Generalidades del protocolo HTTP

Iván Guadalupe Bustamante Cortés
lupitobtte_17@hotmail.com
Universidad de la Sierra Sur

2022/03/14

1. Introducción

El nombre completo del protocolo HTTP es del inglés HyperText Transfer Protocol o en español Protocolo de Transferencia de Hiper Textos y es un protocolo de transmisión de información de la World Wide Web.

HTTP es un protocolo de información de la red de internet, este protocolo es utilizado para solicitar una conexión web que es transmitida por la red. cuenta con cierta flexibilidad para incorporar nuevas peticiones y funcionalidades.

Para poder realizar una conexión con la red, el servidor brinda una respuesta que establecen el inicio, desarrollo y cierre de la transmisión de la información. A estos métodos se les conoce como métodos de petición.

Un ejemplo de esto es cuando en nuestro buscador abrimos una pagina web, el protocolo HTTP sera el encargado del intercambio de información entre el usuario y el servidor, como saber en que posición se mostrara cierta información al igual que, qué imágenes se van a mostrar.

2. Desarrollo

2.1. Mensajes HTTP

"Los mensajes HTTP, son los medios por los cuales se intercambian datos entre servidores y clientes." (mdn web docs, 2022) por lo cual entendemos que los mensajes es la forma en la que se comunica el usuario con servidor para solicitar información sobre alguna pagina web. Entre estos mensajes podemos encontrar dos tipos los cuales son de peticiones que son enviadas por el cliente al servidor, para pedir el inicio de una acción; y de respuestas, que son la respuesta del servidor.

Conforme pasa el tiempo, estos mensajes han ido evolucionando para así poder optimizar su uso, por ejemplo, en HTTP/1.1, y versiones previas del protocolo, estos mensajes

eran enviados de forma abierta a través de la conexión. En HTTP/2.0 los mensajes, que anteriormente eran legibles directamente, se conforman mediante tramas binarias codificadas para aumentar la optimización y rendimiento de la transmisión.

Estos mensajes raramente son programados por los desarrolladores puesto que normalmente son originados mediante archivos de configuración (para proxies, y servidores), APIs (para navegadores) y otros medios.

2.2. Métodos HTTP

Los métodos en HTTP son ocupados la acción que realizara cierto recurso o cierto elemento de la aplicación o pagina web, entre los métodos mas conocidos se encuentran los siguientes:

- GET: "GET es la madre de todas las peticiones de HTTP. Este método de petición existía ya en los inicios de la world wide web y se utiliza para solicitar un recurso, como un archivo HTML, del servidor web.
  Cuando escribes la dirección URL www.ejemplo.com en tu navegador, este se conecta con el servidor web y le envía una petición GET" (IONOS, 2020)

- POST: "Cuando se tienen que enviar al servidor web paquetes grandes de datos, como imágenes o datos de formulario de carácter privado, por ejemplo, el método GET se queda corto, porque todos los datos que se transmiten se escriben en abierto en la barra de direcciones del navegador." (IONOS, 2020)

- HEAD: "El método de petición de HTTP HEAD se utiliza para solicitar que el servidor solo envíe el encabezado de la respuesta, sin el archivo. Esta alternativa es conveniente cuando se han de transferir archivos muy voluminosos, ya que, con esta petición, el cliente conoce primero el tamaño del archivo para luego poder decidir si acepta recibirlo o no." (IONOS, 2020)

- OPTIONS: "Con el método OPTIONS, el cliente puede preguntar al servidor qué métodos soporta para el archivo de que se trata."

- RACE: "Con el método TRACE, puede seguirse la ruta que sigue una HTTP Request hacia el servidor y, desde allí, de regreso al cliente. Este seguimiento puede ejecutarse en Windows con el comando tracert." (IONOS, 2020)

- PUT: "El modo PUT reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición." (mdn web docs, 2022)

- DELETE: "El método DELETE borra un recurso en específico."(mdn web docs, 2022)

- CONNECT: "El método CONNECT establece un túnel hacia el servidor identificado por el recurso." (mdn web docs, 2022)

2.3. Códigos de respuesta HTTP

Los códigos de respuesta en HTTP te ayudan a conocer el estado en el que se encuentra tu página web, esto facilita en muchas ocasiones la resolución de problemas. Los códigos HTTP están especificados por el RFC 2616.

Para funcionar adecuadamente "El primer dígito del código de estado especifica uno de los 5 tipos de respuesta, el mínimo para que un cliente pueda trabajar con HTTP es que reconozca estas 5 clases. La Internet Assigned Numbers Authority (IANA) mantiene el registro oficial de códigos de estado HTTP." (Diego Lazaro, 2008)

- 1XX Respuestas informativas: Este tipo de código de estado indica una respuesta provisional.
- 2XX Peticiones correctas: Este código de estado indica que la acción solicitada por el cliente ha sido recibida, entendida, aceptada y procesada correctamente.
- 3XX Redirecciones: Muchos de estos estados se utilizan para redirecciones.
- 4XX Errores del cliente: Excepto cuando se responde a un HEAD request, el servidor debe incluir una entidad que contiene una explicación del error, y si es temporal o permanente.
- 5XX Errores del servidor: El servidor ha fallado al completar una solicitud aparentemente válida. Cuando los códigos de estado empiezan por 5 indica casos en los que el servidor sabe que tiene un error o realmente es incapaz de procesar el request.

Código de respuesta HTTP	Ejemplo de código de error de WebSphere MQ Managed File Transfer	Ejemplo de descripción
200 OK	Ninguno	Se ha manejado correctamente una solicitud válida y, opcionalmente, se ha proporcionado una respuesta al usuario.
400 Bad Request	BFGWI0001	El URI no es válido porque le falta un tipo de recurso.
403 Forbidden	BFGWI0056	No se ha definido ningún identificador de usuario de IBM® WebSphere MQ Message Descriptor (MQMD) para el usuario.
404 Not Found	BFGWI0015	No se puede encontrar el recurso solicitado.
405 Method Not Allowed	BFGWI0016	El recurso solicitado no da soporte al verbo HTTP que se ha utilizado en la solicitud. Por ejemplo, se ha utilizado GET con un recurso que sólo permite POST o DELETE.
410 Resource Gone	BFGWI0031	El recurso solicitado ya no está disponible. Por ejemplo, el archivo solicitado ha sido suprimido del espacio de archivos.
413 Request Entity Too Large	BFGWI0026	La solicitud contiene un archivo que es demasiado grande para ser manejado por el servidor.
415 Unsupported Media Type	BFGWI0017	Se ha recibido una solicitud con un tipo de medio, especificado por la cabecera HTTP Content-type, que no está soportado.
500 Internal Server Error	BFGWI0018	Se ha encontrado un error interno al gestionar la solicitud. Se ha producido un archivo FFDC o ABEND.
502 Bad Gateway	BFGWI0019	La solicitud no se puede completar porque se ha producido un error fuera de WebSphere MQ Managed File Transfer. Por ejemplo, una cola de IBM WebSphere MQ no está disponible.
503 Service Unavailable	BFGWI0020	El destino no está disponible temporalmente. Por ejemplo, una cola de IBM WebSphere MQ está llena.
504 Gateway Timeout	BFGWI0021	Un intento de completar la solicitud ha sobrepasado el tiempo de espera debido a los límites de tiempo impuestos por WebSphere MQ Managed File Transfer, o debido a los límites de tiempo impuestos por el cliente HTTP.

2.4. Cabeceras HTTP

"Las Cabeceras HTTP son los parámetros que se envían en una petición o respuesta HTTP al cliente o al servidor para proporcionar información esencial sobre la transacción en curso. Estas cabeceras proporcionan información mediante la sintaxis 'Cabecera: Valor' y son enviadas automáticamente por el navegador o el servidor Web." (Wikipedia, 2020)

Las cabeceras son similares a los códigos de respuesta ofrecidos por HTTP, pues indican el estado en el cual se encuentra la petición del servidor. "Las cabeceras HTTP son la parte central de esas solicitudes y respuestas HTTP, y transportan información sobre el navegador cliente, la página solicitada, el servidor y más." (Guzel, 2021)

2.5. Ejemplo de dialogo HTTP

```
GET /index.html HTTP/1.1
Host: www.example.com
Referer: www.google.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0
Connection: keep-alive
[Línea en blanco]
```

La respuesta del servidor está formada por encabezados seguidos del recurso solicitado, en el caso de una página web:

```
HTTP/1.1 200 OK
Date: Fri, 31 Dec 2003 23:59:59 GMT
Content-Type: text/html
Content-Length: 1221

<html lang="eo">
<head>
<meta charset="utf-8">
<title>Título del sitio</title>
</head>
<body>
<h1>Página principal de tuHost</h1>
(Contenido)
  .
  .
  .
</body>
</html>
```

3. Conclusiones

En conclusión conocer todo lo relacionado al protocolo HTTP es de gran importancia debido a que esto puede ayudar a solucionar problemas a futuro ocurridos con los servidores web, también ayuda a conocer que es la información que se intercambia entre el usuario y el servidor. Este protocolo es fácil de comprender y de usar, pues está basado en la estructura de cliente-servidor, debido a su capacidad de usar cabeceras, permite a este protocolo evolucionar con las nuevas y futuras aplicaciones en Internet.

Es preciso comentar sobre los códigos de respuestas que debería ser un tema que todos conozcan, pues facilitaría a los usuarios en gran ocasión para conocer la razón por la cual algunas veces sus páginas no responden, como por ejemplo los códigos 4XX y 5XX.

4. Bibliografía

Diego Lazaro. (5 de Abril de 2018). Obtenido de https://diego.com.es/codigos-de-estado-http

Guzel, B. (12 de Mayo de 2021). *envatotuts*. Obtenido de https://code.tutsplus.com/es/tutorials/http-headers-for-dummies--net-8039

IONOS. (15 de Julio de 2020). Obtenido de https://www.ionos.mx/digitalguide/hosting/cuestiones-tecnicas/http-request/

mdn web docs. (12 de Marzo de 2022). Obtenido de https://developer.mozilla.org/es/docs/Web/HTTP/Overview

mdn web docs. (12 de Marzo de 2022). Obtenido de https://developer.mozilla.org/es/docs/Web/HTTP/Methods