**HTTP Request Methods**

1. Get – request to effect a transfer

* Retrieve a resource
* Required to be supported by all compliant general – purpose

1. Head – entity will not be returned

* Retrieve metadata about the entity
* Link rot- links will be broken for you don’t maintain it periodically
* Supported by server for it to be compliant

1. Post – allowed to modify

* Script, active element

1. Put – create resources, send resources to the server

* Read-write resource

1. Delete – dav\_module

* MKCOL- create resource (folder)

1. Options – query what a server can do
2. Trace – requests a loopback of the request message

* Protocol – what should and shouldn’t be done
* Used for testing/diagnostics of the request/response chain

1. Connect – establishment of an encryption tunnel

* Exchange digital, encryption keys
* Decrypting HTTPS – C F S – man in the middle

**Standard Methods**

1. Safe Method

* No changes being done on server resources
* Get, head, options, trace (don’t modify anything)
* Not Safe – post, put, delete (make changes)

1. Idempotent Method

* Using it repeatedly and expecting the same results
* Get, head, put, delete, options, trace

1. Cacheable Methods

* Can it be stored in caches

**HTTP Message Headers**

General Header Fields(both client and server):

1. Cache Control

* Control caching of requests and responses.
* E.g. origin server

1. Connection

* Establish and control persistent connections

1. Date

* When the response was generated

1. Pragma

* Generic directive

1. Trailer

* Preceed date

1. Transfer-Encoding

* How the body is encoded

1. Upgrade

* Switching to a different protocol
* 101 switching protocols

1. Via

* Where the trace

1. Warning

* Some errors (coding is incorrect, missing header)

Request Header Fields:

* Negotiate with the server (Accept, Accept Charset, Accept-Encoding, Accept-language)
  1. Authorization
* Conjunction with wwww. Authenticate
* Only authorized users can access it.
  1. Proxy-Authorization
* Pairs of header
* Proxy-authentication
  1. Expect
* Two face operation
* 100 continue, Expectation failed
  1. From
* Contact address of user generating the server
  1. Host
* HTTP 1.1 (virtual hosting)
  1. If-Non-Match
* “764-547cd3082f” Entity tag (copy of the resource)
* Can use cache (conditional retrieval)
* E.g. send me the copy of the resource if it’s not yet existing, but if I ever have a copy of it don bother sending it.
  1. If-Modified-Since
* Time-stamp of modification
  1. If-range
* Part of the copy is with you while the other part is mine, send me the full copy (Completion)
  1. Accept-Range
* partial
  1. Max-forwards
* Tracing (for troubleshooting)
* Max-forwards☹indicate possible value
  1. Referer
* Embedded in HTML
* Keep track of where traffic is coming from
  1. TE
* Trailer encoding
* Headers at the end of the data
  1. User-agent
* Identity of requesting client

Response Header Fields(servers):

1. Accept-Range

* Partial retrieval of any source

1. Age

* Response generated by the server
* Used for controlling caching

1. ETag

* Weak identifier for the resource/identity

1. Location

* redirection

1. Proxy-Authenticate
2. Retry-After

* Thing you’re accessing is not available at the moment
* 503 Service Unavailable

1. Server

* Used to indicate information about the server

1. Vary

Entity Header Fields:

1. Allow

* what can you do/ what you are allowed to do

1. Content Encoding

* Encoding
* Encrypted/chunk (zip)

1. Content Language

* Identifiably in that specific language

1. Content Length

* How many bytes

1. Content-Location
2. Content-MD5 (deprecated)

* Hashing algorithm
* Costly, before you send something, you must hash it first

1. Content-Range

* Range request (byte 0-100)

1. Content-Type

* What kind of resource
* Mind type

1. Expires

-used for retrieving another content

1. Last-Modifier

* Last date and time the resource was modified
* Controlling cache