

INTERNET

La **World Wide Web (WWW)**, es un **sistema de acceso y búsqueda de la información** disponible en Internet, cuyas **unidades** informativas son las **páginas web** y que están **enlazadas entre si**, **accesibles vía internet con un navegador web**.

WWW Propuso al **hipertexto** "para **vincular y acceder a información** de diversos tipos **como una red de nodos** en los que el usuario puede navegar. El **hipertexto** es una **herramienta que permite crear, agregar, enlazar y compartir información de diversas fuentes por medio de enlaces** asociativos.

HTTP (Hypertext Transfer Protocol) es un **protocolo de petición/respuesta usado en cada transacción de la World Wide Web**.

TCP/IP son las siglas de **Protocolo de Control de Transmisión/Protocolo de Internet**.

TCP/IP provee **conectividad de extremo a extremo** especificando cómo los **datos** deberían ser **formateados**, **direccionados**, **transmitidos**, **enrutados** y **recibidos** por el destinatario.

El **Protocolo de Internet (IP)** **utiliza direcciones** que son **series de cuatro** números **octetos** (byte) con un formato de punto decimal, por ejemplo: 69.5.163.59

También **proporciona un mecanismo para distinguir distintas aplicaciones dentro de una misma máquina, a través del concepto de Puerto**.

INTERNET

Métodos de petición

HTTP define 8 métodos que indica la acción que desea que se efectúe sobre el recurso identificado.

HEAD, Pide una respuesta idéntica a la que correspondería a una petición GET, pero sin el cuerpo de la respuesta. Esto es útil para la recuperación de meta-información escrita en los encabezados de respuesta, sin tener que transportar todo el contenido.

Solicitar

GET, Pide una representación del recurso especificado (el que invoco). Por seguridad no debería ser usado por aplicaciones que causen efectos ya que transmite información a través de la URI agregando parámetros a la URL.

Enviar

POST, Somete los datos a que sean procesados para el recurso identificado (el que invoco). Los datos se incluirán en el cuerpo de la petición. Esto puede resultar en la creación de un nuevo recurso o de las actualizaciones de los recursos existentes o ambas cosas.

PUT, Sube, carga o realiza un upload de un recurso especificado (archivo), es el camino más eficiente para subir archivos a un servidor.

DELETE, Borra el recurso especificado.

TRACE, Este método solicita al servidor que envíe de vuelta en un mensaje de respuesta. Se utiliza con fines de comprobación y diagnóstico.

OPTIONS, Devuelve los métodos HTTP que el servidor soporta para un URL específico. Esto puede ser utilizado para comprobar la funcionalidad de un servidor web.

CONNECT

Comunicarse
con el
servidor

INTERNET

El protocolo usado para comunicarse con un servidor web es: "HTTP", y normalmente usamos HTTP GET o HTTP POST, para cualquier tecnología de servidor.

method = get | post

El atributo Método especifica qué método HTTP se usará para enviar la información.

GET: envía información a través de la URL

Los valores del formulario se envían al servidor Web como texto a continuación del signo de interrogación de la dirección URL de la petición.

Recibo como Request.QueryString("nombre del objeto")

Cuando utiliza el método GET de HTTP para pasar a un servidor Web valores de un formulario grande y complejo, corre el riesgo de perder información. Algunos servidores Web tienden a restringir el tamaño de la cadena de petición de URL.

POST: envía información con el formulario.

POST, envía los datos de los formularios en el cuerpo de la petición http

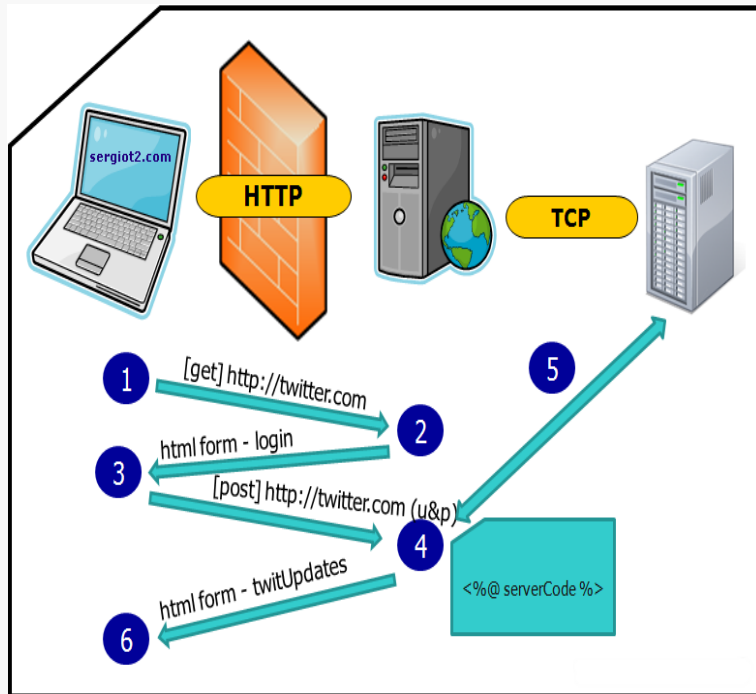
EJEMPLO GET

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
<FORM ACTION="PaginaA.aspx" METHOD="GET">
  Introduzca su nombre:<INPUT TYPE="text" NAME="nombre">
  Introduzca sus apellidos:<INPUT TYPE="text" NAME="apellidos">
  <INPUT TYPE="submit" VALUE="Enviar">&nbsp;&nbsp;&nbsp;
</FORM>
</body>
</html>
```

EJEMPLO POST

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Untitled Page</title>
</head>
<body>
<FORM ACTION="PaginaA.aspx" METHOD="POST">
  Introduzca su nombre:<INPUT TYPE="text" NAME="nombre">
  Introduzca sus apellidos:<INPUT TYPE="text" NAME="apellidos">
  <INPUT TYPE="submit" VALUE="Enviar">&nbsp;&nbsp;&nbsp;
</FORM>
</body>
</html>
```

INTERNET



1. El usuario tiene que ingresar la URL de la página en su navegador (*1). El navegador por detrás se encargará de hacer un request (solicitud) al servidor Web usando el protocolo de comunicación HTTP(*2) (Internet), y en este caso usará el método GET, por que sólo quiere obtener información.

2. El servidor Web recibe el request y envía un response (sólo HTML) al navegador. Los navegadores no entienden el código ASP, PHP, o JSP, ellos sólo muestran contenido en HTML (*3), es por eso que todos los servidores Web después de procesar un request devuelven sólo HTML (que puede incluir JavaScript (*4)), el HTML generado debe ser un formulario en HTML, para que el usuario pueda enviar su información.

3. El usuario llena su información, user y password, y hace clic en el famoso botón "Sign in". El navegador por detrás recolectará esta información, y en este caso que se desea enviar esa información al servidor debe estar usando el método POST. Todos los lenguajes usan POST para enviar información a una página, ya sea ASP.NET, Php, JSP, etc (*5). Con GET también se puede enviar variables, pero no es técnicamente enviar información, es mas bien, un obtener información con estos parámetros.

4. El request llega al servidor Web, y se ejecutará el código de servidor Php, Jsp, o ASP, que se conectará con la base para verificar si existe el usuario y si el password coincide con el enviado por el usuario.

5. Si el usuario y el password son validos, el código de servidor (login.php, login.jsp, o login.aspx), redireccionará el request a otra página showUpdates.php, la cual se conecta nuevamente a la base de datos para traer todos los updates de los amigos del usuario, después de procesar la página, el servidor envía el response (sólo HTML) al usuario.

6. El usuario ve en una página las últimas actualizaciones de sus amigos, y parece que esta semana no habrá @BeerTwit, así que tendrá que inventar alguna excusa para generar uno nuevo.

INTERNET

Ejemplo de un diálogo HTTP

Ingresamos a algún sitio como <http://www.codes.com.ar>. ¿Qué ocurre por debajo?

El navegador realiza varios pasos que son invisibles para nosotros y que vamos a describir a continuación:

1. Traducir el dominio a la dirección IP del servidor.
2. Abrir una conexión con el servidor.
3. Enviar un mensaje HTTP solicitando el recurso.
4. Interpretar el mensaje HTTP de respuesta.

1 - Traducción del dominio

Todo dispositivo conectado a Internet (computador, tablet, smarphone, etc.) está identificado por una dirección IP única (p.e. 31.13.69.160). Los dominios (p.e. www.google.com, www.facebook.com) no son más que un nombre de fácil recordación para los humanos.

Cuando ingresamos el nombre de un dominio en el navegador, se hace un llamado a un DNS (Domain Name Server) que hace la traducción del dominio a la dirección IP. Un DNS es un servicio con una gran base de datos que contiene la dirección IP a la que está asociada cada dominio en Internet. Cuando alguien registra un nuevo dominio, el registrador debe avisar a todos los DNS's (públicos y privados) que actualicen sus bases de datos.

2 - Abrir la conexión

Una vez que el navegador tiene la dirección IP del servidor, el siguiente paso es abrir una conexión a un puerto específico en el servidor. Generalmente omitimos el puerto cuando escribimos una dirección en el navegador; en ese caso se asume el puerto 80. Es posible especificar explícitamente el puerto de la siguiente forma: <http://www.google.com:80/>.

3 - Enviar el mensaje HTTP

Una vez que la conexión con el servidor está abierta, el navegador envía un mensaje de acuerdo al protocolo HTTP.

Un cliente, por ejemplo un navegador, envía un mensaje al servidor en el que especifica el método, URI (la ruta del recurso que estamos solicitando), la versión del protocolo, y, opcionalmente, el cuerpo con el contenido del mensaje. El servidor responde con otro mensaje que incluye el código de estado de la respuesta, algunos encabezados, y el cuerpo con el contenido de la respuesta.

Veamos un ejemplo de un mensaje de petición HTTP (asumiendo que ya tenemos una conexión abierta al servidor de Google):

GET /Doc.html / HTTP/1.1

Host: www.google.com

En la primera línea del mensaje estamos solicitando el recurso /doc.html. La segunda línea es un encabezado. Este mensaje no lleva cuerpo.

INTERNET

4 - Recibir el mensaje de respuesta HTTP

Después de que el navegador ha enviado una petición, el servidor retorna una respuesta de regreso. Por ejemplo, la respuesta de la petición anterior es la siguiente:

```
HTTP/1.1 404 Not Found
Content-Type: text/html; charset=UTF-8
X-Content-Type-Options: nosniff
Date: Sun, 05 Oct 2014 17:49:34 GMT
Server: sffe
Content-Length: 1433
X-XSS-Protection: 1; mode=block
Alternate-Protocol: 80:quic,p=0.002
```

```
<!DOCTYPE html>
<html lang=en>
  <meta charset=utf-8>
  <meta name=viewport content="initial-scale=1, minimum-scale=1, width=device-width">
  <title>Error 404 (Not Found)!!1</title>
  <style>
    ...
  </style>
  <a href=//www.google.com/><span id=logo aria-label=Google></span></a>
  <p><b>404.</b> <ins>That's an error.</ins>
  <p>The requested URL <code>/doc.html</code> was not found on this server. <ins>That's all we know.</ins>
```

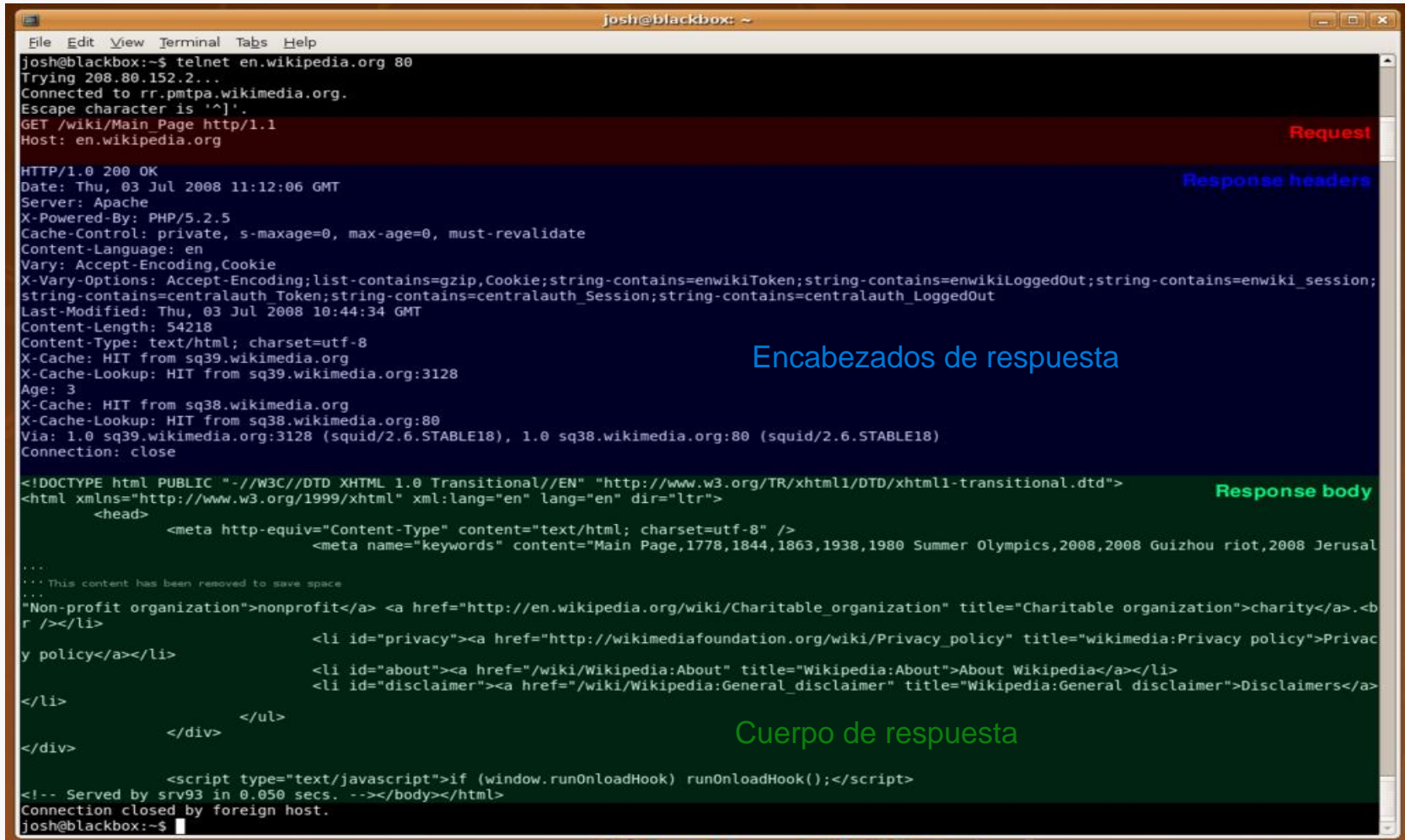
HTML

HTML es un lenguaje que nos permite definir la estructura de un documento que los navegadores interpretan y muestran en la pantalla.

Generalmente, el código HTML (Hyper Text Markup Language) viaja a través de HTTP (Hyper Text Transfer Protocol).

INTERNET

IMAGEN DIALOGO HTTP



```
josh@blackbox: ~  
File Edit View Terminal Tabs Help  
josh@blackbox:~$ telnet en.wikipedia.org 80  
Trying 208.80.152.2...  
Connected to rr.pmta.wikimedia.org.  
Escape character is '^]'.  
GET /wiki/Main Page http/1.1  
Host: en.wikipedia.org  
  
HTTP/1.0 200 OK  
Date: Thu, 03 Jul 2008 11:12:06 GMT  
Server: Apache  
X-Powered-By: PHP/5.2.5  
Cache-Control: private, s-maxage=0, max-age=0, must-revalidate  
Content-Language: en  
Vary: Accept-Encoding, Cookie  
X-Vary-Options: Accept-Encoding;list-contains=gzip, Cookie;string-contains=enwikiToken;string-contains=enwikiLoggedOut;string-contains=enwiki_session;  
string-contains=centralauth_Token;string-contains=centralauth_Session;string-contains=centralauth_LoggedOut  
Last-Modified: Thu, 03 Jul 2008 10:44:34 GMT  
Content-Length: 54218  
Content-Type: text/html; charset=utf-8  
X-Cache: HIT from sq39.wikimedia.org  
X-Cache-Lookup: HIT from sq39.wikimedia.org:3128  
Age: 3  
X-Cache: HIT from sq38.wikimedia.org  
X-Cache-Lookup: HIT from sq38.wikimedia.org:80  
Via: 1.0 sq39.wikimedia.org:3128 (squid/2.6.STABLE18), 1.0 sq38.wikimedia.org:80 (squid/2.6.STABLE18)  
Connection: close  
  
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en" dir="ltr">  
  <head>  
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />  
    <meta name="keywords" content="Main Page,1778,1844,1863,1938,1980 Summer Olympics,2008,2008 Guizhou riot,2008 Jerusal  
    ...  
    ... This content has been removed to save space  
    ...  
    <li id="privacy"><a href="http://en.wikipedia.org/wiki/Charitable_organization" title="Charitable organization">charity</a><br /></li>  
    <li id="about"><a href="http://wikimediafoundation.org/wiki/Privacy_policy" title="wikimedia:Privacy policy">Privac  
y policy</a></li>  
    <li id="disclaimer"><a href="/wiki/Wikipedia:About" title="Wikipedia:About">About Wikipedia</a></li>  
    <li id="disclaimer"><a href="/wiki/Wikipedia:General_disclaimer" title="Wikipedia:General disclaimer">Disclaimers</a>  
  </li>  
  </ul>  
</div>  
  
  <script type="text/javascript">if (window.runOnLoadHook) runOnLoadHook();</script>  
<!-- Served by srv93 in 0.050 secs. --></body></html>  
Connection closed by foreign host.  
josh@blackbox:~$
```

Request

Response headers

Encabezados de respuesta

Response body

Cuerpo de respuesta