Mapillary API Change Overview

This document outlines the rough shape of the “V4” Mapillary APIs with proposed properties, formats, etc.

tl:dr; differences between “V4” and V3

- All spatial APIs are moved to vector tiles from GeoJSON
- No predictable schema for image BLOB URLs (image URLs maybe change but retain image key). Do not depend on image key, there are key schema changes planned.
- All heavy metadata entities can be queried through /:entity/:key endpoints to get the metadata (object detections, for example). One exception: detections - can be queries only by image key to which they belong.
- We will support a specific set of tile zoom levels, yet to be determined
- All collection API endpoints are queried only by /:z/:x/:y, no bounding box or radius search
- Spatial data served as vector tiles endpoints only
- Spatial filtering happens on the client side within vector tile data, so filters cannot be sent to the server
- No paginated total collection endpoints
    - Use /:z/:x/:y vector tile endpoints instead for large amounts of data
    - No arbitrary bbox support as input parameter, client must convert to /:z/:x/:y
- The API enables access to the following entities
    - Users (entity only)
    - Organization (entity only)
    - Images (entity, spatial VT)
    - Sequences (entity, spatial VT)
    - Map features (entity, spatial VT)
    - Detections (aggregated on image key only)
- A VERY SMALL subset of APIs is likely to have spatial responses in the GeoJSON format /:z/:x:y (TBD)

Authorization

The API will support the OAuth 2.0 authorization code grant flow. There will be three scopes for the clients to support: read, write, upload. Most affected clients are ready only.

Scopes

- **read** - read all public data the user can read
- **write** - create/update data on behalf of the user
- **upload** - upload images (in a container format, TBD) on behalf of the user

Endpoints

Content-Type headers

The API supports two types of Content-Type headers:

- application/vnd.mapbox-vector-tile
- application/json

Access to image BLOBs

Image URLs can be accessed through metadata from /images/:key endpoint. We do not support a “predictable” URL schema anymore.

Tile style

Configuration, supported zoom level and schemas to follow (TBD).

Users

Properties

- key
- id
- username
- created_at

GET /users/:key

- **Content-Type: application/json**

Returns public metadata for the given user. This API should be used to get /me metadata with the appropriate key.

Images

Properties

- key
- created_at
- updated_at
- created_by (user)
- owned_by (organization)
- url

GET /images/:key

- **Content-Type: application/json**

GET /tiles/images/:z/:x/:y.mvt

- **Content-Type: application/vnd.mapbox-vector-tile**

Returns public metadata for the images in the tiles. Matches properties outlined in the Properties section.

Sequences

Properties

- key
- id
- creation_time
- update_time
- created_by (user)
- owned_by (organization)
- geometry (GeoJSON)
- length

GET /sequences/:key

- **Content-Type: application/json**

Returns public metadata for the given sequence. The sequence keys are found in the tiles for sequences.

GET /tiles/sequences/:z/:x/:y.mvt

- **Content-Type: application/vnd.mapbox-vector-tile**

Returns public metadata for the sequences in the tiles. Matches properties outlined in the Properties section.

Map features

We distinguish between two types of map features: PF (point feature) and TS (traffic sign).

Properties

- key
- type
- value
- created_at
- updated_at

- image_keys

GET /map_features/:type/:key

- **Content-Type: application/json**

GET /tiles/map_features/:type/:z/:x/:y

- **Content-Type: application/vnd.mapbox-vector-tile**

Returns public metadata for the map features in the tiles. Matches properties outlined in the Properties section.

Detections

Properties

- key
- id
- value
- geometry
- image

GET /images/:key/detections

- **Content-Type: application/json**
- **Return: Array<Detection>**

Vector tile schemas

Images

- key
- captured_at
- created_at
- created_by
- owned_by

Sequences

- key
- captured_at
- created_at
- created_by
- owned_by

Map features

- key
- type
- value
- created_at
- created_by
- image_keys

Q&A

Why you do not provide global collection endpoints with arbitrary filters anymore?

This scenario can be solved more efficiently and better through different means. Majority of the API clients are interested in small areas of the data and we optimized for that use case providing two types of APIs: vector tiles and /:entity/:key endpoints. Querying them in tandem solves the problem for majority of the clients when they filter the tiles as they get them. If you require more complex filters, we will provide a bulk download functionality which will return the data in ONE of the spatial formats (TBD). The download UI will allow to filter by: spatial bounds (REQUIRED), user, timestamps (capture, creation), values, types and keys.

What about detections?

In the current version of the API we only support detections aggregated on an image key. We might offer it in the future through a bulk download functionality in the web UI. Detections are not available as spatial data, but can be retrieved by a spatial query of the images vector tile, then as a second step retrieving all detections which are from an image key in the original query.

How can I filter metadata?

You can filter metadata within vector tiles on the client.

You have removed an endpoint X, therefore I cannot do Y. How can I do Y?

If your use case cannot be covered by one of the new APIs, please let us know and we'll see try to help you out.

Which spatial formats do you support for the download?

Currently we only support GeoJSON.

I have another question or suggestion not covered in this document.

Reach out to cbed@fb.com with your follow up concerns to discuss further.