HTTP - MESSAGES

HTTP is based on the client-server architecture model and a stateless request/response protocol that operates by exchanging messages across a reliable TCP/IP connection.

An HTTP "client" is a program *Webbrowseroranyotherclient* that establishes a connection to a server for the purpose of sending one or more HTTP request messages. An HTTP "server" is a program *generallyawebserverlikeApacheWebServerorInternetInformationServicesIIS, etc.* that accepts connections in order to serve HTTP requests by sending HTTP response messages.

HTTP makes use of the Uniform Resource Identifier *URI* to identify a given resource and to establish a connection. Once the connection is established, **HTTP messages** are passed in a format similar to that used by the Internet mail [RFC5322] and the Multipurpose Internet Mail Extensions *MIME* [RFC2045]. These messages include **requests** from client to server and **responses** from server to client which will have the following format:

```
HTTP-message   = <Request> | <Response> ; HTTP/1.1 messages
```

HTTP requests and HTTP responses use a generic message format of RFC 822 for transferring the required data. This generic message format consists of the following four items.

- A Start-line
- Zero or more header fields followed by CRLF
- An empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields
- Optionally a message-body

In the following sections, we will explain each of the entities used in an HTTP message.

Message Start-Line

A start-line will have the following generic syntax:

```
start-line = Request-Line | Status-Line
```

We will discuss Request-Line and Status-Line while discussing HTTP Request and HTTP Response messages respectively. For now, let's see the examples of start line in case of request and response:

```
GET /hello.htm HTTP/1.1     (This is Request-Line sent by the client)

HTTP/1.1 200 OK             (This is Status-Line sent by the 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.

All the above mentioned headers follow the same generic format and each of the header field consists of a name followed by a colon (**:**) and the field value as follows:

```
message-header = field-name ":" [ field-value ]
```

Following are the examples of various header fields:

```
User-Agent: curl/7.16.3 libcurl/7.16.3 OpenSSL/0.9.7l zlib/1.2.3
Host: www.example.com
Accept-Language: en, mi
Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache
Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT
ETag: "34aa387-d-1568eb00"
Accept-Ranges: bytes
Content-Length: 51
Vary: Accept-Encoding
Content-Type: text/plain
```

Message Body

The message body part is optional for an HTTP message but if it is available, then it is used to carry the entity-body associated with the request or response. If entity body is associated, then usually **Content-Type** and **Content-Length** headers lines specify the nature of the body associated.

A message body is the one which carries the actual HTTP request data *includingformdataanduploaded, etc.* and HTTP response data from the server *includingfiles, images, etc.* . Shown below is the simple content of a message body:

```
<html>
   <body>

      <h1>Hello, World!</h1>

   </body>
</html>
```

Next two chapters will make use of above explained concepts to prepare HTTP Requests and HTTP Responses