OsmOApi Documentation

Release 0.1

Christian Ledermann

Jul 15, 2017

Contents

OsmOApi is short for OpenStreetMap OAuth API.

The authorization is not in the scope of this package, you can use python-social-auth for this

It takes a geo_interface compatible dictionary to create OSM Features.

Contents:

class osmoapi.**ChangeSet**(*id=None*, *created_by='osmoapi v 0.1'*, *comment='Changes via API'*, ***kwargs*)

Changesets.

http://wiki.openstreetmap.org/wiki/API_v0.6#Changesets_2

To make it easier to identify related changes the concept of changesets is introduced. Every modification of the standard OSM elements has to reference an open changeset. A changeset may contain tags just like the other elements. A recommended tag for changesets is the key *comment=** with a short human readable description of the changes being made in that changeset, similar to a commit message in a revision control system. A new changeset can be opened at any time and a changeset may referenced from multiple API calls. Because of this it can be closed manually as the server can't know when one changeset ends and another should begin.

etree_element()

Create a XML representation of this changeset.

class osmoapi.**OSMOAuthAPI**(*client_key*, *client_secret*, *resource_owner_key*, *resource_owner_secret*, *test=True*)

OSM API with OAuth.

close_changeset(*changeset*)

Close: PUT /api/0.6/changeset/#id/close.

http://wiki.openstreetmap.org/wiki/API_v0.6#Close:_PUT_.2Fapi.2F0.6.2Fchangeset.2F.23id.2Fclose

Closes a changeset. A changeset may already have been closed without the owner issuing this API call. In this case an error code is returned.

Parameters:

id

The id of the changeset to close. The user issuing this API call has to be the same that created the changeset.

Response

Nothing is returned upon successful closing of a changeset (HTTP status code 200).

create_changeset(*created_by*, *comment*, ***kwargs*)

Create: PUT /api/0.6/changeset/create.

http://wiki.openstreetmap.org/wiki/API_v0.6#Create:_PUT_.2Fapi.2F0.6.2Fchangeset.2Fcreate

The payload of a changeset creation request has to be one or more changeset elements optionally including an arbitrary number of tags. Any number of possibly editor-specific, tags are allowed. An editor might, for example, automatically include information about which background image was used, or even a bit of internal state information that will make it easier to revisit the changeset with the same editor later, etc.

Clients should include a *created_by=** tag. Clients are advised to make sure that a *comment=** is present, which the user has entered. It is optional at the moment but this might change in later API versions. Clients should not automatically generate the comment tag, as this tag is for the end-user to describe their changes. Clients may add any other tags as they see fit.

create_note(*point*, *text*)

Create a new note: Create: POST /api/0.6/notes

http://wiki.openstreetmap.org/wiki/API_v0.6#Create_a_new_note:_Create:_POST_.2Fapi.2F0.6.2Fnotes

URL: http://api.openstreetmap.org/api/0.6/notes?lat=51.00&lon=0.1&text=ThisIsANote Return type: application/xml

Parameter	Description	Allowed values
lat	Specifies the latitude of the bug	floatingpoint number in degrees
lon	Specifies the longitude of the bug	floatingpoint number in degrees
text	Text containing the note	A text field with arbitrary text

If the request is made as an authenticated user, the note is associated to that user account.

Error codes HTTP status code 400 (Bad Request) if the text field was not present HTTP status code 405 (Method Not Allowed) If the request is not a HTTP POST request HTTP status code 403 If the user did not authorize your application to create/edit notes.

diff_upload(*change*)

Diff upload: POST /api/0.6/changeset/#id/upload.

http://wiki.openstreetmap.org/wiki/API_v0.6#Diff_upload:_POST_.2Fapi.2F0.6.2Fchangeset.2F.23id.2Fupload

With this API call files in the OsmChange format can be uploaded to the server. This is guaranteed to be running in a transaction. So either all the changes are applied or none.

To upload an OSC file it has to conform to the OsmChange specification with the following differences:

Each element must carry a changeset and a version attribute, except when you are creating an element where the version is not required as the server sets that for you. The changeset must be the same as the changeset ID being uploaded to. A <*delete*> block in the OsmChange document may have an *if-unused* attribute (the value of which is ignored). If this attribute is present, then the delete operation(s) in this block are conditional and will only be executed if the object to be deleted is not used by another object. Without the if-unused, such a situation would lead to an error, and the whole diff upload would fail. OsmChange documents generally have user and uid attributes on each element. These are not required in the document uploaded to the API.

Parameters:

id

The ID of the changeset this diff belongs to.

POST data

The OsmChange file data.

class osmoapi.**OsmChange**(*changeset*)

OsmChange.

http://wiki.openstreetmap.org/wiki/OsmChange

osmChange is the file format used by osmosis (and osmconvert) to describe differences between two dumps of OSM data. However, it can also be used as the basis for anything that needs to represent changes. For example, bulk uploads/deletes/changes are also changesets and they can also be described using this format.

create_multipolygon(*multipolygon*, ***kwargs*)

Create a Relation:multipolygon.

http://wiki.openstreetmap.org/wiki/Relation:multipolygon

Any area that is complex (e.g., because its outline consists of several ways joined together, or because the area consists of multiple disjunct parts, or has holes) requires a multipolygon relation.

A multipolygon relation can have any number of ways in the role outer (the outline) and any number of ways in the role inner (the holes), and these must form valid rings to build a multipolygon from.

create_node(*point*, ***kwargs*)
Create a Node.

http://wiki.openstreetmap.org/wiki/Node

A node is one of the core elements in the OpenStreetMap data model. It consists of a single point in space defined by its latitude, longitude and node id.

Nodes can be used to define standalone point features, but are more often used to define the shape or "path" of a way.

create_way(*linestring*, ***kwargs*)
Create a Way.

http://wiki.openstreetmap.org/wiki/Way

A way is an ordered list of nodes which normally also has at least one tag or is included within a Relation. A way can have between 2 and 2,000 nodes. A way can be open or closed. A closed way is one whose last node on the way is also the first on that way. A closed way may be interpreted either as a closed polyline, or an area, or both.

etree_element()
Create a XML representation of this change.

CHAPTER 1

Indices and tables

- genindex
- modindex
- search

Python Module Index

o

Index