Open Curb Assets Data Specification

Version 1.0
Last updated: 2019-03-11
Maintained by Coord (support@coord.co)

Overview

The Open Curb Assets Data Specification defines a format for sharing data about features of curbs. This includes both features of the sidewalk, like bike racks, parking signs, and fire hydrants, as well as features of the street, like lane markings, curb cuts, and crosswalks. What these features all have in common is that they serve to define which activities can happen along the side of a street.

The goal of this Specification is to focus on describing curb assets, not interpreting them. For instance, data that matches this Specification may tell you that a curb is painted yellow, but not what this means. This is in order to make curb asset data easier to verify, maintain, update, and correct.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Data Structure

Open Curb Assets data is provided in ***collections***, which are represented in GeoJSON format, with each collection comprising a GeoJSON `FeatureCollection` object. Please note that,

1

per [RFC 7946](#), GeoJSON feature collections MUST use the WGS 84 spatial referencing system.

A single collection represents data about a single geographic area (such as a city or metropolitan area) as generated by a single *author* over a single *time period*. The author of a collection (which could be a government agency, private company, non-profit group) was responsible for collecting the data and is responsible for ensuring its accuracy. A **metadata** object contains information about the area, author, time period, and scope of the data contained in the collection.

Each GeoJSON `Feature` object represents a single curb asset. Some curb assets are *point features*, meaning that they are represented as a single point along a curb. Others are *linear features*, meaning that they start at one point along a curb and end at another. Linear features MAY overlap each other, and multiple point features MAY occupy exactly the same location.

In addition to their latitude and longitude geometries, both point and linear features MAY contain linear references, defining how far along the curb they start and end, as defined below.

Collection Metadata

Every collection MUST contain a "`metadata`" key, whose value is a Metadata Object.

A Metadata Object is a JSON object containing the following keys. Each key is OPTIONAL and its value MUST be a string unless specified otherwise.

- `author` (REQUIRED): the name of the organization that is responsible for the data in this collection. In the absence of an `owner` field, the `author` field MUST name the organization holding the copyright on the data contained in the collection, unless the data is in the public domain.
- `license_url` (REQUIRED): a URL containing the license governing the use of this data, or the string "`public_domain`" if the data is in the public domain. It is STRONGLY RECOMMENDED that data produced in accordance with this Specification be licensed under the [Open Database License](#).
- `last_updated` (REQUIRED): the latest time any update was made to the data in this collection, as an [RFC 3339](#) timestamp.
- `owner`: the name of the organization that holds the copyright on the data contained in the collection.
- `contact`: an email address to contact with questions, corrections, or licensing inquiries for the data contained in this collection.
- `location`: the name of the location covered by this data. This could be a city or neighborhood name.

- **`asset_types`**: a list of all the types of assets contained in this collection. If present, the value of **`asset_types`** MUST be a list of strings, and there MUST NOT be any assets contained in this collection with types not listed here.
- **`collection_period_start`**: the time when the data in this collection began to be gathered, as an [RFC 3339](#) timestamp. This field is primarily relevant for studies conducted over a short period of time, and SHOULD NOT be present when the data in the collection is derived from, e.g., a city asset management database.
- **`collection_period_end`**: the time when the data gathering process in this collection finished, as an [RFC 3339](#) timestamp. This SHOULD represent the time when the vast majority of the data had been gathered, as opposed to **`last_updated`**, which represents the latest time that any update whatsoever was made. For instance, fixing a small error or updating metadata SHOULD NOT result in a change to **`collection_period_end`**.
- **`notes`**: any information about this collection that would help a reader better understand the data contained within.
- **`map_url`**: the URL of a street map to which this collection's data is referenced. For instance, this could be a link to a city's street database.
- **`Bounding box`**: the boundary of all curbs in this collection defined by **`x_min`**, **`x_max`**, **`y_min`**, **`y_max`**.

Example

```
{
  "type": "FeatureCollection",
  "metadata": {
    "author": "Coord",
    "license": "https://opendatacommons.org/licenses/odbl/1.0/index.html",
    "last_updated": "2019-01-31T17:00:00.000Z",
    "contact": "support@coord.co",
    "location": "Park Slope, Brooklyn, NY",
    "asset_types": ["Parking Sign"],
    "notes": "Parking sign survey of the Park Slope neighborhood"
  },
  "features": [...]
}
```

Features

Each curb asset in the collection is represented as a GeoJSON **`Feature`** object. Features MUST contain a **`properties`** object, as per the GeoJSON specification. Features MUST also contain a **`geometry`** object, and their geometry type MUST be either **`LineString`** or **`Point`**.

Features MAY also have an `id` member representing an identifier for this particular asset. The value of this member is either a string or a number, and MUST be unique within the collection.

Asset Types and Subtypes

The `properties` object MUST contain a key named `asset_type`, This describes the kind of asset this feature represents. properties MAY contain a key named `asset_subtype`, with additional information on the particular kind of feature being identified. Both `asset_type` and `asset_subtype` MUST have values of string type.

While authors MAY use any string for `asset_type` and `asset_subtype`, it is RECOMMENDED that collections use the values below where possible.

- **`Parking Sign`**: a sign with parking rule information on it.

- **`Curb Paint`**: a painted segment of a curb. When `asset_type` is `Curb Paint`, `asset_subtype` MUST describe the color of the paint.

- **`Curb Cut`**: a portion of the curb that has been graded down for pedestrians or vehicles to cross. When asset_type is `Curb Cut`, the following values of `asset_subtype` MAY be used to distinguish different types of curb cut:
  - **`Ramp`**: a pedestrian ramp.
  - **`Driveway`**: a curb cut to allow cars access to a private facility, such as a parking lot or gas station.
  - **`Street`**: a break in the curb allowing vehicles to access a narrow alley or laneway.

- **`Hydrant`**: A fire hydrant. Note that the `Hydrant` feature type SHOULD NOT be used to represent connection points located on or near building walls, but only those located along the curb or in the street.

- **`Bus Stop`**: A marked bus stop. Note that this feature type SHOULD represent the position of the bus stop's pole or sign rather than the position of the bus shelter, if any.

- **`Pavement Marking`**: A marking on the pavement in the curb lane of a street. Note that this `asset_type` SHOULD NOT be used to represent markings on lanes other than the one used for vehicle parking and loading, if such a lane exists. For instance, parking-protected bicycle lanes or offset bus lanes SHOULD NOT be described using this `asset_type`. The following `asset_subtype`s MAY be used to distinguish different types of pavement marking:
  - **`Bicycle`**: a bicycle lane.
  - **`Bus`**: a bus lane.
  - **`Taxi`**: a lane marked for use by taxis, or as a taxi stand.

- ○ **Arrow**: an arrow painted on the pavement. This generally indicates the lane is intended for use as a traffic lane (such as a right-turn lane).
- ○ **Diagonal Lines**: white diagonal lines striped along the pavement.
- ○ **Zigzag**: a zig-zag line striped along the pavement (used to identify driveways and alleys in certain countries).
- ○ **Parallel Parking Spaces**: painted parking spaces for vehicles to parallel park. Note that a single feature with this **asset_subtype** MAY represent an extent of the curb containing multiple individual parking spaces.
- ○ **Perpendicular Parking Spaces**: painted parking spaces for vehicles to park perpendicular to the curb. Note that a single feature with this **asset_subtype** MAY represent an extent of the curb containing multiple individual parking spaces.
- ○ **Yellow**: an area of pavement filled in or outlined in yellow paint.
- ○ **Red**: an area of pavement filled in or outlined in red paint.
- ○ **Blue**: an area of pavement filled in or outlined in blue paint.
- ○ **ISA**: the [International Symbol of Access](#) painted on the pavement (generally to indicate a parking space reserved for people with disabilities)

- **Crosswalk**: a pedestrian crossing (signed or painted). Note that, in some collections, crosswalks MAY be represented by **Curb Cut** features instead.

- **Bicycle Rack**: a rack used for bicycle parking. Bicycle racks used by bike-share systems MAY be indicated with a **asset_subtype** of "**Bike Share**"

- **Curb Extension**: an area where the sidewalk widens, replacing the curb lane of the street (often called a "neckdown", "bulbout", or "curb bulb").

- **Bollards**: a line of bollards placed along the curb.

- **Fence**: a fence along the curb preventing pedestrians from entering or exiting vehicles.

- **Parking Meter**: a machine used to pay for parking on the street. Single, double, and multi-space parking meters MAY be indicated with **asset_subtypes** of "**Single**", "**Double**", and "**Multi-Space**", respectively.

- **No-Parking Curb**: a curb that cannot be used for vehicle parking for an unspecified reason. This **asset_type** MAY used to indicate curbs that are not accessible to pedestrians, on highway on-ramps or off-ramps, in medians of streets, etc.

Linear Referencing

It is often desirable to identify curb assets by their linear positions along the curb rather than their latitude and longitude. The following fields in the properties object SHOULD be used to provide linear references:

- **curb_id**: a string that identifies the particular curb (i.e., the block face) this asset is on. **curb_id** MUST be present if any other linear referencing fields are provided. **curb_id** SHOULD refer to a blockface found in the **map_url** identified in **metadata** (see above).

- **distance_meters**: a number that identifies the distance in meters from the start of the curb in **curb_id** to the feature's location. If distance_meters is present, the feature MUST have a **geometry** of type **Point**.

- **distance_start_meters**: a number that identifies the distance in meters from the start of the curb in **curb_id** to where the feature begins. If **distance_start_meters** is present, **distance_end_meters** MUST be present too, and **distance_end_meters** MUST be larger than **distance_start_meters**. Additionally, the feature MUST have a **geometry** of type **LineString**.

- **distance_end_meters**: a number that identifies the distance in meters from the start of the curb in **curb_id** to where the feature ends. If **distance_start_meters** is present, **distance_end_meters** MUST be present too, and **distance_end_meters** MUST be larger than **distance_start_meters**. Additionally, the feature MUST have a **geometry** of type **LineString**.

In addition, the following fields MAY be provided to give data users additional context about the location of the curb identified by **curb_id**:

- **street_name**: the name of the street this curb is on.

- **from_street_name**: the name of the street where this curb begins, or **"DEAD END"** if it begins at a dead end. If **from_street_name** is present, it MUST represent a street that intersects this curb at the point from which distances are measured.

- **to_street_name**: the name of the street where this curb ends, or **"DEAD END"** if it ends at a dead end.

- **street_side**: the cardinal or intercardinal direction (**"N"**, **"NE"**, **"E"**, etc.) that is closest to the heading of a line drawn to a point on this curb from the nearest point on the centerline of the street. Note that, for some curved streets, this is not well-defined;

additionally, the direction as defined above does not necessarily match the customary directions used within a city (i.e., "street-grid North" may be represented as "`NW`")

Images, Text, Arrows, and Facing

Features MAY have an image associated with them. If so, a URL for the image is provided in the `image_url` member of `properties`.

The interpretation of some assets (particularly parking signs) depends on the direction they are facing. The facing of an asset MAY be indicated using the `facing` member of `properties`. If present, its value MUST be one of the following strings:

- `street`: facing the street.
- `sidewalk`: facing the sidewalk.
- `left`: facing to the left as viewed from the street.
- `right`: facing to the right as viewed from the street.

Signs and other features may also have text and arrows. These MAY be represented using the `text` and `arrows` members of properties, respectively. While `text` may contain any string, the value of `arrows` MUST be one of the following strings:

- `left`: arrow pointing to the left.
- `left_down`: arrow pointing between left and down.
- `down`: arrow pointing straight down.
- `right_down`: arrow pointing between right and down.
- `left_right_down`: arrows pointing both left_down and right_down.
- `left_right`: arrows pointing both left and right.
- `none`: no arrows on this sign.

Some parking signs are double-sided, facing both the street and the sidewalk. In this case:

- The `double_sided` property SHOULD be present, with a value of `true`;
- `facing` SHOULD be `street`;
- `arrows` SHOULD identify the direction of the sign's arrows as viewed from the street.

Additional Properties

The `properties` object MAY contain additional keys representing other information about the asset, such as its condition, height, installation date, etc.

Example

```json
{
  "type": "Feature",
  "properties": {
    "asset_type": "Curb Paint",
    "asset_subtype": "Red",
    "curb_id": "bGE6Njc2Mzk",
    "distance_start_meters": 5.02,
    "distance_end_meters": 13.33
  },
  "geometry": {
    "coordinates": [
      [-118.256495, 34.049384],
      [-118.256549, 34.049324]
    ],
    "type": "LineString"
  },
}
```