OBJETO PETICIÓN

Tiempo estimado: 15min

Un buen conocimiento de HTTP y de los mensajes de petición y respuesta es muy, pero que muy recomendable cuando desarrollamos aplicaciones webs, sobre todo, bajo Express. En esta lección, vamos a centrarnos en las peticiones. En otra lección posterior, presentaremos más detenidamente los de respuesta.

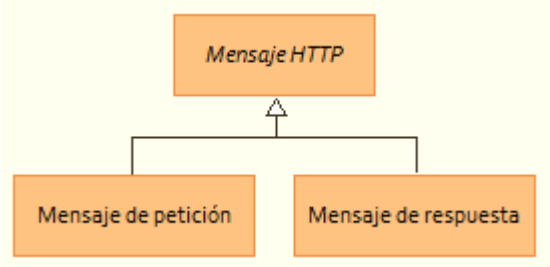
Comenzamos presentando los mensajes HTTP y los dos tipos existentes: petición y respuesta. A continuación, describimos detalladamente los mensajes de petición, principalmente, con objeto de solicitar copias de recursos mediante el método o verbo GET. Después, presentamos el programa cURL para enviar peticiones desde la línea de comandos. Finalmente, presentamos los métodos más utilizados del estándar HTTP, los campos de cabecera más importantes relacionados con las peticiones y el objeto Request con el que acceder, en una aplicación Express, a la información de la petición en curso.

Al finalizar la lección, el estudiante sabrá:

- Qué es un mensaje HTTP.
- Qué es un mensaje de petición HTTP.
- Qué es cURL y para qué usarlo.
- Cuáles son los métodos HTTP más utilizados.
- Cuáles son los principales campos de cabecera de un mensaje de petición HTTP.
- Cómo acceder a la información de la petición en curso en una aplicación Express.

INTRODUCCIÓN

Un mensaje HTTP (*HTTP message*) es una unidad de información a través de la cual dos entidades se comunican entre sí. Son de texto, aunque pueden contener datos binarios. Atendiendo a quién envía el mensaje, se distingue entre mensajes de petición y de respuesta.



Un mensaje de petición (*request message*) es aquel a través del cual se solicita una copia de un recurso como, por ejemplo, una imagen, un vídeo, un audio, un documento HTML, etc. La entidad que lo envía se conoce como cliente siendo generalmente un navegador web a petición del usuario. Y a la entidad que lo recibe y responde como servidor, en nuestro caso, un servidor web como Apache o Nginx o una aplicación Express.

Por otro lado, tenemos el mensaje de respuesta (*response message*), aquel que se envía como contestación a una petición y contiene, generalmente, una copia del recurso solicitado.

MENSAJES DE PETICIÓN

Tal como acabamos de ver, un mensaje de petición (*request message*) es aquel a través del cual se solicita la copia de un recurso web. Aunque no siempre es así. En algunas ocasiones, los clientes pueden remitir información a un servidor o aplicación mediante un mensaje de este tipo. No siempre tienen como objeto solicitar algo, más bien, solicitar o proporcionar algo a la otra entidad. Digamos que es el mensaje a partir del cual comienza una

comunicación entre un cliente y un servidor.

Los clientes redactan este tipo de mensajes y se lo remite a los servidores o aplicaciones webs. Quienes finalmente lo reciben, procesan y responden.

Este tipo de mensajes son de texto y se estructuran como sigue:

```
Línea de petición
Cabecera del mensaje
Línea en blanco
Cuerpo del mensaje
```

Línea de petición

La línea de petición (*request line*) comienza el mensaje y tiene como objeto indicar la versión del protocolo HTTP usada en la redacción del mensaje, el recurso sobre el que se desea hacer algo y qué se desea a hacer con él. Su sintaxis es la siguiente:

```
Método URI Versión-HTTP
```

El método (*method*), también conocido como verbo (*verb*), indica la acción a realizar sobre el recurso como, por ejemplo, obtener una copia, escribir su contenido en el servidor, modificarlo, suprimirlo, etc. A continuación, le sigue un espacio y el identificador del recurso al que aplicar la acción. Finalmente, otro espacio y la versión HTTP bajo la que se redactó el mensaje. La línea de petición acaba con un salto de línea (CRLF).

A guisa de ejemplo, veamos una línea de petición ilustrativa:

```
GET /index.html HTTP/1.1
```

Cabecera del mensaje

La cabecera del mensaje (*message header*) es el fragmento en el que se proporciona información adicional sobre el mensaje y su contenido. Está formado por uno o más campos de cabecera (*header fields*), una línea de texto que proporciona información específica de un aspecto del mensaje. Los campos tienen un nombre, que no distingue entre mayúsculas y minúsculas, y un valor, separados por dos puntos (:):

```
Nombre: Valor
```

Por ejemplo, para indicar el tipo de contenido de un mensaje, se utiliza el campo de cabecera Content-Type. Ejemplo:

```
Content-Type: text/html
```

Son muchos los campos de cabecera de los mensajes HTTP. Algunos son específicos de los mensajes de petición, otros de las respuestas y otros son comunes a ambos tipos. Iremos presentándolos a lo largo del curso a medida que sea necesario, en vez de hacerlo todo de golpe de manera indiscriminada.

Para ir abriendo boca, vamos a mostrar un mensaje de petición que solicita un determinado recurso con varias cabeceras:

```
GET /index.html HTTP/1.1
Host: mi.punto.com
User-Agent: miapp.exe
```

Cuerpo del mensaje

El cuerpo del mensaje (*message body*) es la parte que lleva, tiene o transporta los datos enviados por el cliente al servidor o a la aplicación web. Es opcional, pero necesario cuando deseamos crear o modificar un determinado recurso web. En caso de aparecer en el mensaje, debe separarse de la cabecera por una línea en blanco (CRLF).

cURL

cURL es un programa de línea de comandos a través del cual enviar mensajes de petición HTTP, HTTPS, FTP y otros protocolos. Es muy útil cuando deseamos realizar pruebas sobre nuestra aplicación o simplemente deseamos enviar un determinado mensaje a un servidor HTTP.

Su página oficial es curl.haxx.se. En ella, se puede encontrar ejecutables para distintos sistemas operativos como Linux y Windows. En distribuciones Debian, lo más sencillo es instalar el paquete curl mediante apt. En Windows 10, es posible que disponga de un alias de cURL al comando Invoke-WebRequest. Si desea utilizar el verdadero

cURL, instálelo y suprima este alias mediante:

```
> Remote-Item alias:curl
```

A continuación, vamos a solicitar la página de inicio de www.google.es mediante cURL:

```
> curl www.google.es -v -o archivo.html
```

Con la opción -v, solicitamos a cURL que muestre el mensaje de petición enviado y el mensaje de respuesta recibido del servidor web. Con la opción -o, el archivo donde almacenar la copia del recurso recibido; si no se indica, se mostrará por la consola.

A lo largo del curso, iremos presentando las opciones más utilizadas.

Métodos HTTP

Una de las primeras cosas que debemos conocer de los mensajes de petición HTTP es el concepto de método. Un método (*method*) o verbo (*verb*) es una palabra que indica lo que deseamos hacer con el recurso accedido. El estándar HTTP define varios métodos que los servidores implementan completa o parcialmente. Y por ende, debe implementar nuestra aplicación.

Veamos los más conocidos y utilizados actualmente:

- El método GET se utiliza para solicitar una copia del recurso al servidor o aplicación web.
- El método HEAD tiene como objeto solicitar al servidor o aplicación web que responda como a un GET, pero sin el contenido del recurso.

  Generalmente, se usa para comprobar la accesibilidad al recurso. Muy útil para confirmar la validez actual de enlaces almacenados en una aplicación o base de datos.
- El método POST se utiliza para proporcionar contenido al servidor o aplicación para su almacenamiento. Generalmente, se usa cuando se desea crear un nuevo recurso cuyo contenido es el suministrado en el cuerpo del mensaje de petición.
- El método PUT, por su parte, se utiliza para actualizar el contenido de un recurso. Básicamente, reemplazarlo por otro.
- El método PATCH se utiliza para actualizar el contenido del recurso indicado, pero sólo parcialmente.

  La diferencia entre PUT y PATCH es sutil pero importante: PUT reemplaza *todo* el recurso; mientras que PATCH sólo *parte* de él.
- El método DELETE se utiliza para solicitar la supresión de un determinado recurso.
- El método OPTIONS sirve para obtener la lista de métodos o verbos HTTP que se puede aplicar a un determinado recurso.

De manera predeterminada, cURL envía mensajes de petición GET. Si en algún momento necesitamos fijar otro método, utilizaremos las siguientes opciones:

- -G o --get para GET.
- -I o --head para HEAD.
- -X método o --request método para cualquier otro método.

He aquí un ejemplo de mensaje de petición HEAD:

```
HEAD / HTTP/1.1
Host: www.microsoft.es
User-Agent: curl/7.49.1
Accept: */*
```

Y ahora, cómo enviarlo mediante cURL:

```
> curl --head www.microsoft.es
```

Campos de cabecera

Recordemos que tras la línea de petición se puede indicar campos de cabecera con los que proporcionar

información adicional del mensaje. A continuación, vamos a presentar algunos campos de cabecera que todo desarrollador debe conocer necesariamente. Otros los iremos presentando a lo largo del curso a medida que sea necesario.

Pero antes de hacerlo, vamos a ver cómo indicar campos de cabecera mediante cURL. Para ello, se usa la opción -H o --header, por favor, no confundir con --head. Veamos un ejemplo:

```
> curl --get --header "User-Agent: My Node.js client" http://redis.io
```

Campo de cabecera Host

El campo de cabecera Host indica el dominio donde se encuentra el recurso accedido. No es necesario indicarlo, pero sí muy recomendable. Tengamos en cuenta que un mismo servidor o aplicación web puede servir recursos de distintos dominios como, por ejemplo, de mi.dominio.es y my.domain.com. De alguna manera, el cliente debe indicarle al servidor en cuál, de todos los dominios que aloja y sirve, se encuentra el recurso accedido.

Por ejemplo, si ejecutamos lo siguiente:

```
> curl --head https://mariadb.com/products/mariadb-enterprise
```

cURL enviará un mensaje de solicitud como el siguiente:

```
HEAD /products/mariadb-enterprise HTTP/1.1
Host: mariadb.com
User-Agent: curl/7.49.1
Accept: */*
```

Observe que la URL se descompone en dos partes: la ruta del recurso que se indica en la línea de petición y el dominio en el campo de cabecera Host.

Campo de cabecera User-Agent

Mediante el campo de cabecera User-Agent, el cliente puede indicar, al servidor, la aplicación que ha enviado la petición HTTP. Por ejemplo, Chrome añade algo como lo siguiente a sus peticiones:

```
Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103
Safari/537.36
```

Mientras que cURL algo como sigue:

```
curl/7.49.1
```

La información del cliente usado es muy útil cuando se quiere conocer qué aplicaciones o navegadores utilizan los usuarios a la hora de conectar a nuestro sitio web. Por lo tanto, no es raro que los servidores o aplicaciones webs la almacenen para su posterior análisis.

Campo de cabecera Referer

Mediante Referer, el cliente puede indicar desde qué página está accediendo al recurso. Muy útil para conocer qué sitios webs externos están enlazando con el nuestro. Al igual que User-Agent, las aplicaciones suelen almacenar esta información para su posterior análisis.

Objeto request

Recordemos que un servidor o aplicación web tiene como objeto procesar peticiones HTTP y responder a los remitentes. En una aplicación Express, un buen conocimiento del concepto de mensaje de petición, de su sintaxis y de los campos de cabecera más utilizados es de vital importancia. Por otra parte, no hay que olvidar que en las aplicaciones Express a veces hay que fijar controladores de petición para determinados recursos y/o métodos HTTP.

En Express, por convenio, se utiliza el parámetro request o simplemente req de los controladores de petición para representar la petición en curso. En las funciones de *middleware* normales, este parámetro es el primero. Consiste en una instancia de la clase Request que contiene propiedades y métodos a través de los cuales obtener la información de la petición.

Cómo consultar la línea de petición mediante Express

Recordemos que la línea de petición contiene la información referente al método o verbo, la versión de HTTP

utilizada en la redacción del mensaje y el recurso accedido. Esta información se puede obtener mediante los siguiente miembros.

La propiedad req.method (string) contiene el método especificado en el mensaje en forma de texto, esto es, GET, HEAD, POST, PUT, DELETE, etc.

La versión de HTTP se puede conocer mediante la propiedad req.httpVersion (string).

Para obtener la información del recurso accedido, disponemos de varios miembros:

- La propiedad req.protocol (string) contiene el protocolo utilizado: http o https.
- La propiedad req.originalUrl (string) contiene la ruta completa, con la cadena de consulta, del recurso solicitado.
- La propiedad req.path (string) representa la ruta del recurso accedido, sin cadena de consulta.
- La propiedad req.query (object) contiene los parámetros de la cadena de consulta.

Cabecera del mensaje

Por su parte, la cabecera del mensaje contiene información adicional del mensaje, en forma de campos. Para acceder a esta información disponemos del método req.get(), a través del cual obtener el valor de un determinado campo de cabecera:

```
function get(field) : object
```

Parámetro	Tipo de datos	Descripción
field	string	Campo de cabecera.

He aquí un ejemplo ilustrativo:

```
var host = req.get("Host");
```

Cuerpo del mensaje

El acceso al cuerpo del mensaje se puede realizar mediante la propiedad req.body (object). Más adelante en el curso, hablaremos detenidamente del cuerpo y de cómo acceder a él.

