Peticions REST

Una petició REST a la pràctica simplement és una petició HTTP/S (per tant, client-servidor) que compleix dos requisits:

*Es fa a una URL amb aquest format: *https://nomServidor/cami/fins/el/recurs*. Això és perquè la URL ha d'identificar un determinat recurs proporcionat pel servidor de forma unívoca (òbviament, aquesta URL ha de ser reconeguda pel servidor web a qui va dirigida). De forma opcional al final de la URL es pot afegir una "querystring" per modificar la consulta d'alguna manera (ordenar, filtrar, paginar...).

NOTA: Per "recurs" s'entén qualsevol cosa (no ha de ser una pàgina web necessàriament); de fet, sol representar un determinat element dins d'una col.lecció (un usuari, una mascota...) del qual es vol obtenir / afegir / esborrar / modificar certa informació.

*Es fa mitjançant algun mètode HTTP (GET, POST, PUT, DELETE, principalment) que serveix per indicar l'acció concreta que el servidor ha de realitzar sobre el recurs identificat a la URL. El codi HTTP de resposta retornat pel servidor servirà per saber si aquesta acció s'ha realitzar correctament o no (i si no, per què)

NOTA: En basar-se en HTTP, les peticions REST són igualment catxejables: si la resposta així ho indica, el client emmagatzemarà dita resposta per no haver-la de demanar en un futur proper de nou.

A una petició REST reconeguda per un servidor, aquest pot reaccionar o bé retornant al client una determinada informació associada al recurs indicat (obtinguda d'una font qualsevol, com ara una base de dades, un fitxer, etc i oferida en un format que no té perquè ser HTML, com ara JSON, XML, YAML, etc) o bé afegint-la/modificant-la/esborrant-la d'aquesta font, la qual pot ser al servidor mateix o a un altre servidor extern qualsevol). En el primer cas el mètode HTTP emprat per la petició serà GET; en el segon cas serà POST/PUT/DELETE, respectivament. En qualsevol cas, cada petició és completament independent de les altres (és a dir, no guarda cap "estat").

NOTA: És responsabilitat del client indicar el tipus MIME del recurs desitjat ("text/html", "text/xml", "application/json", etc) per a què el servidor sàpiga quina representació del recurs (si és que la té disponible) és la que donarà com a resposta.

Més en concret, les operacions "CRUD" (de "Create","Read", "Update","Delete") que es poden fer en una petició REST són:

Operació CRUD	Mètode HTTP	Codi resposta	Cos resposta (si no hi ha error)
Obtenir un recurs o un conjunt d'ells	GET	200 o el de l'error ocorregut	El recurs (o recursos) demanats
Crear un nou recurs	POST	201 o el de l'error ocorregut	És recomanable retornar el recurs creat
Actualitzar un recurs o part d'ell	PUT/PATCH	200 o el de l'error ocorregut	És recomanable retornar el recurs modificat
Eliminar un recurs	DELETE	204 o el de l'error ocorregut	Cap

NOTA: En realitat, el mètode POST també es podria fer servir per modificar un recurs ja creat i el mètode PUT es pot fer servir per crear-ne un de nou. L'elecció del mètode POST o PUT dependrà de quins té implementats el servidor en qüestió. Si el servidor ho pot implementar, el mètode PUT és preferible per realitzar les modificacions degut a la seva idempotència (és a dir, degut a què múltiples repeticions seguides de la mateixa petició no donen resultats diferents).

NOTA: El mètode PATCH és similar a PUT però permet modificar part d'un recurs (PUT -i POST, si es fa servir- sempre sobreescriuen el recurs sencer)

Per exemple, una petició GET a http://www.botigamascotes.com/gossos/124 el servidor la podria entendre com una petició REST demanant informació sobre el "recurs" REST identificat com "/gossos/124" (és a dir, sobre la mascota de categoria "gos" amb codi nº 124). En canvi una petició POST a http://www.botigamascotes.com/gossos/124 s'encarregaria de crear al servidor un nou recurs "gos" identificat

pel seu codi (les dades que defineixin aquest recurs "viatjarien" dins del cos de la petició, sota les capçaleres de client). Una petició PUT a http://www.botigamascotes.com/gossos/124 modificaria la informació del gos indicat (la nova informació "viatjaria" també dins del cos de la petició, sota les capçaleres de client). Una petició DELETE a http://www.botigamascotes.com/gossos/124 esborraria l'element indicat del servidor.

Fixar-se que la convenció REST permet establir el protocol HTTP com la "base de transport" per sobre la qual s'estableix un tipus de comunicació que no té res a veure amb la "navegació web" clàssica perquè és molt més genèrica i versàtil. Es tracta, en realitat, d'utilitzar un mecanisme estàndard de petició-resposta per gestionar qualsevol tipus de recurs compartit per un servidor. De fet, avui dia la majoria de peticions REST que es realitzen a través d'"apps" mòbils que no són pas navegadors.

Els noms i jerarquia dels recursos (i el format del contingut existent al cos de les peticions POST i PUT) no estan definits: el servidor pot implementar els que desitgi i com ho desitgi; haurà de ser el client qui hagi de procurar realitzar les peticions convenientment a aquesta implementació. No obstant, sí que hi ha certes "convencions" comunament establertes a totes les implementacions REST actuals:

*Les rutes han de ser plurals (és a dir, per obtenir un usuari amb id=5 cal usar */users/5* i no */user/5*)
*Els finals de les rutes ("endpoints") no han de mostrar cap extensió de fitxer (tot i que la resposta esperada en pugui ser un)
*Els "endpoints" haurien de ser noms, no verbs: les accions són conegudes pel mètode HTTP emprat
*Les rutes haurien d'escriure's en minúscules i guions (en llocs de subratllats)
*L'API hauria de poder gestionar múltiple mètodes HTTP diferents per la mateixa URL

NOTA: Existeixen moltes tecnologies del costat del servidor que permeten desenvolupar un servidor REST: exemples són Tomcat i Spring (si usem llenguatge Java), Laravel (si usem llenguatge PHP), Flask o Django (si usem Python), entre molts d'altres.

A continuació es mostren exemples de diferents tipus de peticions REST fetes amb l'eina *curl*, (suposant que el servidor -fictici- contra el que s'envien té implementada l'API corresponent):

*Obtenir informació sobre una pel.lícula: *curl https://miservidor.com/pelis/superman*

*Obtenir informació sobre totes les pel.lícules: *curl https://miservidor.com/pelis*

*Afegir una nova pel.lícula o modificar (totes) les seves dades (de forma no idempotent):
curl -X POST -H "Content-Type:application/json"
-d '{"Titol":"Superman","Director":"Anònim","Any":1979}' https://miservidor.com/pelis/superman

*Afegir una nova pel.lícula o modificar (totes) les seves dades (de forma idempotent):
curl -X PUT -H "Content-Type:application/json"
-d '{"Titol":"Superman","Director":"Anònim","Any":1979}' https://miservidor.com/pelis/superman

*Esborrar una pel.lícula: *curl -X DELETE https://miservidor.com/pelis/superman*

NOTA: Es pot saber més sobre REST APIs anant a http://restcookbook.com o https://www.restapitutorial.com