HTTP

- **Hypertext Transfer Protocol (HTTP)**
  - Protocolo de Transferencia de Hipertexto
  - Capa de aplicación
  - Transmitir documentos hipermedia como HTML
  - Comunicación entre clientes y servidores
  - Protocolo de transacciones
    - Petición - respesta
  - Protocolo sin estado
    - Stateless
  - Se suele usar con TCP/IP

Como funciona?

- El cliente abre una conexión
  - Por ejemplo TCP
- El cliente inicia una petición HTTP(Request)
- El servidor devuelve la respuesta HTTP (response)
- Se cierra la conexión

HTTP/3

- Versión Draft
- Antes conocido como "HTTP over QUIC"

Conceptos

- User-agent
  - Agente de usuario
  - Programa que representa una persona, el cliente (pej. el navegador)
- Stateless
  - HTTP es un protocolo sin estado
  - No se guarda información sobre conexiones anteriores
  - La respuesta del servidor es la misma para un cliente previo que para uno nuevo
  - Ventajas
    - Escabilidad
    - Menos complejidad
    - Mayor rendimiento
    - Se puede cachear los recursos

    - Desventajas
        - Complica la interacción con el usuario
        - Hace falta información extra para mantener la sesión
        - Susceptible a ataques como DDoS (Distribuited Denial of Services)
- URL
    - Uniform Resource Locator
    - Localizador de Recursos uniforme
    - Recurso (resource)
        - Información que solicita el cliente
        - Por ejemplo
            - Documento HTML
            - Imagen
            - Video
    - Referencia a recuro web especificado la localización en una red y un mecanismo para obtenerlo
    - Son únicas, cada URL identifica unívocamente un recurso

URL – Formato

`scheme://host[:port]/path[?query][#fragment]`

- **scheme**:
  - `http` (HyperText Transport Protocol)
  - `https` (HyperText Transport Protocol Secure)
  - `ftp` (File Transfer Protocol)
  - Y más
- **host:** nombre o la IP del servidor al cual nos queremos conectar
- **port:** puerto al cual nos queremos conectar (opcional). Por defecto:
  - `http` → 80
  - `https` → 443
  - `ftp` → 21

`scheme://host[:port]/path[?query][#fragment]`

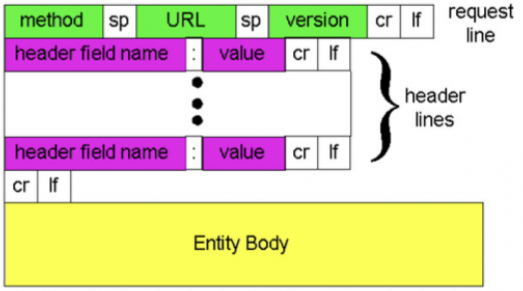
- **path:** ruta en el sistema de archivos de la máquina remota donde se encuentra el recurso. La ruta es relativa al directorio raíz de la web.
- **query:** Parámetros extra, normalmente con el formato "`q=text`"
- **fragment:** Parámetros extra

Tipos

- URI
  - Uniform Resource Identifier
    - RFC
    - Shcemes
- URL
  - Uniform Resource Locator
- URN
  - Uniform Resource Name
- URC
  - Uniform Resource Characteristic
    - Metadatos

Petición HTTP

```
Método SP URL SP Versión Http CRLF
(nombre-cabecera:valor-cabecera(,valor-cabecera)*CRLF)*
CRLF
Cuerpo del mensaje
```



- SP: espacio en blanco
- CRLF: retorno de carro
- (): opcional
- *: se puede repetir
- https://www.w3.org/Protocols/rfc2616/rfc2616-sec5.html

Método

- También llamados verbos
- Acción que deseamos efectuar sobre el recurso indicado en la petición
  - Recurso podría ser un archivo que reside en un servidor, o podría ser un programa que se esta ejecutando en dicho servidor
- Tipos
  - GET
    - Obtener un recurso
    - No modifica nada en el servidor
    - Método obligatorio
    - No tiene cuerpo
  - POST
    - Envía datos al servidor para que sean procesados por el recurso especificadoo en la petición
    - Los datos se incluyen en el cuerpo de la petición
    - Podría crear un nuevo recurso en el servidor, o actualizar un recurso ya existente
  - HEAD
    - Obtener la cabecera de una petición GET sin el contenido
    - Obtener la meta-información del recurso
    - Para conocer el tamaño/versión
    - Método obligatorio
  - PUT

        - Envía un recurso(un archivo) al servidor
        - A diferencia que POST este método crea una nueva conexión (socket) y la emplea para enviar el recurso, lo cual resulta más eficiente que enviarlo dentro del cuerpo del mensaje
    - DELETE
        - Elimina el recurso especificado
    - TRACE
        - Pide al servidor que le envíe un mensaje de respuesta
        - Se suele emplear para diagnosticar posibles problemas de conexión
    - OPTIONS
        - Pide al servidor que le indique los métodos HTTP que soporta para una determinada URL
    - CONNECT
        - Se emplea para transformar una conexión ya existente a una conexión encriptada (HTTPS)
    - PATCH
        - Modificar parcialmente un recurso ya existente en el servidor
- Siempre van en MAYÚSCULAS

Safe methods

- Aquellos que no deberían cambiar el estado del servidor
- Recuperan información

Idempotent methods

- Múltiples peticiones debrían tener el mismo resultado

URL

- URL del recurso que se esta pidiendo

Versión

- Del prótocolo HTTP que se esta empleando

Cabeceras

- Una o más
- nomrbe:valor(,valor)*
- Los espacios en blanco forman parte del valor, no son separadores
- Cada cabecera es una linea

Respuesta HTTP - Códigos de estado

- Números de tres dígitos
- Forman parte de las respuestas HTTP
- Explcian qué ha sucedido al intentar llevar a cabo una petición
- Varias Categorías
    - 1xx Mensajes
        - 100 Continue: Conexión aceptada, continua la petición
        - 101 Switching Protocols: Cmaiando Protocolos
        - 102 Procesing: Procesado todavía la petición
    - 2xx Operación realizada con éxito
        - 200 Ok: la petición se ha realizado con éxito
        - 201: Created: Petición OK y se ha creado un nuevo recurso
        - 202 Accepted: Petición aceptada pero el proceso no ha terminado todavía
        - 204 No content: Sin contenido
    - 3xx Redirección
        - 301 Moved Permanently: Contiene la nueva URL
        - 302 Found: EL cambio es temportal
        - 304 Not Modified: No modificado, se puede usar la versión cacheada
    - 4xx Error por parte del cliente
        - 400 Bad Request: Petición malformada o invalida
        - 401 Unauthorized: El cliente tiene que registrarse
        - 403: Forbidden: El cliente no tiene derechos de acceso
        - 404 Not Found: No existe el recurso indicado
    - 5xx Error del Servidor
        - 500 Internal Server Error: El servidor no sabe gestionar la situación
        - 502 Bad Gateway

HTTPS

- Hypertext Transfer Protocol Secure
- Protocolo de aplicación basado en el protocolo HTTP
- Destinado a la transferencia segura de datos
- Utiliza un cifrado basado en SSL/TLS
    - Puerto 443

Postman

- Plataforma de desarrollo API
    - API
        - Application Programm Interfaces

        - Conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el SW de las aplicaciones permitiendo la comunicación entre dos aplicaciones de SW a través de un conjunto de reglas
    - Enviar peticiones HTTP
    - Testing
    - Simular Endpoints
- Antes era una extensión del navegador