API REST

DÉFINITION

C'est l'un des standards de création d'API les plus logiques, efficaces, et utilisés. Plus de 80 % des API sont des API REST.

REST signifie ***Re****presentational* ***S****tate* ***T****ransfer* (ou *transfert d'état de représentation*, en français), et constitue un ensemble de normes, ou de lignes directrices architecturales qui structurent la façon de communiquer les données entre votre application et le reste du monde, ou entre différents composants de votre application.

Les API RESTful se basent sur le protocole HTTP pour transférer les informations – le même protocole sur lequel la communication web est fondée ! Donc, lorsque vous voyez http au début d'une URL, votre navigateur utilise HTTP pour faire une requête de ce site web au serveur. REST fonctionne de la même façon.

AVANTAGES DE REST

Client-serveur séparation

L'une des normes de REST est la séparation du client et du serveur.
Un client est celui qui va utiliser l'API. Cela peut être une application, un navigateur ou un logiciel.
Un serveur est un ordinateur distant capable de récupérer des données depuis la base de données, de les manipuler si besoin et de les renvoyer à l'API.
De façon générale, il existe une séparation entre le client et le serveur. Cette séparation permet au client de s'occuper uniquement de la récupération et de l'affichage de l'information et permet au serveur de se concentrer sur le stockage et la manipulation des données.
Les API REST offrent un moyen de communication standardisé entre le client et les données. Peu importe comment le serveur est construit ou comment le client est codé, du moment qu'ils structurent tous les deux leur communication selon les lignes directrices architecturales REST, en utilisant le protocole HTTP, ils pourront communiquer entre eux.

Stateless

Un des autre aspect des API REST est qu'elles sont stateless – *sans état*, en français – ce qui signifie que le serveur ne sauvegarde aucune des requêtes ou réponses précédentes.

Étant donné que chaque message est isolé et indépendant du reste, il vous faudra vous assurer d'envoyer avec la requête que vous formulez toutes les données nécessaires pour être sûr d'avoir la réponse la plus précise possible.

Le fait d'être stateless rend chaque requête et chaque réponse très déterminée et compréhensible. Donc, si vous êtes développeur et que vous voyez la requête API de quelqu'un d'autre dans un code déjà existant, vous serez capable de comprendre l'objet de la requête sans contexte.

Cacheable (ou *sauvegardable*, en français)

La réponse doit contenir l'information sur la capacité ou non du client de mettre les données en cache, ou de les sauvegarder. Si les données peuvent être mises en cache, la réponse doit être accompagnée d'un numéro de version. Ainsi, si votre utilisateur formule deux fois la même requête (c'est-à-dire s'il veut revoir une page) et que les informations n'ont pas changé, alors votre serveur n'a pas besoin de rechercher les informations une deuxième fois. À la place, le client peut simplement mettre en cache les données la première fois, puis charger à nouveau les mêmes données la seconde fois.

Uniforme Interface (interface uniforme)

Lors de la création d'une API REST, les développeurs acceptent d'utiliser les mêmes normes. Ainsi, chaque API a une interface uniforme. L'interface constitue un contrat entre le client et le service, que partagent toutes les API REST. C'est utile, car lorsque les développeurs utilisent des API, cela leur permet d'être sûrs qu'ils se comprennent entre eux.

Une API REST d'une application peut communiquer *de la même façon* avec une autre application entièrement différente.

Layered system (système de couches)

Chaque composant qui utilise REST n'a pas accès aux composants au-delà du composant précis avec lequel il interagit.

Cela signifie qu'un client qui se connecte à un composant intermédiaire n'a aucune idée de ce avec quoi ce composant interagit ensuite. Par

exemple, si vous faites une requête à l'API Facebook pour récupérer les derniers posts : vous n'avez aucune idée des composants avec lesquels l'API Facebook communique.

Cela encourage les développeurs à créer des composants indépendants, facilitant le remplacement ou la mise à jour de chacun d'entre eux.

ALTERNATIVE AUX API REST

REST n'est qu'un type d'API. Il existe des alternatives qui vous seront également utiles à connaître, notamment les API SOAP.

SOAP est l'acronyme de *Simple Object Access Protocol*, ou *protocole simple d'accès aux objets*, en français. Contrairement à REST, il est considéré comme un protocole, et non comme un style d'architecture.

Les API SOAP étaient les API les plus courantes avant l'arrivée de REST. REST utilise le protocole HTTP pour communiquer, SOAP d'un autre côté peut utiliser de multiples moyens de communication. Le souci, c'est la complexité qui en ressort, car les développeurs doivent se coordonner pour s'assurer qu'ils communiquent de la même manière afin d'éviter les problèmes. De plus, le SOAP peut demander plus de bande passante, ce qui entraîne des temps de chargement beaucoup plus longs. REST a été créé pour résoudre certains de ces problèmes grâce à sa nature plus légère et plus flexible.

De nos jours, le SOAP est plus fréquemment utilisé dans les applications de grandes entreprises, puisqu'on peut y ajouter des couches de sécurité, de confidentialité des données, et d'intégrité supplémentaires. REST peut être tout aussi sécurisé, mais a besoin d'être implémenté, c'est-à-dire d'être développé au lieu d'être juste intégré comme avec le SOAP.

EN RÉSUMÉ

- Toutes les API ne sont pas REST et les API REST ont des lignes directrices architecturales spécifiques.
- Les avantages clés des API REST sont les suivants :
    - la séparation du client et du serveur, qui aide à scaler plus facilement les applications ;
    - le fait d'être stateless, ce qui rend les requêtes API très spécifiques et orientées vers le détail ;

    - la possibilité de mise en cache, qui permet aux clients de sauvegarder les données, et donc de ne pas devoir constamment faire des requêtes aux serveurs.
- SOAP est un autre type d’API, mais est plus utilisé dans les grandes entreprises.