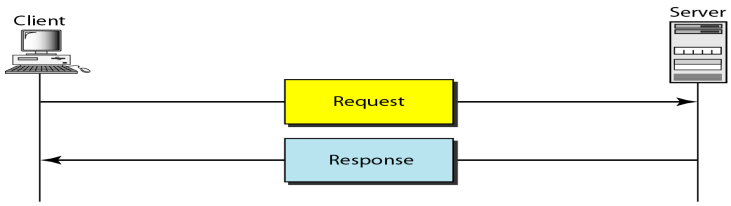
Chapter 4: HTTP and the Web Services

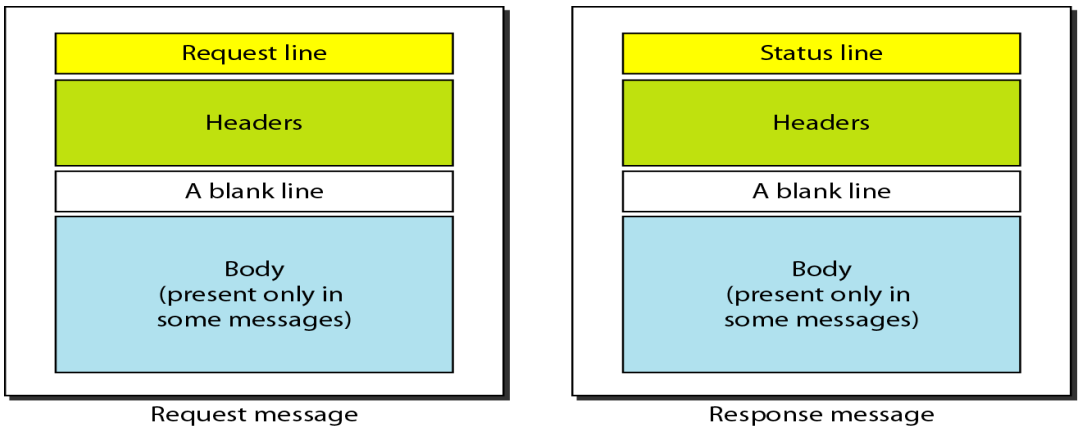
HTTP

- Hyper Text Transfer Protocol used mainly to access data on WWW.
- HTTP functions as a combination of FTP and SMTP.
- It is similar to FTP because it transfer files and use services of TCP. However, it is much simpler than FTP because it uses only one TCP connection. There is not separate control connection; 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 message. In addition, the format of the message is controlled by MIME like headers. Unlike SMTP, the HTTP messages are not designed to be read by humans; they are read and interpreted by HTTP server and HTTP client (browser).
- Web clients communicate with web server using HTTP protocol
- HTTP is based on a Request/Response Model



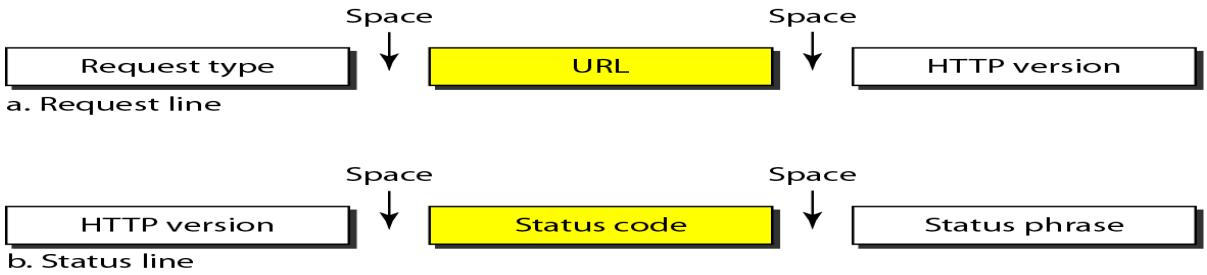
- HTTP uses a TCP connection through well known port 80
- HTTP - a stateless protocol

- **HTTP Transaction**
  - Browser opens a connection
  - Browser sends a request to a server
  - Server sends a response
  - Connection is closed
- **Request and Response Messages**
  - HTTP Transaction is initialized by sending a request message.
  - The server replies by sending a response message.
  - The formats of request and response messages are similar as shown below.
  - A request message consists of a request line, header, and sometimes body.
  - A response message consists of a status line, a header, and sometimes body.



- **Request and status lines**

The first line in a request message is called a request line; the first line in the response message is called the status line. There is one common field, as shown below.



- **Request Type:** This field is used in the request message. In version 1.1 of HTTP, several request types are defined. The request type is categorized into methods as shown below.

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

- **URL:** Uniform resource locater (search note for more info)
- **Version:** The most current version of HTTP is 1.1.
  Status code: This field is used in the response message. The status code is similar to those in the FTP and SMTP protocols. It consists of three digits, Whereas the code in the 100 range are only informational, the code in the 200 range indicates a successful request. The code in the 300 range redirect the client to another URL, and the code in

the 400 range indicate an error at the client site. Finally, the codes in the 500 range indicate an error at the server site. Figure below shows the status codes.

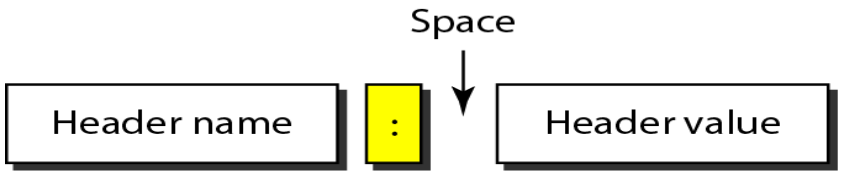
Code	*Phrase*	*Description*
	Informational	
100	Continue	The initial part of the request has been received, and the client may continue with its request.
101	Switching	The server is complying with a client request to switch protocols defined in the upgrade header.
	Success	
200	OK	The request is successful.
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.

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.

- **Status Phrase**: This field is used in the response message. It explains the ststus code in the text form. Above figure also contains status phrase.

➢ Header

The header exchanges additional information between the client and the server. e.g. the document format, file types etc. The header can contains one or more header lines. Each header line has header name, a colon, a space, and a header value. A header belongs to one of the four categories: general header, request header, response header and entity header. A request message can contain only general, request and entity headers. A response message can contain only general, response and entity headers.



- **General Headers**
  A general header gives general information about the message and can be present in both a request and response. Figure below shows general headers with descriptions.

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

- **Request headers**
  The request header specifies the client's configuration and the client's preferred document format. Figure below shows the request headers and their descriptions.

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

- **Response Headers**
  Response header specifies the server's configuration and special information about the request. Figure below shows the response headers and their descriptions.

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

- **Entity Headers**
  The entity header gives information about the body of the document. Although it is mostly present in response message, some request messages, such as POST or PUT methods, that contain a body also use this type of header. Figure below shows the entity headers and their descriptions.

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

➢ **Body:** The body can be present in a request or response message. Usually, it contains the document to be received.

Persistent Versus Nonpersistent Connection

HTTP prior to version 1.1 specified a nonpersistent connection, while a persistent connection is the default in version 1.1.

Nonpersistent Connection

In a **nonpersistent connection,** one TCP connection is made for each request/response. The following lists the steps in this strategy:

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.

In this strategy, for N different pictures in different files, the connection must be opened and closed N times. The nonpersistent 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

HTTP version 1.1 specifies a **persistent connection** by default. 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. This is the case 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.

HTTP version 1.1 specifies a persistent connection by default.
