

PROTOCOLO HTTP

Salvador Mira

IES Pere Maria Orts



INTRODUCCIÓN

HTTP significa Hypertext Transfer Protocol

Es el protocolo fundamental de la WEB

Ampliamente utilizado por distintos tipos de dispositivo

Funciona en la capa de aplicación de OSI

HTTP/1.1 está definido en el **RFC 2616**, y actualmente actualizado en los **RFC 7230 – 7237**

HTTP/2 está definido en el **RFC 7540** y actualiza las características de **HTTP/1.1** sobre todo en el **manejo de streams**

HTTP funciona sobre **TCP**



INTRODUCCIÓN

Se utiliza para transferir recursos entre un servidor y un cliente

Un recurso es un fragmento de información que puede identificarse mediante una URL

Los recursos suelen ser archivos albergados en un servidor, pero también pueden ser datos generados dinámicamente



TRANSACCIONES HTTP

HTTP hace uso de un modelo **cliente/servidor**

- Los clientes abren conexiones con los servidores para solicitar recursos (**http request**)
- Los servidores envían mensajes en respuesta a las peticiones de recurso (**http response**)
- Cuando finalizan cada respuesta, cierran la conexión

HTTP es un **protocolo sin estado**

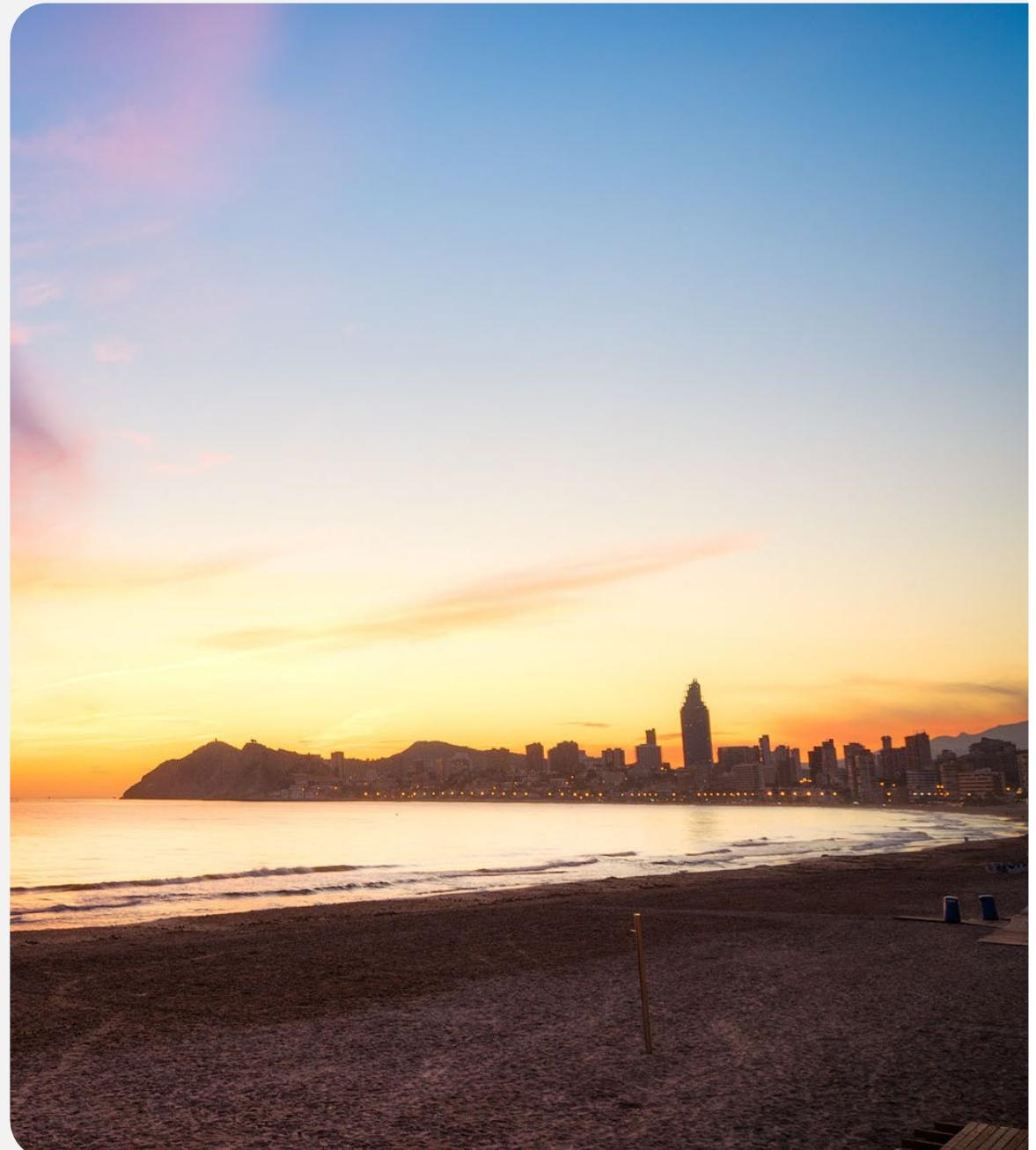
- Cada transacción entre un cliente y un servidor es independiente de todas las anteriores



MENSAJES HTTP/1.1

Formato del mensaje :

- *Línea inicial* (diferente en solicitudes y respuestas)
- Encabezado 1: valor 1
- Encabezado 2: valor 2
- ...
- Encabezado n: valor n
- *Línea en blanco*
- *Cuerpo del mensaje* opcional / contenidos de un archivo / datos binarios / etc ...



MENSAJES HTTP/1.1

LINEA INICIAL (Request)

La línea inicial se compone de **tres elementos**, separados por espacios:

- Método (GET, PUT, POST, OPTIONS, TRACE, DELETE ...)
- Identificador de recurso (URI)
- Versión del protocolo HTTP

Ejemplos:

- GET /directorio1/directorio2/index.html HTTP/1.0
- GET / HTTP/1.1



MENSAJES HTTP/1.1

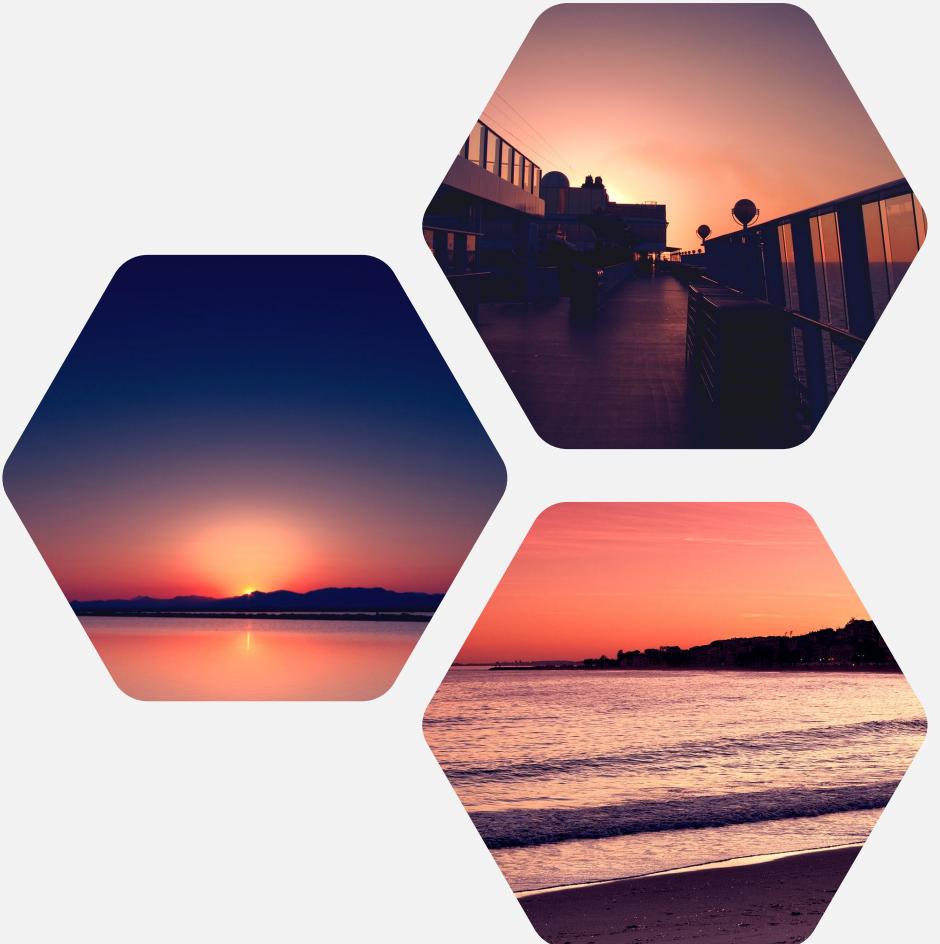
LINEA INICIAL (Response)

La línea inicial se compone de **tres elementos**, separados por espacios. Representa un estado de la transacción:

- Versión del protocolo HTTP
- Código de estado
- Frase explicativa

Ejemplos :

- HTTP/1.0 200 OK
- HTTP/1.1 404 Not Found



MENSAJES HTTP/1.1

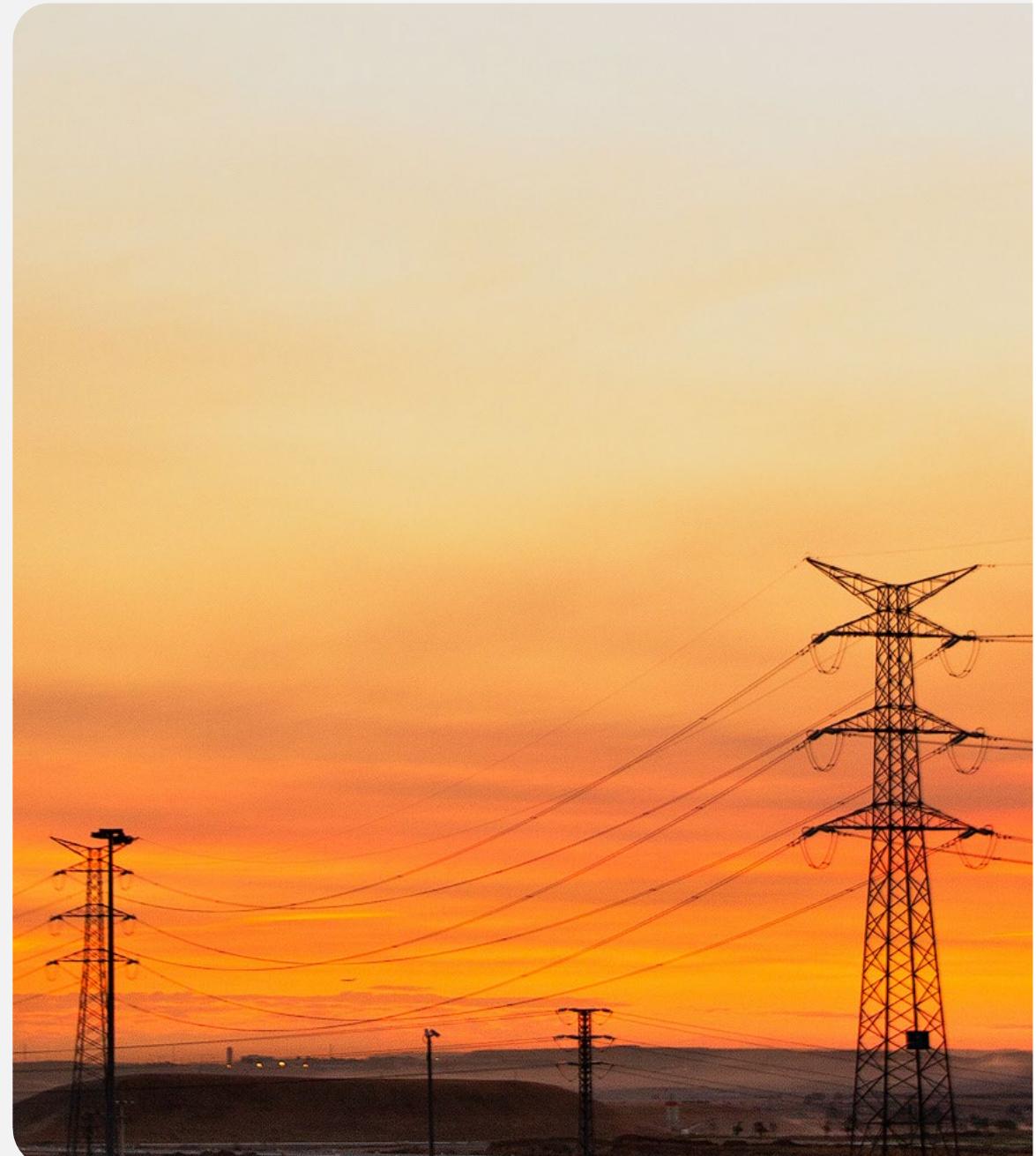
CÓDIGOS DE ESTADO

Son números enteros de tres dígitos

- 1xx: informativos
- 2xx: éxito
- 3xx: redirección
- 4xx: error de cliente
- 5xx: error de servidor

Los más típicos :

- **200 OK:** Éxito, la respuesta se envía en el cuerpo
- **404 Not Found:** El recurso solicitado no existe
- **303 See Other:** El recurso se ha movido a otra URL dada en el header **location field**
- **500 Server Error:** Error no esperado en el servidor



MENSAJES HTTP/1.1

LÍNEAS DE ENCABEZADO

Proporcionan información acerca de la solicitud o de la respuesta

Cada línea está compuesta por:
nombre_campo: valor

El orden de las líneas de encabezado no importa

Ejemplos:

- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.198 Safari/537.36
- Last-Modified: Mon, 16 Nov 2020 09:36:31 GMT
- Content-Type: text/html; charset=utf-8



MENSAJES HTTP

LÍNEAS DE ENCABEZADO

HTTP/1.0 define **16** tipos de header (ninguno obligatorio)

HTTP/1.1 define **46** tipos de header (sólo Host: obligatorio)

HTTP/2 funciona en binario, intercambiando tramas HTTP/2

En las **solicitudes** es normal que se incluya:

- User-Agent: identifica al navegador y su versión

En las **respuestas** se suele incluir:

- Server: identifica la versión del servidor web
- Last-Modified: fecha de la última modificación del recurso. Se usa para mantener las cachés



MENSAJES HTTP

CUERPO DEL MENSAJE

La mayor parte de las **respuestas** contendrá un **cuerpo del mensaje HTTP**

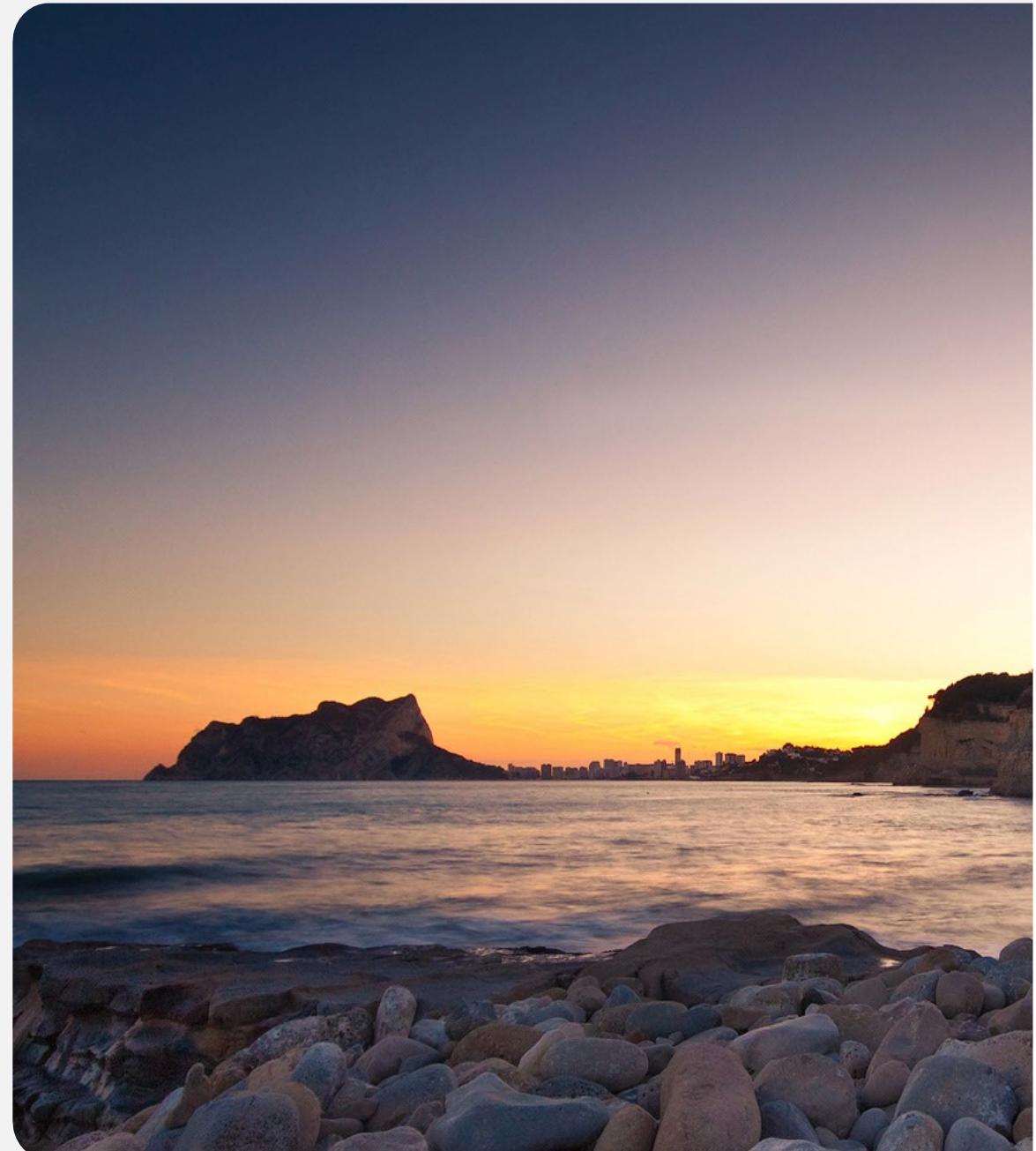
- Contiene el recurso solicitado

Puede haber cuerpo también en las solicitudes

- Envío de datos del cliente al servidor

Las líneas de encabezado definirán el tipo de cuerpo con:

- **Content-Type:** Indica el tipo MIME como por ejemplo **text/html image/png**
- **Content-Length:** Indica el número de bytes en el cuerpo



MENSAJES HTTP

EJEMPLO SESIÓN HTTP

El cliente quiere obtener <http://campus.somtic.net/ejemplo/ejemplo.html>

- El cliente establece una conexión TCP al **puerto 80** de campus.somtic.net y envía una **solicitud GET** a través del protocolo **HTTP**

```
> Transmission Control Protocol, Src Port: 59986, Dst Port: 80, Seq: 1, Ack: 1, Len: 524
  Hypertext Transfer Protocol
    > GET /ejemplo/ejemplo.html HTTP/1.1\r\n
      Host: campus.somtic.net\r\n
      Connection: keep-alive\r\n
      Upgrade-Insecure-Requests: 1\r\n
      User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.198 Safari/537.36\r\n
      Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0
      Accept-Encoding: gzip, deflate\r\n
      Accept-Language: ca-ES,ca;q=0.9,en;q=0.8,es;q=0.7\r\n
    > Cookie: MoodleSession=i33i1vbs6f6l1frqntevu52efv\r\n
    \r\n
```

MENSAJES HTTP

EJEMPLO SESIÓN HTTP

El servidor responde con:

```
> Transmission Control Protocol, Src Port: 80, Dst Port: 60642, Seq: 1, Ack: 524, Len: 317
‐ Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Mon, 16 Nov 2020 10:46:06 GMT\r\n
    Server: Apache/2.4.18 (Ubuntu)\r\n
    Last-Modified: Mon, 16 Nov 2020 10:43:47 GMT\r\n
    ETag: "22-5b4370d8c230e"\r\n
    Accept-Ranges: bytes\r\n
  > Content-Length: 34\r\n
    Keep-Alive: timeout=5, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html\r\n
    \r\n
    [HTTP response 1/2]
    [Time since request: 0.044040000 seconds]
    [Request in frame: 696]
    [Next request in frame: 706]
    [Next response in frame: 710]
    [Request URI: http://arxiu.somtic.net/favicon.ico]
    File Data: 34 bytes
‐ Line-based text data: text/html (1 lines)
  <html><body>Ejemplo</body></html>\n
```

MÉTODO HEAD

Las solicitudes HEAD son parecidas a las GET

Las respuestas a HEAD sólo contienen encabezados, nunca cuerpo

Son útiles para obtener las características de un recurso, sin llegar a transferirlo

Se usan en las peticiones de actualización en REST, para la implementación de un CRUD



MÉTODO POST

Se utiliza para **enviar datos a un servidor**, por ejemplo los recogidos en un formulario

Se diferencia de **GET** en:

- Hay un bloque de datos que se envía en el cuerpo de la solicitud
- Normalmente hay headers que describen el cuerpo del mensaje enviado (Ej. **Content-Type** y **Content-Length**)
- El URI solicitado normalmente es un script que deberá procesar los datos
- La respuesta HTTP normalmente se genera de forma dinámica



MÉTODO POST

El método **POST** se usa de forma habitual para enviar los **datos recogidos de un formulario** a un script en el servidor para que los procese

El **Content-Type** toma el valor **application/x-www-form-urlencoded** y el **Content-Length** indica la longitud en bytes

Se usa en las peticiones de **creación** en REST, para la implementación de un **CRUD**



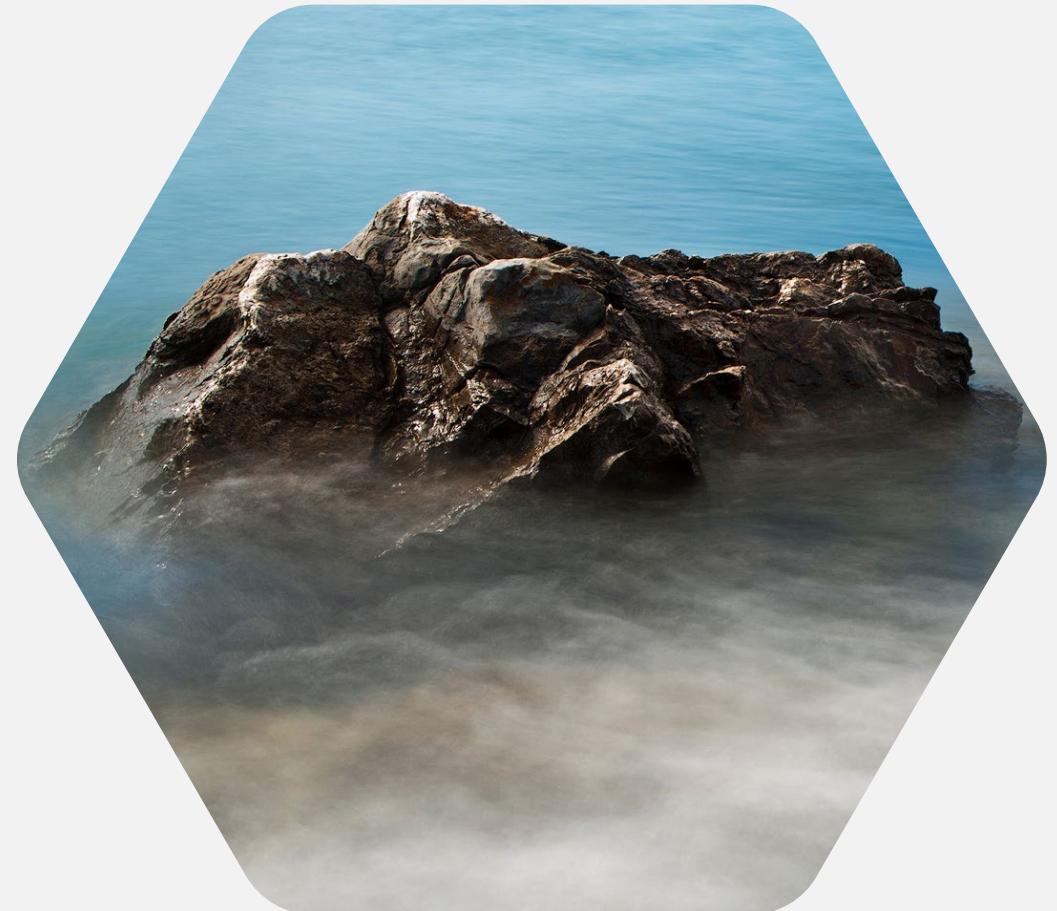
MÉTODO POST

```
Connection: keep-alive\r\n
> Content-Length: 87\r\n
Cache-Control: max-age=0\r\n
Upgrade-Insecure-Requests: 1\r\n
Origin: http://arxiu.somtic.net\r\n
Content-Type: application/x-www-form-urlencoded\r\n
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/86.0.4240.198 Safari/537.36\r\n
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0
Referer: http://arxiu.somtic.net/login/index.php\r\n
Accept-Encoding: gzip, deflate\r\n
Accept-Language: ca-ES,ca;q=0.9,en;q=0.8,es;q=0.7\r\n
> Cookie: MoodleSession=eapg39fqiqh3qffur4kvvnh6su\r\n
\r\n
\[Full request URI: http://arxiu.somtic.net/login/index.php\]
[HTTP request 1/2]
\[Response in frame: 306\]
\[Next request in frame: 307\]
File Data: 87 bytes
HTML Form URL Encoded: application/x-www-form-urlencoded
> Form item: "anchor" =
> Form item: "logintoken" = "uyykv2mAeeggFp1fn83d59iRSFNVi1a"
> Form item: "username" = "
> Form item: "password" = "
```

HTTP/1.1

Incluye muchas **mejoras** respecto a HTTP/1.0

- Respuesta más **veloz**
- Permite **múltiples transacciones** en una única solicitud/respuesta
- Permite **ahorro de ancho de banda** mediante uso de caché
- **Respuesta más rápida** en páginas dinámicas: permite el envío de la respuesta antes de saber la longitud total (chunked response)
- Uso eficiente de direcciones IP: permite **servidores virtuales** basados en nombre



HTTP/1.1

CLIENTE

Los clientes deben cumplir con los siguientes requisitos:

- Incluir el encabezado **Host**: en cada solicitud
- Aceptar respuestas en modo *chunk*
- Soportar conexiones persistentes o incluir el encabezado "**Connection: close**" en cada solicitud
- Ser capaces de manejar la respuesta "**100 Continue**"



HTTP/1.1

ENCABEZADO Host

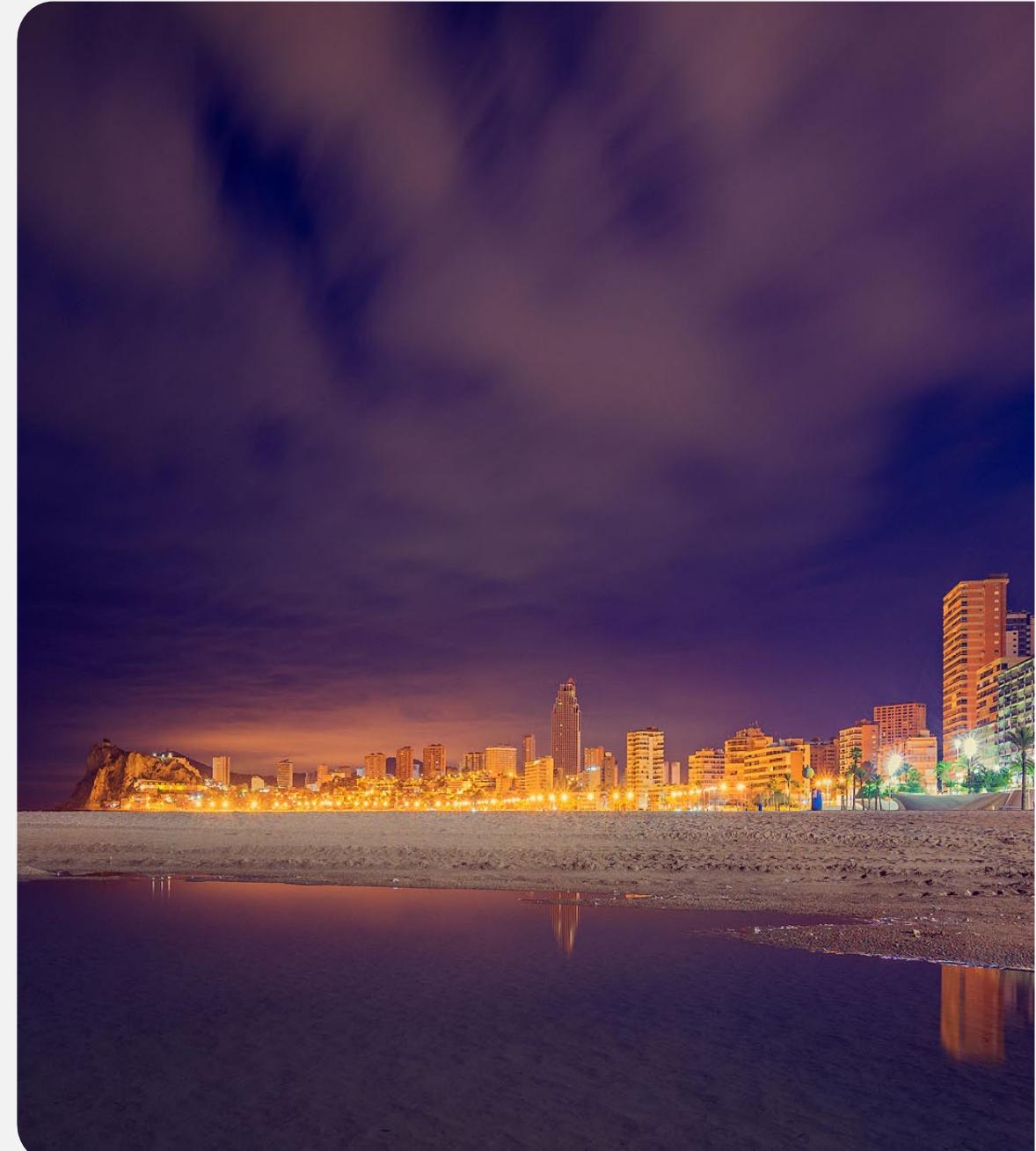
A partir de HTTP/1.1 los servidores pueden manejar distintos **sitios web virtuales**

- Es obligatorio incluir el encabezado Host: en cada solicitud

Ejemplo:

```
▼ Hypertext Transfer Protocol
  > POST /login/index.php HTTP/1.1\r\n
    Host: arxiu.somtic.net\r\n
    Connection: keep-alive\r\n
  > Content-Length: 87\r\n
    Cache-Control: max-age=0\r\n
    Upgrade-Insecure-Requests: 1\r\n
```

Host: es el único encabezado obligatorio en HTTP/1.1



HTTP/1.1

Chunked Transfer-Encoding

Si un servidor quiere comenzar a enviar la respuesta antes de conocer la longitud total debe incluir el encabezado **Transfer-Encoding: chunked**

El cuerpo de un mensaje de este tipo contiene una serie de trozos (chunks) seguidos de una línea con un cero "0"

Cada trozo tiene dos partes:

- Una línea con el tamaño del trozo
- Los datos del trozo



HTTP/1.1

Chunked Transfer-Encoding

```
▼ HTTP chunked response
  ▼ Data chunk (28682 octets)
    Chunk size: 28682 octets
    ▼ Data (28682 bytes)
      Data: 1f8b08000000000000003ecbd5b93e3c6952efaae5f814d9d...
      [Length: 28682]
      Chunk boundary
  ▼ Data chunk (6775 octets)
    Chunk size: 6775 octets
    ▼ Data (6775 bytes)
      Data: fea1aa19cda38bb1d4a7713alebb906d27281b86ba3e5214...
      [Length: 6775]
      Chunk boundary
  ▼ End of chunked encoding
    Chunk size: 0 octets
    Chunk boundary
Content-encoded entity body (gzip): 35457 bytes -> 274083 bytes
```

HTTP/1.1

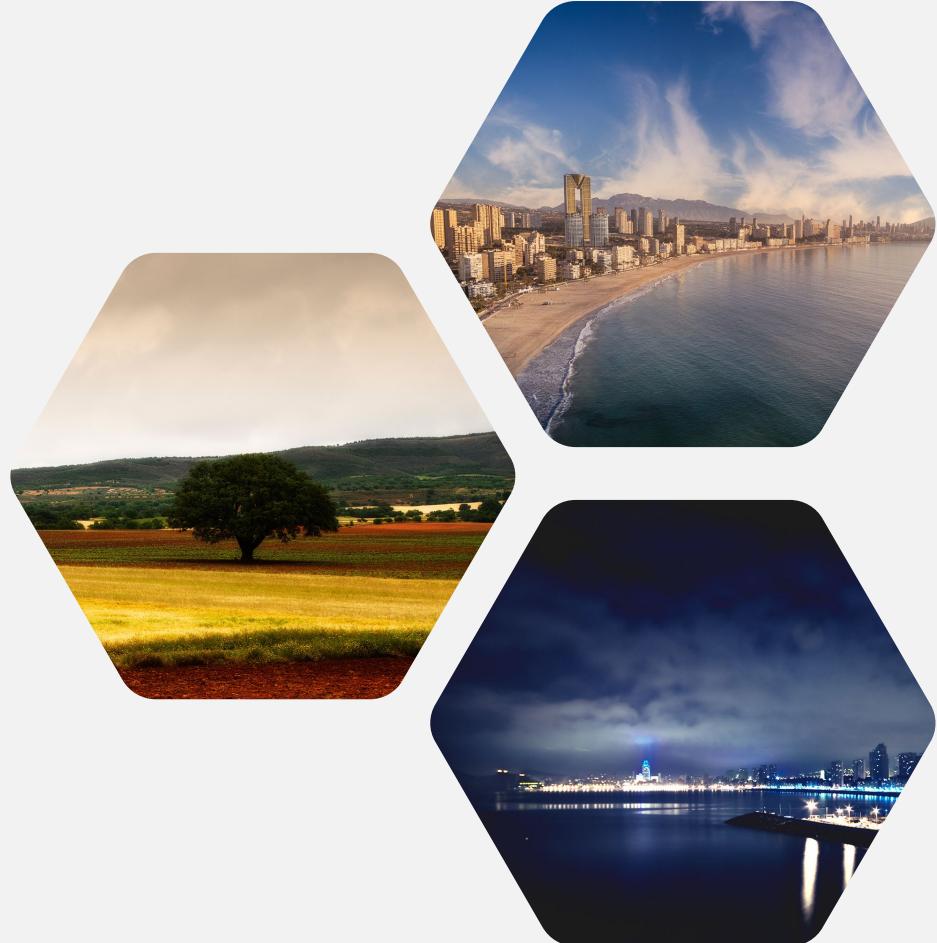
CONEXIONES PERSISTENTES

En **HTTP/1.1** las conexiones son **persistentes** por omisión

- Tras una transacción el servidor no cierra la conexión, sino espera otra solicitud

El cliente puede incluir el encabezado **Connection: close** en una solicitud para indicarle al servidor que cierre la conexión tras la respuesta

- Si el cliente no soporta la persistencia de sesiones, está obligado a incluir siempre el encabezado anterior



HTTP/1.1

ENCABEZADO Date

Para poder implementar **cachés**, es necesario registrar las fechas y horas de creación y modificación de los recursos (timestamps)

Los servidores incluyen el encabezado "**Date**". Los servidores deben incluir la fecha y hora del recurso en cuestión con el encabezado

▼ Hypertext Transfer Protocol

› HTTP/1.1 200 OK\r\n

Date: Mon, 16 Nov 2020 11:00:08 GMT\r\n

Server: Apache/2.4.18 (Ubuntu)\r\n

Expires: \r\n

Cache-Control: private, pre-check=0, post-check=0, max-age=0, no-transform\r\n

HTTP/1.1

If-(Un)Modified-Since

Se utilizan para ahorrar ancho de banda y hacer uso de cachés

If-Modified-Since

- Indica que hay que enviar el recurso sólo si ha cambiado después de la fecha especificada
- Si no ha cambiado, el servidor responde con **304 Not Modified**

```
✓ Hypertext Transfer Protocol
  > HTTP/1.1 304 Not Modified\r\n
    Date: Mon, 16 Nov 2020 12:31:49 GMT\r\n
    Server: Apache/2.4.18 (Ubuntu)\r\n
    Connection: Keep-Alive\r\n
    Keep-Alive: timeout=5, max=100\r\n
    ETag: "22-5b4370d8c230e"\r\n
    \r\n
  [HTTP response 1/1]
  [Time since request: 0.043679000 seconds]
  \[Request in frame: 1920\]
  [Request URI: http://arxiu.somtic.net/ejemplo/ejemplo.html]
```



HTTP/1.1

If-(Un)Modified-Since

If-Unmodified-Since

- Indica que hay que enviar el recurso sólo si no ha cambiado después de la fecha especificada
- Si el recurso ha cambiado el servidor contesta con **412 Precondition Failed**

