

---

# **commons-api Documentation**

**mySociety**

**Feb 15, 2019**



---

## Contents:

---

<b>1</b>	<b>Architecture</b>	<b>3</b>
1.1	Data ingest . . . . .	3
1.2	Boundaries . . . . .	3
1.3	Models . . . . .	3
1.4	API . . . . .	4
<b>2</b>	<b>Reference</b>	<b>5</b>
2.1	Tasks . . . . .	5
2.2	Querying Wikidata . . . . .	5
2.3	Renderers . . . . .	5
<b>3</b>	<b>Indices and tables</b>	<b>7</b>
	<b>Python Module Index</b>	<b>9</b>



`commons-api` provides a view onto democracy data maintained in Wikidata.

It periodically pulls in data using SPARQL queries against the [Wikidata Query Service \(WDQS\)](#), run as [celery](#) tasks. Data are stored using the Django ORM, and exposed through views and templates, and a [django-rest-framework](#) API.



### 1.1 Data ingest

Data are pulled in with a number of celery tasks, listed at *Updating from Wikidata*.

### 1.2 Boundaries

`commons-api` embeds `represent-boundaries` for storing boundary data. Boundaries are pulled in using celery tasks from two sources:

- `boundaries/` directories in proto-commons repositories
- GeoShapes (linked with `wdt:P3896` properties in Wikidata) from Wikimedia Commons

Represent Boundaries has two models, `boundaries.models.BoundarySet` and `boundaries.models.Boundary`. For boundaries sourced from proto-commons repositories, each shapefile is mapped to its own `BoundarySet`. Commons geoshapes have a `boundaries.models.BoundarySet` per source URL, and so have a single `boundaries.models.Boundary` per `boundaries.models.BoundarySet`.

The `represent-boundaries` views are included in `commons_api.urls.urlpatterns` at `/boundary-sets/` and `/boundaries/`.

### 1.3 Models

Models representing Wikidata items all subclass `commons_api.wikidata.models.WikidataItem`, which implements common functionality around labels, links to Wikipedia, etc. Wikidata IDs are used as primary keys.

Some models are timebound, i.e. they potentially have start/inception and end/dissolution dates in Wikidata. These subclass `commons_api.wikidata.models.Timebound`.

Some Wikidata IDs will have representations in different models, e.g. Andorra is both a `commons_api.wikidata.models.Country` and an `commons_api.wikidata.models.AdministrativeArea` for its national legislature.

Memberships correspond to statements — not items — in Wikidata, so `commons_api.wikidata.models.LegislativeMembership` doesn't subclass `commons_api.wikidata.models.WikidataItem`.

## 1.4 API

`commons-api` has an API built on top of `django-rest-framework` and exposed at `/api/`. The API is a standard use of `django-rest-framework`, split across:

- `commons_api.wikidata.api` (the `rest_framework.routers.DefaultRouter` definition, where viewsets get wired in)
- `commons_api.wikidata.viewsets` (these handle the list and detail views for API objects, using serializers and renderers)
- `commons_api.wikidata.serializers` (these extract data from ORM objects into a simple Python data structure)
- `commons_api.wikidata.renderers` (these turn the simple Python data structure into representations (e.g. JSON, CSV) to send to clients).



## 2.1 Tasks

### 2.1.1 Updating from Wikidata

```
commons_api.wikidata.tasks.refresh_labels(app_label, model, ids=None,  
                                           queued_at=None)
```

Refreshes all labels for the given model

```
commons_api.wikidata.tasks.refresh_country_list()  
commons_api.wikidata.tasks.refresh_legislatures(id, queued_at)  
commons_api.wikidata.tasks.refresh_members(id, queued_at)  
commons_api.wikidata.tasks.refresh_districts(id, queued_at)
```

### 2.1.2 Updating boundaries from Wikimedia Commons

### 2.1.3 Updating boundaries from proto-commons repositories

## 2.2 Querying Wikidata

```
commons_api.wikidata.utils.templated_wikidata_query(query_name, context)
```

## 2.3 Renderers

```
class commons_api.wikidata.renderers.GeoJSONRenderer  
  
    format = 'geojson'
```

```
media_type = 'application/vnd.geo+json'

render (data, accepted_media_type=None, renderer_context=None)
    Render data into JSON, returning a bytestring.

to_geojson (data)

class commons_api.wikidata.renderers.PopoloJSONRenderer

    format = 'popolo-json'
    media_type = 'application/popolo+json'
    membership_to_popolo (data)
    organization_to_popolo (data, classification=None)
    person_to_popolo (data)
    render (data, accepted_media_type=None, renderer_context=None)
        Render data into JSON, returning a bytestring.
    to_m17n (data)
```

## CHAPTER 3

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### C

`commons_api.wikidata.renderers`, [5](#)



## C

commons\_api.wikidata.renderers (module), 5

## F

format (commons\_api.wikidata.renderers.GeoJSONRenderer attribute), 5

format (commons\_api.wikidata.renderers.PopoloJSONRenderer attribute), 6

## G

GeoJSONRenderer (class in commons\_api.wikidata.renderers), 5

## M

media\_type (commons\_api.wikidata.renderers.GeoJSONRenderer attribute), 5

media\_type (commons\_api.wikidata.renderers.PopoloJSONRenderer attribute), 6

membership\_to\_popolo() (commons\_api.wikidata.renderers.PopoloJSONRenderer method), 6

## O

organization\_to\_popolo() (commons\_api.wikidata.renderers.PopoloJSONRenderer method), 6

## P

person\_to\_popolo() (commons\_api.wikidata.renderers.PopoloJSONRenderer method), 6

PopoloJSONRenderer (class in commons\_api.wikidata.renderers), 6

## R

refresh\_country\_list() (in commons\_api.wikidata.tasks), 5

refresh\_districts() (in commons\_api.wikidata.tasks), 5

refresh\_labels() (in commons\_api.wikidata.tasks), 5

refresh\_legislatures() (in commons\_api.wikidata.tasks), 5

refresh\_members() (in commons\_api.wikidata.tasks), 5

render() (commons\_api.wikidata.renderers.GeoJSONRenderer method), 6

render() (commons\_api.wikidata.renderers.PopoloJSONRenderer method), 6

## T

templated\_wikidata\_query() (in commons\_api.wikidata.utils), 5

to\_geojson() (commons\_api.wikidata.renderers.GeoJSONRenderer method), 6

to\_m17n() (commons\_api.wikidata.renderers.PopoloJSONRenderer method), 6