What is HTTP?

HTTP stands for Hyper Text Transfer Protocol which is an application level protocol for transferring textual data between client and server.

HTTP is stateless protocol.

Web Client such as web browser sends the request to server, server responds and sends the requested data (such as a HTML document) or returns error code and closes the connection. Once the response is returned, server doesn't remember anything about the client.(even the very next request)

Http request methods and Headers
eg : URL -- http://www.server.com:8080/bank/login.html
Whenever the client sends the request to server for any resource such as a HTML document, it specifies
1. HTTP method (get/post/put/delete....)
2. Request URI (eg : /bank/login.html)
3. Protocol version
4. Optional Header information(eg : accept , cookies)

After the client sends the request, server processes the request and sends the response.

HTTP Response contains

1. Response status information(eg : 404/200)
2. Response headers (eg : cookies/resp cont type
eg : text/html, application/json image/gif.... /cont length)
3. Response data.

Following is an example of HTTP request which uses GET request method to ask for a HTML document named tutorial.html using HTTP protocol version 1.1

```
GET /tutorial.html HTTP/1.1
```

Following is the example of request header, which provides additional information about the request.

```
User-Agent: Mozilla/4.0 (compatible; MSIE 4.0; Windows 95)
Accept: image/gif, image/jpeg, text/*, */*
eg : text/html, text/xml,application/json
```

Above header specifies the information about the client software and what MIME(Multi-purpose Internet Mail Extension) type the client accepts in response, using User-Agent and Accept headers.

Http request methods

The request method indicates to the server what action to perform on the resource identified by the request-URI.

HTTP 1.1 specifies these request methods:
GET, POST , HEAD,OPTIONS, PUT, TRACE, DELETE, CONNECT.
Servlets can process any of the above requests.

But GET and POST methods are most commonly used.

The GET request method -- safe(doesn't change state of the resource) & idempotent (repeated reqs do not have any further side effects after the first request)

The GET request is used typically for getting static information from the server such as HTML document, images, and CSS files.
It can be used to retrieve dynamic information also by appending query string at the end of request URI.

Query String

Query string is used to pass additional request parameters along with the request.

Query string format

URL?name1=value1&name2=value&name3=value3 ….

eg -- http://www.abc.com/test/login.jsp?userid=10&name=abc&age=25

The POST request method

The POST request method is typically used to access OR create new dynamic resources.

POST request is used to
1. send the large amount of data to the server.
2. upload files to servers
3. send serializable Java objects or raw bytes.

GET Vs POST

1.
GET request sends the request parameters as query string appended at the end of the request URL, whereas POST request sends the request parameters as part of the HTTP request body.

2.Unlike GET, POST request sends all the data as part of the HTTP request body, so it doesn’t appear in browser address bar and cannot be bookmarked.

3. GET -- non-secure, POST -secure
GET -- idempotent

POST---non-idempotent

Typically use GET -- to retrieve stuff from server.
use POST -- for changing some state on server(something like update)

4. Some web servers limit the length of the URL.
So if in GET request -- too many parameters are appended in query string and URL exceeds this length, some web servers might not accept the request.
For POST -- no such limitations.

Safe HTTP methods
HTTP methods are considered safe if they do not alter the server state. So safe methods can only be used for read-only operations.

eg : GET, HEAD, OPTIONS and TRACE.

In practice it is often not possible to implement safe methods in a way they do not alter any server state.

eg : a GET request might create log or update statistic values or trigger a cache refresh on the server.

Idempotent HTTP methods

Idempotency means that multiple identical requests will have the same outcome. So it does not matter if a request is sent once or multiple times.

The following HTTP methods are idempotent: GET, HEAD, OPTIONS, TRACE, PUT and DELETE.

All safe HTTP methods are idempotent but PUT and DELETE are idempotent but not safe.

Note that idempotency does not mean that the server has to respond in the same way on each request.

eg : If you delete emp details by an id using DELETE request:
1st time , it will succeed(server will send resp status code 200) then next time onwards ,since emp details are already deleted , server will send resp code 404 (indicating resource not found!)

HTTP method overview

The following table summarizes which HTTP methods are safe and idempotent:

HTTP Method	Safe	Idempotent
GET	Yes	Yes
HEAD	Yes	Yes

OPTIONS	Yes	Yes
TRACE	Yes	Yes
PUT	No	Yes
DELETE	No	Yes
POST	No	No
PATCH	No	No