HTTP y HTTPS

HTTP

- HTTP Hypertext Transfer Protocol
- Es un protocolo para distribuir, colaborar sistemas de información hypermedia.
- Fue creado por la World Wide Web desde 1990.
- Es un protocolo sin estado.
- Esta basado en TCP/IP

Características básicas

- HTTP is connectionless
- HTTP is media independent
- HTTP is stateless

HTTP Version

- HTTP uses a <major>.<minor> numbering scheme to indicate versions of the protocol. The version of an HTTP message is indicated by an HTTP-Version field in the first line. Here is the general syntax of specifying HTTP version number:

- HTTP-Version = "HTTP" "/" 1*DIGIT "." 1*DIGIT

- Example

- HTTP/1.0

  or

- HTTP/1.1

Uniform Resource Identifiers

- Uniform Resource Identifiers (URI) are simply formatted, case-insensitive string containing name, location, etc. to identify a resource, for example, a website, a web service, etc. A general syntax of URI used for HTTP is as follows:

- URI = "http:" "//" host [":" port] [abs_path ["?" query]]

Date/Time Formats

All HTTP date/time stamps MUST be represented in Greenwich Mean Time (GMT), without exception. HTTP applications are allowed to use any of the following three representations of date/time stamps:

Sun, 06 Nov 1994 08:49:37 GMT ; RFC 822, updated by RFC 1123
Sunday, 06-Nov-94 08:49:37 GMT ; RFC 850, obsoleted by RFC 1036
Sun Nov 6 08:49:37 1994 ; ANSI C's asctime() format

Character Sets

We use character sets to specify the character sets that the client prefers. Multiple character sets can be listed separated by commas. If a value is not specified, the default is the US-ASCII.

US-ASCII

or

ISO-8859-1

or

ISO-8859-7

Content Encodings

A content encoding value indicates that an encoding algorithm has been used to encode the content before passing it over the network. Content coding are primarily used to allow a document to be compressed or otherwise usefully transformed without losing the identity.

Accept-encoding: gzip

or

Accept-encoding: compress

or

Accept-encoding: deflate

Media Types

- HTTP uses Internet Media Types in the Content-Type and Accept header fields in order to provide open and extensible data typing and type negotiation. All the Media-type values are registered with the Internet Assigned Number Authority (IANA). The general syntax to specify media type is as follows:

```
media-type     = type "/" subtype *( ";" parameter )
```

```
Accept: image/gif
```

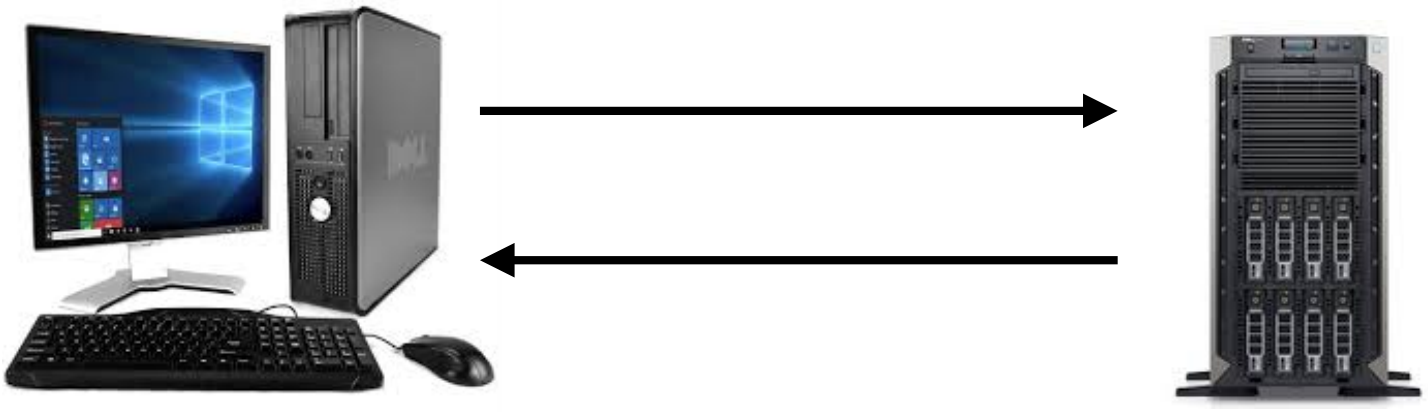
Language Tags

- HTTP uses language tags within the Accept-Language and Content-Language fields. A language tag is composed of one or more parts: a primary language tag and a possibly empty series of subtags:

```
language-tag  = primary-tag *( "-" subtag )
```

```
en, en-US, en-cockney, i-cherokee, x-pig-latin
```

HTTP-Messages



Cliente **Servidor**

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.

```
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.

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

Message Start-Line

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

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

- **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.

- 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

HTTP-Request

The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by space SP characters.

```
Request-Line = Method SP Request-URI SP HTTP-Version CRLF
```

1.GET
2.HEAD
3.POST
4.PUT
5.DELETE
6.CONNECT
7.OPTIONS
8.TRACE

HTTP - Satuts Codes

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

Validador HTTP

- https://validator.w3.org/i18n-checker/