

R-Pi

Team Emertxe



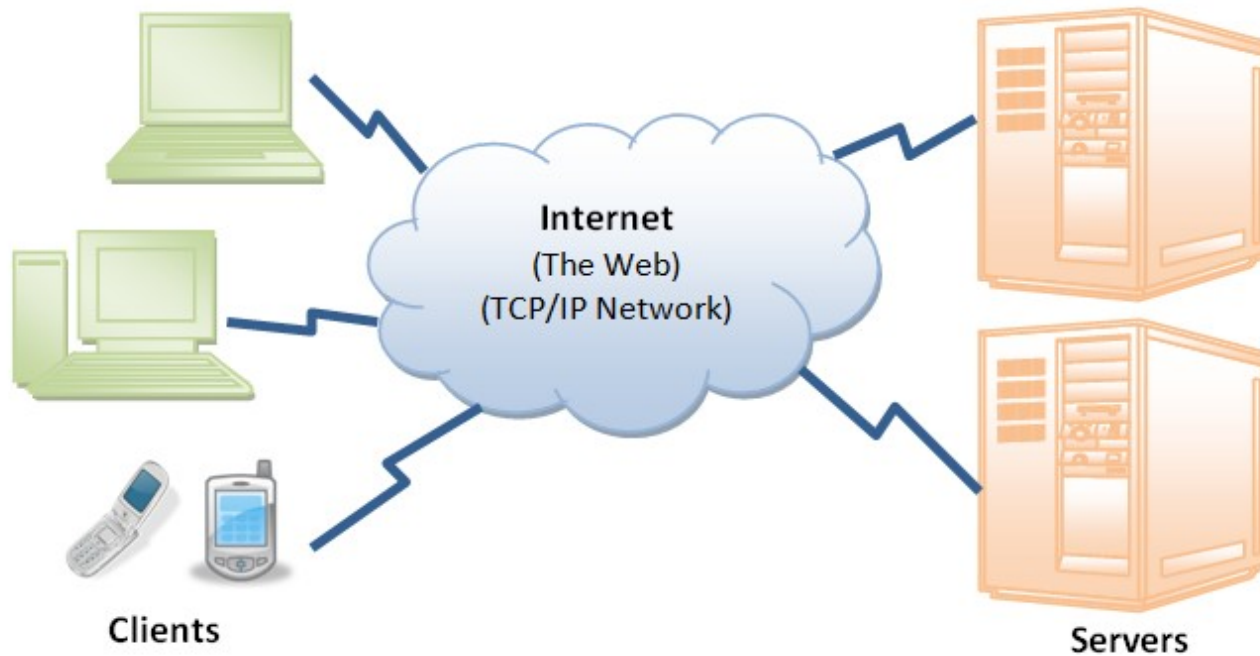
IoT Protocols

HTTP

HTTP

The Web

- Internet (or The Web) is a massive distributed client/server information system.



HTTP

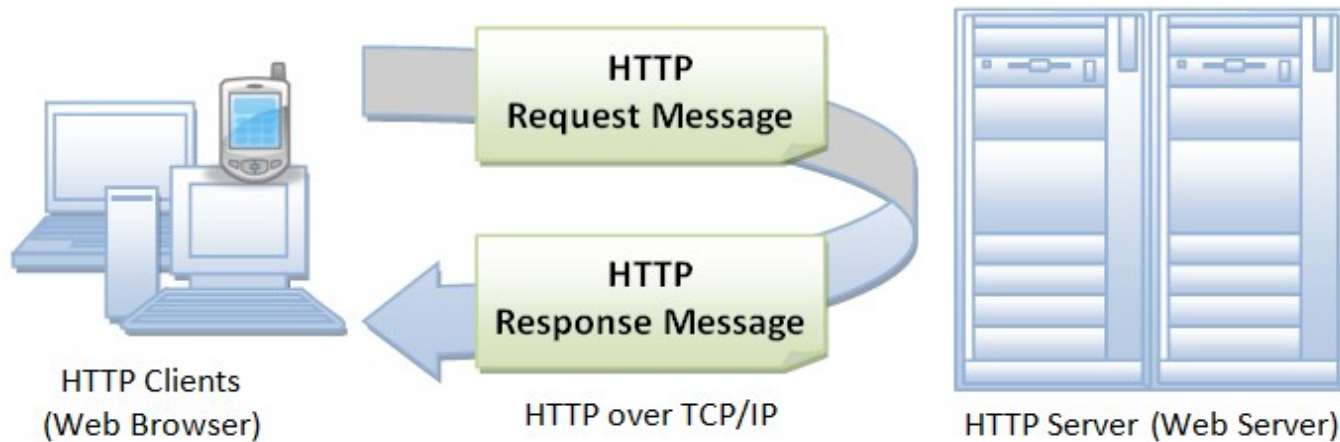
The Web

- Many applications are running concurrently over the Web, such as web browsing/surfing, e-mail, file transfer, audio & video streaming, and so on.
- In order for proper communication, the applications must agree on a specific application-level protocol such as HTTP, FTP, SMTP, POP, and etc.

HTTP

Introduction

- The most popular application protocol used in the Internet (or The WEB).
- Asymmetric request-response client-server protocol.
- An HTTP client sends a request message to an HTTP server.
- The server, in turn, returns a response message.



HTTP

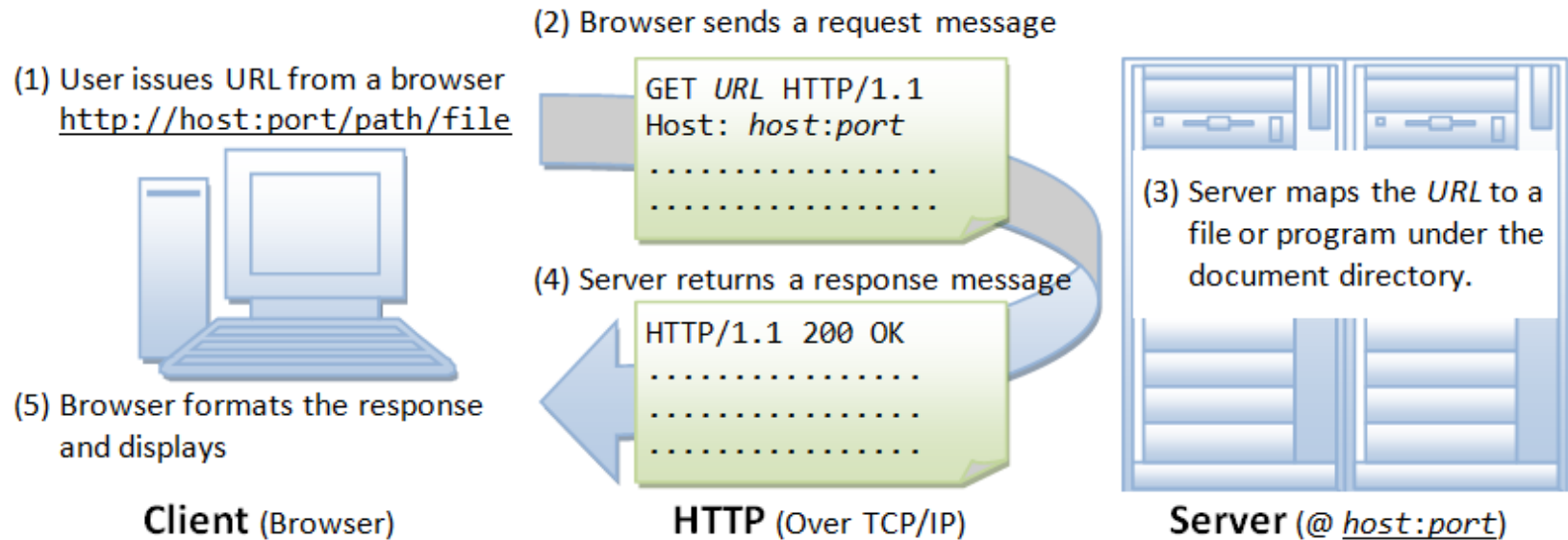
Introduction

- Stateless protocol
 - The current request does not know what has been done in the previous requests.
- Permits negotiating of data type and representation, so as to allow systems to be built independently of the data being transferred.

HTTP

Browser

- The browser turns the URL into a request message and sends it to the HTTP server.
- The HTTP server interprets the request message, and returns you an appropriate response message, which is either the resource you requested or an error message.



HTTP

URL(Uniform Resource Locator)

- Used to uniquely identify a resource over the web.

Syntax: `protocol://hostname:port/path-and-file-name`

Protocol	The application-level protocol used by the client and server, e.g., HTTP, FTP, and telnet.
Hostname	The DNS domain name (e.g., <code>www.emertxe.com</code>) or IP address (e.g., <code>192.128.1.2</code>) of the server.
Port	The TCP port number that the server is listening for incoming requests from the clients.
Path-and-file-name	The name and location of the requested resource, under the server document base directory.

HTTP

URL(Uniform Resource Locator)

- Used to uniquely identify a resource over the web.

URL:

`http://www.emertxe.com/docs/index.html`

Protocol

HTTP

Hostname

`www.emertxe.com`

Port

The port number was not specified in the URL, and takes on the default number, which is TCP port 80 for HTTP.

Path-and-
file-name

`"/docs/index.html".`

HTTP

URL(Uniform Resource Locator)

- More Examples

```
ftp://www.ftp.org/docs/test.txt
```

```
mailto:user@test101.com
```

```
news:soc.culture.Singapore
```

```
telnet://www.nowhere123.com/
```

HTTP

URL(Uniform Resource Locator)

- For example, the browser translated the URL `http://www.emertxe.com/` into the following request message:

The screenshot shows a Mozilla Firefox browser window with the Emertxe website loaded. The website header includes the Emertxe logo, the tagline "The future - ready IT institute", and a navigation menu with links for Courses, Placements, Course Materials, News, Blog, Careers, About Us, and Contact Us. A prominent banner advertises "510+ PLACEMENT DRIVES" and "India's Best Embedded systems & IoT Training & Placement Program". A WhatsApp chat button is visible on the right.

The browser's developer tools are open, displaying the Network tab. A table lists the requests made by the browser:

Sta...	Me...	Domain	File	Cause	Type	Transferred	Size	0 ms	10.24 s	20.48 s
200	GET	www.em...	google-logo-small.png	img	png	cached	1.54...			
200	GET	www.em...	mystery-man.png	img	png	cached	1.15...			
200	GET	www.em...	whats_app_icon.png	img	png	cached	6.33...			
200	GET	www.em...	47d3e.js	script	js	cached	1.83...			
200	GET	www.em...	ca57b.js	script	js	cached	10.6...			
200	GET	www.em...	2f606.js	script	js	cached	3.69...			
200	GET	www.em...	7efe8.js	script	js	cached	6.53...			
200	GET	www.em...	25116.js	script	js	cached	8.17...			
200	GET	www.em...	5a595.js	script	js	cached	566 B			
	GET	www.em...	d07e0.js	script	js	836 B	2.51...	0 ms		
	GET	www.em...	7e5dc.js	script	js	2.23 KB	6.16...	0 ms		
200	GET	www.em...	000df.js	script	js	cached	1.75...			
200	GET	www.em...	bbaff.js	script	js	cached	8.96...			
200	GET	www.em...	ca87e.js	script	js	cached	1.64...			

At the bottom of the network tab, a summary shows: 28 requests, 1.47 MB / 4.27 KB transferred, Finish: 1.60 s, DOMContentLoaded: 236 ms, load: 17.43 s.

The right-hand pane of the developer tools shows the details for the selected request (google-logo-small.png):

- Request URL: `https://www.emertxe.com/wp-content/plugins/google-places-reviews-pro/dist/images/google-logo-small.png`
- Request method: GET
- Status code: 200 OK
- Version: HTTP/2.0
- Referrer Policy: no-referrer-when-downgrade

The Headers tab is selected, showing the following response headers:

- `accept-ranges: bytes`
- `alt-svc: quic=":443"; ma=2592000; v="35,39,43,44"`
- `cache-control: max-age=604800, public`
- `content-length: 1577`
- `content-type: image/png`
- `date: Mon, 03 Jun 2019 08:50:38 GMT`
- `etag: "629-5a445c9c-6f6f73c70a048c39;:"`
- `expires: Mon, 10 Jun 2019 08:50:36 GMT`
- `last-modified: Thu, 28 Dec 2017 02:53:16 GMT`
- `server: LiteSpeed`
- `strict-transport-security: max-age=63072000; includeSubDomains`

HTTP

URL(Uniform Resource Locator)

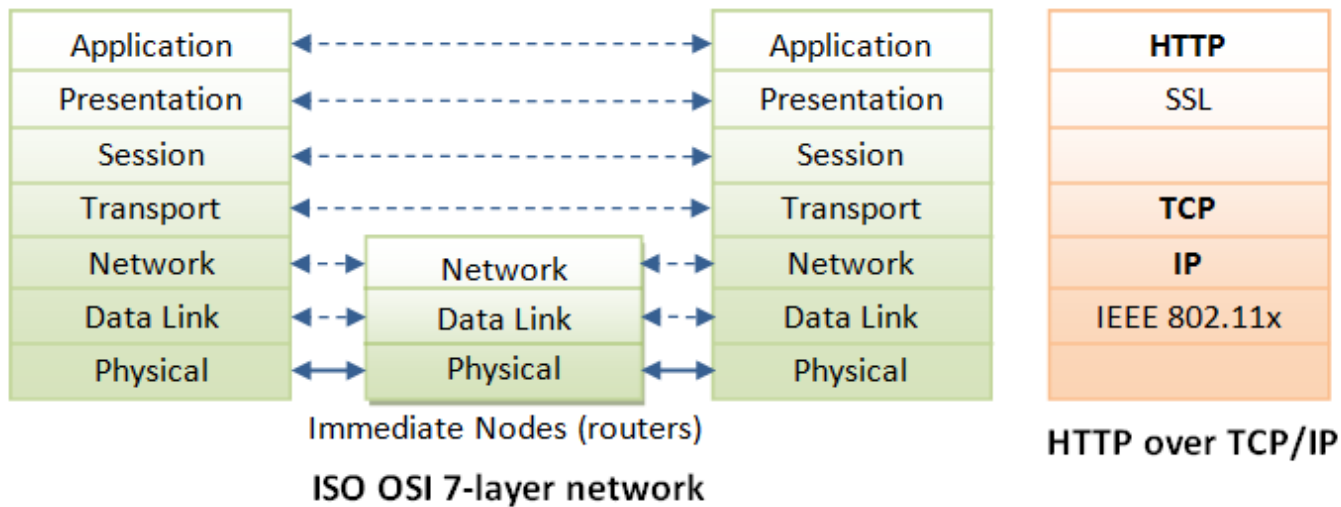


- When this request message reaches the server, the server can take either one of these actions:
 - The server interprets the request received, maps the request into a file under the server's document directory, and returns the file requested to the client.
 - The server interprets the request received, maps the request into a program kept in the server, executes the program, and returns the output of the program to the client.
 - The request cannot be satisfied, the server returns an error message.
- The browser receives the response message, interprets the message and displays the contents of the message on the browser's window according to the media type of the response (as in the Content-Type response header).
- Common media type include "text/plain", "text/html", "image/gif", "image/jpeg", "audio/mpeg", "video/mpeg", "application/msword", and "application/pdf".

HTTP

HTTP over TCP/IP

- HTTP is a client-server application-level protocol.
- It typically runs over a TCP/IP connection.

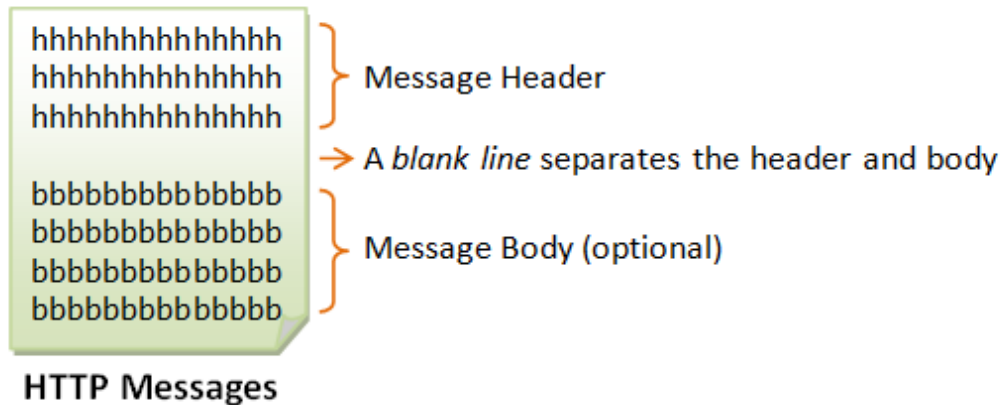


To communicate over TCP/IP, you need to know (a) IP address or hostname, (b) Port number.

HTTP

Request + Response Messages

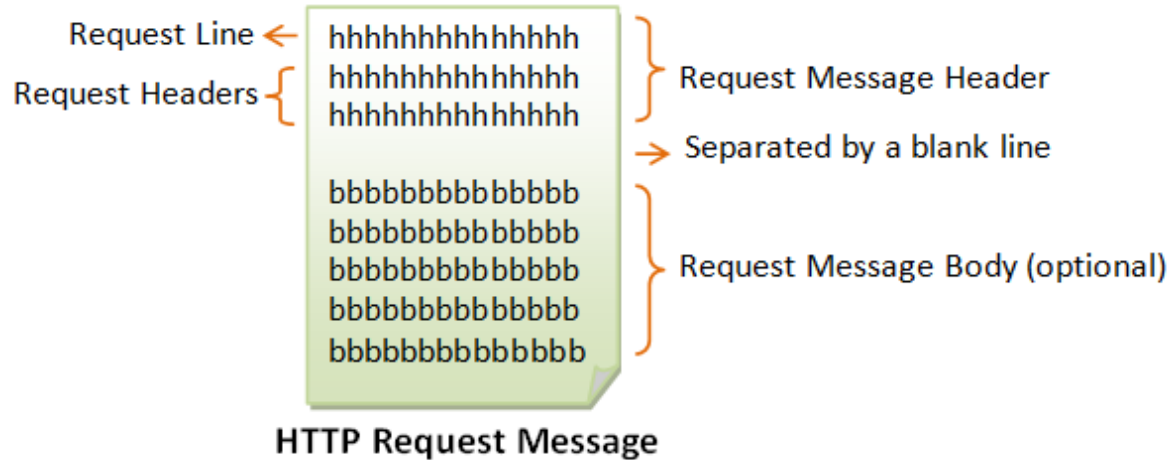
- HTTP client and server communicate by sending text messages.
- The client sends a request message to the server. The server, in turn, returns a response message.
- An HTTP message consists of a message header and an optional message body, separated by a blank line, as illustrated below:



HTTP

Request Messages

- The format of an HTTP request message is as follow:



HTTP: Request Messages

Request Line



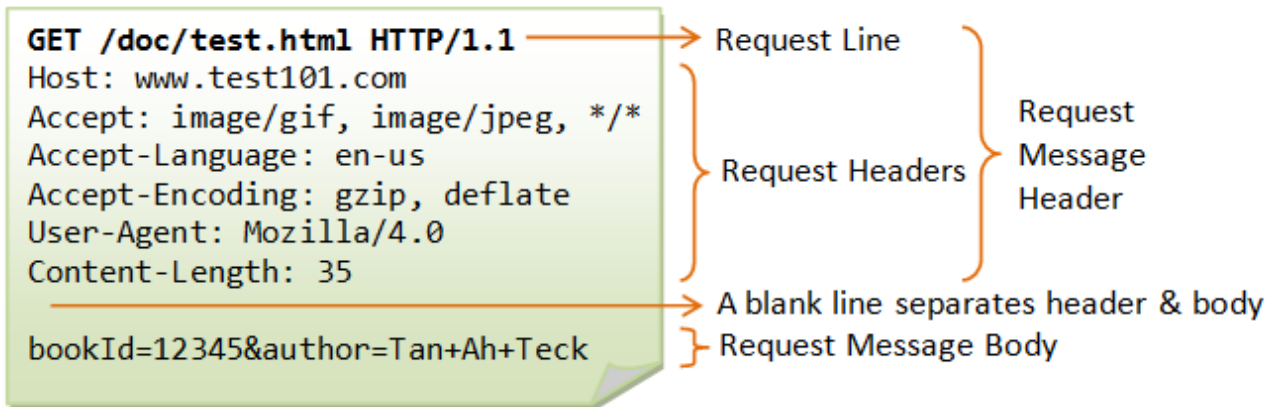
Syntax	<code>request-method-name request-URI HTTP-version</code>
request-method-name	<ul style="list-style-type: none">• HTTP protocol defines a set of request methods, e.g., GET, POST, HEAD, and OPTIONS.• The client can use one of these methods to send a request to the server.
request-URI	specifies the resource requested.
HTTP-version	Two versions are currently in use: HTTP/1.0 and HTTP/1.1.
Examples	<code>GET /test.html HTTP/1.1</code> <code>HEAD /query.html HTTP/1.0</code> <code>POST /index.html HTTP/1.1</code>

HTTP: Request Messages

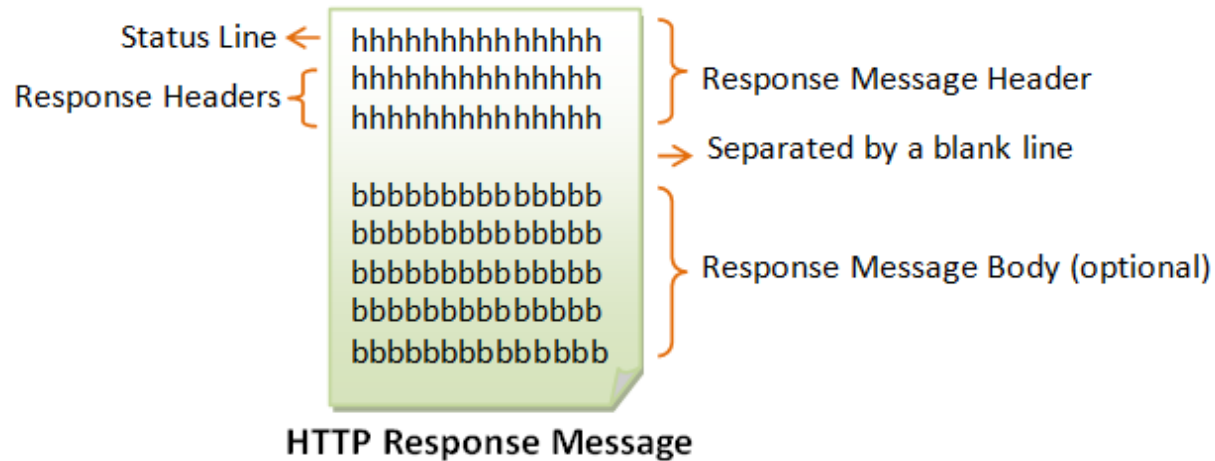
Request Headers



Syntax	<code>request-header-name: request-header-value1, request-header-value2, ...</code>
Examples	<code>Host: www.xyz.com</code> <code>Connection: Keep-Alive</code> <code>Accept: image/gif, image/jpeg, */*</code> <code>Accept-Language: us-en, fr, cn</code>



HTTP: Response Messages Format



HTTP: Response Messages

Status Line



Syntax	<code>HTTP-version status-code reason-phrase</code>
HTTP-version	<ul style="list-style-type: none">• The HTTP version used in this session. Either HTTP/1.0 and HTTP/1.1.
status-code	a 3-digit number generated by the server to reflect the outcome of the request.
Examples	<code>HTTP/1.1 200 OK</code> <code>HTTP/1.0 404 Not Found</code> <code>HTTP/1.1 403 Forbidden</code>

Common status code and reason phrase are "200 OK", "404 Not Found", "403 Forbidden", "500 Internal Server Error".

HTTP: Response Messages

Response Headers

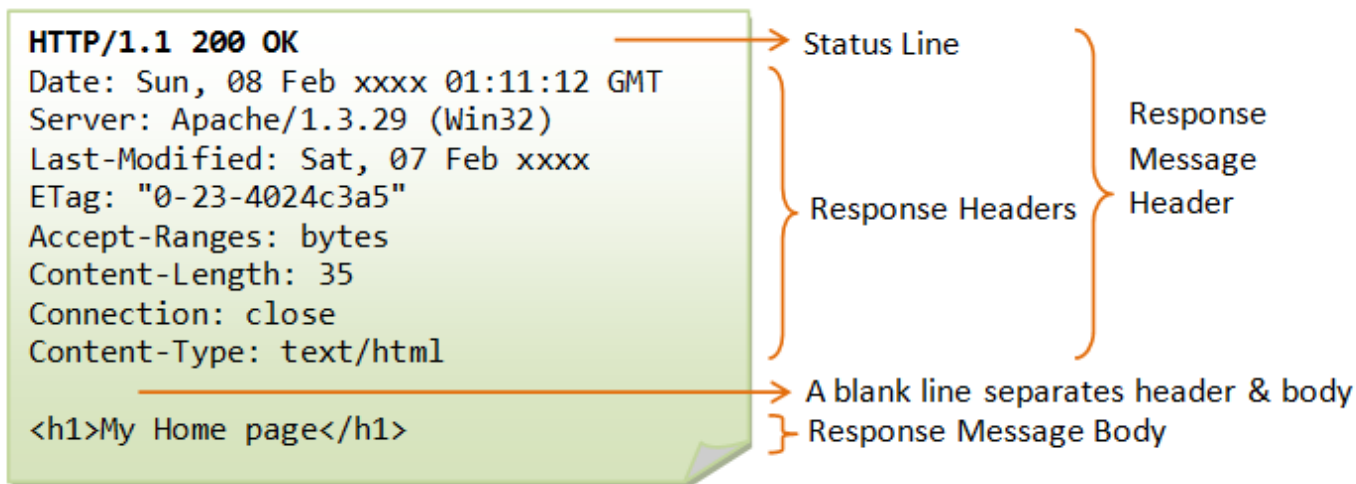


Syntax	<code>response-header-name: response-header-value1, response-header-value2, ...</code>
Examples	<code>Content-Type: text/html</code> <code>Content-Length: 35</code> <code>Connection: Keep-Alive</code> <code>Keep-Alive: timeout=15, max=100</code>

HTTP: Response Messages

Response Body

- The response message body contains the resource data requested.



HTTP

Request Methods

- HTTP protocol defines a set of request methods.
- A client can use one of these request methods to send a request message to an HTTP server.

HTTP:

Request Methods



GET	A client can use the GET request to get a web resource from the server.
HEAD	<ul style="list-style-type: none">• A client can use the HEAD request to get the header that a GET request would have obtained.• Since the header contains the last-modified date of the data, this can be used to check against the local cache copy.
POST	Used to post data up to the web server.
PUT	Ask the server to store the data.
DELETE	Ask the server to delete the data.
TRACE	Ask the server to return a diagnostic trace of the actions it takes.
OPTIONS	Ask the server to return the list of request methods it supports.
CONNECT	<p>Used to tell a proxy to make a connection to another host and simply reply the content, without attempting to parse or cache it.</p> <p>This is often used to make SSL connection through the proxy.</p>

HTTP

Request Methods: GET



GET	<ul style="list-style-type: none">• GET is the most common HTTP request method.• A client can use the GET request method to request (or "get") for a piece of resource from an HTTP server.
SYNTAX	<pre>GET request-URI HTTP-version (optional request headers) (blank line) (optional request body)</pre>
request-URI	specifies the path of resource requested, which must begin from the root "/" of the document base directory.
HTTP-version:	Version of the protocol
(optional request headers)	The client uses the optional request headers (such as Accept, Accept-Language, and etc) to negotiate with the server and ask the server to deliver the preferred contents (e.g., in the language that the client preferred).

HTTP

Request Methods: GET- Testing

- There are many way to test out the HTTP requests.
- "telnet" can be used to test the GET or write you own network program to send raw request message to an HTTP server to test out the various HTTP requests.
- TELNET
 - "Telnet" is a very useful networking utility.
 - use telnet to establish a TCP connection with a server; and issue raw HTTP requests.
 - Example: suppose that you have started your HTTP server in the localhost (IP address 127.0.0.1) at port 8000:

HTTP: Request Methods

HTTP/1.0 GET Request

- GET /index.html HTTP/1.0
 - (enter twice to create a blank line)

```
HTTP/1.1 200 OK
Date: Sun, 18 Oct 2009 08:56:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Sat, 20 Nov 2004 07:16:26 GMT
ETag: "10000000565a5-2c-3e94b66c2e680"
Accept-Ranges: bytes
Content-Length: 44
Connection: close
Content-Type: text/html
X-Pad: avoid browser bug

<html><body><h1>It works!</h1></body></html>

Connection to host lost.
```

HTTP:

Response status code



- The first line of the response message (i.e., the status line) contains the response status code, which is generated by the server to indicate the outcome of the request.
- The status code is a 3-digit number:
 - 1xx (Informational): Request received, server is continuing the process.
 - 2xx (Success): The request was successfully received, understood, accepted and serviced.
 - 3xx (Redirection): Further action must be taken in order to complete the request.
 - 4xx (Client Error): The request contains bad syntax or cannot be understood.
 - 5xx (Server Error): The server failed to fulfill an apparently valid request.
- Example:
 - 200 OK: The request is fulfilled.
 - 404 Not Found: The requested resource cannot be found in the server.

HTTP

Request Methods: HEAD

HEAD

- HEAD request is similar to GET request.
- However, the server returns only the response header without the response body, which contains the actual document.
- HEAD request is useful for checking the headers, such as Last-Modified, Content-Type, Content-Length, before sending a proper GET request to retrieve the document.

SYNTAX

```
HEAD request-URI HTTP-version
(other optional request headers)
(blank line)
(optional request body)
```

Example

```
HEAD /index.html HTTP/1.0
(blank line)

HTTP/1.1 200 OK
Date: Sun, 18 Oct 2009 14:09:16 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Sat, 20 Nov 2004 07:16:26 GMT
ETag: "10000000565a5-2c-3e94b66c2e680"
Accept-Ranges: bytes
Content-Length: 44
Connection: close
Content-Type: text/html
X-Pad: avoid browser bug
```

HTTP

Request Methods: OPTIONS

HEAD	<ul style="list-style-type: none">A client can use an OPTIONS request method to query the server which request methods are supported.
SYNTAX	<pre>OPTIONS request-URI * HTTP-version (other optional headers) (blank line)</pre>
Example	<pre>OPTIONS http://www.amazon.com/ HTTP/1.1 Host: www.amazon.com Connection: Close (blank line) HTTP/1.1 200 OK Date: Fri, 27 Feb 2004 09:42:46 GMT Content-Length: 0 Connection: close Server: Stronghold/2.4.2 Apache/1.3.6 C2NetEU/2412 (Unix) Allow: GET, HEAD, POST, OPTIONS, TRACE Connection: close Via: 1.1 xproxy (NetCache NetApp/5.3.1R4D5) (blank line)</pre>

THANK YOU