HTTP Fundamentals

The **Hypertext Transfer Protocol (HTTP)** is an application-level protocol for distributed, collaborative, hypermedia information systems. This is the foundation for data communication for the World Wide Web since 1990. **HTTP** is a generic and Stateless and Connectionless protocol which can be used for other purposes as well using extensions of its request methods, error codes, and headers.

E.g. http:// Food.com/recipe/salad

Where,

Http: is a URL scheme,

Food.com is a host, it will go to DNS server and getting replace with its IP address

/recipe/salad... is a URL path

TCP/IP (Transmission Control Protocol/Internet Protocol)

- Application Layer (HTTP)

  It is visible layer among users.It is responsible to communicate between users.
- Transport Layer (TCP)

  It is responsible session and datagram communication services.
- Internet Layer (IP)
- Link Layer

Each Layer has its separate responsibility in whole procedure.

Basic Features of HTTP

There are three basic features that make HTTP a simple but powerful protocol:

- **HTTP is connectionless:** The HTTP client, i.e., a browser initiates an HTTP request and after a request is made, the client disconnects from the server and

waits for a response. The server processes the request and re-establishes the connection with the client to send a response back.

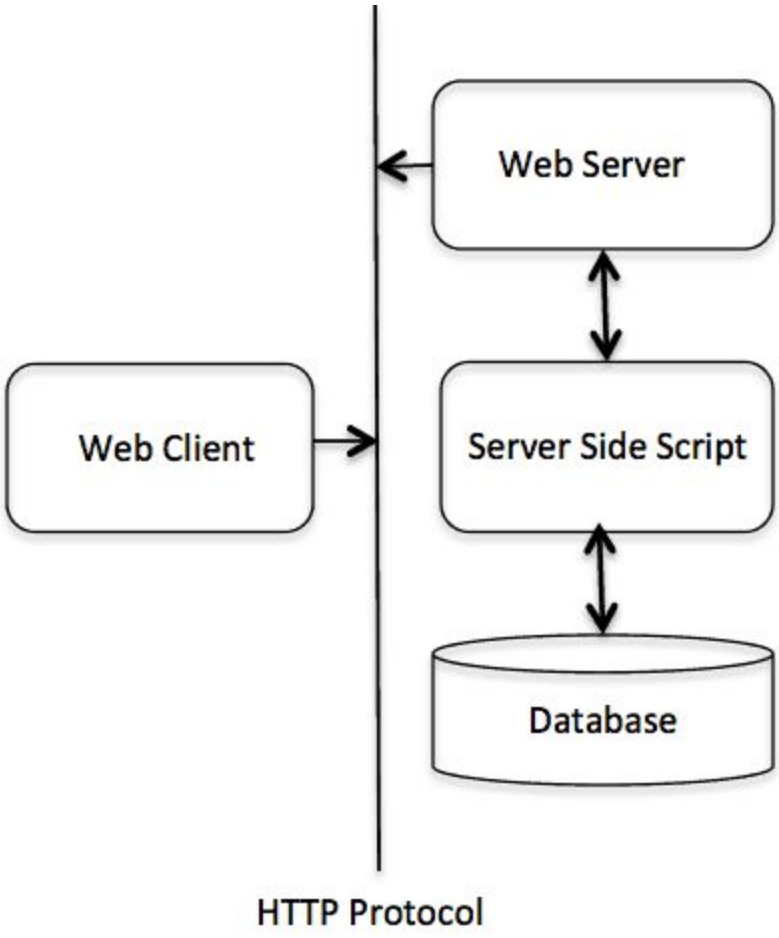
- **HTTP is media independent:** It means, any type of data can be sent by HTTP as long as both the client and the server know how to handle the data content. It is required for the client as well as the server to specify the content type using appropriate MIME-type.
- **HTTP is stateless:** As mentioned above, HTTP is connectionless and it is a direct result of HTTP being a stateless protocol. The server and client are aware of each other only during a current request. Afterwards, both of them forget about each other. Due to this nature of the protocol, neither the client nor the browser can retain information between different requests across the web pages.

Client

The HTTP client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content over a TCP/IP connection.

Server

The HTTP server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta information, and possible entity-body content.



As above diagram shows HTTP is the media between Client and Server.

Header Fields

HTTP header fields provide required information about the request or response, or about the object sent in the message body. There are four types of HTTP message headers:

- **General-header:** These header fields have general applicability for both request and response messages.
- **Request-header:** These header fields have applicability only for request messages.

- **Response-header:** These header fields have applicability only for response messages.
- **Entity-header:** These header fields define meta information about the entity-body or, if no body is present, about the resource identified by the request.

HTTP Headers

HTTP headers carry information about behavior and application state between the browser and the server.

The most important browser headers, in terms of end-user performance, are:

1. HTTP version (HTTP/1.0 or HTTP/1.1)
2. Accept-Encoding: gzip, deflate
3. Connection: Keep-Alive
4. If-* headers
5. Cache-Control or Pragma no-cache

Web Request Methods

GET

The GET method is used to retrieve information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data.

HEAD

Same as GET, but it transfers the status line and the header section only.

POST

A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.

PUT

Replaces all the current representations of the target resource with the uploaded content.

DELETE

Removes all the current representations of the target resource given by URI.

CONNECT

Establishes a tunnel to the server identified by a given URI.

OPTIONS

Describe the communication options for the target resource.

TRACE

Performs a message loop back test along with the path to the target resource.