El protocol HTTP

Defineix com ha de ser el format i l'intercanvi de peticions i respostes transferides entre un client HTTP i un servidor HTTP, per tal de què aquest pugui entendre quin recurs vol el client i el pugui oferir adientment.

Normalment, els recursos oferts pels servidors HTTP són pàgines web emmagatzemades al seu disc dur, però no té perquè ser així: poden ser imatges, documents JSON i, en general, qualsevol tipus d'informació.

Exemples de clients HTTP poden ser qualsevol navegador web (Chrome, Firefox, Safari, Edge, etc), la comanda *wget*, la comanda *curl*, etc. Exemples de servidors HTTP poden ser Apache, Nginx, IIS, etc

Peticions HTTP (1/2)

Totes les peticions HTTP tenen la següent estructura:

- Una primera línia amb el format **<METODE> <PATH> <VERSIO>**
- Següents línies anomenades de "capçalera de client" amb el format **NomCapçalera:Valor** (case-insensitive)
- Una línia en blanc
- Només en peticions de tipus POST: el cos de la petició (conté dades, les quals poden estar codificades en diversos formats, que aporten informació addicional al servidor)

Peticions HTTP (2/2)

- El valor de *<METODE>* sol ser la paraula **GET** o **POST** (entre d'altres possibles com **HEAD**, **PUT**, **DELETE** o **OPTIONS**, etc). Aquesta paraula representa l'acció que el client vol executar sobre un determinat recurs allotjat al servidor. Concretament, el mètode GET serveix per demanar obtenir un determinat recurs (una pàgina web, una foto, un document JSON, etc) i el mètode POST serveix per omplir de dades un recurs (és a dir, per enviar dades al servidor, com per exemple les indicades a un formulari web)
- El valor de *<PATH>* indica la ruta concreta dins del servidor on es troba el recurs sobre el qual es vol executar el mètode indicat anteriorment
- El valor de *<VERSIO>* indica la versió del protocol utilitzat i actualment pot ser **1.1** (on la transmisió de les dades es fa en format textual) o **2** (on aquesta transmisió es fa de forma binària)

Per exemple, si el client (un navegador) vol demanar obtenir una pàgina web allotjada al servidor, el mètode que haurà de fer servir és GET i el PATH serà la ruta de la pàgina web a aconseguir (la qual se sol escriure formant part de la URL indicada a la barra de direccions del navegador, concretament, després del nom DNS del servidor). Cal fer notar que aquest PATH no té per què correspondre's amb la ruta real del recurs en el sistema de fitxers propi del servidor web.

Estructura d'una URL

L'estructura d'una URL és la següent:

protocol://maquinaremota:port/ruta/fins/el/recurs?clau1=valor1&clau2=valor2...

- El protocol pot ser la cadena "https" o "http" segons si la connexió és sobre un canal xifrat o no, respectivament. Actualment per defecte tots els navegadors usen el primer
- El nom de "maquinaremota" és el nom DNS del servidor HTTP al que s'hi connecta
- El valor del port és el nombre on estarà escoltant el servidor. Si el protocol emprat pel servidor per rebre les peticions és el "http", aquest port per defecte és el nº80 i si és el "https", serà el nº443. Si el servidor empra efectivament aquests ports, llavors no caldrà indicar-ho a la URL perquè els clients ja s'hi connecten per defecte a ells.
- El valor "/ruta/fins/el/recurs" és el PATH. És a dir, és específicament la ruta on es troba el recurs en qüestió dins de la jerarquia interna de carpetes existents al servidor
- Si el recurs fos dinàmic (és a dir, si fos un programa que cada cop que s'executés pogués retornar un resultat variable, com és el cas, per exemple, de les pàgines PHP) i per executar-lo s'usés el mètode GET, se li poden passar paràmetres d'entrada mitjançant una cadena final precedida del símbol **"?"** (que fa de separador respecte el PATH precedent) i que ha de tenir el format **nom1=valor1&nom2=valor2**... A aquesta cadena se li anomena **"querystring"**.

Cal tenir present que no tots els caràcters estan permesos en una URL: només es poden incloure les lletres minúscules i majúscules de l'alfabet anglès, els dígits i certs símbols. A més, en el cas de voler "escapar" algun dels símbols permesos amb significat especial (com per exemple ?, &, =, /, : ...) cal codificar-los seguint una taula indicada als apunts de teoria.

Les peticions POST

Aquest mètode està dissenyat per enviar informació des del client (continguda al cos de la petició) cap al recurs identificat per <PATH> (ubicat al servidor), perquè aquest la processi segons el tingui programat. Un cas típic és enviar dades des d'un formulari web cap a una pàgina PHP.

Però hem vist que el mètode GET també permet que un client pugui enviar dades al servidor mitjançant la creació d'una querystring al final de l'URL del recurs sol·licitat. Aleshores, quina és la diferència entre tots dos mètodes a l'hora d'enviar dades al servidor? La diferència principal és al lloc dins de la petició on es troben les dades a enviar al servidor: **amb el mètode GET s'envien, tal com hem dit, dins de la pròpia URL sol·licitada (en forma de querystring**, i per tant, visibles per a tothom a la barra d'adreces del navegador) i **amb el mètode POST s'envien dins del cos de la petició** (i per tant, romanen una mica més interns).

Les dades enviades mitjançant peticions GET estan escrites en un format anomenat **"application/x-www-form-urlencoded"** (o, per escurçar, "URL-encode"); això vol dir que compleixen dos requisits: segueixen l'estructura -ja vista- d'una querystring i usen les regles de codificació de caràcters mostrades a la taula anterior.

Les dades enviades mitjançant POST poden estar escrites igualment en el format **"URL-encode"** (és a dir, seguir l'estructura de querystring i estar codificades convenientment -això sí, dins del cos de la petició-) però també poden tenir un altre format més específic anomenat **"multipart/form-data"**, dissenyat especialment per a la transferència de dades binàries (útil, doncs, per a la "pujada" de fitxers) o altres més recents, com **"application/json"**, per transmetre dades en format JSON. El format utilitzat per defecte en totes les peticions POST, de totes maneres, és sempre el "URL-encode".

Capçaleres de les peticions

Les línies de capçalera enviades en les peticions dels clients serveixen per informar al servidor sobre detalls técnics d'aquesta per a què en pugui respondre més adequadament. L'única capçalera de petició que és obligatòria, però, és *Host:*, la resta són opcionals:

- **Host:** Indica el nom DNS (seguit de ":" i un nº de port si aquest fos diferent del 80/443) del servidor HTTP/S al que se li envia la petició. Aquesta capçalera pot semblar redundant, però en casos on un servidor HTTP amb una única adreça IP té associats diversos noms DNS (una cosa força habitual) aquesta capçalera permet localitzar correctament el recurs indicat a la URL, ja que recordeu que la petició arriba al servidor *un cop s'ha fet ja la resolució DNS*
- **Accept:** Indica al servidor els tipus MIME (separats per comes) dels recursos sol·licitats que el client és capaç d'acceptar (els recursos el tipus MIME dels quals no estigui inclòs a la llista especificada en aquesta capçalera seran rebutjats pel client). Un exemple (que només accepta pàgines HTML) podria ser aquest: *Accept:text/html*

Un tipus MIME és una cadena que serveix per classificar un recurs segons la seva naturalesa: text, imatge, àudio.. Cada tipus MIME consta d'un tipus principal genèric ("text", "image", "àudio"...) i un subtipus més concret (adequat al tipus principal) que indica el format específic d'aquest recurs. Alguns dels tipus MIME més habituals són *text/plain* (text pla genèric), *text/html* (codi HTML), *text/css* (codi CSS), *application/javascript* (codi Javascript), *application/json* (dades de tipus JSON), *application/x-www-urlencoded* (dades en format "URL-encode"), *text/csv* (dades en format CSV), *image/gif* (imatge GIF), *image/jpeg* (imatge JPG), *image/png* (imatge PNG), *application/pdf* (document PDF), *application/gzip* (dades comprimides de tipus Gzip) o *application/octet-stream* (dades sense estructura reconeguda), entre d'altres. Per indicar en conjunt tots els subtipus d'un tipus donat, es pot fer servir el símbol especial "*", així, per exemple: image/*. Igualment, per indicar tots els tipus MIME possibles, podeu escriure */*.

- **Content-Type:** Només apareix en peticions POST. Informa al servidor del tipus MIME de les dades enviades al cos de la petició. Per exemple, si aquestes estan en format "URL-encode" el valor serà: *Content-Type: application/x-www-form-urlencoded*. Ve acompanyada d'una altra capçalera anomenada **Content-Length:**, la qual serveix per informar al servidor de la mida (en bytes) del cos de la petició
- **Date:** Informa al servidor de la data i hora en què la petició va ser enviada. El seu valor s'ha d'expressar en un format definit al document RFC 1123
- **Referer:** Informa al servidor de l'URL del recurs que es va sol·licitar just abans del recurs actual. A la pràctica, el seu valor sol ser l'adreça de la pàgina web que contenia lenllaç que ha portat a la petició vigent cap a la pàgina actual.
- **User-Agent:** Informa al servidor sobre quin programa concret (i la seva versió) és el client que realitza la petició. Per exemple, *User-Agent: Mozilla/5.0 (X11; Linux x86_64;*

rv:32.0) Firefox/32.0 . Aquesta línia de capçalera ens pot anar bé per simular ser un determinat navegador i comprovar el comportament del servidor segons aquesta dada.

Respostes HTTP

Tota resposta HTTP enviada per un servidor a un client (corresponent a la petició feta anteriorment per aquest) té la següent estructura:

- Una primera línia amb el format **<VERSIÓ> <CODI> <FRASE>**
- Següents línies anomenades de "capçalera de resposta" amb el format **NomCapçalera:Valor** (case-insensitive)
- Una línia en blanc
- Només en peticions de tipus GET: següents línies, amb format lliure, anomenades "cos de la resposta"

La primera línia defineix el tipus de resposta oferta, la capçalera de resposta aporta informació sobre detalls més tècnis de la resposta i el cos de la resposta aporta el contingut pròpiament dit d'aquesta (per exemple, en el cas d'haver demanat el client una pàgina web, el cos de la resposta contindria el codi HTML d'aquesta, el qual un navegador interpretaria i renderitzaria visualment a pantalla).

Codis de resposta

El camp <*VERSIÓ*> de la primera línia simplement indica la versió del protocol utilitzada per fer la resposta, i sempre tindrà com a valor o bé la cadena "HTTP/1.1" o bé "HTTP/2".

El camp <*CODI*> és un nombre de tres dígits que defineix quin tipus de resposta s'està proporcionant. El camp <*FRASE*> conté una breu explicació textual del valor de <CODI>

Els valors possibles del camp <CODI> es poden classificar en cinc categories, depenent del resultat obtingut al servidor després del processament de la petició. Aquestes categories són identificades pel dígit de més a l'esquerra, servint els altres dos per especificar més al detall el tipus de resposta concreta. A continuació presentem alguns dels codis de resposta comuns:

- **1xx** *(Informació)*: El servidor indica que la petició ha estat rebuda i que la processarà. Es fa servir en casos molt específics que no veurem.
- **2xx** *(Èxit)*: El servidor indica que la petició ha estat rebuda i processada amb èxit. El codi més típic d'aquesta categoria és **200**, obtingut quan la consulta ha estat satisfactòria i el recurs sol·licitat és tornat correctament (suposant que el mètode de la petició sigui GET; si el mètode és POST el codi llavors és 201).

- **3xx** *(Redirecció)*: El servidor indica que la petició no s'ha completat i s'han de fer més accions per aconseguir finalitzar-la. Generalment es fa servir com a resposta a peticions GET per assenyalar un canvi de localització del recurs sol·licitat (és a dir, una nova URL d'una pàgina ja existent). Aquesta nova localització s'indica a una capçalera enviada pel servidor just per a aquest propòsit: **Location:**. Si el canvi és permanent, s'utilitza el codi **301** i per a peticions següents el client ha de fer servir la ubicació proporcionada a Location; si el canvi és temporal hi ha altres codis (**302, 303, 307…**) i les peticions següents han de continuar fent-se a la URL original. Normalment aquests codis són interpretats automàticament per qualsevol navegador actual, realitzant lacció pertinent sense necessitat de notificar-ho a lusuari.
- **4xx** *(Error de client)*: El servidor indica que la petició procedent del client està mal formada o conté errors. El codi tornat més comú en aquests casos és el **404**, que indica que el servidor no ha trobat el recurs sol·licitat a la URL de la petició. Un altre força comú és el codi **401**, el qual indica que l'usuari no ha presentat les credencials adequades per poder estar autoritzat a accedir a un recurs protegit.
- **5xx** *(Error de servidor)*: El servidor indica que, encara que la petició és correcta, ha fallat en processar-la. Els diferents motius s'assenyalen amb un codi concret. Per exemple, **500** avisa d'un error indeterminat o **501** indica que el servidor no suporta el mètode de petició, entre d'altres.

Capçaleres de les respostes

Les línies de la capçalera de la resposta serveixen per informar el client sobre detalls més tècnics de la resposta, permetent que aquest pugui obtenir una informació més completa sobre el recurs en qüestió. A continuació es llista una breu selecció:

- **Content-Type:** Informa al client, en respostes a peticions GET, sobre quin és el tipus MIME del contingut del cos del recurs retornat. També informa (però només si el tipus MIME és textual, com text/plain o text/html) del sistema de codificació (ASCII, UTF-8, etc), per a què el client pugui mostrar aquest contingut correctament. Això darrer s'indica mitjançant la paraula especial "charset" després d'un punt i coma, així: *Content-Type: text/html; charset=UTF-8*
- **Connection:** Informa al client sobre si la connexió TCP creada pel servidor per enviar la resposta HTTP continuarà oberta o no. Si sí (valor *keep-alive*), aquesta connexió TCP podrà ser reutilitzada per rebre més peticions HTTP posteriors del mateix client; si no (valor *close*), serà tancada i per enviar una nova resposta HTTP el servidor haurà de crear una nova connexió.

- **Content-Length:** Informa al client de la mida (en bytes) del contingut del cos del recurs retornat (una vegada comprimit, si és el cas). Si el valor de la línia Connection és *close*, el valor d'aquesta línia és fàcilment calculable i es pot indicar directament, així: *Content-Length:348* però si Connection és *keep-alive* (per exemple quan s'ofereix un fitxer a "streaming"), el servidor no sap per endavant aquesta dada, per la qual cosa aquesta línia de capçalera sol ser substituïda per la línia **Transfer-Encoding: chunked** , la qual indica al client que el recurs és enviat "a trossos" progressivament fins a enviar una marca final.
- **Date:** Informa al client de la data i hora en què la resposta va ser generada. El seu valor s'ha d'expressar en un format definit al document RFC 1123
- **Expires:** Informa el client de la data i hora en què la resposta donada es considerarà caducada. El seu valor s'ha d'expressar en un format definit al document RFC 1123.
- **Location:** Indica al client la nova URL d'un recurs quan la URL sol·licitada (corresponent a aquest recurs) ja no és vàlida.
- **Server:** Informa el client del nom i sistema del servidor. Per exemple, *Server:Apache/2.4 (Unix)*