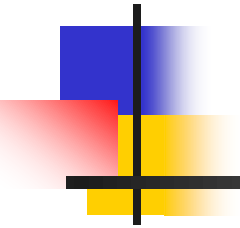
UNIT 5- APPLICATION LAYER
WWW and HTTP



Prepared By

Haseeba Yaseen

What is World Wide Web ?

- The World Wide Web is a way of exchanging information between computers on the Internet.
- The World Wide Web is the network of pages of images, texts and sounds on the Internet which can be viewed using browser software .

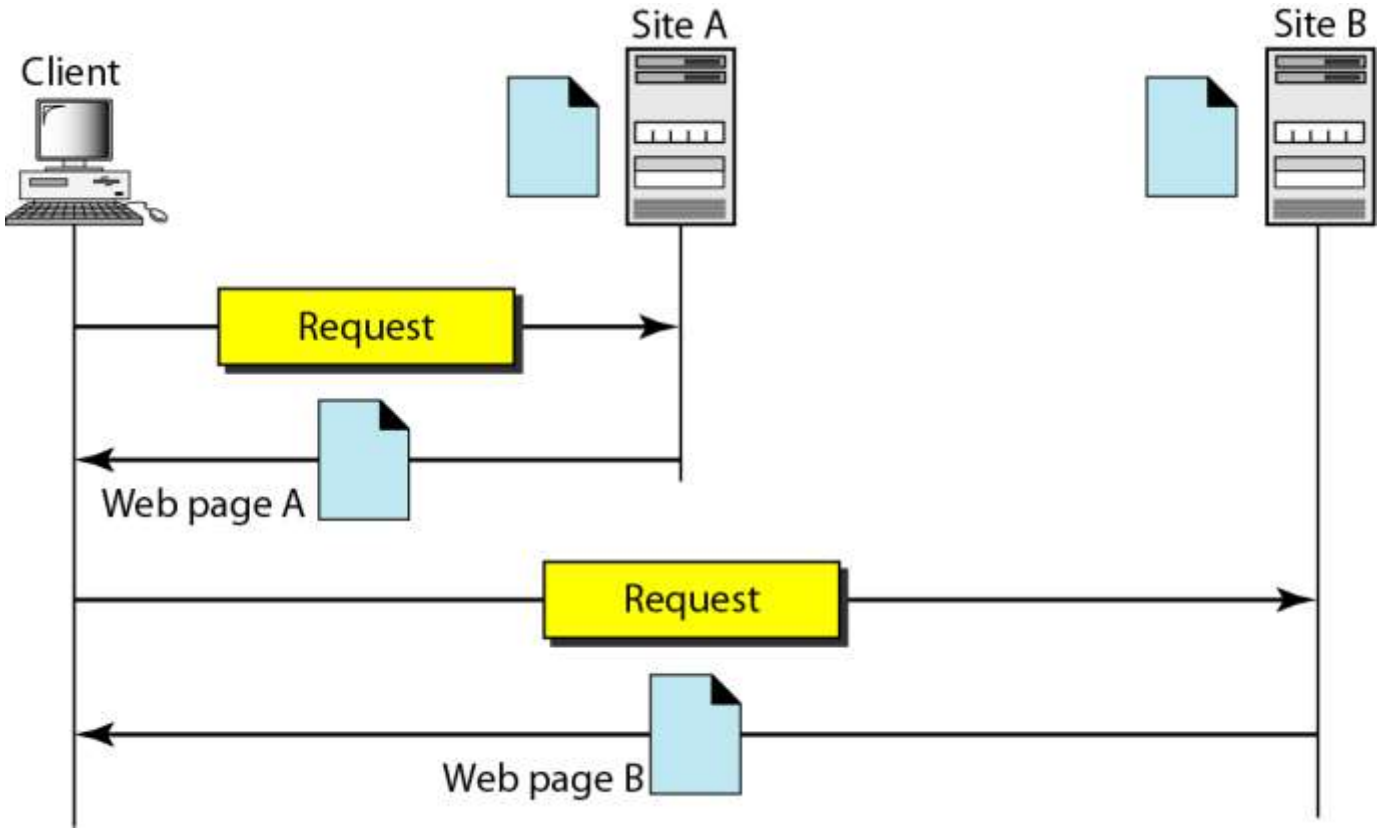
27-1 ARCHITECTURE

The WWW today is a distributed client/server service, in which a client using a browser can access a service using a server. However, the service provided is distributed over many locations called sites.

Topics discussed in this section:

Client (Browser)
Server
Uniform Resource Locator
Cookies

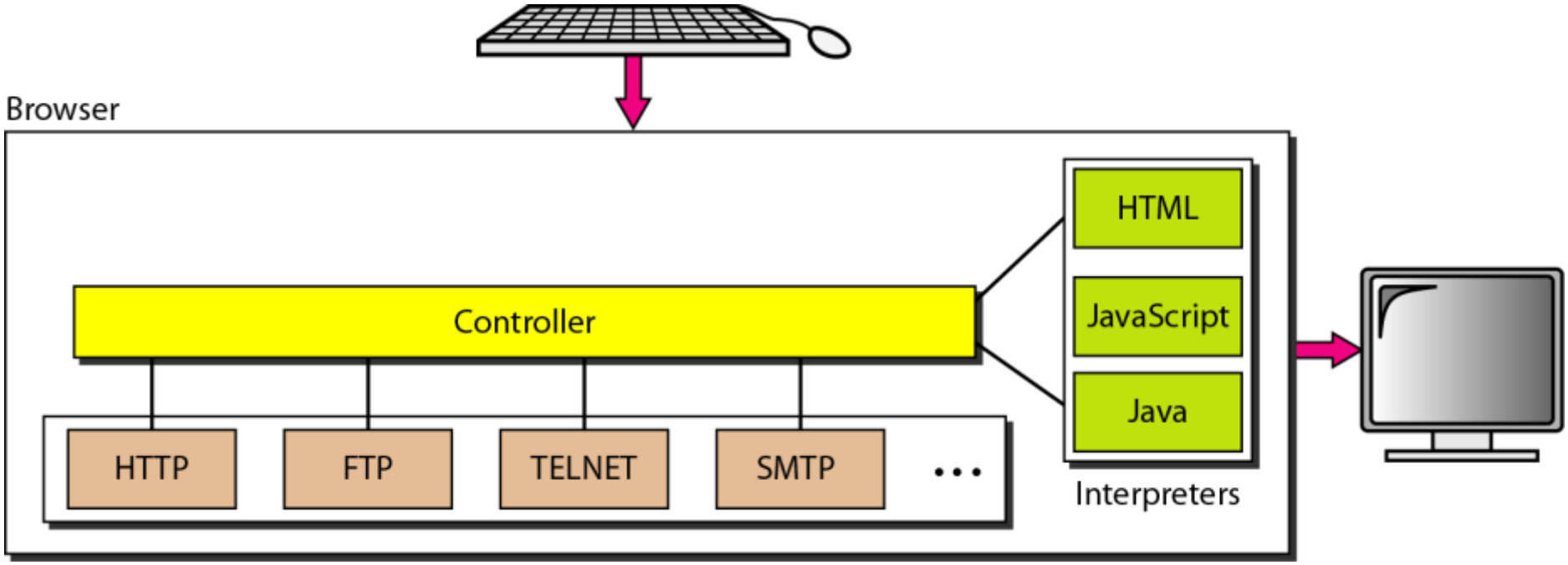
Figure 27.1 *Architecture of WWW*



Client(Browser)

- It consists of three parts:
  a controller, client protocol, and interpreters.
- The controller receives input from the keyboard or the mouse and uses the client programs to access the document.
- After the document has been accessed, the controller uses one of the interpreters to display the document on the screen.

Figure 27.2 Client- *Browser*

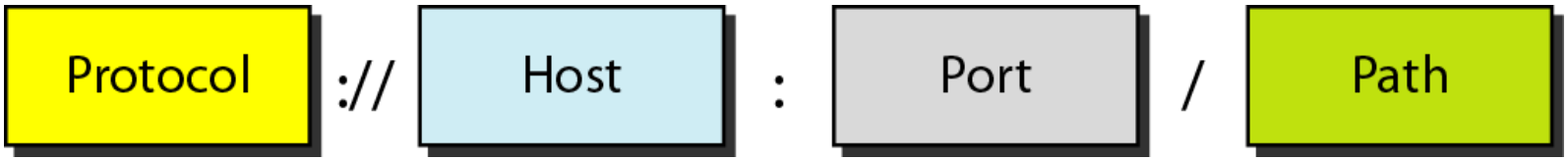


URL-(Uniform Resource Locator)

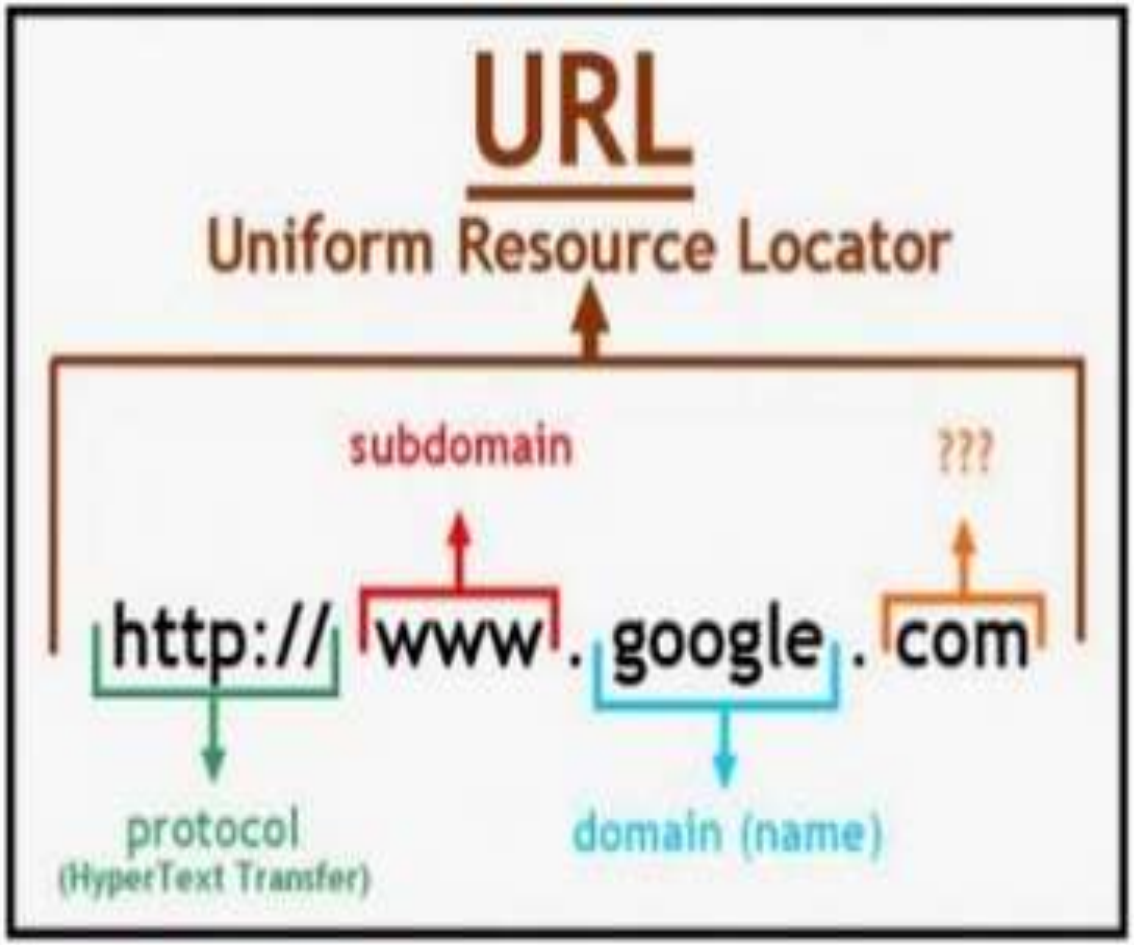
- The uniform resource locator (URL) is a standard for specifying any kind of information on the Internet.
- The URL defines four things:

  protocol, host computer, port, and path

Figure 27.3 *URL- standard for specifying any kind of information on the internet*



What is URL?


URL
Uniform Resource Locator
subdomain
???
http://
www
.
google
.
com
protocol
(HyperText Transfer)
domain (name)

Cookies

- The World Wide Web was originally designed as a stateless entity. A client sends a request; a server responds. Their relationship is over.

1. Some websites need to allow access to registered clients only.
2. Websites are being used as electronic stores that allow users to browse through the store, select wanted items, put them in an electronic cart, and pay at the end with a credit card.
3. Some websites are used as portals: the user selects the Web pages he wants to see.

Cookies

1. When a server receives a request from a client, it stores information about the client in a file or a string.
2. The information may include the domain name of the client, the contents of the cookie (information the server has gathered about the client such as name, registration number, and so on), a timestamp, and other information depending on the implementation.

Cookies

3. The server includes the cookie in the response that it sends to the client.

4. When the client receives the response, the browser stores the cookie in the cookie directory, which is sorted by the domain server name

27-2 WEB DOCUMENTS

The documents in the WWW can be grouped into three broad categories: static, dynamic, and active. The category is based on the time at which the contents of the document are determined.

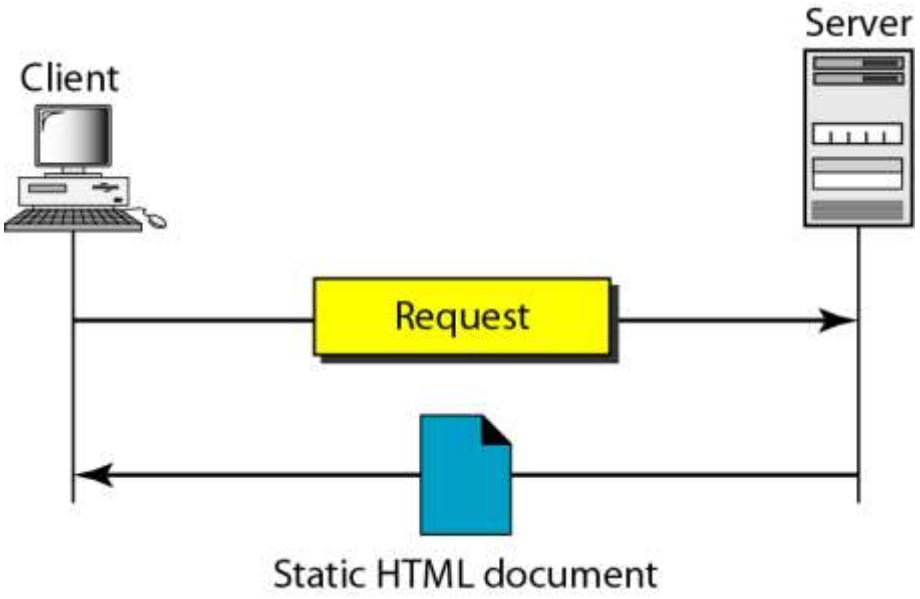
Topics discussed in this section:

Static Documents
Dynamic Documents
Active Documents

1. Static Documents

- Static documents are fixed-content documents that are created and stored in a server. The client can get only a copy of the document.
- The contents of the file are determined when the file is created, not when it is used.
- The contents in the server can be changed, but the user cannot change them.
- When a client accesses the document, a copy of the document is sent. The user can then use a browsing program to display the document

Figure 27.4 *Static document- HTML- To create Web Pages*



HTML FILE

- HTML is a format that tells a computer how to display a web page. The documents themselves are plain text files with special "tags" or codes that a web browser uses to interpret and display information on your computer screen.
- HTML stands for Hyper Text Markup Language
- An HTML file is a text file containing small markup tags.
- The markup tags tell the Web browser how to display the page .

Example

- <html>
- <head>
- <title>My First Webpage</title>
- </head>
- <body>
- This is my first homepage. This text is bold
- </body>
- </html>

Figure 27.5 *Boldface tags*

Figure 27.6 *Effect of boldface tags*

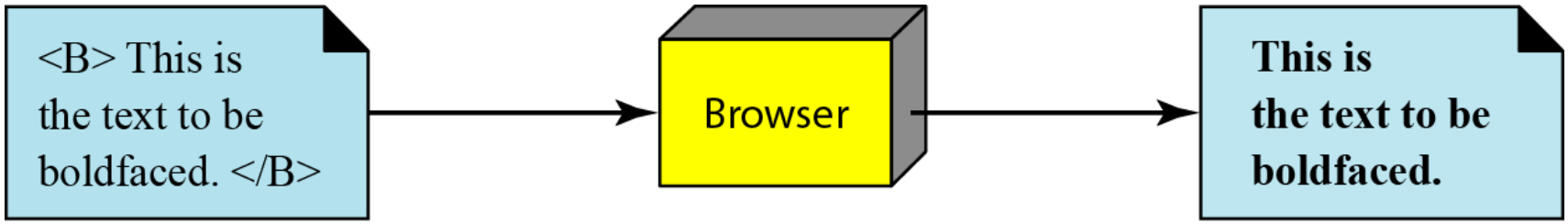


Figure 27.7 *Beginning and ending tags*

a. Beginning tag

b. Ending tag

2. Dynamic Documents

- A dynamic document is created by a Web server whenever a browser requests the document.
- When a request arrives, the Web server runs an application program or a script that creates the dynamic document. The server returns the output of the program or script as a response to the browser that requested the document.
- Because a fresh document is created for each request, the contents of a dynamic document can vary from one request to another.

Figure 27.8 *Dynamic document- created whenever the browser requests the document*

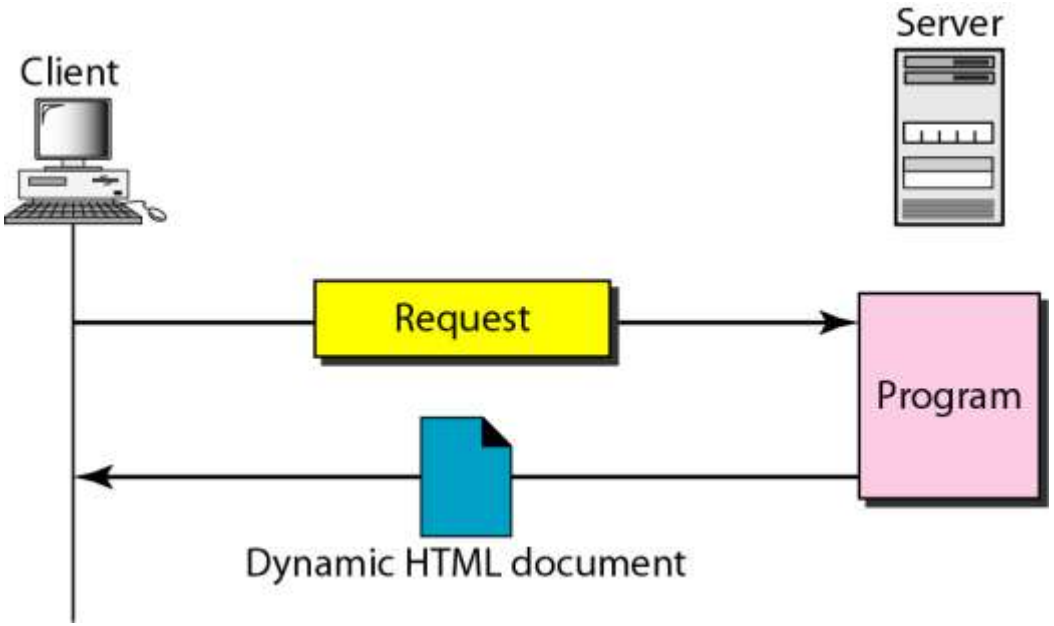
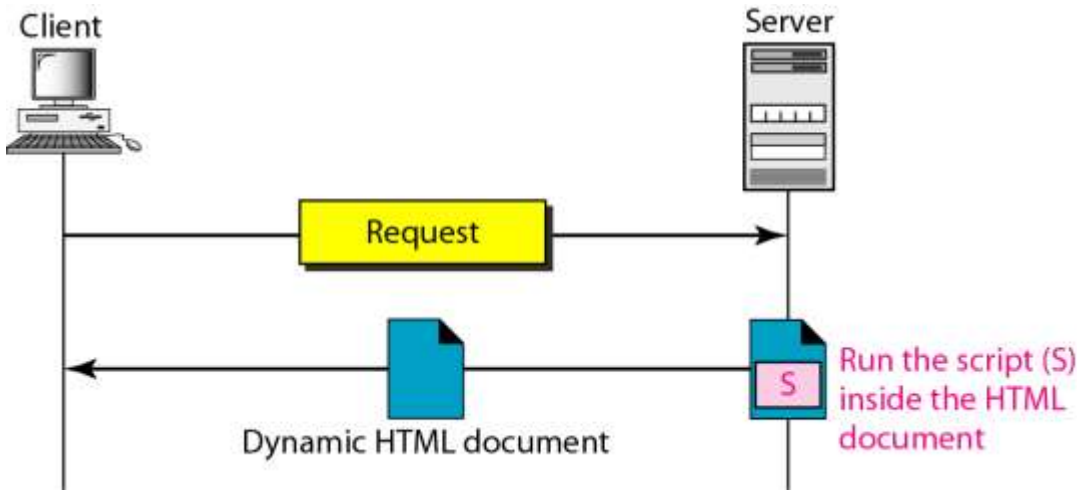


Figure 27.9 *Dynamic document using server-site script*



Note

Dynamic documents are sometimes referred to as server-site dynamic documents.

3. Active Documents:-

- An active web document consists of a computer program that the server sends to the browser and that the browser must run locally.
- When it runs, the active document program can interact with the user and change the display continuously.
- **Advantages:**
- simplicity, reliability and performance.

Active Documents:-

- Suppose we want to run a program that creates animated graphics on the screen or a program that interacts with the user.
- The program definitely needs to be run at the client site where the animation or interaction takes place.
- When a browser requests an active document, the server sends a copy of the document or a script. The document is then run at the client (browser) site.

Figure 27.10 *Active document using Java applet*

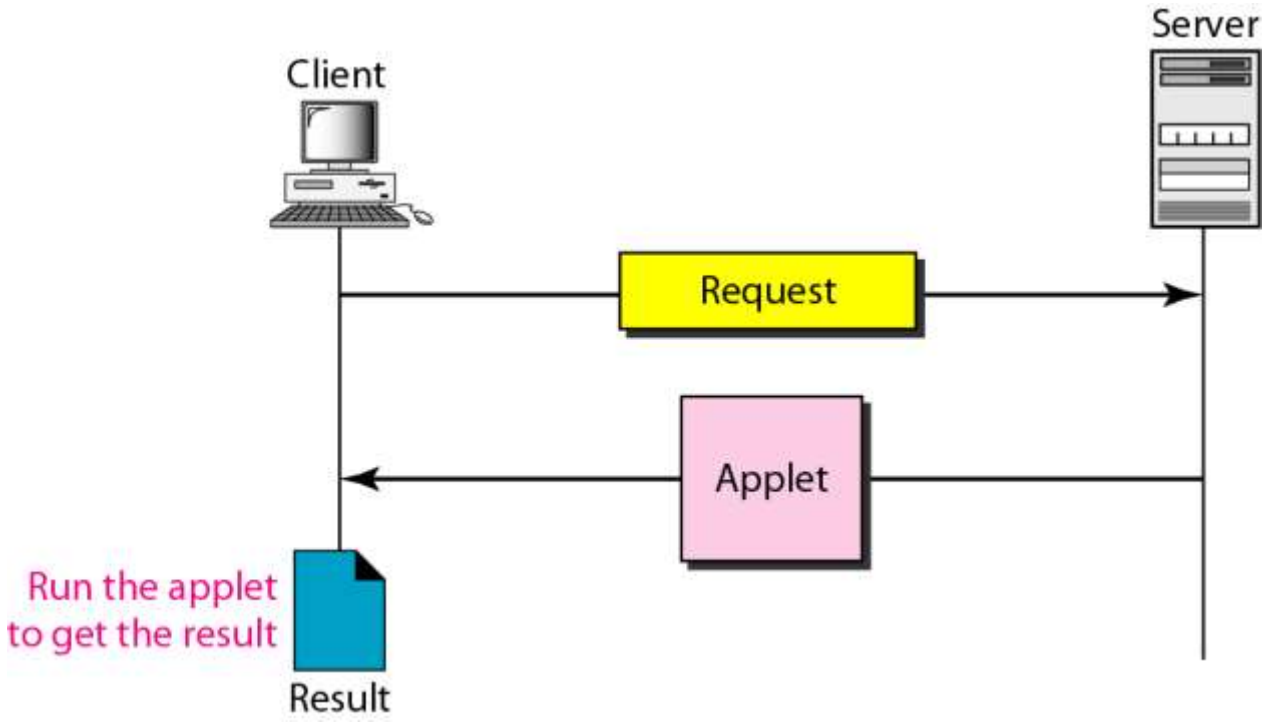
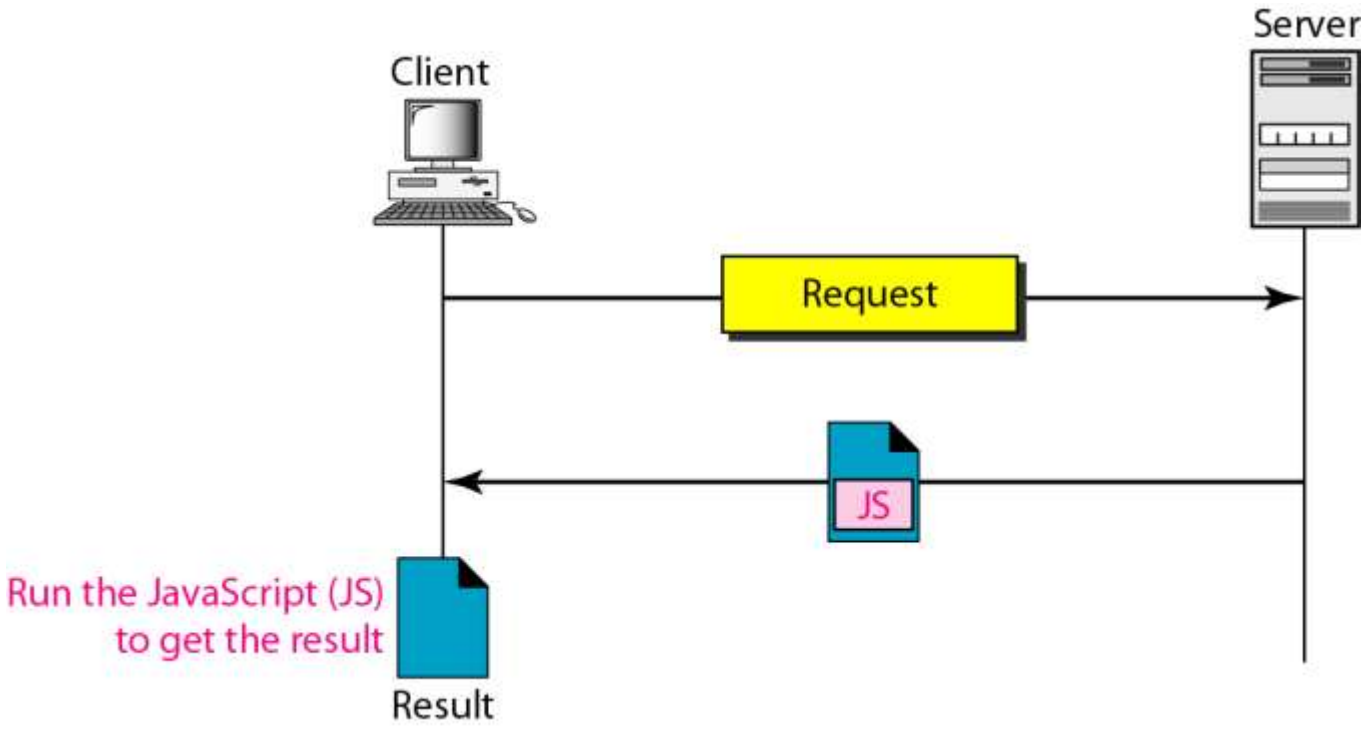


Figure 27.11 *Active document using client-site script*



Note

Active documents are sometimes referred to as client-site dynamic documents.

27-3 HTTP

The Hypertext Transfer Protocol (HTTP) is a protocol used mainly to access data on the World Wide Web. HTTP functions as a combination of FTP and SMTP.

Topics discussed in this section:

HTTP Transaction

Persistent Versus Nonpersistent Connection

HTTP:-

- It is similar to FTP because it transfers files and uses the services of TCP.
- Only data are transferred between the client and the server.
- HTTP is like SMTP because the data transferred between the client and the server look like SMTP messages.
- Read and interpreted by the HTTP server and HTTP client (browser)

HTTP:-

- SMTP messages are stored and forwarded, but HTTP messages are delivered immediately.
- The commands from the client to the server are embedded in a request message.
- The contents of the requested file or other information are embedded in a response message.

Note

HTTP uses the services of TCP on well-known port 80.

Figure 27.12 *HTTP transaction*

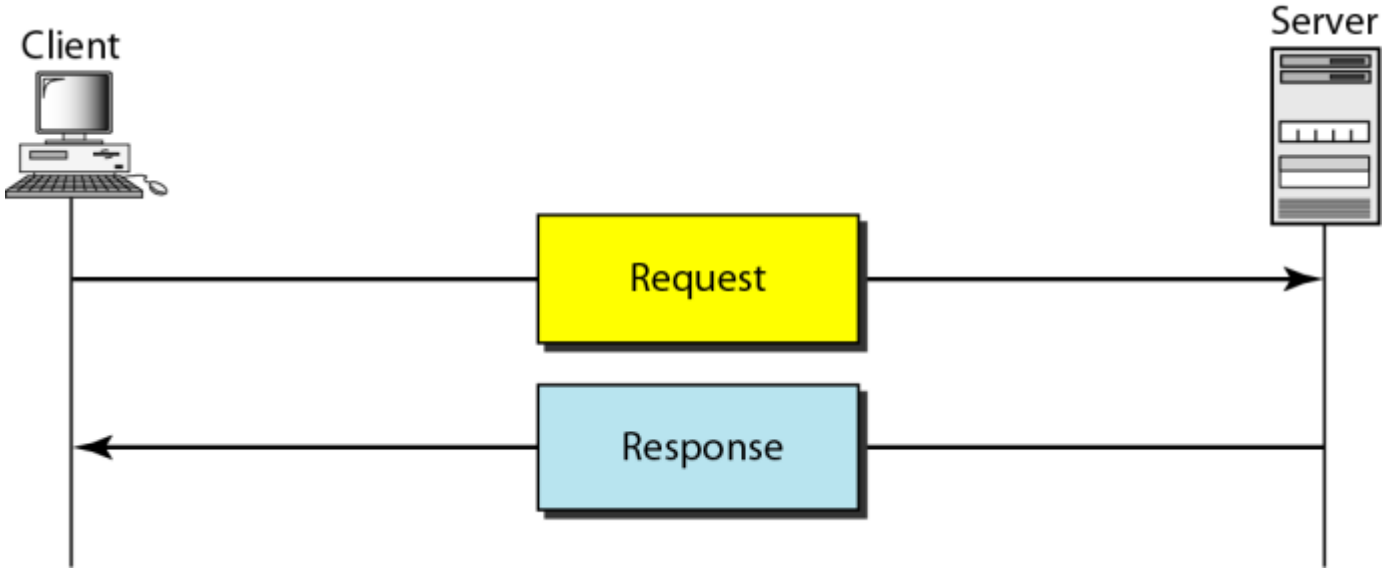


Figure 27.13 *Request and response messages*

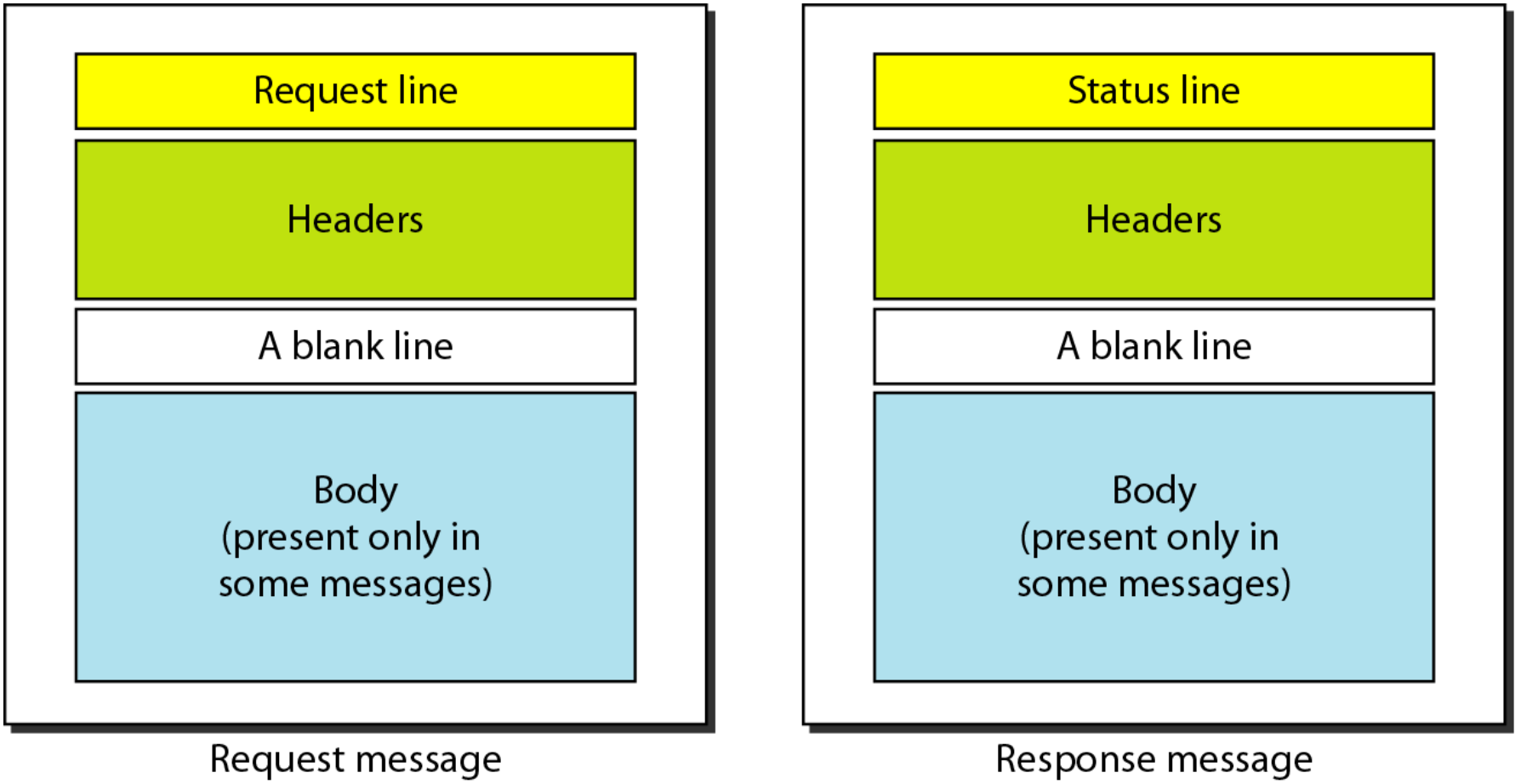
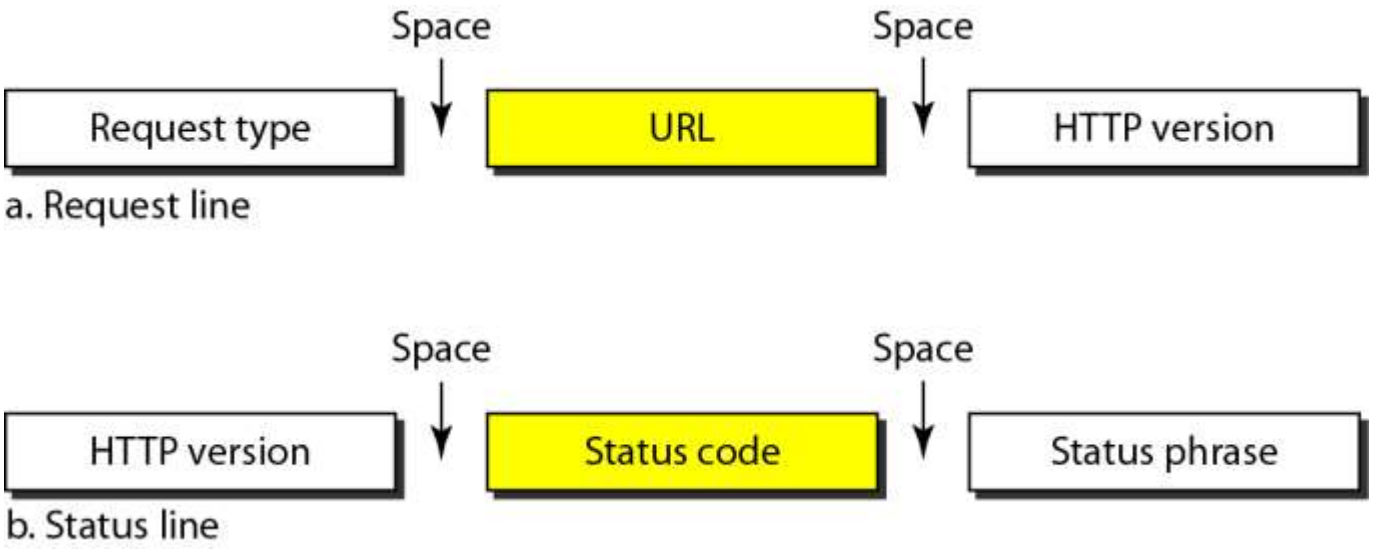


Figure 27.14 ***Request Line- First Line in the request message***
Status lines- First Line in the response message



1. **Status code.** This field is used in the response message.

- Codes in the 100 range are informational
- Codes in the 200 range indicate a successful request.
- Codes in the 300 range redirect the client to another URL, and
- Codes in the 400 range indicate an error at the client
- Codes in the 500 range indicate an error at the server

2. Status phrase. This field is used in the response message. It explains the status code in text form.

Table 27.1 Request Type:- *Methods*

Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	Inquires about available options

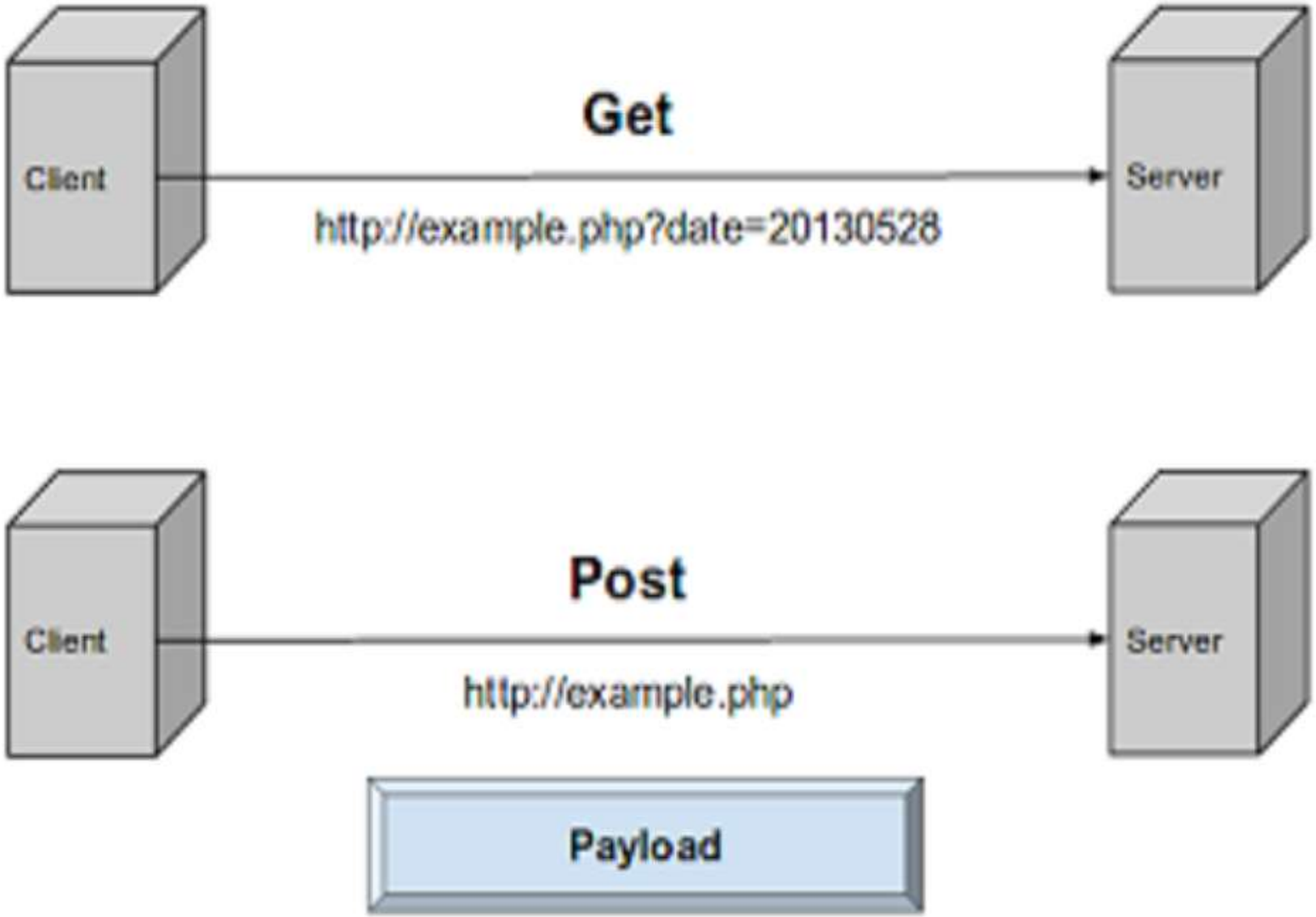

Client
Get
http://example.php?date=20130528
Server
Client
Post
http://example.php
Server
Payload

Table 27.2 *Status codes*

Code	*Phrase*	*Description*
	Informational	
100	Continue	The initial part of the request has been received, and the client may continue with its request.
101	Switching	The server is complying with a client request to switch protocols defined in the upgrade header.
	Success	
200	OK	The request is successful.
201	Created	A new URL is created.
202	Accepted	The request is accepted, but it is not immediately acted upon.
204	No content	There is no content in the body.

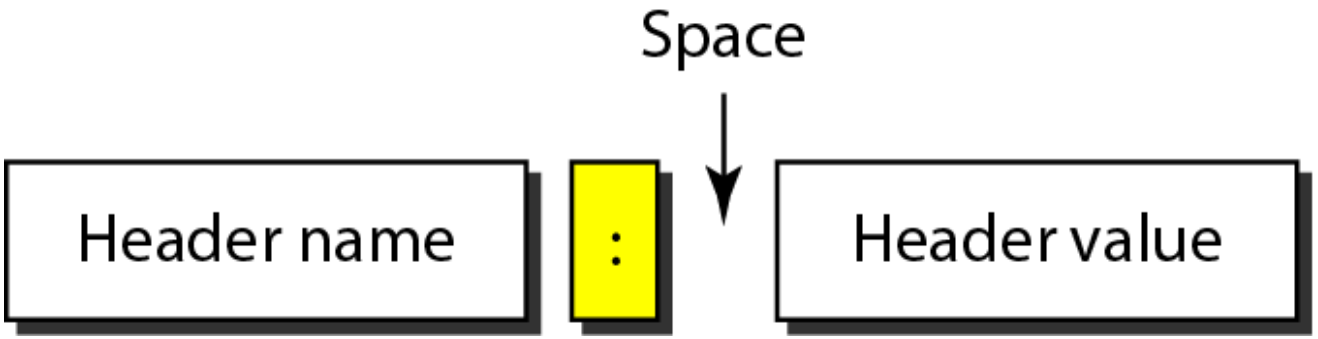
Table 27.2 *Status codes (continued)*

Code	*Phrase*	*Description*
	Redirection	
301	Moved permanently	The requested URL is no longer used by the server.
302	Moved temporarily	The requested URL has moved temporarily.
304	Not modified	The document has not been modified.
	Client Error	
400	Bad request	There is a syntax error in the request.
401	Unauthorized	The request lacks proper authorization.
403	Forbidden	Service is denied.
404	Not found	The document is not found.
405	Method not allowed	The method is not supported in this URL.
406	Not acceptable	The format requested is not acceptable.
	Server Error	
500	Internal server error	There is an error, such as a crash, at the server site.
501	Not implemented	The action requested cannot be performed.
503	Service unavailable	The service is temporarily unavailable, but may be requested in the future.

Header:

- The header exchanges additional information between the client and the server.
- For example, the client can request that the document be sent in a special format, or the server can send extra information about the document.
- The header can consist of one or more header lines. Each header line has a header name, a colon, a space, and a header value

Figure 27.15 *Header format*



method path protocol

```
GET /tutorials/other/top-20-mysql-best-practices/ HTTP/1.1
Host: net.tutsplus.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Cookie: PHPSESSID=r2t5uvjq435r4q7ib3vtdjq120
Pragma: no-cache
Cache-Control: no-cache
```

HTTP headers as Name: Value

Table 27.3 *General headers*

Header	*Description*
Cache-control	Specifies information about caching
Connection	Shows whether the connection should be closed or not
Date	Shows the current date
MIME-version	Shows the MIME version used
Upgrade	Specifies the preferred communication protocol

Table 27.4 *Request headers- Present in a request message*

Header	*Description*
Accept	Shows the medium format the client can accept
Accept-charset	Shows the character set the client can handle
Accept-encoding	Shows the encoding scheme the client can handle
Accept-language	Shows the language the client can accept
Authorization	Shows what permissions the client has
From	Shows the e-mail address of the user
Host	Shows the host and port number of the server
If-modified-since	Sends the document if newer than specified date
If-match	Sends the document only if it matches given tag
If-non-match	Sends the document only if it does not match given tag
If-range	Sends only the portion of the document that is missing
If-unmodified-since	Sends the document if not changed since specified date
Referrer	Specifies the URL of the linked document
User-agent	Identifies the client program

Table 27.5 *Response headers- Present in a response message*

Header	*Description*
Accept-range	Shows if server accepts the range requested by client
Age	Shows the age of the document
Public	Shows the supported list of methods
Retry-after	Specifies the date after which the server is available
Server	Shows the server name and version number


HTTP/1.1 200 OK
Status Line
Date: Thu, 20 May 2004 21:12:58 GMT
Connection: close
General Headers
Server: Apache/1.3.27
Accept-Ranges: bytes
Response Headers
Content-Type: text/html
Content-Length: 170
Last-Modified: Tue, 18 May 2004 10:14:49 GMT
Entity Headers
HTTP Response
<html>
<head>
<title>Welcome to the Amazing Site!</title>
</head>
<body>
This site is under construction. Please come back later. Sorry!
</body>
</html>
Message Body

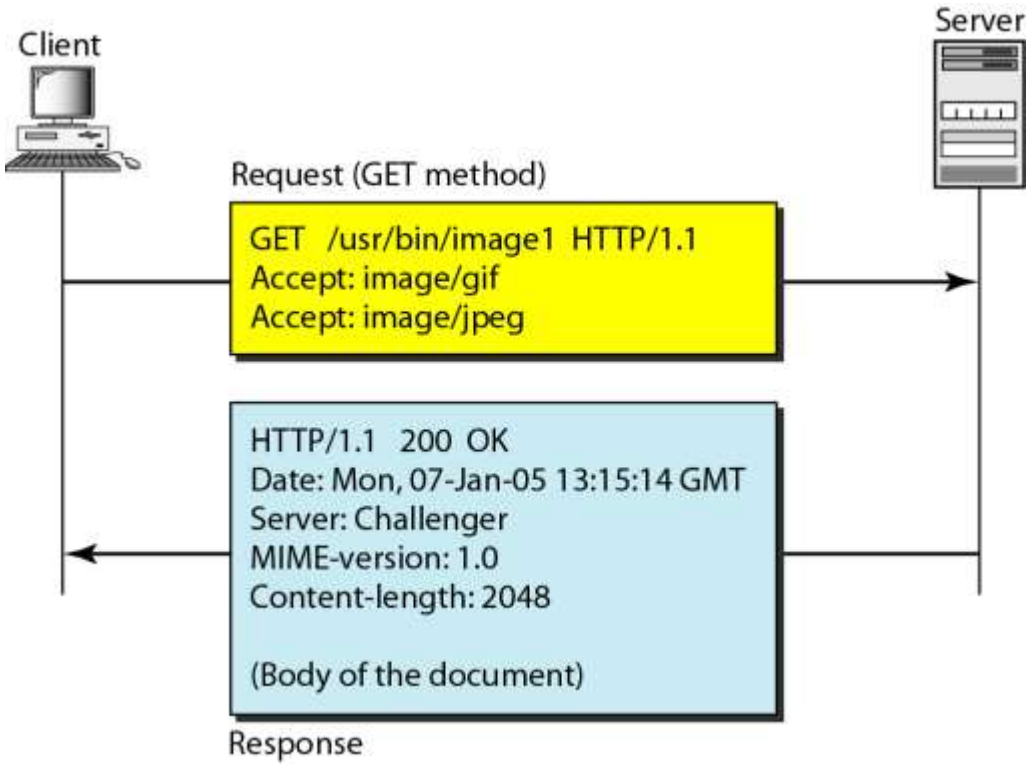
Table 27.6 *Entity headers- Information about Body of the document*

Header	*Description*
Allow	Lists valid methods that can be used with a URL
Content-encoding	Specifies the encoding scheme
Content-language	Specifies the language
Content-length	Shows the length of the document
Content-range	Specifies the range of the document
Content-type	Specifies the medium type
Etag	Gives an entity tag
Expires	Gives the date and time when contents may change
Last-modified	Gives the date and time of the last change
Location	Specifies the location of the created or moved document

Example 27.1

This example retrieves a document. We use the GET method to retrieve an image with the path /usr/bin/image1. The request line shows the method (GET), the URL, and the HTTP version (1.1). The header has two lines that show that the client can accept images in the GIF or JPEG format. The request does not have a body. The response message contains the status line and four lines of header. The header lines define the date, server, MIME version, and length of the document. The body of the document follows the header (see Figure 27.16).

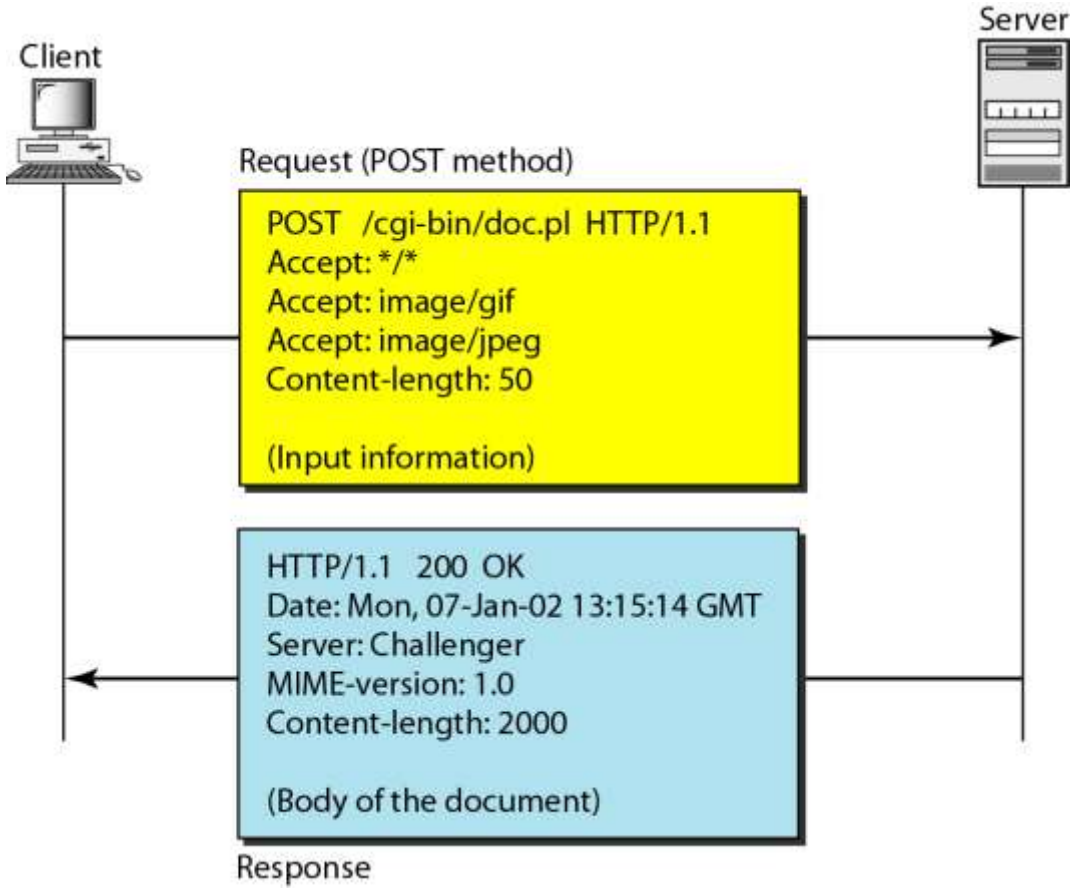
Figure 27.16 *Example 27.1*



Example 27.2

In this example, the client wants to send data to the server. We use the POST method. The request line shows the method (POST), URL, and HTTP version (1.1). There are four lines of headers. The request body contains the input information. The response message contains the status line and four lines of headers. The created document, which is a CGI document, is included as the body (see Figure 27.17).

Figure 27.17 *Example 27.2*



Example 27.3

HTTP uses ASCII characters. A client can directly connect to a server using TELNET, which logs into port 80 (see next slide). The next three lines show that the connection is successful. We then type three lines. The first shows the request line (GET method), the second is the header (defining the host), the third is a blank, terminating the request. The server response is seven lines starting with the status line. The blank line at the end terminates the server response. The file of 14,230 lines is received after the blank line (not shown here). The last line is the output by the client.

Example 27.3 (continued)

```
$ telnet www.mhhe.com 80
Trying 198.45.24.104 . . .
Connected to www.mhhe.com (198.45.24.104).
Escape character is '^]'.
GET /engcs/compsci/forouzan HTTP/1.1
From: forouzanbehrouz@fhda.edu

HTTP/1.1 200 OK
Date: Thu, 28 Oct 2004 16:27:46 GMT
Server: Apache/1.3.9 (Unix) ApacheJServ/1.1.2 PHP/4.1.2 PHP/3.0.18
MIME-version:1.0
Content-Type: text/html
```

Note

HTTP version 1.1 specifies a persistent connection by default.

Persistent Versus Nonpersistent Connection

- HTTP prior to version 1.1 specified a nonpersistent connection, persistent connection default in version 1.1.
- Nonpersistent Connection

  In a nonpersistent connection, one TCP connection is made for each request/response.

  1. The client opens a TCP connection and sends a request.
  2. The server sends the response and closes the connection.
  3. The client reads the data until it encounters an end-of-file marker; it then closes the connection.

Nonpersistent Connection

- In this strategy, for N different pictures in different files, the connection must be opened and closed N times.
- The non persistent strategy imposes high overhead on the server because the server needs N different buffers and requires a slow start procedure each time a connection is opened.

Persistent Connection

- In a persistent connection, the server leaves the connection open for more requests after sending a response.
- The server can close the connection at the request of a client or if a time-out has been reached.
- The sender usually sends the length of the data with each response.
- However, there are some occasions when the sender does not know the length of the data.

Persistent Connection

- When a document is created dynamically or actively. In these cases, the server informs the client that the length is not known and closes the connection after sending the data so the client knows that the end of the data has been reached.