EL PROTOCOLO HTTP

Eduard Lara

INDICE

1. EVOLUCIÓN HISTÓRICA DE LA WEB

- La Web nace al 1989 a partir de un proyecto del CERN (Consejo Europeo para la Investigación Nuclear)
- Tim Berners-Lee construye el prototipo que dio lugar a lo que hoy en día se conoce como www (World Wide Web).
- La intención original era hacer más fácil compartir textos de investigación entre científicos y permitir al lector revisar las referencias de un articulo mientras lo iba leyendo.
- Un sistema de hipertexto enlazaría todos los documentos entre si para que el lector pudiera revisar las referencias de un artículo mientras fuera leyendo

1. EVOLUCIÓN HISTÓRICA DE LA WEB

- Inicialmente el programa del CERN, sólo consideraba texto. Viola de Pei Wei (1992) añadió la capacidad de presentar gráficos.
- Marc Andreessen presentó un navegador web llamado "Mosaic" en 1993 que disparó la popularidad de la web.
- Andreesen fundó Mosaic Communications Corporation (después Netscape) añadiendo características como contenido dinámico, música y animaciones.
- La capacidad de los navegadores y servidores fue más rápida que la evolución de los estándares. Al inicio las nuevas funciones no estaban soportadas en todos los navegadores: IE, Mozilla Firefox, Opera, Safari, Amaya, Galeon, Konqueror, Lynx, Netscape Navigator

2. ESTANDARES DE LA WEB

La web se basa en tres estándares:

- **El Localizador Uniforme de Recursos (URL),** que especifica como cada página de información se asocia a una dirección única donde se puede localizar;
- **El protocolo de Transferencia de Hipertexto (HTTP),** que especifica cómo el navegador y el servidor intercambian información en forma de peticiones y respuestas.
- **El lenguaje de Marcación de Hipertexto (HTML),** un método para codificar la información de los documentos y sus enlaces. Berners-Lee dirige en la actualidad el World Wide Web Consortium, que desarrolla y mantiene éste y otros estándares

3. SERVICIOS WEB DE INTERNET

- El servicio web de Internet se basa en la transmisión de páginas web.
- Las páginas web se encuentran almacenadas en un ordenador (o en varios) capaz de funcionar como un servidor web.
- El servicio Web funciona siguiendo el denominado modelo cliente - servidor: habitual en las aplicaciones que funcionan en una red.
  - Servidor → Quien presta el servicio.
  - Cliente → Quien lo recibe.

3. SERVICIOS WEB DE INTERNET

SERVIDOR WEB

- Un servidor web recibe peticiones de clientes y responde con el envío de ficheros solicitados, texto plano (html, php) o binarios (gif, jpeg).
- Permanentemente escucha las peticiones de conexión de los clientes en determinados puertos: 80 para HTTP, 443 para el HTTPS
- La atención a la petición del cliente consiste en buscar el archivo solicitado. Si lo encuentra, lo transmite; sino envía un mensaje de error.
- El servidor web comprueba si el usuario tiene acceso a los documentos.

3. SERVICIOS WEB DE INTERNET

CLIENTE WEB

- Programa con el que el usuario interacciona para solicitar a un servidor el envío de páginas web.
- Utiliza los protocolos HTTP o FTP
- Las páginas web están codificadas en HTML. El cliente web interpreta estos documentos para mostrárselos al usuario en el formato adecuado.
- Cuando un documento recibido es un objeto multimedia (vídeo o sonido), el cliente activa una aplicación externa capaz de gestionarlo.
- Clientes web o navegadores mas usuales: Netscape, IE Explorer, Mozilla

3. SERVICIOS WEB DE INTERNET

PROCESO TRANSFERENCIA

- El usuario especifica en el cliente web la URL de la pagina que desea consultar.
- El cliente establece la conexión con el servidor web y solicita la pagina deseada.
- El servidor busca la pagina solicitada en su sistema de ficheros. Si la encuentra la transfiere, sino devuelve un código de error.
- El cliente interpreta el código HTML y muestra la pagina al usuario.
- Se cierra la conexión. La conexión siempre se libera al terminar la transmisión de la pagina.

3. SERVICIOS WEB DE INTERNET

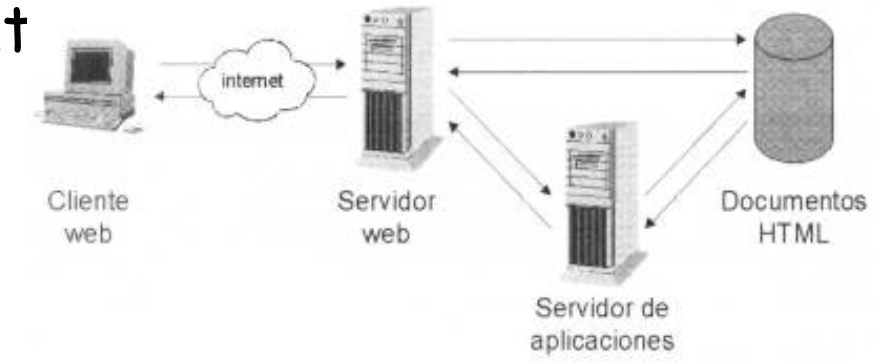
MODELO TRANSACCIONAL

- Las aplicaciones cliente-servidor siguen el denominado modelo de transacciones: el cliente realiza una petición, el servidor la atiende e inmediatamente se cierra la comunicación.
- Se establece una transacción independiente para cada documento u objeto que se transmite.
- No se mantiene memoria entre las sucesivas peticiones (acceso banco, sitios seguros, etc)
- El uso de cookies o acceso a base de datos resuelve este problema

3. SERVICIOS WEB DE INTERNET

PÁGINAS DINÁMICAS

- Son páginas creadas en el momento de su petición (combinan una plantilla de documento con los resultados de la consulta a una BD.
- El servidor web cede el control al denominado **servidor de aplicaciones**, que es quien se encarga de construir la pagina.
- Una vez creada la pasa al servidor web, que a su vez la envía al client



4. PROTOCOLO HTTP

- HTTP (HyperText Transfer Protocol) es el protocolo utilizado en cada transacción de la web (www).
- A través de dicho protocolo podemos emitir peticiones HTTP de acceso a una página web desde el cliente (navegador web) hasta el servidor.
- Una vez tramitada la petición, el servidor devolverá una respuesta HTTP al cliente, cuyo contenido estará en Hipertexto (HTML), y que éste interpretará mostrando el resultado al usuario.
- HTTP sirve para enviar información con mensajes mediante formularios.
- La versión actual de HTTP es la 1.1, y su especificación está en el documento RFC2616.

4. PROTOCOLO HTTP

- HTTP es un protocolo transaccional: Es un protocolo sin estado, es decir, que no almacena información sobre conexiones anteriores.
- Está basado en el modelo cliente - servidor: Un cliente HTTP abre una conexión y realiza una solicitud al servidor. Este responde al cliente y al finalizar la transacción, se cierra la conexión y se pierden los datos
- Por este motivo se han popularizado las **cookies**, pequeños ficheros almacenados en el propio ordenador, que puede leer una página web al establecer conexión con él, y de esta forma reconocer un visitante antiguo.

5. URL

- Para poder emitir peticiones necesitamos de una URL.
- Una URL es un nombre canónico que localiza un recurso específico en Internet. Está formado por:

```
protocolo://host:puerto/ruta/archivo
```

  - **protocolo**; normalmente usaremos http o https para peticiones seguras.
  - **host**; es el nombre o IP del servidor
  - **puerto**; puerto de escucha para las aplicaciones web. Suele ser 8080 o únicamente 80.
  - **ruta**; es el context path de la aplicación. Es un nombre que hace referencia a la carpeta raíz de la aplicación.
  - **archivo**; puede ser una página web, un servlet o cualquier recurso de la aplicación.

6. TIPOS DE PETICIONES HTTP

- HTTP es un protocolo de internet que regula la transmisión y visualización de las páginas web.
- Implementa diferentes formas de solicitar las páginas a los servidores en los que están alojadas.
- Aparte de GET y POST, existen más métodos en HTTP:
  - HEAD Es similar a GET, pero sólo envía las cabeceras de respuesta ignorando el contenido.
  - PUT: Permite subir ficheros al servidor
  - DELETE: Borra un archivo en el servidor
  - TRACE: Permite comprobar si ha habido alguna modificación del contenido, desde que sale del servidor hasta que llega al cliente.
  - OPTIONS: Muestra los métodos aceptados por el servidor.
  - CONNECT: Normalmente usado para pasar a modo seguro.
  - PATCH: Para modificar un recurso determinado.

6.1. MÉTODO GET

- Es el más simple de los dos.

- Cuando el usuario pulsa cualquier enlace de la página que está visualizando, se envía al servidor una cadena de texto con información sobre la nueva página solicitada para que nos la envíe.
- Si le hemos solicitado una página que existe, nos devolverá un código 200 y la página en cuestión.
- Si no existe esa página, nos devolverá un 404 indicando una página de error con un mensaje similar a "recurso no encontrado".

6.1. MÉTODO GET

- Al hacer click sobre un enlace, el navegador traduce esa acción en la petición de la página al servidor.
- En esa petición se utiliza GET. Le indica al servidor que le de el fichero index.html para mostrárselo al usuario.
- Cuando pulsamos en un link a www.aaa.com/pagina.html, la petición GET que se envía al servidor es similar a:

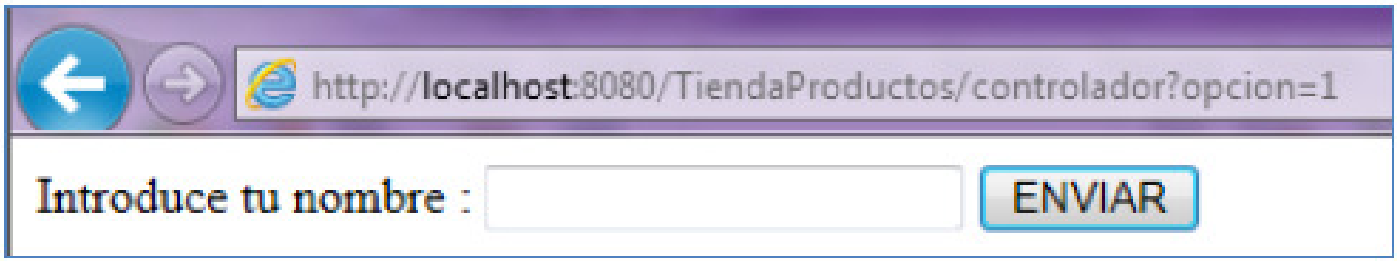
GET /pagina.html HTTP/1.1 Host: www.aaa.com

  - GET le indica al servidor el método HTP que queremos ejecutar
  - /index.html le dice qué página queremos visualizar
  - HTTP/1.1 indica al servidor la versión de HTTP que nuestro navegador utiliza
  - Host: www.aaa.com indica el nombre del servidor al que le estamos solicitando la página

6.1. MÉTODO GET

Paso de parámetros entre páginas web con GET

- Se envían los datos en la url para su procesado.
- Los parámetros se ven en la URL.
- En la siguiente imagen vemos como en la url se separa la lista de parámetros con el carácter "?". El parámetro opción viaja con el valor 1. Si hubiese más de un parámetro estos irían unidos con el carácter "&".



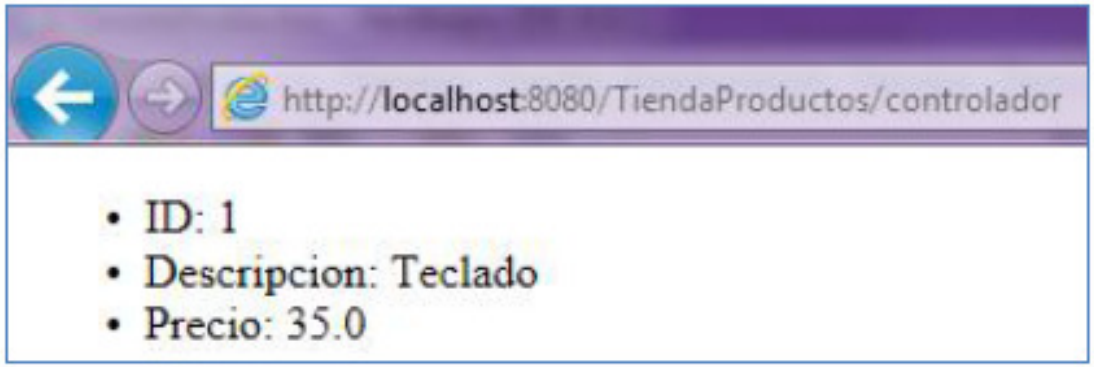
6.2. MÉTODO POST

- Es similar al funcionamiento de GET, pero si GET es para pedir datos al servidor, POST está más orientado a ENVIAR información HACIA el servidor.

- Como se puede ver en el diagrama, hay un paso extra en el caso de POST que corresponde con el guardado de la información que el usuario ha enviado.
- Se suele decir, que después de una petición GET el servidor no cambia de estado, y que después de una petición POST sí que cambia de estado.

6.2. MÉTODO POST

- El método POST es más avanzado.
- Podemos solicitar algo y al mismo tiempo enviar datos de un formulario al servidor. Los parámetros están ocultos, se envían en el cuerpo del mensaje.
- Ejemplo:



- Vemos que se ha enviado el código del producto a buscar utilizando el método post. Esta vez no vemos el parámetro con su valor correspondiente ya que este viaja dentro del cuerpo de la petición.

6.2. MÉTODO POST

Razones para usar POST en vez de GET:

- El numero de caracteres es limitado usando GET (dependiendo del servidor).
- Los datos enviados con GET se añaden al final de la URL.
- No permite almacenar los argumentos del formulario

7. ENVIO DE PARÁMETROS EN UNA URL

Podemos adjuntar nuestros propios parámetros sobreescribiendo una url. Esto consiste en añadir el carácter separación (?) y a continuación detallar los parámetros necesarios.

```
<a href="controlador?opcion=1">Consultar todos los productos</a><br>
```

8. ENVIO DE PARÁMETROS DE UN FORMULARIO

- A través del atributo name podemos establecer el nombre del parámetro. El valor será el que rellene el usuario.
- También podemos hacer uso de los campos ocultos de esta forma el usuario no verá el dato que enviamos.
- Lo aconsejable con campos ocultos es utilizar el método POST ya que si utilizásemos GET estos se verían en la url.

```
<form action="controlador" method="post">
    Introduce Id del producto:
    <input type="text" name="codigo" />
    <input type="hidden" name="opcion" value="2" />
    <input type="submit" value="ENVIAR" />
</form>
```

9. CÓDIGOS DE RESPUESTA HTTP

La línea inicial es diferente en las solicitudes y en las respuestas.

Solicitudes HTTP

En las solicitudes vemos campos separados por un espacio en blanco:

Método recurso versión_del_protocolo P.e.: "GET /recursos/notas/notas.html HTTP/1.1"

Respuesta HTTP

La línea inicial de respuesta tiene 3 campos separados por un espacio:

Versión_del_protocolo codigo_respuesta mensaje P.e.: " HTTP/1.0 200 Ok" ó "HTTP/1.1 404 Not found"

9. CÓDIGOS DE RESPUESTA HTTP

1xx Mensaje de información 100 Continua 101 Cambio de protocolo	**3xx Redirección hacia otra URL** 300 Múltiples posibilidades 301 Mudado permanentemente 302 Encontrado 303 Ver otros 304 No modificado 305 Utiliza un proxy 307 Redirección temporal
2xx Operación satisfactoria 200 OK 201 Creado 202 Aceptado 203 Información no oficial 204 Sin Contenido 205 Contenido para recargar 206 Contenido parcial	**5xx Error por parte del servidor** 500 Error interno 501 No implementado 502 Pasarela incorrecta 503 Servicio no disponible 504 Tiempo de espera de la pasarela agotado 505 Versión de HTTP no soportada

9. CÓDIGOS DE RESPUESTA HTTP

4xx Error por parte del cliente	408 Tiempo de espera agotado
400 Solicitud incorrecta	409 Conflicto
401 No autorizado	410 Ya no está disponible
402 Pago requerido	411 Requiere longitud
403 Prohibido	412 Error precondición
404 No encontrado	413 Entidad de solicitud demasiado larga
405 Método no permitido	414 URL de solicitud demasiado larga
406 No aceptable	415 Tipo de medio no soportado
407 Proxy requerido	416 Rango solicitado no disponible
	417 Error expectative

RECUERDA QUE...

- Podemos emitir peticiones a través de los métodos GET o POST.
- Con el método GET los parámetros de la petición se envían en la propia url por lo cual son visibles.
- Con el método POST los parámetros viajan en el cuerpo de la petición por lo cual no son visibles.