

PYGEOAPI



API FOR GEOSPATIAL DATA

**@tomkralidis @JMendesDeJesus
@normanbarker @perrygeo @justb4
@pvgenuchten**

version: June 19, 2019

I. IN THE BEGINNING THERE WAS... WFS V1/V2

CHARACTERISTICS

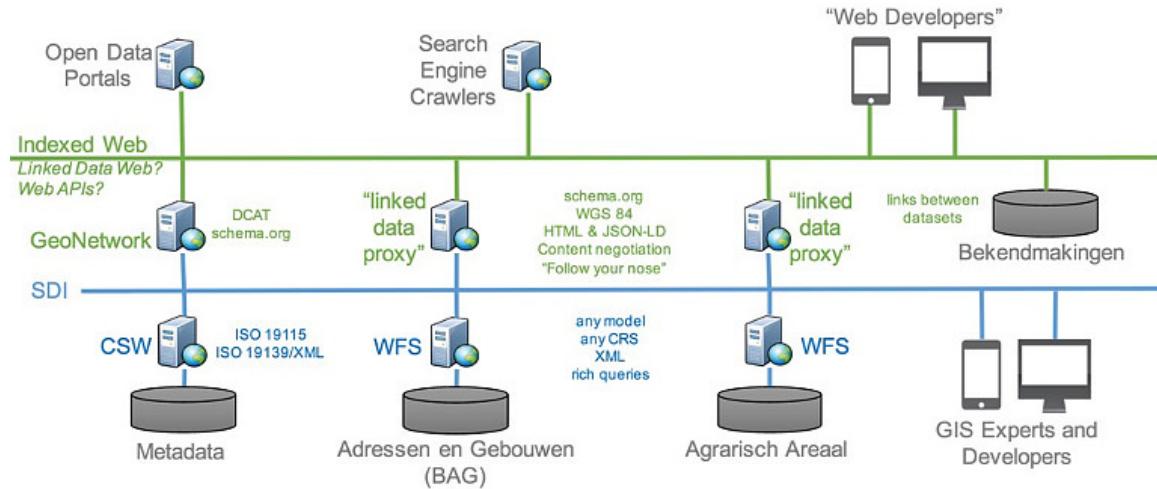
- Key Value Parameters (KVP) or XML encodings
- Requests using HTTP GET, HTTP POST and SOAP
- Responses as XML (GML)
- Error handling ExceptionReport in response

```
<wfs:FeatureCollection
    xmlns:ms="http://mapserver.gis.umn.edu/mapserver"
    xmlns:gml="http://www.opengis.net/gml" [...]>
    <gml:boundedBy>
        <gml:Envelope srsName="EPSG:32661">[ ... ]
            </gml:Envelope>
    </gml:boundedBy>
    <gml:featureMember>
        <ms:north_poles_wfs gml:id="north_poles_wfs.0">
            <gml:boundedBy>[ ... ]</gml:boundedBy>
            <ms:msGeometry>[ ... ]</ms:msGeometry>
            <ms:Id>0</ms:Id>[ ... ]
            </gml:featureMember>[ ... ]
    </gml:featureMember>
</wfs:FeatureCollection>
```

ISSUES WITH WFS V1/2

- Highly verbose
- Highly dependent on XML technologies
- Verbose documentation (WFS 2.0 has 239 pages)
- OGC document from 2010
- No unique urls for features
- Features hard to index by search engine
- "In-band" error reporting

MORE ISSUES



- ldproxy for services to be crawlable
- OGC world is NOT W3C-/Web- friendly

II. A NEW ERA IS UPON US

RESTFUL

- REST is becoming a MUST
- Fast standards' development
- Opinion:
Standards (-development) should be more developer-friendly

WHAT IS REST ???

- REpresentational State Transfer (REST)
- Uses HTTP's verbs (GET/POST/DELETE)
- Uses HTTP's codes (200, 404, 201, etc)
- Uses URI to identify resources
- Content negotiation (ask for media-type)
- Stateless

{rest}

- Very popular among developers
- Easy to understand
- JSON is king in REST webservices
- No rigid standards (good or bad)

OGC AND W3C

Spatial Data on the Web Best Practices

W3C Working Group Note 28 September 2017



<https://www.w3.org/TR/sdw-bp/>

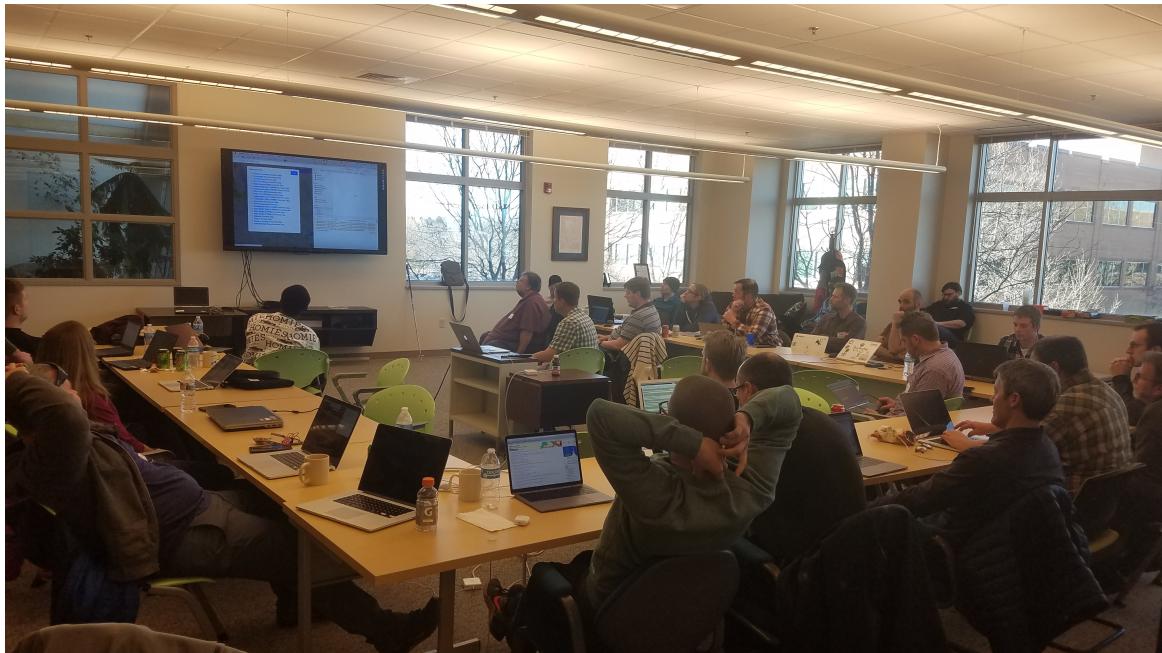
MUST READ DOCUMENT

III. KICKSTARTING WFS 3.0

WFS 3.0 HACKATHON

- March 2018 - **Fort Collins, CO, USA**
- OGC-sponsored
- Develop WFS 3.0 based on Swagger/OpenAPI (v3)
- Write specification using GitHub
- First implementations
- **pygeoapi was born on Valentine 2018!**









Matthew Douglas Wiggett aka Jivan Amara

1975 - 2018

(Go-lang client/server for WFS3.0)

IV. OPENAPI / SWAGGER

OPENAPI / SWAGGER

- Specification on how to describe a REST web service
- Originally was referred as Swagger
- Language agnostic

<http://swagger.io>



- OpenAPI is based on YAML Syntax
- Describes REST endpoints
- HTTP Verbs to use
- Format of Responses

Example yaml

```
openapi: 3.0.1
info:
  title: OGC Web Feature Service standard API
  version: 0.0.1
  description: 'This is a sample OpenAPI definition for OGC'
  contact:
    name: La oreja de van Gogh
```

```
paths:  
/:  
  get:  
    summary: landing page of this API  
    description: 'Description of dataset'  
    operationId: getLandingPage  
    tags:  
      - Capabilities  
  responses:  
    '200':  
      description: links to the API capabilities  
      content:  
        - application/json:
```

V. THE WFS 3.0 SPEC

NB: renamed as: "**OGC API - Features**"

SPEC

github.com/opengeospatial/WFS_FES

Maintained in GitHub

By C. Portele - [link](#)

OGC API – Features – Part 1: Core



Table 1. Overview of resources, applicable HTTP methods and links to the document sections

Resource	Path	HTTP method	Document reference
Landing page	/	GET	7.2 API landing page
API definition	/api	GET	7.3 API definition
Conformance declaration	/conformance	GET	7.4 Declaration of conformance classes
Feature collections	/collections	GET	7.12 Feature collections
Feature collection	/collections/{collectionId}	GET	7.13 Feature collection
Features	/collections/{collectionId}/items	GET	7.14 Features
Feature	/collections/{collectionId}/items/{featureId}	GET	7.15 Feature the features

<http://docs.opengeospatial.org/DRAFTS/17-069r1.html#tldr>



A sample API conforming to the OGC Web Feature Service standard

[M1](#)[OGC3](#)

This is a sample OpenAPI definition that conforms to the OGC Web Feature Service specification (conformance classes: "Core", "GeoJSON", "HTML" and "OpenAPI 3.0").

[Acme Corporation - Website](#)
[Send email to Acme Corporation](#)
[CC-BY 4.0 license](#)

<https://dev.example.org/> ▾

Capabilities Essential characteristics of this API including information about the data.

GET	/ landing page of this API	↔
GET	/conformance information about standards that this API conforms to	↔
GET	/collections describe the feature collections in the dataset	↔
GET	/collections/{collectionId} describe the {collectionId} feature collection	↔

Features Access to data (features).

GET	/collections/{collectionId}/items retrieve features of feature collection {collectionId}	↔
GET	/collections/{collectionId}/items/{featureId} retrieve a feature; use content negotiation to request HTML or GeoJSON	↔

Check it in SwaggerHub

EXAMPLE

```
/collections  
/collections/countries  
/collections/countries/items  
/collections/countries/items/The_Netherlands
```

Data is structured and each Feature is a REST representation

OGC API - FEATURES

LINKS

- [**GitHub Repository**](#)
- [**OpenAPI Schemas**](#)
- [**Draft 17-069.html**](#)
- [**Gitter Chat**](#)
- [**OGC-API-Hackathon-2019**](#)

VI. PYGEOAPI

PYGEOAPI



Part of the amazing Geopython community in GitHub

geopython.github.io

The screenshot shows a web browser window with the URL "geopython.github.io" in the address bar. At the top right is a "View on GitHub" button with a GitHub icon. The main content area has a dark header with the text "geopython" in white. Below the header is a logo consisting of four colored squares (blue, yellow, red, green) arranged in a 2x2 grid. To the right of the logo, the text "geopython is a GitHub organization comprised of [Python](#) projects related to geospatial." is displayed. A bulleted list follows, including "OWSLib", "pycsw", "PyWPS", "MetaSearch", "GeoHealthCheck", "MapSlicer", "CadTools", "Stell", and "pygeoapi". Below this list are links to "Gitter" and the "mailing list". A note about the "Toblerity Project" is also present. At the bottom of the page, a dark footer bar contains the text "Published with [GitHub Pages](#)".

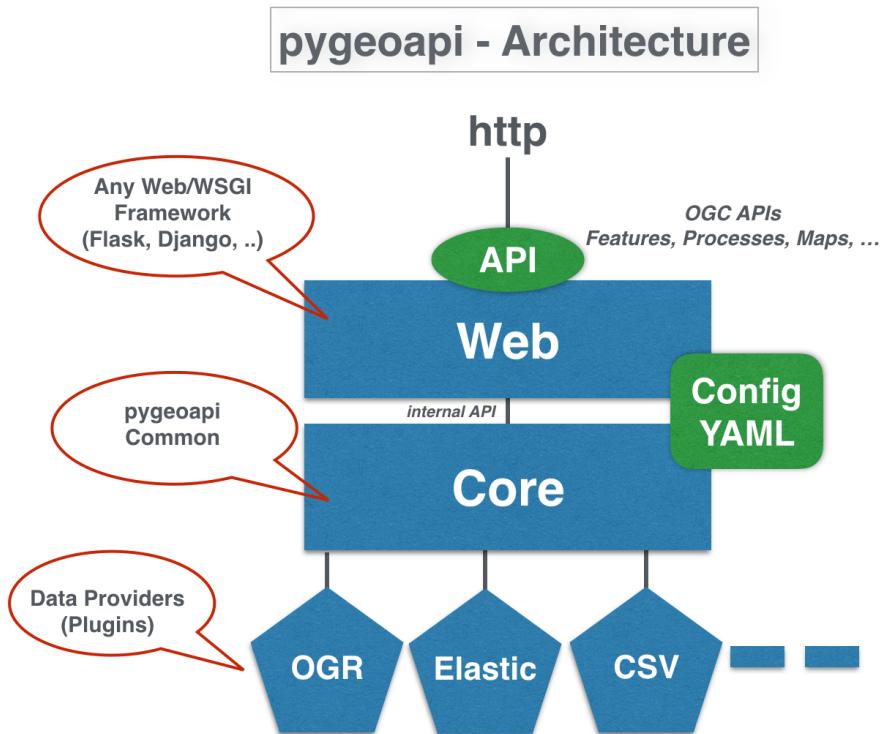
PYGEOAPI

- More than just a WFS v3 implementation
- Access geospatial data via OGC APIs
- OSGeo Community Project (in Motion)
- Powered (default) by:



PYGEOAPI - TECHNICAL

- Flask with REST support
- OpenAPI endpoint automatically generated
- Data provider agnostic (plugins)
- Docker Images



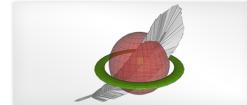
DATA PROVIDERS



elasticsearch



CSV



Spatialite*



GeoJSON

Elasticsearch

CSV

Spatialite*

GeoJSON*

* also via GDAL-OGR Provider

DATA PROVIDERS - CONT



GDAL-OGR

DATA PROVIDERS - GDAL-OGR

- via OGR-Python bindings
- in theory **~100 Vector Formats!**
- thus: WFS2, GeoPackage, Shapefile, GeoJSON, PostGIS, ...
- paging, reprojection, bbox/attr/id-query, auth
- ==> unlock existing WFS2s!

<https://pygeoapi.io>

The screenshot shows the homepage of the pygeoapi website. At the top, there's a browser header with the URL 'pygeoapi.io' and a refresh icon. Below the header is the pygeoapi logo, which consists of a blue and yellow stylized map or compass design. The word 'pygeoapi' is written in a lowercase sans-serif font directly beneath the logo. Underneath the logo, a brief description reads: 'pygeoapi is a Python server implementation of the emerging OGC WFS 3.0 standard'. The main content area is organized into several sections, each enclosed in a rounded rectangular box:

- Code and Docs**
 - GitHub repository
 - Repository with code, install and README and Docker compositions
 - [GitHub](#)
- Demo**
 - Live demonstration
 - Demonstration of pygeoapi in action
 - [demo.pygeoapi.io](#)
- Docker Images**
 - Images hosted in Docker Hub
 - Docker images/composition to run pygeoapi
 - [Docker Hub](#)
- WFS 3.0**
 - Proto-WFS 3.0 docs
 - Live documentation of future WFS 3.0 standard
 - [WFS 3.0 repository](#)
- Client Tools**
 - WFS clients
 - Client tools to work with WFS 3.0
 - [GDAL driver](#)
 - [OWSLib](#)
- Community**
 - Support
 - Talk to the developers or help us with code and suggestions
 - [Gitter](#)
 - [Mailing list](#)

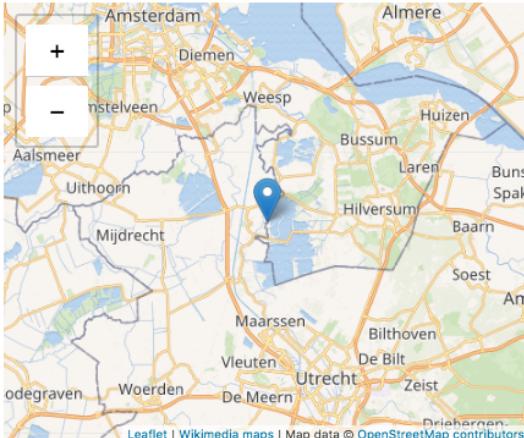
DEMO

C https://demo.pygeoapi.io/master/collections/dutch_windmills/items/Molens.3 |

pygeoapi Demo instance - running latest GitHub version [Contact](#)

Home / Collections / Windmills Within The Netherlands / Items / Molens.3 [JSON](#)

Item Molens.3



A map of the Netherlands with a blue marker indicating the location of Molens.3 in the Utrecht area. Labels for various towns and cities like Amsterdam, Diemen, Weesp, Bussum, Laren, Baarn, Soest, Utrecht, and Woerden are visible.

Leaflet | Wikimedia maps | Map data © OpenStreetMap contributors

[Prev](#) [Next](#)

Property	Value
INFOLINK	https://zoeken.allemolens...
THUMBNAIL	
HFDFUNCTIE	poldermolen
FOTOGRAAF	Frank Terpstra
FOTO_GROOT	

Powered by [pygeoapi 0.6.0](#)

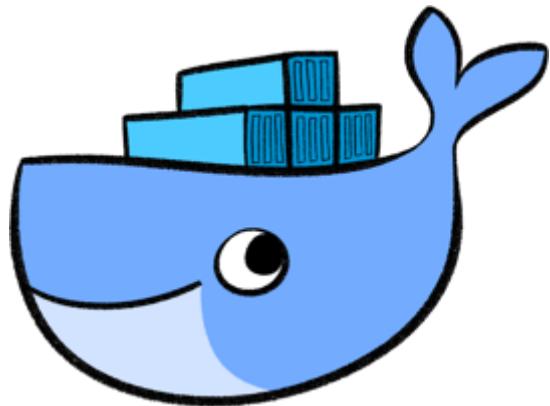
PYGEOAPI - ROADMAP

- More data providers
- Content negotiation (e.g. Response as GeoPackage, GML,...)
- Advanced filters (CQL)
- More OGC APIs: Maps, Tiles, Coverages, Processes

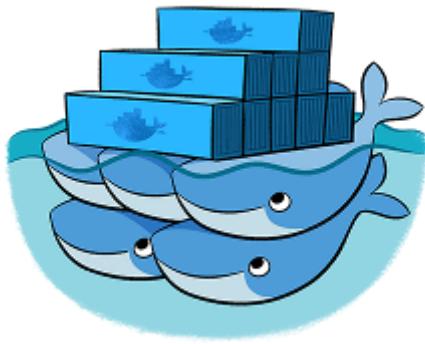
PYGEOAPI - LINKS

- Home: [**https://pygeoapi.io**](https://pygeoapi.io)
- Demo: [**https://demo.pygeoapi.io**](https://demo.pygeoapi.io)
- GitHub: [**https://github.com/geopython/pygeoapi**](https://github.com/geopython/pygeoapi)
- Docker:
[**https://cloud.docker.com/u/geopython/repository/docker/geopython/pygeoapi**](https://cloud.docker.com/u/geopython/repository/docker/geopython/pygeoapi)
- Chat: [**https://gitter.im/geopython/pygeoapi**](https://gitter.im/geopython/pygeoapi)
- Mail: [**https://lists.osgeo.org/mailman/listinfo/pygeoapi**](https://lists.osgeo.org/mailman/listinfo/pygeoapi)

VII. DOCKER AND PYGEOAPI



- pygeoapi Docker Images on DockerHub: [**Link**](#)
- Images create linux containers running content
- Use Docker Images to test locally or deploy in server
- See READMEs



Docker Images for geopython/pygeoapi

UP AND RUNNING IN 2 MINUTES

Pull from DockerHub

```
docker pull geopython/pygeoapi:latest
```

Run (default config)

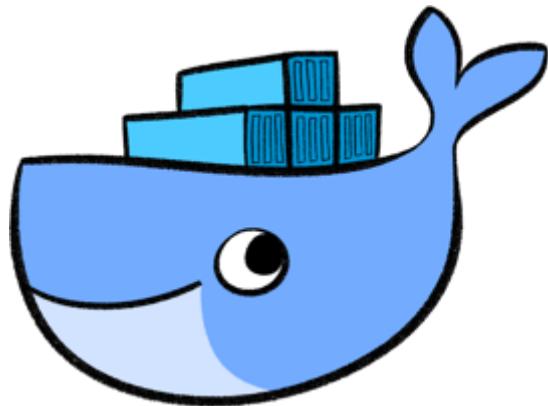
```
docker run -p 5000:80 -it geopython/pygeoapi:latest
```

Test:

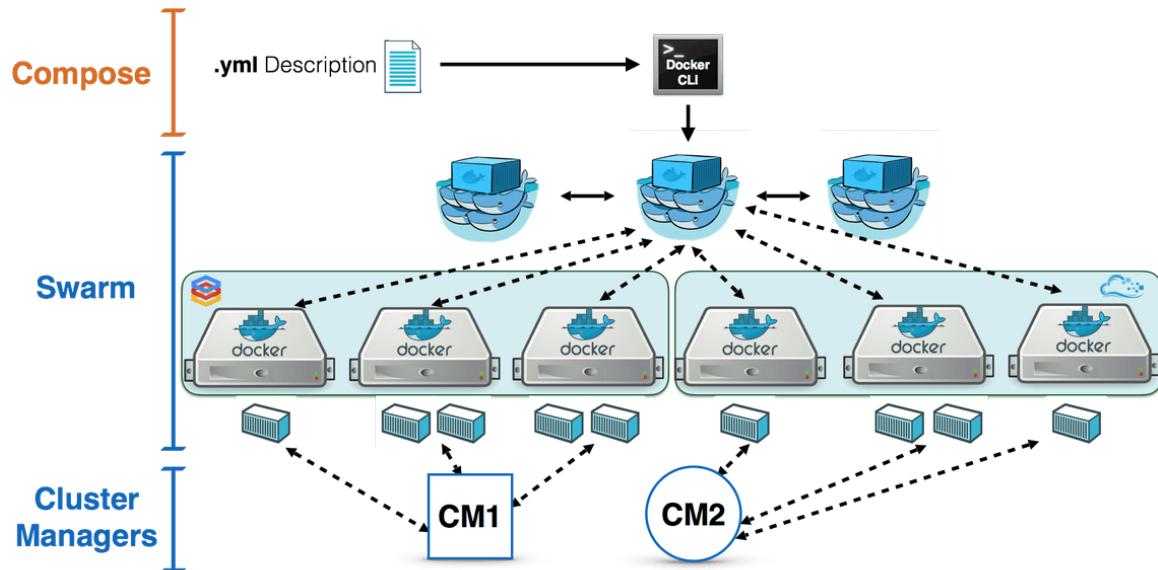
```
http://localhost:5000
http://localhost:5000/collections
http://localhost:5000/collections/lakes/items?limit=2
http://localhost:5000/collections/lakes/items?f=json
```

DOCKER - NEXT

- Custom config via Docker Volume Mapping
- Run with Docker Compose
- Subpath running via Flask "SCRIPT_NAME" env var
- Examples in **pygeoapi GitHub**



- Dockerhub image without Elasticsearch
- Docker-compose for Elasticsearch in github
- Easy to scale pygeoapi with docker swarm



- Cloud clustering with high availability

VIII. PYGEOAPI IN PRODUCTION

(Some) Production instances of pygeoapi



Contact: [**@tomkralidis**](#)

The screenshot shows the official website of the Canadian Government. At the top, there is a navigation bar with links for "Jobs", "Immigration", "Travel", "Business", "Benefits", "Health", "Taxes", and "More services". A search bar is located on the right side of the navigation bar. The main content area features a heading "National Water Data Archive: HYDAT" and a sub-section titled "National Water Data Archive". Below this, there is a paragraph of text describing the data collection and compilation process, mentioning HYDEX and HYDAT databases. The URL "http://bit.do/hydat" is overlaid on the left side of the main content area.

Government of Canada Gouvernement du Canada

Search Canada.ca

French

Jobs ▾ Immigration ▾ Travel ▾ Business ▾ Benefits ▾ Health ▾ Taxes ▾ More services ▾

Home → Environment and natural resources → Natural resources → Water and the environment → Water quantity → Water Survey of Canada → Water survey data products and services

National Water Data Archive: HYDAT

National Water Data Archive

Hydrometric data are collected and compiled by Water Survey of Canada's eight regional offices. The information is housed in two centrally-managed databases: HYDEX and HYDAT.

HYDEX is the relational database that contains inventory information on the various streamflow, water level, and sediment stations (both active and discontinued) in Canada. This database contains information about the stations themselves such as; location, equipment, and type(s) of data collected.

http://bit.do/hydat

Canadian National Water Data Archive

numberMatched: 60 770 985

```
http://geo.weather.gc.ca/geomet-beta/features/collections/  
hydrometric-daily-mean/items?resulttype=hits
```

100 Features

```
http://geo.weather.gc.ca/geomet-beta/features/collections/  
hydrometric-daily-mean/items?limit=100
```

Fetching 1 feature (200ms)

```
http://geo.weather.gc.ca/geomet-beta/features/collections/  
hydrometric-daily-mean/items/10SB001.1992-01-11
```

IX. STAC

SPATIOTEMPORAL ASSET CATALOG (STAC)

- Simple catalog of spatiotemporal items (JSON)
- (stac != csw || csw => stac)
- More oriented for sat imagery
- Parallel work to WFS 3.0
- Pull from storages like S3
- Geospatial assets openly searchable and crawlable

The screenshot shows the GitHub repository page for `radiantearth/stac-spec`. The repository name is at the top left, followed by navigation links for Watch (35), Star (71), Fork (27), and Insights. Below this is a header bar with tabs for Code, Issues (18), Pull requests (2), Projects (0), and Insights. A sub-header below the tabs reads "SpatioTemporal Asset Catalog specification - making geospatial assets openly searchable and crawlable" and provides a link to <https://gitter.im/SpatioTemporal-Asse...>. Below the header are several buttons: satellites, satellite-imagery, earth-observation, and earth-observation-catalogue. A summary bar shows 342 commits, 5 branches, 3 releases, 18 contributors, and Apache-2.0 license. A dropdown for the branch is set to master, and there is a "New pull request" button. The main content area displays a list of recent commits:

Author	Commit Message	Time Ago
cholmes	Merge branch 'dev' to align master to 0.5.0 release	Latest commit 75d6739 5 days ago
api-spec	Merge pull request #104 from radiantearth/transaction	25 days ago
extensions	updated examples to adjust to new prefix	7 days ago
json-spec	Merge branch 'dev' to align master to 0.5.0 release	5 days ago
lonos	Added nnn lonos	a month ago

<https://github.com/radiantearth/stac-spec>

LANDSAT8 - STAC EXAMPLE

```
{  
  "id": "LC81530252014153LGN00",  
  "type": "Feature",  
  "bbox": [ 49.16354,72.27502,51.36812,75.67662  
    ],  
  "geometry": {  
    "type": "MultiPolygon",  
  "coordinates": [...]}
```

LANDSAT8 - STAC EXAMPLE

```
"properties": {  
    "datetime": "2014-06-02T09:22:02Z",  
    "provider": "USGS",  
    "eo:gsd": 30.0,  
    "eo:cloud_cover": 10,  
    "eo:off_nadir": 0.000,  
    "eo:azimuth": 0.00,  
    "eo:sun_azimuth": 149.01607154,  
    "eo:sun_elevation": 59.21424700,  
}
```

LANDSAT8 - STAC EXAMPLE

```
"assets" :{
    "thumbnail": {
        "href": "http://landsat[...]/LC81530252014153LGN00_thumb_large.jpg",
        "required": true,
        "type": "jpeg"
    },
    "metadata": {
        "href": "http://landsat[...]/LC81530252014153LGN00_MTL.txt",
        "required": true,
        "type": "mtl"
    }
},
```

X. PYGEOAPI SUPPORT



Need support for WFS3.0???

@Geocat can help you





justobjects.nl