Generalidades del protocolo HTTP

Floriceli Gonzalez Ruiz
gonzalezruizfloriceli183@gmail.com
Universidad de la Sierra Sur

14/03/2022

1 Introduction

En este ensayo principalmente se habla de las generalidades del protocolo HTTP ya que este ensayo nos permitirá tener un mejor conocimiento del protocolo HTTP que es muy importante y necesario hoy en día, el cual es un protocolo que nos sirve para transmisión de información de la World Wide Web y con este protocolo también se establecen criterios de sintaxis y semántica informática para poder establecer comunicación y este protocolo se podría decir que funciona por medio de peticiones y respuestas entre el servidor web y el cliente y estos ocupan como medios para intercambiar datos a los mensajes, pero este protocolo define un conjunto de métodos de petición para poder indicar que acciones se quieren realizar y el cual también contienen códigos de respuesta que nos muestras si la solicitud que realizamos se hizo exitosamente y cabeceras que se pueden definir como parámetros que tienen información importante sobre la transacción.

2 Desarrollo

2.1 Generalidades del protocolo HTTP

HTTP es un protocolo el cual 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, y un protocolo de estructura cliente-servidor, esto quiere decir que una petición de datos es iniciada por el elemento que recibirá los datos (el cliente), normalmente un navegador Web. Así, una página web completa resulta de la unión de distintos sub-documentos recibidos, como, por ejemplo: un documento que especifique el estilo de maquetación de la página web (CSS), el texto, las imágenes, vídeos, scripts como otros. *(Generalidades del protocolo HTTP,*(2022))

2.2 Mensajes HTTP

Los mensajes HTTP, son los medios por los cuales se intercambian datos entre servidores y clientes. Hay dos tipos de mensajes: peticiones, enviadas por el cliente al servidor, para pedir el inicio de una acción; y respuestas, que son la respuesta del servidor.
Los mensajes HTTP están compuestos de texto, codificado en ASCII, y pueden comprender múltiples líneas. 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.

Los desarrolladores de páginas Web, o administradores de sitios Web, desarrolladores a veces codifican directamente estos mensajes HTTP. Normalmente especifican estos mensajes HTTP, mediante archivos de configuración (para proxies, y servidores), APIs (para navegadores) y otros medios.

El mecanismo de tramas binarias de HTTP/2 ha sido diseñado para que no necesite ninguna modificación de las APIs o archivos de configuración utilizados: es totalmente transparente para el usuario.
Las peticiones y respuestas HTTP, comparten una estructura similar, compuesta de:

Una línea de inicio ('start-line' en inglés) describiendo la petición a ser implementada, o su estado, sea de éxito o fracaso. Esta línea de comienzo, es siempre una única línea. Un grupo opcional de cabeceras HTTP, indicando la petición o describiendo el cuerpo ('body' en inglés) que se incluye en el mensaje. Una línea vacía ('empty-line' en inglés) indicando toda la meta-información ha sido enviada. Un campo de cuerpo de mensaje opcional ('body' en inglés) que lleva los datos asociados con la petición (como contenido de un formulario HTML), o los archivos o documentos asociados a una respuesta (como una página HTML, o un archivo de audio, vídeo ...) . La presencia del cuerpo y su tamaño es indicada en la línea de inicio y las cabeceras HTTP.

La línea de inicio y las cabeceras HTTP, del mensaje, son conocidas como la cabeza de la peticiones, mientras que su contenido en datos se conoce como el cuerpo del mensaje. *(Mensajes HTTP*,(2022))

2.3 Peticiones HTTP

Las peticiones HTTP son mensajes enviados por un cliente, para iniciar una acción en el servidor. Su línea de inicio está formada por tres elementos:
Un método HTTP, un verbo como: GET, PUT o POST) o un nombre como: HEAD (en-US) o OPTIONS (en-US)), que describan la acción que se pide sea realizada. Por ejemplo, GET indica que un archivo ha de ser enviado hacia el cliente, o POST indica que hay datos que van a ser enviados hacia el servidor (creando o modificando un recurso, o generando un documento temporal para ser enviado). El objetivo de una petición, normalmente es una URL, o la dirección completa del protocolo, puerto y dominio también suelen ser especificados por el contexto de la petición. El formato del objetivo de la petición varia según los distintos métodos HTTP. Puede ser:
Una dirección absoluta, seguida de un signo de cierre de interrogación '?' y un texto de consulta. Este es el formato

más comun, conocido como el formato original ('origin form' en inglés), se usa en los métodos GET, POST, HEAD, y OPTIONS . POST / HTTP 1.1
GET /background.png HTTP/1.0
HEAD /test.html?query=alibaba HTTP/1.1
OPTIONS /anypage.html HTTP/1.0
Una URL completa; conocido como el formato absoluto, usado mayormente con GET cuando se conecta a un proxy.
GET http://developer.mozilla.org/en-US/docs/Web/HTTP/Messages HTTP/1.1
El componente de autoriade de una URL, formado por el nombre del domínio y opcionalmente el puerto (el puerto precedido por el simbolo ':'), se denomina a este formato como el formato de autoridad. Unicamente se usa con CONNECT cuando se establece un tunel HTTP.
CONNECT developer.mozilla.org:80 HTTP/1.1
El formato de asterisco, se utliza un asterisco ('*') junto con las opciones: OPTIONS , representando al servidor entero en conjunto.
OPTIONS * HTTP/1.1
la versión de HTTP, la cual define la estructura de los mensajes, actuando como indicador, de la versión que espera que se use para la respuesta. *(Métodos de petición HTTP*,(2022))

2.4 Códigos de respuesta HTTP

El código de respuesta o retorno es un número que indica que ha pasado con la petición. El resto del contenido de la respuesta dependerá del valor de este código. El sistema es flexible y de hecho la lista de códigos ha ido aumentando para así adaptarse a los cambios e identificar nuevas situaciones. Cada código tiene un significado concreto. Sin embargo el número de los códigos están elegidos de tal forma que según si pertenece a una centena u otra se pueda identificar el tipo de respuesta que ha dado el servidor.
Los códigos HTTP están estandarizados y se recogen en el registro de códigos de estado HTTP de la IANA (Internet Assigned Numbers Authority).

Estos códigos HTTP se clasifican en 5 tipos. El primer dígito del código es el que corresponde al tipo de respuesta a la que nos enfrentamos: respuestas informativas, respuestas satisfactorias, redirecciones, errores del navegador y errores de los servidores.

Códigos con formato 1xx. Son respuestas de carácter informativo e indican que el navegador puede continuar con la petición. Como no reflejan ningún error, no se muestran al usuario.
Códigos con formato 2xx. El conjunto de códigos de estado HTTP del tipo 2xx son respuestas satisfactorias, Simplemente indican que la petición fue procesada correctamente, por lo que lo ideal es que todas las webs devuelvan este código HTTP. Generalmente, como la petición fue exitosa, no se muestra el código de estado HTTP, el navegador

únicamente devuelve el contenido que el usuario solicitó.

Códigos con formato 3xx. Estos códigos HTTP hacen referencia a cuando el navegador tiene que realizar una acción adicional como, por ejemplo, una redirección 301.

Códigos con formato 4xx Los códigos de estado que comienzan con el dígito 4 hacen referencia a errores producidos por el navegador web. En estos casos, el usuario recibe una página en HTML en la que es informado del error.

Códigos con formato 5xx Estos códigos HTTP también muestran errores, pero por el lado del servidor web. *(Códigos de respuesta HTTP,*(2022))

2.5 Cabeceras HTTP

Las cabeceras (en inglés headers) HTTP permiten al cliente y al servidor enviar información adicional junto a una petición o respuesta. Una cabecera de petición esta compuesta por su nombre (no sensible a las mayusculas) seguido de dos puntos ':', y a continuación su valor (sin saltos de línea). Los espacios en blanco a la izquierda del valor son ignorados. Se pueden agregar cabeceras propietarias personalizadas usando el prefijo 'X-', pero esta convención se encuentra desfasada desde Julio de 2012, debido a los inconvenientes causados cuando se estandarizaron campos no estandar en el RFC 6648; otras están listadas en un registro IANA, cuyo contenido original fue definido en el RFC 4229, IANA tambien mantiene un registro de propuestas para nuevas cabeceras HTTP
Las Cabeceras pueden ser agrupadas de acuerdo a sus contextos:

Cabecera general: Cabeceras que se aplican tanto a las peticiones como a las respuestas, pero sin relación con los datos que finalmente se transmiten en el cuerpo.

Cabecera de consulta: Cabeceras que contienen más información sobre el contenido que va a obtenerse o sobre el cliente.

Cabecera de respuesta:: Cabeceras que contienen más información sobre el contenido, como su origen o el servidor (nombre, versión, etc.).

Cabecera de entidad: Cabeceras que contienen más información sobre el cuerpo de la entidad, como el tamaño del contenido o su tipo MIME.

Las cabeceras también pueden clasificarse de acuerdo a cómo se comportan frente a ellas los proxies:

Cabeceras de extremo a extremo: Estas cabeceras deben ser enviadas al recipiente final del mensaje; esto es, el servidor (para una petición) o el cliente (para una respuesta). Los proxies intermediarios deben transmitir las cabeceras de extremo-a-extremo sin modificar, y las cachés deben guardarlas tal y como son recibidas.

Cabeceras de paso: Estas cabeceras sólo son significativas para conexiones de un paso, y no deben ser transmitidas por proxies o almacenarse en caché. Éstas cabeceras son: Connection (en-US), Keep-Alive, Proxy-Authenticate (en-US), Proxy-Authorization (en-US), TE (en-US), Trailer (en-US), Transfer-Encoding and Upgrade (en-US). La

cabecera general Connection (en-US) sólo puede usarse para este tipo de cabeceras.
(HTTP headers,(2022))

2.6 Ejemplos de dialogo HTTP

Un ejemplo de petición GET es:

```
    GET  /dir/cargaPagina.php?id=21&nombre=Pepe  HTTP/1.1
    <cabeceras>
```

Este ejemplo, convertido a petición POST es:

```
    POST  /dir/cargaPagina.php  HTTP/1.1
    <cabeceras>
    id=21&nombre=Pepe
```

Vemos que los parámetros se pasan en el cuerpo de la petición, fuera de la línea del comando.
Comúnmente existen 3 formas de enviar una petición GET:

```
    Teclear la petición directamente en la barra del navegador:
    http://www.xx.com/pag.html?id=123&nombre=pepe
    Colocar la petición en un enlace y pinchar el enlace para realizarla:
    <a href="http://www.xx.com/pag.html?id=123&nombre=pepe">Pulsa Aqui</a>
    Enviar la petición tras rellenar un formulario con method="get" (o sin method) con los dos parámetros
    <html><body>
        <form action="http://www.xx.com/pag.html">
            <input type="text" name="id" value="123">
            <input type="text" name="nombre" value="pepe">
            <input type="submit" value="Enviar">
        </form>
    </body></html>
```

Para enviar una petición POST, normalmente se utiliza un formulario con method="post":

```
<html><body>
    <form action="http://www.xx.com/pag.html" METHOD=POST>
        <input type="text" name="id" value="123">
        <input type="text" name="nombre" value="pepe">
        <input type="submit" value="Enviar">
    </form>
</body></html>
\textit{(Ejemplo de dialogo HTTP},(2022))
```

3 Conclusiones

Se llego a la conclusión que HTTP es uno de los protocolos mas importante en la trasmisión de información de datos en la World Wide Wed ya que hace diferentes funciones importantes en la Web ya que es un protocolo estructura cliente-servidor en donde se comunican intercambiando mensajes individuales ya que también se llega a entender que este protocolo no solo nos sirve para transmitir documentos de hipertext, sino también para transmitir imágenes, vídeos, enviar datos o contenido a los servidores los cuales son necesario en la actualidad ya que esto nos facilitan las diversas actividades que se realizan diariamente.

4 Referencias

Mensajes HTTP - HTTP | MDN. Developer.mozilla.org. (2022). Retrieved 15 March 2022, from https://developer.mozilla.org/es/docs/Web/HTTP/Messages.

HTTP headers - HTTP | MDN. Developer.mozilla.org. (2022). Retrieved 15 March 2022, from https://developer.mozilla.org/es/docs/Web/HTTP/Headers.

Códigos de respuesta HTTP. (2022). Retrieved 15 March 2022, from https://www.lucushost.com/blog/codigos-http-mas-comunes/.

Métodos de petición HTTP - HTTP | MDN. Developer.mozilla.org. (2022). Retrieved 15 March 2022, from https://developer.mozilla.org/es/docs/Web/HTTP/Methods.

Generalidades del protocolo HTTP - HTTP | MDN. Developer.mozilla.org. (2022). Retrieved 15 March 2022, from https://developer.mozilla.org/es/docs/Web/HTTP/Overview.

perfil, V. (2022). Ejemplo de dialogo HTTP. Retrieved 15 March 2022, from http://danihttp.blogspot.com/2012/06/ejemplo-de-dialogo-http.html