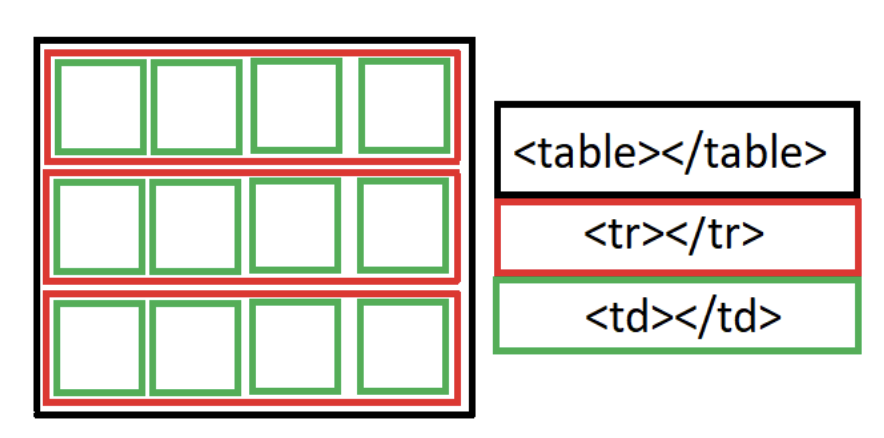
**Tablas**

Las tablas se usan para representar datos.

El ejemplo más común de tablas son los documentos de Excel.

En HTML hay que definir una etiqueta para cada parte de la tabla.

Las tablas no se usan para maquetar.



Captura de pantalla de un celular con texto

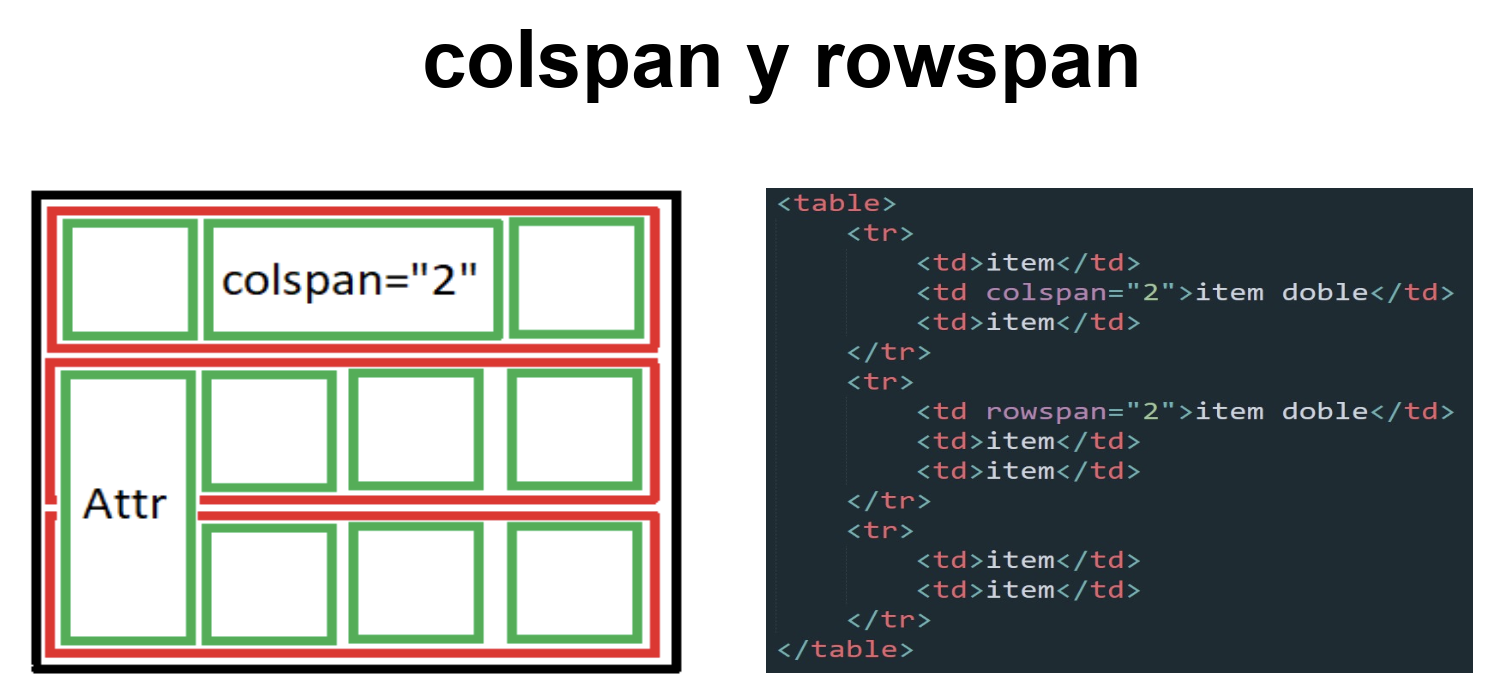
Descripción generada automáticamente

Captura de pantalla de un celular

Descripción generada automáticamente

Captura de pantalla de un celular

Descripción generada automáticamente



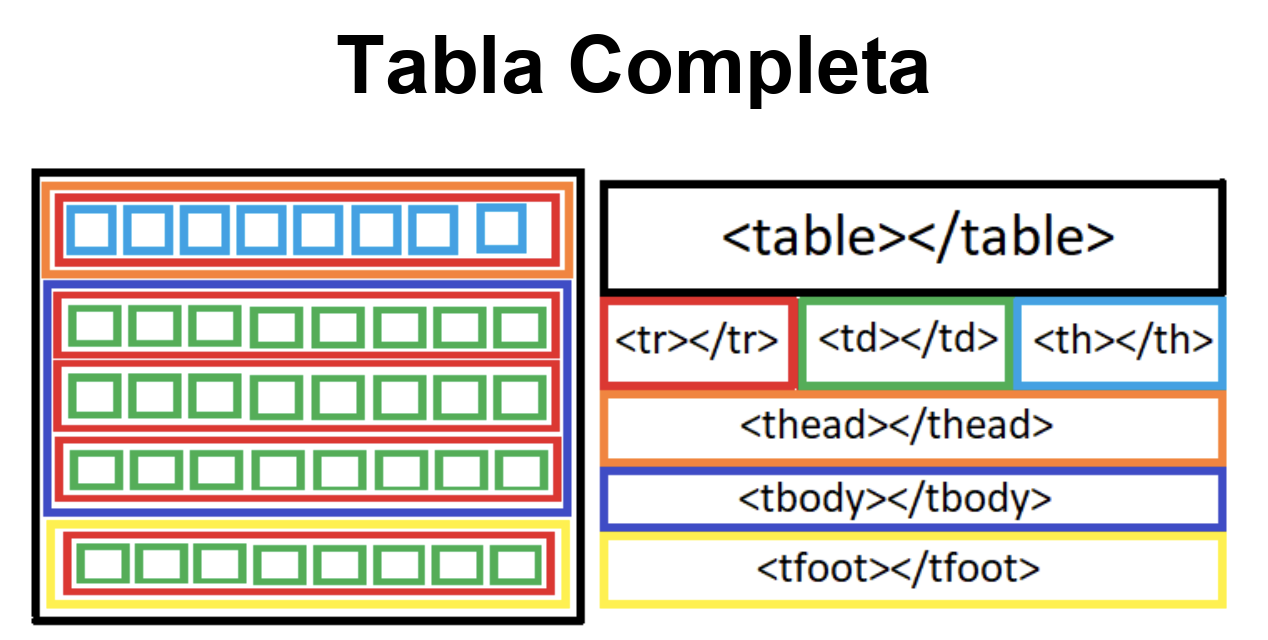


Imagen que contiene dibujo

Descripción generada automáticamente

**IFRAME**

Captura de pantalla de un celular

Descripción generada automáticamente**Iconos**

**Captura de pantalla de un celular

Descripción generada automáticamente**

****

**<audio> y <video>**

**Captura de pantalla de un celular con texto

Descripción generada automáticamente**

**PARTE 3 HTML**

**Formularios**

Se usan para recolectar datos. Podemos enviar los datos recolectados a un servidor.

Para crear un formulario, primero hay que indicar una etiqueta contenedora <form>, que va a incluir toda la Información que se quiere recolectar en ese formulario.

Sus elementos y atributos:

<form> será el elemento padre que anide todos los elementos HTML que representarán los campos de nuestro formulario, incluido el botón de enviar.

- action: indica la URL a la que se enviará la petición HTTP con toda la información del formulario.

- method: indica si la petición HTTP será GET o POST.



Sus elementos y propiedades:

<input>: permite introducir diferentes tipos de campo de formulario en base

al valor del atributo type. En función del valor indicado en type dispondremos

de unos atributos u otros:

* type: este valor puede tener muchos valores: text, email, checkbox, color, date, file, hidden, etc. en función del tipo de campo que queramos, los nombres son bastante auto-explicativos.
* id: este atributo es obligatorio si en el elemento label tiene un atributo for,

en tal caso deberá contener un identificador único en la página.

* name: representa el nombre asignado al campo cuando se envíe la

petición HTTP.

* value: representa el valor que se asignará al campo cuando se envíe la

petición HTTP.

* <label>: se usa para especificar la etiqueta (o nombre) del campo del formulario.
* for: tiene que tener el mismo valor que el atributo id del campo (input, select o

textarea) al que hace referencia la etiqueta.

* <textarea>: representa un campo que nos permite introducir textos con saltos de

línea incluidos, normalmente se usa cuando hay que introducir: descripciones,

biografías, etc.

* Id: igual que el elemento input y select.
* name: igual que el campo input y select.
* <button>: representa un botón y el texto del botón está representado por su

contenido.

* type: define el comportamiento del botón cuando está activado y puede

contener tres valores: submit, reset, button.

* <select>: nos permite crear una lista desplegable de opciones, cada opción

estará contenida como hija dentro de un elemento <option> donde vamos a

encapsular cada opción de la lista.

* id: igual que el elemento input.
* name: igual que el campo input.
* value: igual que el atributo value del campo input.
* <fieldset>: permite organizar en grupos los campos de un formulario. Tiene un elemento <legend> que puede contener a otros elementos o se puede utilizar en forma de título.
* id: igual que el elemento input.
* name: igual que el campo input.
* value: igual que el atributo value del campo input.
* Textarea: Define un entrada de texto más amplia.
* Label: Define un nombre o referencia para un input.
* For: Attr que especifica a qué input se vincula.
* Fieldset: Define una sección en un formulario.
* Legend: Define un título para una fieldset.
* Select: Similar al input radio.
* Option: Define las opciones para el select.

En muchos de los elementos podemos añadir (opcionalmente) otros

atributos como:

* required a un elemento para que el navegador se encargue de

validar que este campo está relleno.

* placeholder si queremos que aparezca un texto de ayuda para

rellenar el campo.

* value para introducir un valor por defecto en el campo.
* readonly si queremos que sea de sólo lectura.

Atributo type de la etiqueta <input>

<input type="text">

<input type="password">

<input type="radio">

<input type="checkbox">

<input type="number">

<input type="email">

<input type="date">

<input type="url">

<input type="file“>

<input type=“button">

<input type=“range">



