**HTTP Request Methods:**

The two most common HTTP methods are: GET and POST.

**What is HTTP**:

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

**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

**Important 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 are 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.

**important 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

**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 have side effects of creating the same resource multiple times.

**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.

**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 |