Nombre: Adrian Iza

NRC:3700

Protocolo http

HTTP define un conjunto de **métodos de petición** para indicar la acción que se desea realizar para un recurso determinado. Aunque estos también pueden ser sustantivos, estos métodos de solicitud a veces son llamados *HTTP verbs*. Cada uno de ellos implementan una semántica diferente

GET

El método get solicita una representación de un recurso específico. Las peticiones que usan el método GET sólo deben recuperar datos.

HEAD

El método head pide una respuesta idéntica a la de una petición GET, pero sin el cuerpo de la respuesta.

POST

El método post se utiliza para enviar una entidad a un recurso en específico, causando a menudo un cambio en el estado o efectos secundarios en el servidor.

PUT

El modo put reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.

DELETE

El método delete borra un recurso en específico.

CONNECT

El método connect establece un túnel hacia el servidor identificado por el recurso.

OPTIONS

El método options es utilizado para describir las opciones de comunicación para el recurso de destino.

TRACE

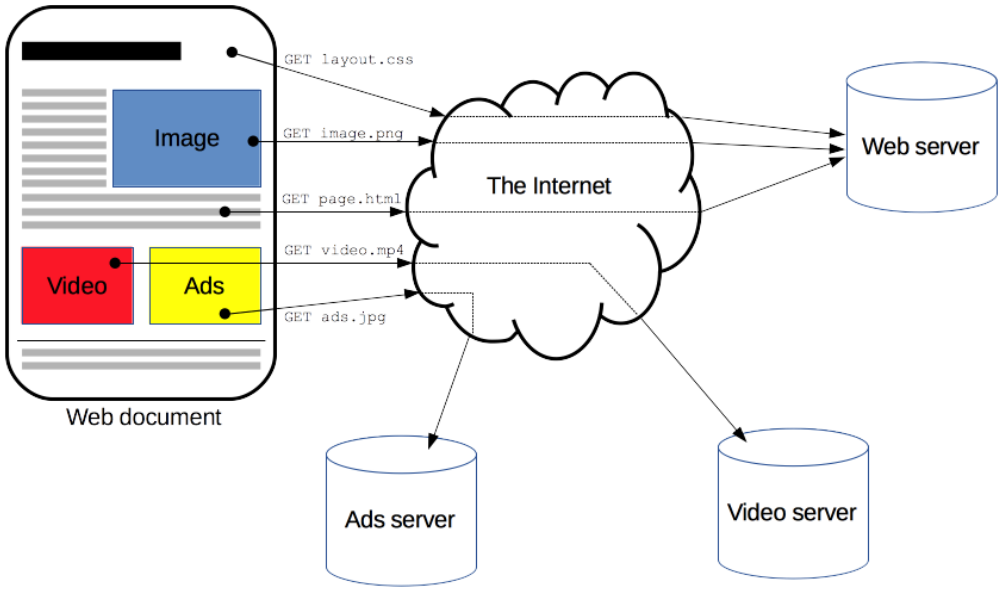
El método trace realiza una prueba de bucle de retorno de mensaje a lo largo de la ruta al recurso de destino.

PATCH

El método patch es utilizado para aplicar modificaciones parciales a un recurso.

GENERALIDADES DEL PROTOCOLO HTTP

HTTP, de sus siglas en inglés: "Hypertext Transfer Protocol", es el nombre de 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, etc...



Clientes y servidores se comunican intercambiando mensajes individuales (en contraposición a las comunicaciones que utilizan flujos continuos de datos). Los mensajes que envía el cliente, normalmente un navegador Web, se llaman *peticiones*, y los mensajes enviados por el servidor se llaman *respuestas*.

CARACTERÍSTICAS CLAVE DEL PROTOCOLO HTTP

HTTP es sencillo

Incluso con el incremento de complejidad, que se produjo en el desarrollo de la versión del protocolo HTTP/2, en la que se encapsularon los mensajes, HTTP esta pensado y desarrollado para ser leído y fácilmente interpretado por las personas, haciendo de esta manera más facil la depuración de errores, y reduciendo la curva de aprendizaje para las personan que empieza a trabajar con él.

HTTP es extensible

Presentadas en la versión HTTP/1.0, las cabeceras de HTTP, han hecho que este protocolo sea fácil de ampliar y de experimentar con él. Funcionalidades nuevas pueden desarrollarse, sin más que un cliente y su servidor, comprendan la misma semántica sobre las cabeceras de HTTP.

HTTP es un protocolo con sesiones, pero sin estados

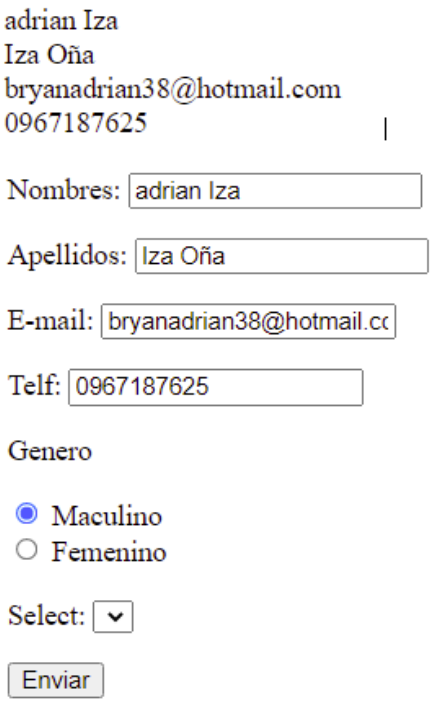
HTTP es un protocolo sin estado, es decir: no guarda ningún dato entre dos peticiones en la mísma sesión. Esto crea problemáticas, en caso de que los usuarios requieran interactuar con determinadas páginas Web de forma ordenada y coherente

HTTP y conexiones

Una conexión se gestiona al nivel de la capa de trasporte, y por tanto queda fuera del alcance del protocolo HTTP. Aún con este factor, HTTP no necesita que el protocolo que lo sustenta mantenga una conexión continua entre los participantes en la comunicación, solamente necesita que sea un protocolo fiable o que no pierda mensajes (como mínimo, en todo caso, un protocolo que sea capaz de detectar que se ha pedido un mensaje y reporte un error).

EJEMPLO

Post




```php
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Documento sin título</title>
</head>

<body>
    <?php
    //declarar variables
    $nombre=$_POST['nombre'];
    $apellido=$_POST['apellido'];
    $email=$_POST['email'];
    $telefono=$_POST['telefono'];
    $genero=$_POST['genero'];

    echo $nombre;
    echo("<br>");
    echo $apellido;
    echo("<br>");
    echo $email;
    echo("<br>");
    echo $telefono;
    echo("<br>");
    echo $genero;
        echo("<br>");

    ?>
</body>
</html>
```

Get

adrian Iza
Iza Oña
bryanadrian38@hotmail.com
0967187625

Nombres: adrian Iza

Apellidos: Iza Oña

E-mail: bryanadrian38@hotmail.cc

Telf: 0967187625

Genero

- (•) Maculino
- ( ) Femenino

Select:

Enviar

HEAD

```
1  HEAD /downloads/video1.mpeg HTTP/1.0
```

En el encabezado que el servidor le envía de respuesta, el cliente encuentra los datos sobre el tamaño del archivo en el campo “content-length”:

age	96529
cache-control	max-age=604800
content-length	1495
content-type	text/html; charset=UTF-8
date	Fri, 06 Mar 2020 14:53:39 GMT
etag	"3147526947+gzip"
expires	Fri, 13 Mar 2020 14:53:39 GMT
last-modified	Thu, 17 Oct 2019 07:18:26 GMT
server	ECS (dcb/7F83)
vary	Accept-Encoding
x-cache	HIT

OPTIONS

```
1 OPTIONS /download.php
```

La respuesta podría consistir en algo parecido a esto:

allow	OPTIONS, GET, HEAD, POST
cache-control	max-age=604800
content-length	0
content-type	text/html; charset=UTF-8
date	Fri, 06 Mar 2020 14:06:08 GMT
expires	Fri, 13 Mar 2020 14:06:08 GMT
server	EOS (vny/0452)

Respuesta del servidor a la petición OPTIONS

Tracert

```
1 tracert www.example.com
```

PROTOCOLO HTTPS

HTTPS (HyperText Transfer Protocol Secure, Protocolo de transferencia de hipertexto) es un protocolo de comunicación de Internet que protege la integridad y la confidencialidad de los datos de los usuarios entre sus ordenadores y el sitio web.

El envío de datos mediante el protocolo HTTPS está protegido con el protocolo *Seguridad en la capa de transporte* (Transport Layer Security, TLS), que proporciona estas tres capas de seguridad principales:

1. **Cifrado:** se cifran los datos intercambiados para mantenerlos a salvo de miradas indiscretas. Eso significa que cuando un usuario está navegando por un sitio web, nadie puede "escuchar" sus conversaciones, hacer un seguimiento de sus actividades por las diferentes páginas ni robarle información.
2. **Integridad de los datos:** los datos no pueden modificarse ni dañarse durante las transferencias, ni de forma intencionada ni de otros modos, sin que esto se detecte.
3. **Autenticación:** demuestra que tus usuarios se comunican con el sitio web previsto. Proporciona protección frente a los ataques de intermediario y fomenta la confianza de los usuarios, lo que se traduce en otros beneficios empresariales.

CARACTERÍSTICAS HTTPS

-Una conexión HTTPS a un website puede ser validada si y solo si todo siguiente es verdad:

-El usuario confía en la Autoridad de certificación para dar fe solo para websites legitimos sin nombres engañosos.

-El website proporciona un certificado válido (y un certificado inválido muestra una alerta en la mayoría de los navegadores), lo que significa que está firmado por una autoridad confiable. El certificado identifica correctamente al website (p.e. visitando

https://algunsitio y recibiendo un certificado para "AlgunSitio S.A."

-Cada uno de los nodos involucrados en internet son dignos de confianza, o que el usuario confíe en que la capa de cifrado del protocolo (TLS o SSL) es inquebrantable por un eaves dropper.

-HTTPS utiliza un cifrado basado en SSL/TLS para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP. De este modo se consigue que la información sensible (usuario y claves de paso normalmente) no pueda ser usada por un atacante que haya conseguido interceptar la transferencia de datos de la conexión, ya que lo único que obtendrá será un flujo de datos cifrados que le resultará imposible de descifrar

El puerto estándar para este protocolo es el 443

CAPAZ DE RED

HTTP opera en la capa más alta del Modelo OSI, la Capa de Aplicación; pero el protocolo HTTPS opera en una subcapa más baja, cifrando un mensaje HTTP previo a la transmisión y descifrando un mensaje una vez recibido

Estrictamente hablando, HTTPS no es un protocolo separado, pero refiere el uso del HTTP ordinario sobre una Capa de Conexión Segura cifrada Secure Sockets Layer (SSL) o una conexión con Seguridad de la Capa de Transporte (TLS)

Bibliografia

https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/http-request/

https://developers.google.com/search/docs/advanced/security/https?hl=es

https://developer.mozilla.org/es/docs/Web/HTTP/Methods