



REST - Le nommage

[Mise à jour le 11/11/2021]



- **Source** : Mooc Fun "Programmer l'internet des objets"
- **Vidéo** sur YouTube: [Web et REST](#)

1. Le modèle client-serveur du Web

Le Web et ses extensions sont basés sur un **modèle client-serveur**. Les serveurs possèdent des ressources et les clients peuvent y accéder ou les modifier grâce à un protocole tel que HTTP. Le **modèle client-serveur** est quelque chose de courant dans les réseaux informatiques, mais le Web suit certaines directives de conception connues sous le nom de **REST** (**RE**presentational **S**tate **T**ransfer).



Selon Roy Fielding, qui a défini ce modèle, REST est un ensemble de principes, de propriétés et de contraintes. REST utilise le modèle de communication client-serveur et utilise généralement le protocole HTTP (Hypertext Transfer Protocol).

Les **6 propriétés** qui définissent REST :

- **architecture client-serveur** : les **serveurs** sont les entités qui **fournissent des services** tels que l'heure, le partage de données, etc. ; les clients sont les entités qui interrogent ces services.
- **sans état** (ou stateless) : cela signifie que l'état de l'interaction entre le client et le serveur n'est pas stocké. Chaque requête client est traitée comme une nouvelle demande indépendante. De cette façon, **le serveur n'a pas besoin de stocker, suivre ou analyser les requêtes précédentes du même client**.
- **cacheability** : les périphériques intermédiaires ainsi que les clients peuvent mettre en cache les réponses pour une utilisation ultérieure. Cela conduit à améliorer les performances.
- **système en couches** : cela signifie que les services peuvent être réalisés en utilisant des couches telles que différents serveurs responsables de différentes parties de services (stockage, recherche, etc.), mais le client ne sait pas s'il est connecté au serveur final ou intermédiaire.
- **code à la demande** (facultatif) : les serveurs peuvent éventuellement envoyer du code exécutable aux clients.
- **interface uniforme** : elle fait référence aux principes selon lesquelles une ressource doit avoir une seule représentation telle que les **URI (Uniform Resource Identifier)**, et suivre certaines directives pour la dénomination, le format de lien et de données, etc. Il doit avoir suffisamment d'informations sur le traitement. Les ressources initiales telles que la page d'accueil d'un site Web, devraient avoir d'autres liens leur permettant de découvrir d'autres ressources.



Le respect de ces règles permet l'évolutivité du système en ajoutant continuellement des acteurs et de nouvelles données. L'objectif principal est de simplifier le comportement du serveur afin de servir le plus grand nombre de requêtes possible.

2. Le nommage

Chaque ressource du Web est identifiée par une valeur unique appelée URI (Uniform Resource Identifier). Si l'URI contient des caractères internationaux, (comme les lettres accentuées, ...) il est appelé IRI (International Resource Identifier).

Les URI permettent de désigner une ressource de manière non ambiguë, c'est-à-dire que l'on ne retrouvera pas le même URI pour désigner deux ressources différentes. Par construction, la structure de l'URI est hiérarchique, ce qui permet de créer des identifiants uniques de manière distribuée. Si vous voulez identifier une ressource, vous devez posséder une séquence unique : un numéro de téléphone, un numéro de sécurité sociale, un nom de domaine. En y ajoutant quelque chose d'unique pour nous, cela crée un identifiant globalement unique.

Par exemple, on a une image que l'on veut identifier, on peut l'appeler

"image"

mais il y a peu de chance que ce nom soit unique, d'autres personnes sur Terre ont sûrement eu la même idée que nous. En revanche, si je la fais précéder de mon numéro de téléphone, on aura :

33667789078image

qui sera unique si je ne nomme qu'une seule ressource "image". Un autre utilisateur sur le même principe pourra nommer sa ressource :

33667239018image

sans ambiguïté possible. Cependant, comme le numéro de téléphone est unique dans l'espace des numéros de téléphone, d'autres numéros uniques pourraient entrer en conflit dans d'autres espaces de numérotation.

Pour éviter les conflits, il est intéressant de donner, au début l'espace de numérotation, par exemple :

tel:33667789078image

et

ss:33667789078image

On aura donc deux identifiants uniques, même si le hasard a fait que ce numéro de téléphone et ce numéro de sécurité sociale coïncident.

Les URI formalisent ce principe. Le [RFC 3986] explique comment ils peuvent être construits.

Un URI commence par un schéma indiquant l'autorité de nommage, suivi d'une valeur d'autorité puis d'un chemin dans l'espace d'autorité. Des caractères comme les ":" ou les "/" sont utilisés pour améliorer la lisibilité de l'URI.

Par exemple :

`mailto:mduerst@ifi.unizh.ch` `ssh:utilisateur@example.com` `ftp://ftp.is.co.za/rfc/rfc1808.txt` Ainsi, si je mets une ressource sur mon site Web, celui-ci est identifié par un nom de domaine, par exemple `example.com`. Je suis propriétaire de ce nom. Je peux donc l'utiliser pour identifier de manière unique ma ressource. Si on reprend le principe de construction d'un URI, j'aurai : `http://example.com/ma_ressource` Personne d'autre dans l'univers ne pourra identifier ses ressources avec cette chaîne de caractères puisque `example.com` m'appartient. Je dispose donc d'un espace de nommage infini sous `example.com` qui me permet de désigner l'ensemble infini de ressources sans que personne d'autre ne puisse prendre les mêmes noms. Un URI est une construction administrative permettant d'attribuer un identifiant unique global à une ressource spécifique. L'URI a pour but de facilement nommer une ressource, de pouvoir lier les ressources entre elles pour former cette toile d'araignée mondiale. Le schéma définit à la fois l'espace de nommage de l'autorité et son format. Une adresse IP ou un nom de domaine comme autorité est à la fois un moyen d'assurer l'unicité globale, mais également de savoir comment accéder à la ressource. Un sous-ensemble d'URI peut être directement utilisé pour localiser la ressource, c'est-à-dire trouver sur quel serveur se trouve la ressource et comment y accéder. Il s'agit d'une URL (Uniform Resource Locator) bien connue utilisée par les navigateurs Web. Le schéma `http` est bien pratique car il peut se lire également comme un URL. Ce schéma donne : le protocole à utiliser pour accéder à la ressource (`http`), l'autorité qui indique l'adresse du serveur (et son port), et enfin, le chemin d'accès de ce que l'on va demander au serveur et qui peut parfois correspondre à une arborescence de fichiers sur un serveur.

From:

<https://webge.fr/dokuwiki/> - **WEBGE Wikis**

Permanent link:

<https://webge.fr/dokuwiki/doku.php?id=reseaux:internet:rest&rev=1636621430>

Last update: **2021/11/11 10:03**

