

# Cartographie et OpenStreetMap: API REST

Institut Galilée - Master 2 P2S

---

Jones Magloire

6 Janvier 2025



# Qui suis-je ?

## CTO / DevOPS / Carto & Geocoding Expert

- Diplômé Master PLS 2015 (Mention Très Bien)
- Stage de 6 Mois chez takima
- Création du projet Jawg Maps 2015 (juillet)
- Création de la start-up Jawg Maps 2017
- Developpeur backend (Kotlin, NodeJS, Rust, Java)



# Qui suis-je ?

## Passionné de Développement et Photographie

- Site web: <https://joxit.dev>
- Contributions journalières sur Github @Joxit
  - Pelias Geocoder ([github.com/pelias](https://github.com/pelias))
  - Docker Registry UI ([github.com/Joxit/docker-registry-ui](https://github.com/Joxit/docker-registry-ui))
  - Vert.x ([github.com/eclipse-vertx](https://github.com/eclipse-vertx))
- Partage photos sur instagram @jox.it



① API REST/JSON

② Règles d'une API REST

# API REST/JSON

---

## Définition: API

- Une API (Application Programming Interface) désigne un contrat ou une spécification abstraite ou la standardisation d'un composant logiciel.
- Ce terme est utilisé pour le développement logiciel (on implémente une API) ainsi que pour la communication web (une API web qui expose des endpoints).
- Étant un contrat, plusieurs personnes peuvent l'implémenter de façon différente tant que le contrat est respecté.

## Définition: API REST

- Une API REST est une interface de programmation d'application qui respecte les principes de conception du style d'architecture REST, abréviation de « REpresentational State Transfer ».
- L'API ici est à la fois le contrat et son implémentation.

## Définition: JSON


Le JSON (JavaScript Object Notation) est un format d'échange de données basé sur du texte lisible pour stocker et transmettre des objets de données au format clé-valeur et de tableaux (ou d'autres valeurs sérialisables). Couramment utilisé en JavaScript et alternative au XML dans les APIs.



# Règles d'une API REST

---

# Règles d'une API REST

 *Il existe beaucoup de règles différentes, le choix de celles que nous suivons est subjectif. Je vais montrer uniquement celles qui me semblent les plus pertinentes.*

# Règles d'une API REST

## CRUD (Create, Read, Update, Delete)

Utiliser les verbes HTTP pour effectuer vos actions.

Méthode	Utilisation
GET	Récupérer la donnée
POST	Créer de la donnée
PUT	Mettre à jour de la donnée
PATCH	Mise à jour partielle de la donnée
DELETE	Supprimer de la donnée

# Règles d'une API REST

## Utiliser des noms au pluriel

Étant donné que vous utilisez les verbes HTTP, vous devez utiliser que des noms pour vos chemins. Pour POST, PUT et PATCH la donnée doit être mise dans le body de la requête.

Chemin	Description
GET /users	Récupérer tous les utilisateurs
POST /users	Créer un utilisateur
PUT /users/{id}/avatar	Mettre à jour l'avatar d'un utilisateur
PATCH /users/{id}	Mise à jour partiellement d'un utilisateur
DELETE /users/{id}	Supprimer un utilisateur

# Règles d'une API REST

## Utiliser les Status Code HTTP

En cas d'erreur de votre API ou d'erreur client, utilisez les status codes HTTP en plus de vos messages. Regardez la liste complète sur Wikipedia. Voici les grandes familles

Status Code	Description
100-199	Information, peu utilisée
200-299	Requête en succès (200 par défaut)
300-399	Redirection sur d'autres URL
400-499	Erreur venant du client
500-599	Erreur venant du serveur

# Règles d'une API REST

## Utiliser des query parameters

Vous aurez besoin de pagination, de filtres, de méta-données dans vos requêtes, tout se passe via les query parameters. Par exemple: `GET /users?limit=10` pour récupérer que 10 utilisateurs.

## Être idempotent

Vos requêtes doivent être de préférence idempotentes, c'est-à-dire qu'à chaque exécution, le résultat doit être le même. Effectuer deux `DELETE` du même élément à la suite doit renvoyer le même résultat.