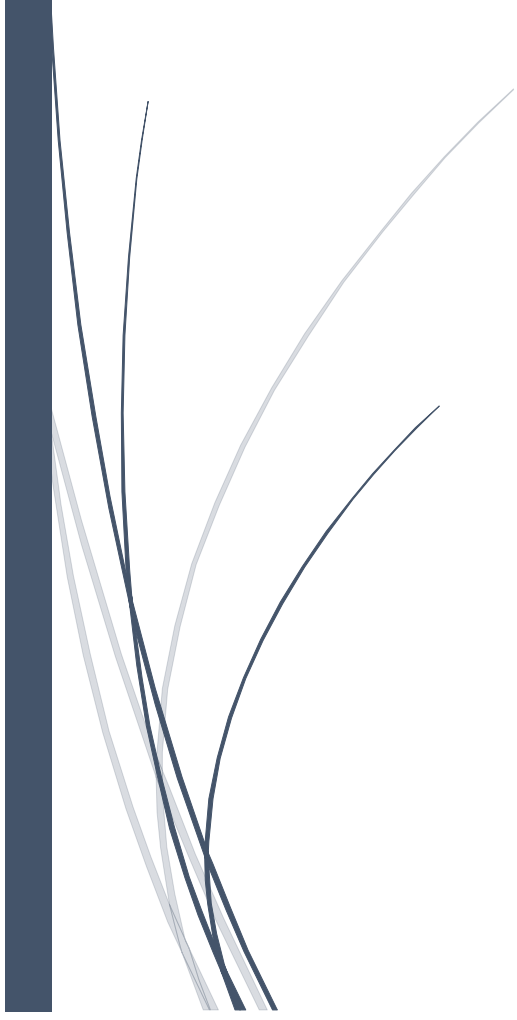
07/01/2025

REST API



HP

MICROSOFT

Table des matières

II. Le protocole HTTP

1.1. Définition

L'**http** est l'abréviation de **Hypertext Transfer Protocol**, on le traduit littéralement en français par **protocole de transfert hypertexte**.

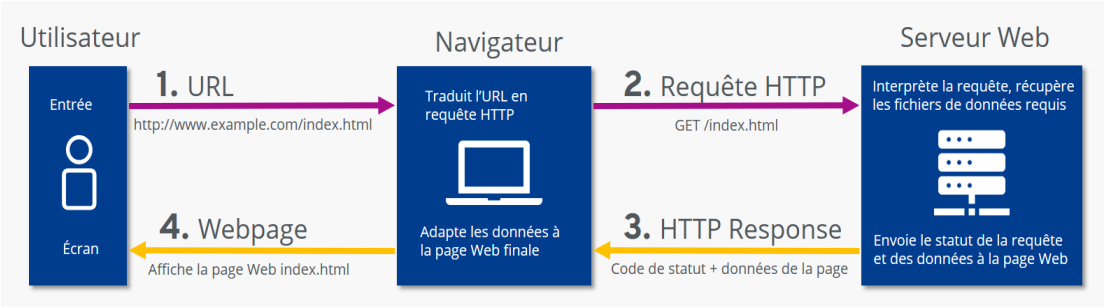
Http est un protocole de communication permettant de récupérer et d'envoyer des ressources hypermédia.

Au départ, HTTP a été créé pour permettre de transfert de documents HTML uniquement puis ses usages se sont rapidement étendus pour permettre de transférer d'autres types de ressources comme des images, des vidéos ou d'autres types de fichiers (XML, JSON, …).

1.2. Le fonctionnement de http

Le fonctionnement du HTTP peut être expliqué très simplement à travers la consultation d'un site Internet :

1. L'utilisateur saisit dans la barre d'adresse de son navigateur Internet ***example.com***.
2. Le navigateur envoie une requête correspondante, appelée **requête HTTP**, au serveur Web qui administre le domaine *example.com*. Normalement, cette requête est de type : « Merci de m'envoyer le fichier ».
3. Le serveur web traite la requête reçue et envoie une réponse au navigateur.
4. Le navigateur reçoit le fichier et l'affiche sous forme de site Internet.



1.3. Les requêtes http

Une requête HTTP est un ensemble de lignes envoyé au serveur par le navigateur. Elle comprend :

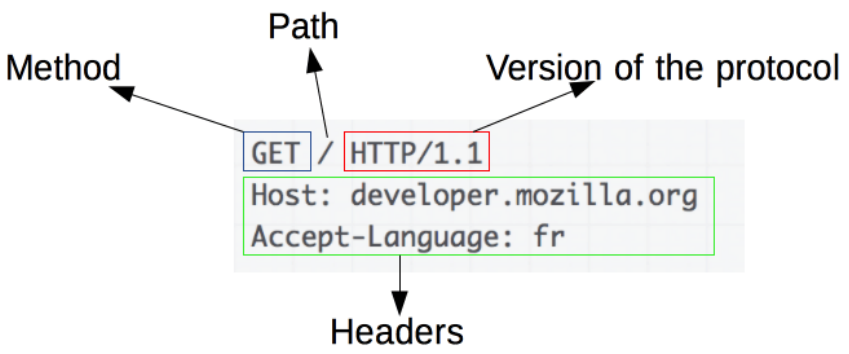
1. **Une ligne de requête :** La première ligne d'une requête HTTP permet de préciser la requête HTTP. Celle-ci est composée de :
   - La méthode de requête utilisée qui sert à indiquer le type de requête effectuée : simple récupération de ressources, envoi de données sur le serveur, etc. ;

- La cible de la requête (si applicable) qui va généralement prendre la forme d'une URL ou d'un chemin absolu ;
- La version HTTP utilisée pour la requête (qui sert également à indiquer la version attendue pour la réponse).

2. **Les champs d'en-tête de la requête :** il s'agit d'un ensemble de lignes facultatives permettant de donner des informations supplémentaires sur la requête et/ou le client (Navigateur, système d'exploitation, ...). Chacune de ces lignes est composée d'un nom qualifiant le type d'en tête, suivi de deux points (:) et de la valeur de l'en-tête

3. **Le corps de la requête :** c'est un ensemble de lignes optionnelles devant être séparées des lignes précédentes par une ligne vide et permettant par exemple un envoi de données par une commande POST lors de l'envoi de données au serveur par un formulaire

Voici un exemple de requête http :



1.3.1. Les méthode d'une requête http :

Il existe 9 méthodes distinctes parmi lesquelles il faut choisir lorsque l'on décide d'envoyer une requête HTTP à un serveur.

Voici la description des méthodes http les plus utilises :

a. GET

La méthode de requête HTTP la plus fréquemment utilisée.

Une requête GET demande au serveur un élément d'information ou une ressource spécifique. Lorsque vous vous connectez à un site web, votre navigateur envoie généralement plusieurs requêtes GET pour recevoir les données dont il a besoin pour le chargement de la page.

b. POST

Votre navigateur utilise la méthode de requête POST lorsqu'il doit envoyer des données au serveur. Par exemple, si vous remplissez un formulaire de contact sur un site web et que vous l'envoyez, vous utilisez une requête POST pour que le serveur reçoive ces informations.

c. PUT

Les requêtes PUT ont une fonctionnalité similaire à celle de la méthode POST. Toutefois, au lieu de soumettre des données, vous utilisez des demandes PUT pour mettre à jour des informations qui existent déjà sur le serveur final.

d. DELETE

Même chose que le PUT, mais pour supprimer une entité, et sans corps dans la requête (toutes les informations doivent être passées dans les en-têtes ou l'URI).

1.4. Les réponses http

Une réponse HTTP est un ensemble de lignes envoyées au navigateur par le serveur. Elle comprend :

- **Une ligne de statut** : c'est une ligne précisant la version du protocole utilisé et l'état du traitement de la requête à l'aide d'un code et d'un texte explicatif. La ligne comprend trois éléments devant être séparés par un espace :
- **La version du protocole utilisé**
- **Le code de statut**
- **La signification du code**
- **Les champs d'en-tête de la réponse** : il s'agit d'un ensemble de lignes facultatives permettant de donner des informations supplémentaires sur la réponse et/ou le serveur. Chacune de ces lignes est composée d'un nom qualifiant le type d'en-tête, suivi de deux points (:) et de la valeur de l'en-tête
- **Le corps de la réponse** : il contient le document demandé

Voici un exemple de réponse http :

```
HTTP/1.0 200 OK
Date : Sat, 15 Jan 2000 14:37:12 GMT Server : Microsoft-IIS/2.0 Content-Type :
text/HTML
Content-Length : 1245 Last-Modified : Fri, 14 Jan 2000 08:25:13 GMT
```

Les statuts d'une réponse http :

Code	Message	Description
10x	**Message d'information**	**Ces codes ne sont pas utilisés dans la version 1.0 du protocole**
20x	**Réussite**	**Ces codes indiquent le bon déroulement de la transaction**
200	OK	La requête a été accomplie correctement
201	CREATED	Elle suit une commande POST, elle indique la réussite, le corps du reste du document est sensé indiquer l'URL à laquelle le document nouvellement créé devrait se trouver.
202	ACCEPTED	La requête a été acceptée, mais la procédure qui suit n'a pas été accomplie
203	PARTIAL INFORMATION	Lorsque ce code est reçu en réponse à une commande GET, cela indique que la réponse n'est pas complète.
204	NO RESPONSE	Le serveur a reçu la requête mais il n'y a pas d'information à renvoyer
205	RESET CONTENT	Le serveur indique au navigateur de supprimer le contenu des champs d'un formulaire
30x	**Redirection**	**Ces codes indiquent que la ressource n'est plus à l'emplacement indiqué**
301	MOVED	Les données demandées ont été transférées à une nouvelle adresse
40x	**Erreur due au client**	**Ces codes indiquent que la requête est incorrecte**
400	BAD REQUEST	La syntaxe de la requête est mal formulée ou est impossible à satisfaire
402	PAYMENT REQUIRED	Le client doit reformuler sa demande avec les bonnes données de paiement
403	FORBIDDEN	L'accès à la ressource est tout simplement interdit
404	NOT FOUND	Classique ! Le serveur n'a rien trouvé à l'adresse spécifiée. Parti sans laisser d'adresse... :)
50x	**Erreur due au serveur**	**Ces codes indiquent qu'il y a eu une erreur interne du serveur**
500	INTERNAL ERROR	Le serveur a rencontré une condition inattendue qui l'a empêché de donner suite à la demande (comme quoi il leur en arrive des trucs aux serveurs...)
501	NOT IMPLEMENTED	Le serveur ne supporte pas le service demandé (on ne peut pas tout savoir faire...)
502	BAD GATEWAY	Le serveur a reçu une réponse invalide de la part du serveur auquel il essayait d'accéder en agissant comme une passerelle ou un proxy
503	SERVICE UNAVAILABLE	Le serveur ne peut pas vous répondre à l'instant présent, car le trafic est trop dense (toutes les lignes de votre correspondant sont occupées veuillez rappeler ultérieurement)
504	GATEWAY TIMEOUT	La réponse du serveur a été trop longue vis-à-vis du temps pendant lequel la passerelle était préparée à l'attendre (le temps qui vous était imparti est maintenant écoulé...)

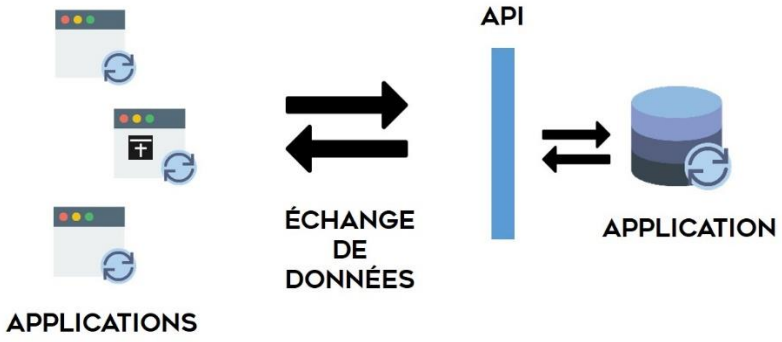
III. API REST

3.1. Définition de l'API

Une **Application Programming Interface (API)** est une Interface Applicative de Programmation qui permet d'établir des connexions entre plusieurs logiciels pour échanger des données. Plus techniquement, il s'agit d'un ensemble de fonctions qui vont permettre à un développeur d'utiliser simplement une application dans son programme.

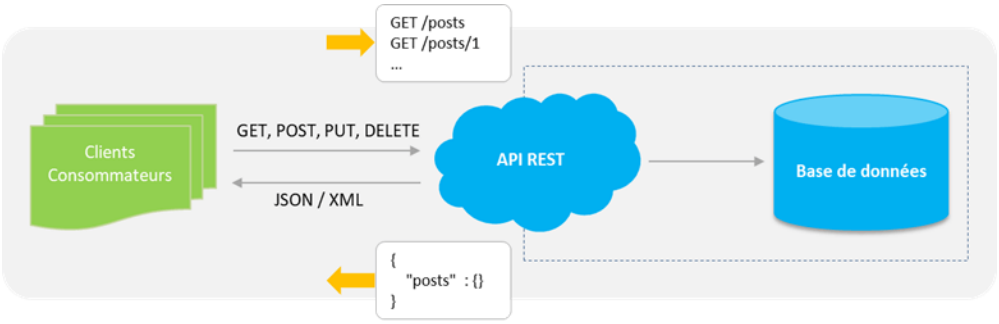
Une API est considérée comme un contrat entre un fournisseur d'informations et un utilisateur d'informations, qui permet de définir le contenu demandé au consommateur (l'appel) et le contenu demandé au producteur (la réponse).

Par exemple, l'API conçue pour un service de météo peut demander à l'utilisateur de fournir un code postal et au producteur de renvoyer une réponse en deux parties : la première concernant la température maximale et la seconde la température minimale.



3.2. Définition API REST

Une API RESTful est un style architectural pour une interface de programme d'application (API) qui utilise des requêtes HTTP pour accéder à des données et les utiliser. Ces données peuvent être utilisées pour les types de données GET, PUT, POST et DELETE, qui font référence à la lecture, la mise à jour, la création et la suppression d'opérations concernant des ressources.



3.3. Les règles de l'API Rest

a. Règle n°1 : l'URI comme identifiant des ressources

REST se base sur les **URI (Uniform Resource Identifier)** afin d'identifier une ressource. Ainsi une application se doit de construire ses URI (et donc ses URL) de manière précise, en tenant compte des contraintes REST.

Par exemple :

http://www.ismo.com/livre	permet de récupérer la liste des livres
http://www.ismo.com/livre?filtre=javascript&tri=asc	Récupère les livres sur JavaScript trie par ordre croissant
http://www.ismo.com/livre/34	Récupère le livre dont ID égale à 34
http://www.ismo.com/livre/34/comments	Récupère les commentaires du livre 34

b. Règle n°2 : les méthodes HTTP comme identifiant des opérations

La seconde règle d'une architecture REST est d'utiliser les verbes HTTP existants plutôt que d'inclure l'opération dans l'URI de la ressource. Ainsi, généralement pour une ressource, il y a 4 opérations possibles (CRUD) :

- Créer (create)
- Afficher (read)
- Mettre à jour (update)
- Supprimer (delete)

HTTP propose les verbes correspondant :

- Créer (create) => **POST**
- Afficher (read) => **GET**
- Mettre à jour (update) => **PUT**
- Supprimer (delete) => **DELETE**

Exemple d'URL pour une ressource donnée (un livre par exemple) :

POST - http://www.ismo.com/livre	Créer un livre

GET – http://www.ismo.com/livre/87	Récupère le livre dont l'ID égale à 87
PUT – http://www.ismo.com/livre/87	Mettre à jour le livre dont l'ID égale à 87
DELETE – http://www.ismo.com/livre/87	supprime le livre dont l'ID égale à 87

c. Règle n°3 : les réponses HTTP comme représentation des ressources

Il faut noter que la réponse envoyée n'est pas une ressource, c'est la représentation d'une ressource. Ainsi, une ressource peut avoir plusieurs représentations dans des formats divers : **HTML, XML, CSV, JSON, etc.**