What is HTTP?

The Hypertext Transfer Protocol (HTTP) is designed to enable communications between clients and servers.

HTTP is stateless protocol.

HTTP works as a request-response protocol between a client and server.

A web browser may be the client, and an application on a computer that hosts a web site may be the server.

Example: A client (browser) submits an HTTP request to the server; then the server returns a response to the client. The response contains status information about the request and may also contain the requested content.

Difference Between HTTP and HTTPS?

|  |  |
| --- | --- |
| HTTP | HTTPS |
| URL starts with http:// | URL starts with https:// |
| HTTP uses port 80 | HTTPS uses port 443 |
| HTTP is unsecure | HTTPS is secure |
| Does not require any certificate | Need SSL or TLS certificates |
| Encryption is absent | Encryption is present |
|  |  |

## Difference Between SSL and TLS:

|  |  |
| --- | --- |
| SSL | TLS |
| SSL stands for Secure Socket layer | TLS stands for Transport Layer Security |
| Less secure | More secure (TLS is just and updated or more secure version of SSL) |
|  |  |

## HTTP Methods

* **GET**
* **POST**
* **PUT**
* **HEAD**
* **DELETE**
* **PATCH**
* **OPTIONS**

## The GET Method

**GET is used to request data from a specified resource.**

**GET is one of the most common HTTP methods.**

Note that the query string (name/value pairs) is sent in the URL of a GET request:

/test/demo\_form.php?name1=value1&name2=value2

**Some other notes on GET requests:**

* GET requests can be cached
* GET requests remain in the browser history
* GET requests can be bookmarked
* GET requests should never be used when dealing with sensitive data
* GET requests have length restrictions
* GET requests is only used to request data (not modify)

## The POST Method

**POST is used to send data to a server to create/update a resource.**

The data sent to the server with POST is stored in the request body of the HTTP request:

POST /test/demo\_form.php HTTP/1.1  
Host: w3schools.com  
name1=value1&name2=value2

**POST is one of the most common HTTP methods.**

**Some other notes on POST requests:**

* POST requests are never cached
* POST requests do not remain in the browser history
* POST requests cannot be bookmarked
* POST requests have no restrictions on data length
* POST requests are ***not idempotent.*** means when you call POST request repeatedly make have side effect of creating the same resource multiple times.

## The PUT Method

**PUT is used to send data to a server to create/update a resource.**

The difference between POST and PUT is that PUT requests are ***idempotent. That is, calling the same PUT request multiple times will always produce the same result.*** In contrast, calling a POST request repeatedly make have side effects of creating the same resource multiple times.

***What is mean idempotent?***

***Idempotence is the property of certain operations in mathematics and computer science, that can be applied multiple times without changing the result beyond the initial application.***

***Idempotent Methods: Get, Put, Delete***

***Non Idempotent Methods: Post***

## The HEAD Method

**HEAD is almost identical to GET, but without the response body.**

In other words, if GET /users returns a list of users, then HEAD /users will make the same request but will not return the list of users.

HEAD requests are useful for checking what a GET request will return before actually making a GET request - like before downloading a large file or response body.

## The DELETE Method

**The DELETE method deletes the specified resource.**

## The OPTIONS Method

**The OPTIONS method describes the communication options for the target resource.**

## Compare GET vs. POST

The following table compares the two HTTP methods: GET and POST.

|  |  |  |
| --- | --- | --- |
|  | **GET** | **POST** |
| BACK button/Reload | Harmless | Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted) |
| Bookmarked | Can be bookmarked | Cannot be bookmarked |
| Cached | Can be cached | Not cached |
| Encoding type | application/x-www-form-urlencoded | application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data |
| History | Parameters remain in browser history | Parameters are not saved in browser history |
| Restrictions on data length | Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters) | No restrictions |
| Restrictions on data type | Only ASCII characters allowed | No restrictions. Binary data is also allowed |
| Security | GET is less secure compared to POST because data sent is part of the URL  Never use GET when sending passwords or other sensitive information! | POST is a little safer than GET because the parameters are not stored in browser history or in web server logs |
| Visibility | Data is visible to everyone in the URL | Data is not displayed in the URL |

## Compare PUT vs. POST

* PUT request is idempotent, POST request is Not idempotent.
* When you know the exact URL of the resource/thing you want to create or overwrite, a PUT method should be used. Alternatively, if you only know the URL of the category or sub-section of the resource/thing you want to create something within, use the POST method (A POST method should also be used if you do not know the specific URL of where your newly created resource should reside).

## HTTP – Status Codes

The Status-Code element in a server response, is a 3-digit integer where the first digit of the Status-Code defines the class of response and the last two digits do not have any categorization role. There are 5 values for the first digit:

|  |  |
| --- | --- |
| **S.N.** | **Code and Description** |
| 1 | **1xx: Informational**  It means the request has been received and the process is continuing. |
| 2 | **2xx: Success**  It means the action was successfully received, understood, and accepted. |
| 3 | **3xx: Redirection**  It means further action must be taken in order to complete the request. |
| 4 | **4xx: Client Error**  It means the request contains incorrect syntax or cannot be fulfilled. |
| 5 | **5xx: Server Error**  It means the server failed to fulfil an apparently valid request. |

HTTP status codes are extensible and HTTP applications are not required to understand the meaning of all the registered status codes. Given below is a list of all the status codes.

## 1xx: Information

|  |  |
| --- | --- |
| **Message** | **Description** |
| 100 Continue | Only a part of the request has been received by the server, but as long as it has not been rejected, the client should continue with the request. |
| 101 Switching Protocols | The server switches protocol. |

## 2xx: Successful

|  |  |
| --- | --- |
| **Message** | **Description** |
| 200 OK | The request is OK. |
| 201 Created | The request is complete, and a new resource is created. |
| 202 Accepted | The request is accepted for processing, but the processing is not complete. |
| 203 Non-authoritative Information | The information in the entity header is from a local or third-party copy, not from the original server. |
| 204 No Content | A status code and a header are given in the response, but there is no entity-body in the reply. |
| 205 Reset Content | The browser should clear the form used for this transaction for additional input. |
| 206 Partial Content | The server is returning partial data of the size requested. Used in response to a request specifying a *Range* header. The server must specify the range included in the response with the *Content-Range*header. |

## 3xx: Redirection

|  |  |
| --- | --- |
| **Message** | **Description** |
| 300 Multiple Choices | A link list. The user can select a link and go to that location. Maximum five addresses. |
| 301 Moved Permanently | The requested page has moved to a new URL. |
| 302 Found | The requested page has moved temporarily to a new URL. |
| 303 See Other | The requested page can be found under a different URL. |
| 304 Not Modified | This is the response code to an *If-Modified-Since* or *If-None-Match*header, where the URL has not been modified since the specified date. |
| 305 Use Proxy | The requested URL must be accessed through the proxy mentioned in the *Location* header. |
| 306 *Unused* | This code was used in a previous version. It is no longer used, but the code is reserved. |
| 307 Temporary Redirect | The requested page has moved temporarily to a new URL. |

## 4xx: Client Error

|  |  |
| --- | --- |
| **Message** | **Description** |
| 400 Bad Request | The server did not understand the request. |
| 401 Unauthorized | The requested page needs a username and a password. |
| 402 Payment Required | *You can not use this code yet*. |
| 403 Forbidden | Access is forbidden to the requested page. |
| 404 Not Found | The server can not find the requested page. |
| 405 Method Not Allowed | The method specified in the request is not allowed. |
| 406 Not Acceptable | The server can only generate a response that is not accepted by the client. |
| 407 Proxy Authentication Required | You must authenticate with a proxy server before this request can be served. |
| 408 Request Timeout | The request took longer than the server was prepared to wait. |
| 409 Conflict | The request could not be completed because of a conflict. |
| 410 Gone | The requested page is no longer available. |
| 411 Length Required | The "Content-Length" is not defined. The server will not accept the request without it. |
| 412 Precondition Failed | The pre condition given in the request evaluated to false by the server. |
| 413 Request Entity Too Large | The server will not accept the request, because the request entity is too large. |
| 414 Request-url Too Long | The server will not accept the request, because the url is too long. Occurs when you convert a "post" request to a "get" request with a long query information. |
| 415 Unsupported Media Type | The server will not accept the request, because the mediatype is not supported. |
| 416 Requested Range Not Satisfiable | The requested byte range is not available and is out of bounds. |
| 417 Expectation Failed | The expectation given in an Expect request-header field could not be met by this server. |

## 5xx: Server Error

|  |  |
| --- | --- |
| **Message** | **Description** |
| 500 Internal Server Error | The request was not completed. The server met an unexpected condition. |
| 501 Not Implemented | The request was not completed. The server did not support the functionality required. |
| 502 Bad Gateway | The request was not completed. The server received an invalid response from the upstream server. |
| 503 Service Unavailable | The request was not completed. The server is temporarily overloading or down. |
| 504 Gateway Timeout | The gateway has timed out. |
| 505 HTTP Version Not Supported | The server does not support the "http protocol" version. |