***Francisco Javier Otero Herrero***

***Grupo ATU***

***27-3-2025***

***Manejo del Protocolo HTTP***

**Contenido**

[**Protocolo HTTP 3**](#_Toc194003531)

[**Para qué se utiliza el protocolo HTTP 3**](#_Toc194003532)

[**Principales propiedades 3**](#_Toc194003533)

[**Etapas de una transacción HTTP 5**](#_Toc194003534)

[**Grupos de Códigos de estado HTTP 6**](#_Toc194003535)

[Códigos 1xx (Informativos) 6](#_Toc194003536)

[Códigos 2xx (Éxito) 6](#_Toc194003537)

[Códigos 3xx (Redirección) 6](#_Toc194003538)

[Códigos 4xx (Errores del cliente) 7](#_Toc194003539)

[Códigos 5xx (Errores del servidor) 7](#_Toc194003540)

# *Protocolo HTTP*

## Para qué se utiliza el protocolo HTTP

El protocolo ***HTTP (Hypertext Transfer Protocol)*** se utiliza para la comunicación entre un cliente (navegador web) y un servidor web. Sus utilidades:

* *Permite solicitar y recibir páginas web, imágenes, videos y otros recursos de Internet.*
* *Funciona en un modelo cliente-servidor donde el cliente envía una petición GET y el servidor responde con el contenido solicitado.*
* *Es un protocolo sin estado, lo que significa que cada solicitud es independiente y el servidor no recuerda información de solicitudes anteriores.*

Un ejemplo de uso en una página web:

*Cuando escribes en el navegador:*

*http://www.ejemplo.com/index.html*

El navegador envía una solicitud HTTP al servidor *www.ejemplo.com* para obtener la página *index.html*

En definitiva, HTTP se usa para transmitir páginas web y otros recursos desde servidores hasta los navegadores, permitiendo la navegación en Internet.

## Principales propiedades

Las principales propiedades del protocolo HTTP son:

1. ***Protocolo sin estado (Stateless)***

* *Cada solicitud HTTP es independiente, es decir, el servidor no recuerda información de peticiones anteriores.*
* *Si se necesita persistencia, se usan tecnologías como cookies, sesiones o tokens.*

1. ***Basado en texto y fácil de leer***

* HTTP utiliza un formato de texto claro y entendible, lo que facilita la depuración y análisis.

1. ***Modelo Cliente-Servidor***

* Funciona en una estructura donde un cliente (navegador) envía peticiones y un servidor web responde.
* Ejemplo: cuando escribes "http://google.com", tu navegador solicita la página y el servidor la devuelve.

1. ***Comunicación basada en peticiones y respuestas***

* El cliente envía una solicitud (GET, POST, PUT, DELETE, etc.) y el servidor responde con código de estado y contenido.

1. ***Usa puertos estándar***

* HTTP utiliza el puerto 80 por defecto.
* HTTPS (versión segura) usa el puerto 443 con cifrado SSL/TLS.

1. ***Soporta múltiples métodos de petición***

* ***GET →*** Solicita recursos.
* ***POST →*** Envía datos al servidor.
* ***PUT →*** Modifica un recurso existente.
* ***DELETE →*** Elimina un recurso.
* ***HEAD, OPTIONS, PATCH, etc.***

1. ***No es seguro por sí mismo***

* HTTP transmite datos sin cifrar, por lo que puede ser interceptado.
* Para seguridad, se usa HTTPS (HTTP + SSL/TLS) que cifra la comunicación.

En resumen, HTTP es rápido y simple, pero no guarda estados ni es seguro por sí solo. Se usa para navegación web, APIs y transferencia de información.

## Etapas de una transacción HTTP

Una transacción HTTP sigue varias etapas desde que el cliente (navegador) solicita un recurso hasta que el servidor responde:

1. ***Conexión con el servidor***

* *El cliente (navegador) escribe una URL o hace clic en un enlace.*
* *Se resuelve la dirección IP del servidor mediante DNS.*
* *Se establece una conexión TCP con el servidor en el puerto 80 (HTTP).*

1. ***Envío de la solicitud HTTP***

* *El cliente envía una petición HTTP al servidor.*
* *La petición incluye:*
* *Método HTTP (GET, POST, PUT, DELETE, etc.).*
* *Ruta del recurso (/index.html).*
* *Versión de HTTP (HTTP/1.1).*
* *Encabezados (User-Agent, Host, Cookies, etc.).*

1. ***Procesamiento de la solicitud***

* *El servidor web recibe la petición.*
* *Busca el recurso solicitado (index.html).*
* *Si el recurso existe, lo prepara para enviarlo al cliente.*
* *Si el recurso no existe, genera un error (404 Not Found).*

1. ***Respuesta del servidor***

* *El servidor envía una respuesta HTTP con:*
* *Código de estado (200 OK, 404 Not Found, 500 Internal Server Error...).*
* *Encabezados (tipo de contenido, fecha, tamaño del recurso, etc.).*
* *Cuerpo del mensaje (HTML, JSON, imagen...).*

1. ***Renderización y carga de recursos adicionales***

* *El navegador interpreta el HTML recibido.*
* *Si la página contiene imágenes, CSS o scripts, se envían nuevas peticiones HTTP por cada recurso.*
* *Cada recurso sigue la misma secuencia de solicitud-respuesta.*

1. ***Cierre de la conexión***

* *En HTTP 1.0, la conexión TCP se cierra después de cada transacción.*
* *En HTTP 1.1, se puede reutilizar la conexión (Keep-Alive) para mejorar el rendimiento.*

## Grupos de Códigos de estado HTTP

Los códigos de estado HTTP indican el resultado de una solicitud realizada al servidor. Se dividen en 5 grupos, cada uno con un propósito específico:

### ***Códigos 1xx (Informativos)***

Indican que la solicitud fue recibida y el proceso continúa.

* *Ejemplos:*
  + **100 Continue →** El servidor ha recibido los encabezados y el cliente puede enviar el cuerpo de la solicitud.
  + **101 Switching Protocols →** El servidor acepta cambiar a un protocolo diferente (ej., de HTTP/1.1 a WebSockets).

### ***Códigos 2xx (Éxito)***

Significa que la solicitud fue recibida, entendida y procesada correctamente.

* *Ejemplos:*
  + **200 OK →** La solicitud fue exitosa y el servidor devuelve la respuesta esperada.
  + **201 Created →** El servidor ha creado un recurso como respuesta a la solicitud.
  + **204 No Content →** La solicitud fue exitosa, pero no hay contenido en la respuesta.

### ***Códigos 3xx (Redirección)***

Indican que se requiere una acción adicional para completar la solicitud.

* *Ejemplos:*
  + **301 Moved Permanently →** El recurso ha sido movido a otra URL de forma permanente.
  + **302 Found →** Redirección temporal a otra URL.
  + **304 Not Modified →** Indica que el recurso no ha cambiado y el cliente puede usar su copia en caché.

### ***Códigos 4xx (Errores del cliente)***

Errores causados por solicitudes incorrectas del cliente.

* *Ejemplos:*
  + **400 Bad Request →** La solicitud es inválida o está malformada.
  + **401 Unauthorized →** Se necesita autenticación para acceder al recurso.
  + **403 Forbidden →** El servidor rechaza la solicitud, aunque el usuario esté autenticado.
  + **404 Not Found →** El recurso solicitado no existe en el servidor.
  + **405 Method Not Allowed →** El método HTTP usado no está permitido en este recurso.

### ***Códigos 5xx (Errores del servidor)***

Errores causados por fallos en el servidor.

* *Ejemplos:*
  + **500 Internal Server Error →** Error interno en el servidor.
  + **502 Bad Gateway →** El servidor actúa como puerta de enlace y recibió una respuesta inválida de otro servidor.
  + **503 Service Unavailable →** El servidor no puede atender la solicitud por sobrecarga o mantenimiento.
  + **504 Gateway Timeout →** El servidor actuando como puerta de enlace no recibió respuesta a tiempo.

En resumen, los grupos de los códigos de estado de HTTP son:

***1xx: Informativos.***

***2xx: Éxito***

***3xx: Redirección***

***4xx: Error del cliente***

***5xx: Error del servidor***