<codoacodo/>

Agencia de
Aprendizaje
a lo largo
de la vida

Desarrollo Fullstack

BA

Les damos la bienvenida

Vamos a comenzar a grabar la clase

Clase 21

JAVASCRIPT

- Workshop

Clase 22

Introducción a Backend

- Relación Cliente/Servidor
- Protocolo HTTP
- Status Codes

Clase 23

Patrones de Arquitectura

- ¿Qué son?
- Tipos de patrones
- MVC
- REST

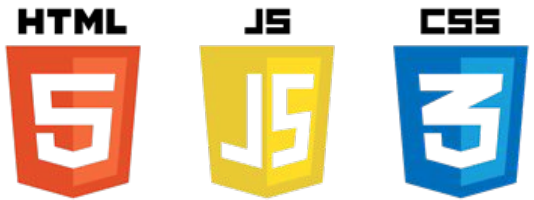
Antes de arrancar, un repaso.

Frontend 🎨

Es todo lo que pasa del lado del cliente (en el navegador).

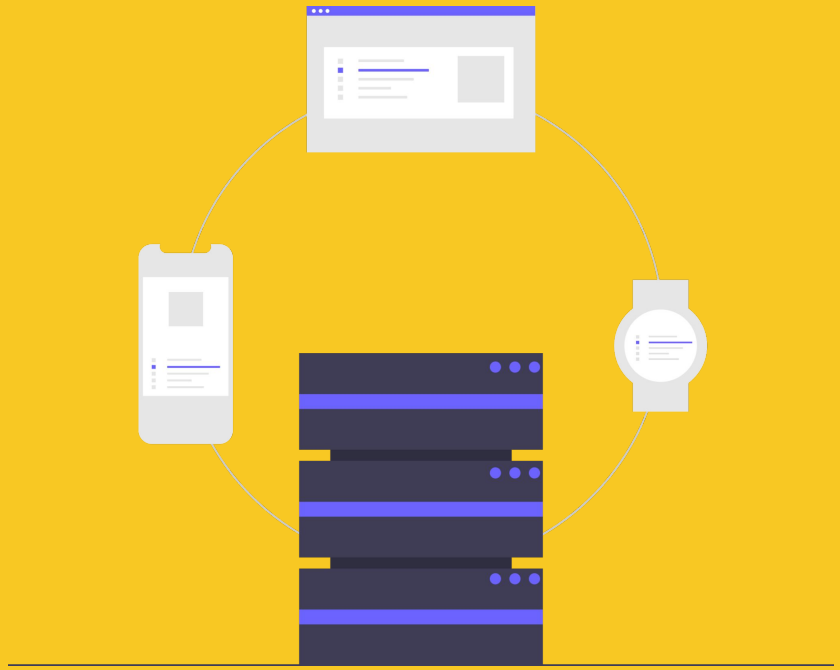
Está compuesto por archivos estáticos en los lenguajes que el navegador puede interpretar. Estos lenguajes son **HTML** para la estructura de información, **CSS** para los estilos y **JavaScript** para la interacción dentro nuestro sitio web.

El desarrollador de **front-end** se encarga de implementar todo lo relacionado con la parte visible, lo que “toca" el usuario cuando navega por la web.



BACKEND

Introducción



Backend ⚙️

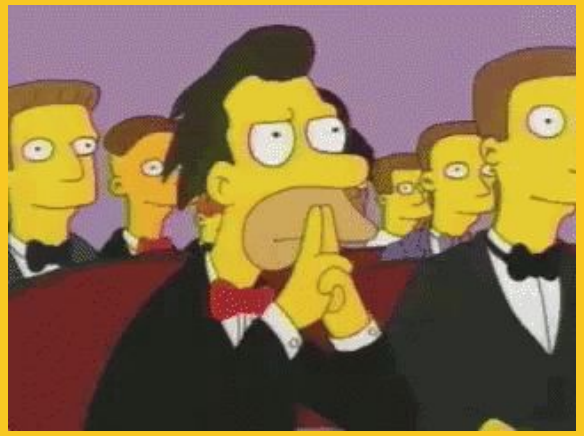
Es todo lo que pasa del lado del servidor.

El backend es la parte del desarrollo web que se encarga de que toda la lógica de una página web funcione. Se trata del conjunto de acciones que pasan en una web pero que no vemos como, por ejemplo, la consulta de datos a una BBDD y posterior envío al cliente.

Algunas de las funciones que se gestionan en la parte del back-end son:

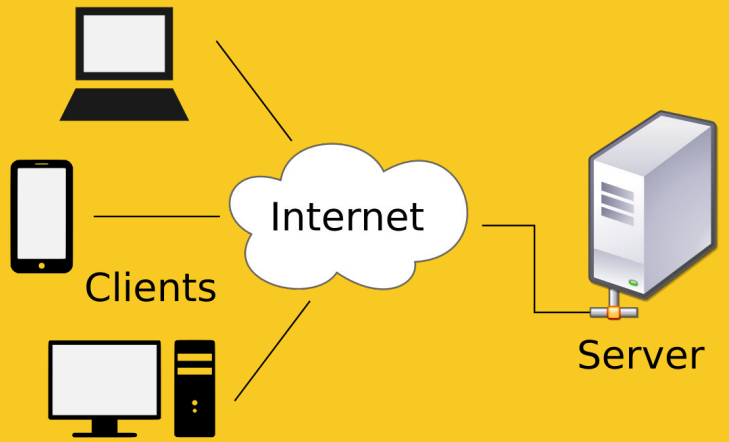
- El desarrollo de funciones que simplifiquen el proceso de desarrollo.
- Acciones de lógica.
- Conexión con bases de datos.
- Uso de librerías del servidor web (por ejemplo para implementar temas de caché o para comprimir las imágenes de la web).

Pero... ¿cómo se comunican?



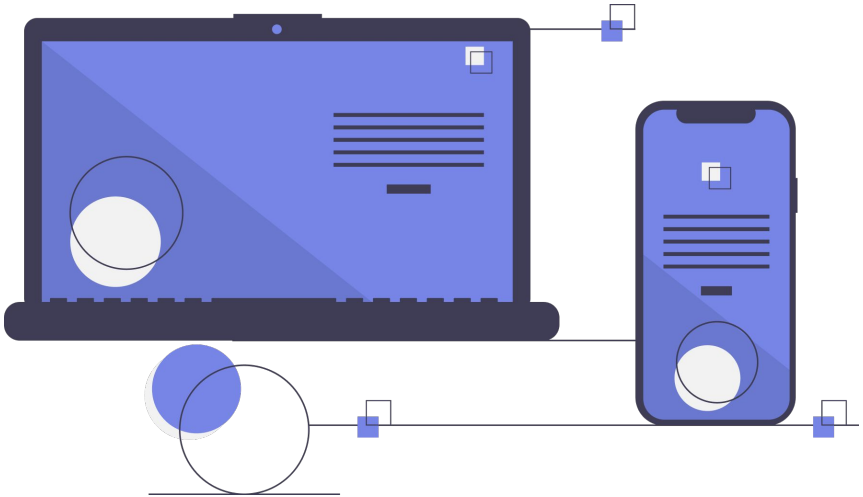
Flujo Cliente/Servidor

Es una relación en la cual un programa (el cliente) solicita un servicio o recurso de otro programa (el servidor). Este esquema refleja el flujo de información entre uno o varios dispositivos con un servidor web.



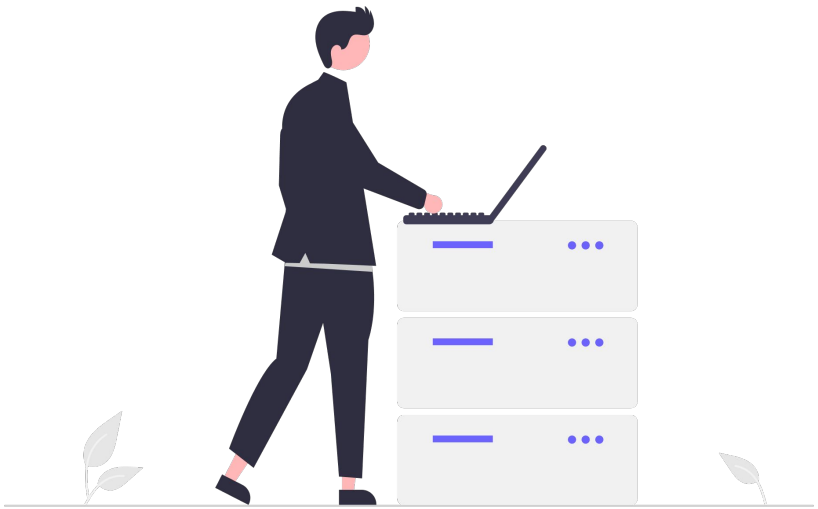
Cliente

- Envía una petición al servidor y se queda esperando por una respuesta.
- Una vez que son servidas sus solicitudes, termina el trabajo.
- Un cliente accede a un servidor y recupera servicios especiales o datos de él.



Es tarea del cliente estandarizar las solicitudes, transmitirlas al servidor y procesar los datos obtenidos para que puedan visualizarse en un dispositivo de salida como una pantalla.

Servidor



Es un programa que ofrece un servicio que se puede obtener en una red.

- Acepta la petición desde la red, realiza el servicio y devuelve el resultado al solicitante.
- El servidor comienza su ejecución antes de comenzar la interacción con el cliente.
- Su tiempo de vida o de interacción es “interminable”, una vez comienza a correr, se queda esperando las solicitudes que pudieran llegar desde los diversos clientes.

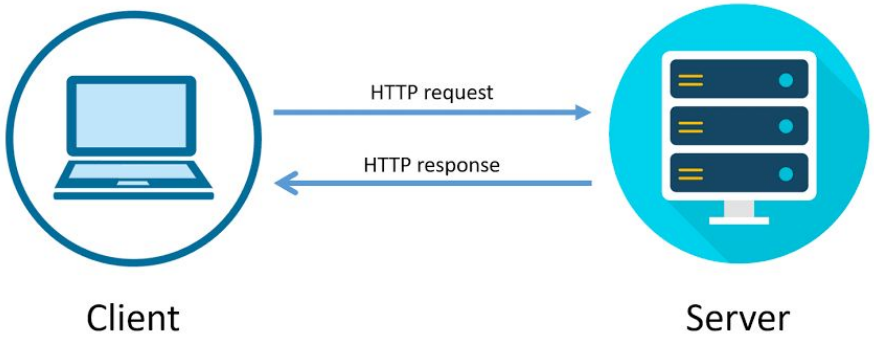
Request/Response

REQUEST

Son las solicitudes o peticiones que hacemos a través del navegador (el cliente) a un servidor.

RESPONSE

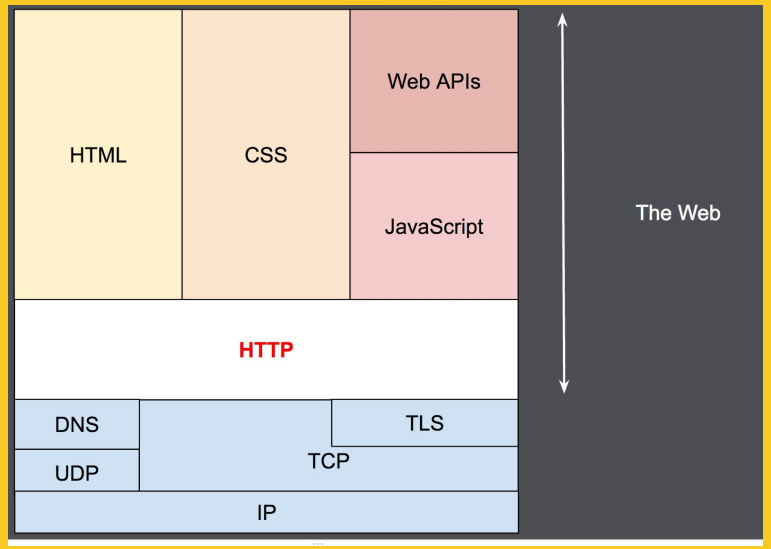
El servidor recibe nuestra solicitud, la procesa y envía como resultado una respuesta al cliente (navegador).



Protocolo HTTP

HTTP o Hyper Text Transfer Protocol es el nombre de un protocolo que nos permite realizar una petición de datos y recursos, como pueden ser documentos HTML.

Es la base de cualquier intercambio de datos en la web para la comunicación de un cliente con un servidor.



Manejo de información

HTTP es un protocolo sin estado, por lo que no guarda ninguna información sobre conexiones anteriores.

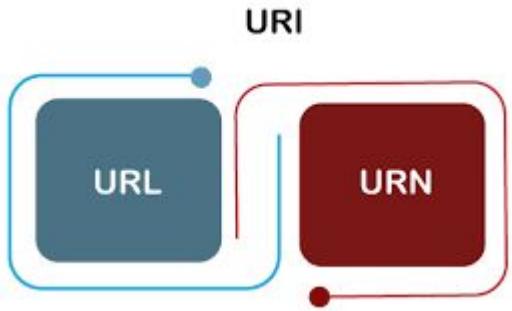
El desarrollo de aplicaciones web necesita frecuentemente mantener estado, por ejemplo, para el uso de "CARRITOS de compra" en páginas de comercio electrónico. Para esto se usan las cookies, que es información que un servidor puede almacenar en el sistema cliente.

IDENTIFICACIÓN DE RECURSOS WEB

El objetivo de una solicitud HTTP se denomina "recurso", (es decir: datos) y dicho recurso, no posee un tipo definido por defecto; puede ser un documento, o una foto, o cualquier otra posibilidad.

Cada recurso es identificado por un Identificador Uniforme de Recursos (URI) y es utilizado a través de HTTP, para la identificación del tipo de recurso.

Una URI (identificador de recursos uniformes) es un bloque de texto que se escribe en la barra de direcciones de un navegador web y está compuesto por dos partes: la URL y la URN.



URI: URL + URN

Un **URI** consta de un máximo de cinco partes, de las cuales solo dos son obligatorias:

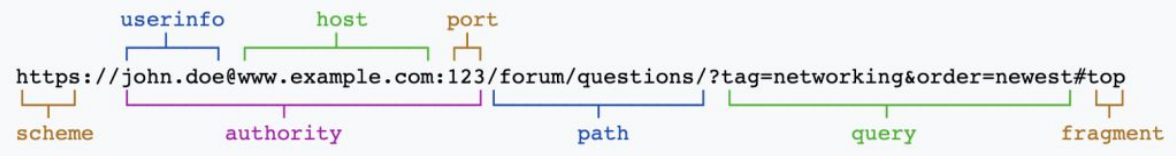
- scheme (esquema): proporciona información sobre el protocolo utilizado.
- authority (autoridad): identifica el dominio.
- path (ruta): muestra la ruta exacta al recurso.
- query (consulta): representa la acción de consulta.
- fragment (fragmento): designa una parte del recurso principal.

URL

Uniform resource locator se utiliza para indicar dónde se encuentra un recurso. Por lo tanto, también sirve para acceder a algunas páginas web por Internet.

URN

Uniform resource name es independiente de la ubicación y designa un recurso de forma permanente.



FASES HTTP

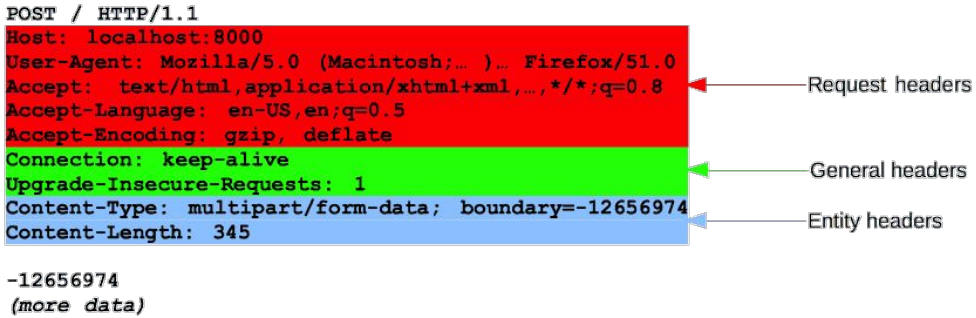
En los protocolos basados en el modelo cliente-servidor, como es el caso del HTTP, una sesión consta de tres fases:

- El cliente establece una conexión TCP.
- El cliente manda su petición, y espera por la respuesta.
- El servidor procesa la petición, y responde con un código de estado y los datos correspondientes.

HTTP HEADERS

Las cabeceras (en inglés headers) HTTP permiten al cliente y al servidor enviar información adicional junto a una petición o respuesta.

- Headers generales, (General headers), como Via (en-US), afectan al mensaje como una unidad completa.
- Headers de petición, (Request headers), como User-Agent, Accept-Type, modifican la petición especificando con mayor detalle (como: Accept-Language (en-US), o dándole un contexto, como: Referer, o restringiéndola condicionalmente, como: If-None.
- Headers de entidad, (Entity headers'), cómo Content-Length las cuales se aplican al cuerpo de la petición. Por supuesto, esta cabecera no necesita ser transmitida si el mensaje no tiene cuerpo ('body' en inglés).

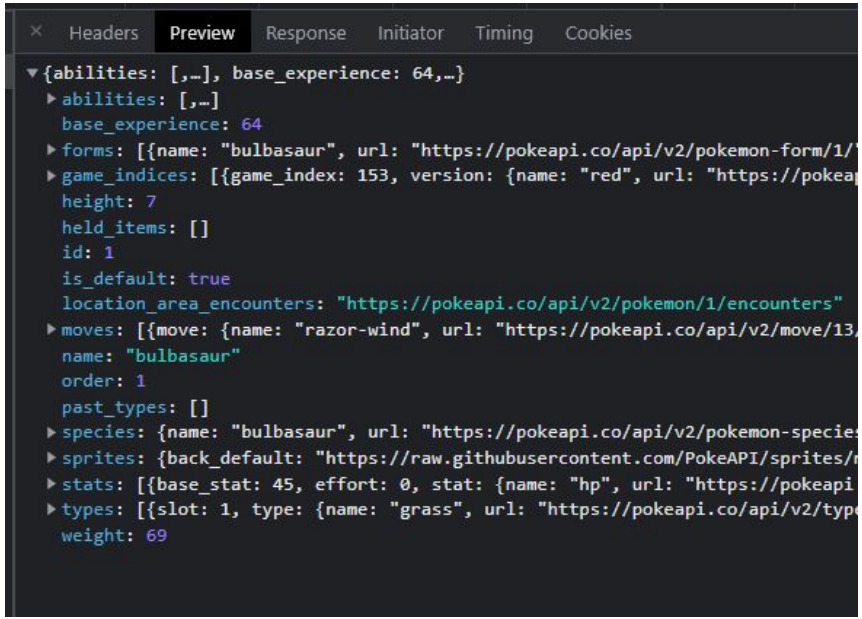


HTTP BODY

La última parte del mensaje de respuesta es el body.

No siempre se envía o se recibe uno y se utiliza para enviar información desde un cliente (navegador) o para recibirla desde el servidor.

Para ello se utiliza el METHOD HTTP conocido como POST que manipula la información en el cuerpo del mensaje a diferencia del método GET que lo hace directamente en la URL a través de los query params.



HTTP METHODS

HTTP define un conjunto de métodos de petición para indicar la acción que se desea realizar para un recurso determinado. Estos métodos son utilizados principalmente en la creación de servicios REST.

GET	Leer información
PUT	Reemplazar información
POST	Enviar/Solicitar/Crear información
PATCH	Modificar/Actualizar información
DELETE	Eliminar información

HTTP STATUS CODES

Los códigos de estado de respuesta HTTP indican si se ha completado satisfactoriamente una solicitud HTTP específica. Las respuestas se agrupan en cinco clases:

100	Respuestas informativas (100–199)
200	Respuestas satisfactorias (200–299)
300	Redirecciones (300–399)
400	Errores de los clientes (400–499)
500	Errores de los servidores (500–599)

HTTP STATUS CODES

Status Code HTTP más comunes:

200	La solicitud ha tenido éxito.
302	El recurso de la URI solicitada ha sido cambiado temporalmente
400	El servidor no pudo interpretar la solicitud dada una sintaxis inválida.
401	Es necesario autenticar para obtener la respuesta solicitada.
403	El cliente no posee los permisos necesarios para cierto contenido
404	El servidor no pudo encontrar el contenido solicitado.
500	El servidor ha encontrado una situación que no sabe cómo manejarla.
503	El servidor está caído por mantenimiento o está sobrecargado

No te olvides de dar el presente

Recordá:

- Revisar la Cartelera de Novedades.
- Hacer tus consultas en el Foro.

Todo en el Aula Virtual.

Gracias