Recordemos. Que HTTP. Funciona en modo cliente/Servidor. Un cliente (como el navegador web) envía una solicitud y un servidor (donde se aloja una página web) le responde.

En este Proceso hay 2 elementos clave: Los metodos HTTP y los codigos de estado

Los metodos HTTP mas comunes:

- GET → Solicita datos o recursos específicos al servidor, es un metodo seguro e idempotente, lo que significa que no modifica los datos y Puede repetir la Peticion sin Problema
- POST → Envia datos al servidor, a diferencia del metodo GET, este metodo Puede modificar el estado del servidor

Despues de que el cliente envia una solicitud con un metodo (GET, Post), el servidor responde con un codigo de estado de 3 digitos, el Primer digito es el más importante ya que nos dice el tipo de respuesta, los tipos (5) mayus comunes son:

1XX (Informativa) → Tu solicitud ha sido recibida y esta en Proceso

2XX (Exitosa) → Tu solicitud fue recibida, entendida y aceptada

3XX (Redireccion) → se necesita una accion adicional Para completar tu solicitud, Por ejemplo que el recurso se haya movido a otra direccion

4XX (Error del Cliente) → La solicitud tiene un error, el más famoso es el 404

5XX (Error del servidor) → Tu solicitud es valida, Pero el servidor no Pudo completarla, es un Pedo del servidor

Cuando escribes una URL, tu navegador hace una Solicitud GET, Pidiendo una web especifica, El servidor recibe tu solicitud y tiene 3 opciones Principales Para responder:

- Si tiene el recurso, lo entrega con un codigo 200 OK
- Si ha sido movido a otro lugar, la tienda te dice con un codigo 301 Moved Permanently que vayas a la new direccion
- Si Pediste un recurso que no existe, el servidor responde un 404 NOT Found

Codigos de estado más comunes

200 OK → La solicitud ha tenido exito

201 Created → La solicitud ha tenido exito y se ha creado un nuevo recurso, comun en Peticiones POST

301 Moved Permanently → El recurso solicitado ha sido asignado a una nueva URL Permanente

302 Found → o sea moved temporarily, el recurso se encuentra temporalmente en otra ubicacion, el cliente debe seguir usando la misma URL

400 Bad Reqest: El servidor no puede entender la solicitud debido a una sintaxis incorrecta del cliente
401 Unauthorized: La solicitud requiere autenticacion
404 Not Found: El servidor no puede encontrar el recurso solicitado
500 Internal Server Error: El servidor se ha encontrado con una situacion que no sabe manejar. Es un error generico del servidor
503 Service Unavailable: El servidor no esta listo para manejarl la solicitud. Es comun cuando un servidor esta sobrecargado o en mantenimiento

Versiones de HTTP

HTTP/1.1, HTTP/2 y HTTP/3

HTTP/1.1 El clasico

Aunque antes de esta tambien hubo un 1.0
Este era capaz (el 1.1) de enviar el nombre de una pagina web en las cabeceras, lo que permitia que el servidor recibiera el nombre y lo utilizara para enviar los ficheros en funcion de la url

Ha sido el estandar durante mucho tiempo. Su modelo es simple: el cliente abre una conexion TCP, envia una solicitud, espera la respuesta del servidor y, en la mayoria de los casos la cierra

Problemas: El principal inconveniente es que cada recurso(una imagen, un archivo CSS, Js) necesita una solicitud y una respuesta esperada. Para cargar una pagina web moderna con muchas imagenes y scripts, el navegador tiene que hacer peticiones una tras otra, lo que puede ser lento

Al principiop cuando las paginas web eran muy sencillas, no se tenian que cargar tantos archivos ni imagenes, en comparacion con las actuales

HTTP/2 LA EVOLUCION

Para solucionar los problemas de HTTP/1.1 nacio HTTP/2

- Multiplexacion: Es su caracteristica mas importante. A diferencia de HHTP/1.1 que procesa las solicitudes una por una, HHTP2 pude enviar multiples solicitudes y recibir multiples respuestas a traves de una sola conexion. Esto evita la "congestion" de la red
- Compresion de cabeceras: las cabeceras HTTP suelen ser grandes y repetitivas. HTTP2 las comprime para reducir el tamaño de la informacion que se envia
- Server Push: el servidor puede enviar recursos al cliente sin que este los haya pedido. Esto es util para enviar por ejemplo los archivos CSS o JS que la pagina va a necesitar antes de que el navegadorl los solicite. En la practica era fragil, solo vivia por conexion y causaba problemas con la cache y proxies. Navegadores y servidores lo retiraron sustituyendolo por Link: rel=preload y 103 Early Hint

HTTP/3: LA REVOLUCION

Es la version mas reciente y representa un cambio fundamental
Protocolo QUIC: Minetras que las versiones anteriores se basanj en TCP, HTTP/3 utiliza un nuevo protocolo de transporte llamado QUIC(Quick UDP Internet Conection)

El problema de TCP es que si un solo paquete de datos se pierde, la conexioon entera se detiene hasta que se retransmite. Con QUIC, si un paquete se pierde,, solo afecta a ese stream de datos especifico, no a toda la conexion. Esto es crucial para conexiones moviles o inestables.