API4INSPIRE

Sylvain Grellet, Abdelfettah Feliachi

Welcome and some hints for participants

Mute your mic!
To mute and unmute, click the microphone icon next to your name or at the bottom of the screen.
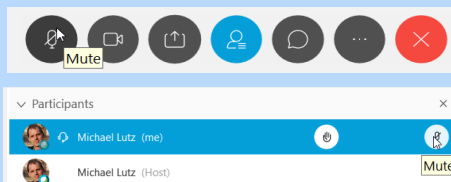
Turn off video
Share your webcam video **only** when you are talking. To do this, click video icon next to your name.

Ask a question
Use "raise hand" functionality to ask a question. Click the hand icon next to your name in the participant list. If this is not available write 'hand' in the chat.
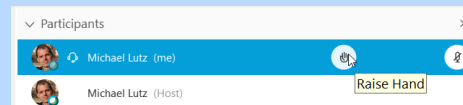
API4INSPIRE

Overview

- API4INSPIRE
  - Why APIs in INSPIRE?
  - Project details
- OGC API - Features
  - GeoServer Extension
  - OGC API - Simple
- OGC SensorThings API
  - FROST
- Data Nests
  - Airy Austria
  - Urban Data Platform Hamburg
  - Franco-Germanic Flow
  - Covid-19
- Use!

Why APIs in INSPIRE?

- ## State-of-play
  - Most of the services in the INSPIRE geoportal are based on W*S/OWS services
  - Huge interest in APIs within the INSPIRE community
  - Few known implementations of APIs in INSPIRE
    - different approaches
      - encodings, documentation, standards
    - costs and benefits not very well known
- ## European agenda
  - European Strategy for Data
    - Sector-specific data spaces
    - Connecting different actors
  - Open Data Directive
    - High-Value Datasets (harmonised, documented and available through APIs)
  - INSPIRE MIWP 2020 - 2024
    - Standard-based APIs are one of the means for modernising INSPIRE's technological stack

Why APIs in INSPIRE?

APIs provide an excellent opportunity for INSPIRE

1) Increase the use of the infrastructure
   - INSPIRE → mainstream ICT
   - Improve the discoverability through search
     - DWBP and SDWBP
       - *DWBP Best Practice 23: Make data available through an API*
       - *SDWBP Best Practice 12: Expose spatial data through 'convenience APIs'*

2) Leverage on grassroots standardisation
   - Novel approaches at the OGC (OGC API-Features and SensorThingsAPI)
     - Hackathons
     - Multiple early implementations
     - Co-creation of specifications

API4INSPIRE

- Implemented under the ISA^2 ELISE Action
  - European Location Interoperability Solutions for e-Government
- Based on demand
  - Requested by MS at the ISA^2 working group on geospatial solutions
- Novel approach
  - Providers on board
  - Learning from hands-on experiences
- Tasks
  - Evaluation Methodology
    - Benefits & Efforts
  - Deployment Strategies for
    - OGC API - Features
    - OGC SensorThings API
  - Deployment and API endpoints
  - Guidelines / technologies, lessons learned
  - Provide evidence for INSPIRE Good Practices with these APIs

API4INSPIRE - Objectives of the workshop

In line with our project objectives of promoting the (future) use of APIs in INSPIRE via good practices, this workshop will:

- Explain and demonstrate OGC API - Features and SensorThings API
- Provide information on endpoints made available by this project
- Collect feedback on strengths and weaknesses of the API approach identified by:
  - Data providers
  - Users
- Identify MS willing to participate in establishing the INSPIRE Good Practices

API4INSPIRE - Project Partners

- Fraunhofer IOSB: developers of FROST server, experts on SensorThings API
- DataCove e.U.: INSPIRE expertise
- GeoSolutions: developers of GeoServer, experts on OGC API

API4INSPIRE - Data Providers

- Austrian Meteorological Agency (ZAMG) [AT]
- Austro Control (ACG) [AT]
- Austrian Environment Agency (UBA) [AT]
- European Environment Agency (EEA) [EU]
- City of Hamburg (CH) [DE]
- French Geological Survey (BRGM) [FR]
- Office for Biodiversity (OFB) + "INSIDE" - environmental information systems research center (BRGM+OFB) [FR]
- Environment Agency Baden-Württemberg (LUBW) [DE]

OGC APIs

- OGC is defining a new set of services to replace the currently used OWS, that date back 20 years.

- These new services are collectively identified as OGC APIs.

- They are Web APIs, sometimes referred to as **RESTful services**.

- Each service is described by an **OpenAPI document**.

- Each of them is geared towards JSON based representations of resources:

  - Both other encodings are supported, **e.g. HTML**.

- Each service has a minimal core, and **numerous optional extensions to add functionality**.

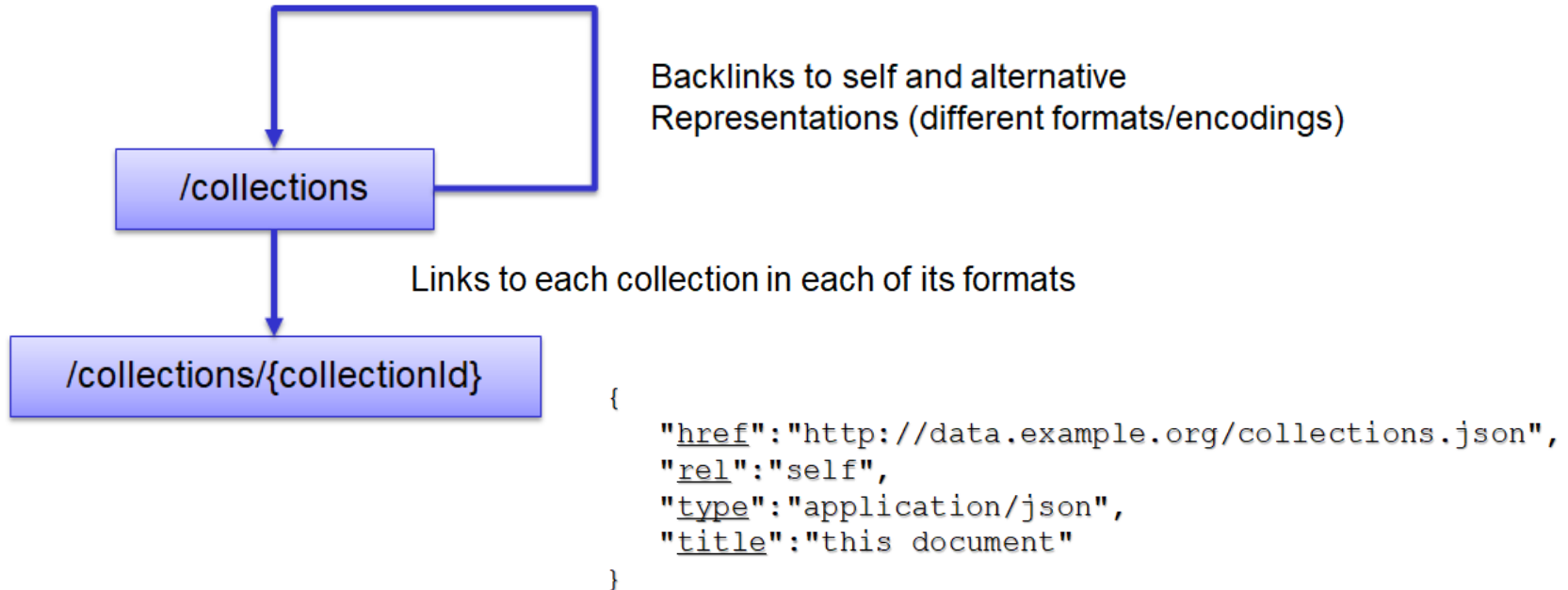**Service Capabilities**

FEATURES
1.0

IMAGES
1.0

STYLES
1.0

TILES
1.0

OGC API - Common

- Can be found at: https://github.com/opengeospatial/oapi_common
- **Resources**:
  - / (landing page)
  - /conformance (conformance classes implemented)
  - /api (API definition)
  - **/collections** (list of collections published)
  - **/collection/{collectionId}** (metadata of the single collection
- Encodings:
  - HTML, **which is browsable!**
  - **JSON/GeoJSON**

Links, links everywhere!

- All resources are linked to the others, in the various encodings:

Backlinks to self and alternative
Representations (different formats/encodings)

/collections

Links to each collection in each of its formats

/collections/{collectionId}

```
{
    "href":"http://data.example.org/collections.json",
    "rel":"self",
    "type":"application/json",
    "title":"this document"
}
```

Very little is mandatory (and that's good!)

- **None of the encodings are mandatory**, a server could do XML or protocol buffers and still be compliant.

- If present, the usage of OpenAPI is suggested but not mandatory!

- A client could work by following links between resources, **that's the way of the ~~modern~~ Web!**

- How does a client work then? By checking the conformance declaration at "/conformance":

```
{
  "conformsTo": [
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/core",
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/oas30",
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/html",
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/geojson"
  ]
}
```

OGC API - Features

- Can be found at: https://github.com/opengeospatial/ogcapi-features

- In addition to OGC API (OAPI) core:

  - **/collection/{collectionId}/items (features)**

  - **/collection/{collectionId}/items/{itemId}**

- Only supported CRS are:

  - CRS84 (WGS84 lon/lat)

  - CRS84h (WGS84 lon/lat/height)

- No mandated schema, features can be anything:

  - Simple (SF-0)
  - Complex (SF-1, SF-2e)

Extensions! Some are already <u>in the making</u>:

- cql
- crs
- transactions
- README.md

OGC API – Features

GeoServer

GeoServer Feature Collections

This document lists all the collections available in the Features service.
This document is also available as application/x-yaml, application/json.

eposb:Borehole

- **Title:** Borehole
- **Geographic extents:**
  - 41.325, -5.15, 51.118, 9.624.
- Data as HTML. Collection items are also available in the following formats: --Please choose an option--
- Queryables as HTML.

gsmlp:BoreholeView

- **Title:** BoreholeView
- **Geographic extents:**
  - -64.656, -21.388, 139.535, 87.933.
- Data as HTML. Collection items are also available in the following formats: --Please choose an option--
- Queryables as HTML.

```
{
  type: "FeatureCollection",
  features: [
    {
      type: "Feature",
      id: "0001000001",
      geometry: {
        type: "Point",
        coordinates: [
          ███████,
          ███████
        ]
      },
      properties: {
        "@featureType": "Borehole",
        identifier: {
          value: "BSS000AAAA",
          "@codeSpace": "http://www.ietf.org/rfc/rfc2616"
        },
        type: {
          "@href": "http://vocabulary.odm2.org/samplingfeaturetype/borehole/",
          "@title": "borehole"
        },
        sampledFeature: [
          {
            "@href": "https://sweet.jpl.nasa.gov/2.3/realmEarthReference.owl#EarthLithosphere",
            "@title": "Lithosphere"
          }
        ],
        shape: {
          "@dataType": "shape",
          Point: {
            type: "Point",
            coordinates: [
              ███████,
              ███████
            ]
          }
        },
        custodian: {
          "@dataType": "Custodian",
          custodian: {
            "@dataType": "CI_ResponsibleParty",
            organisationName: "BRGM",
            role: {
              "@codeListValue": "custodian",
              "@codeList": "http://www.isotc211.org/2005/resources/Codelist/gmxCodelists.xml"
            }
          }
        }
```
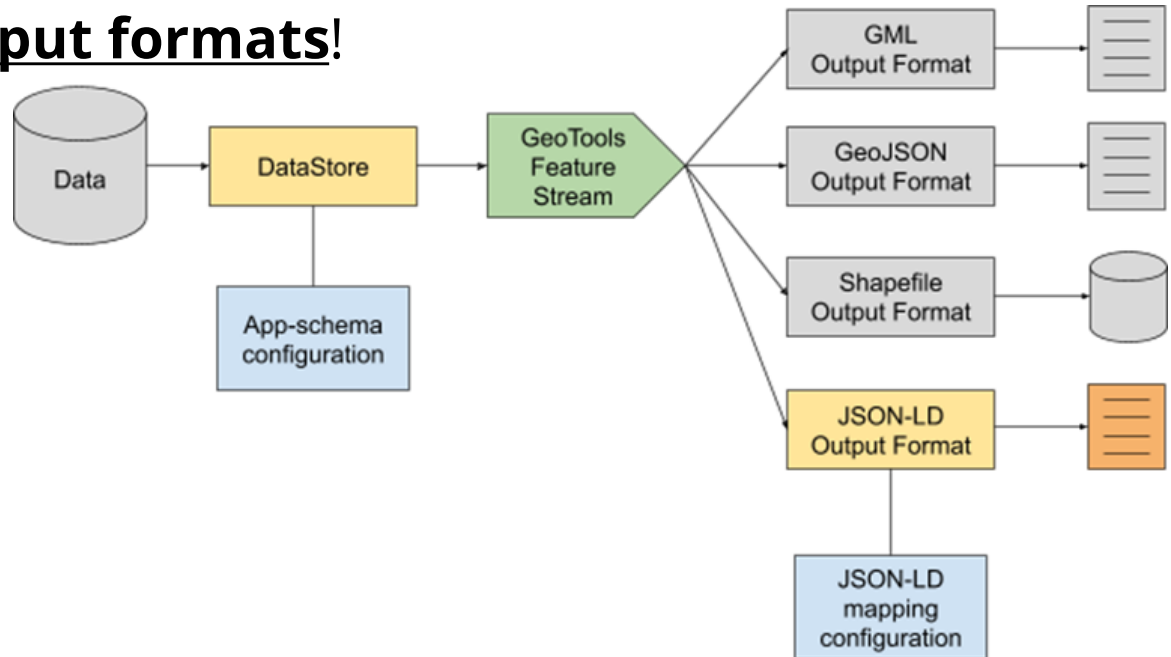
GeoServer Extensions
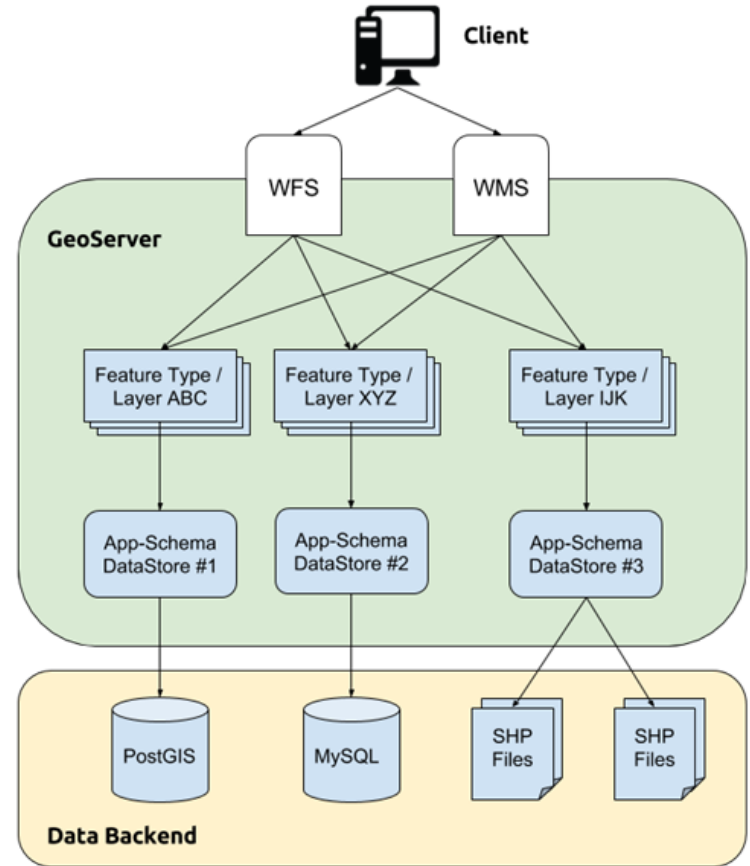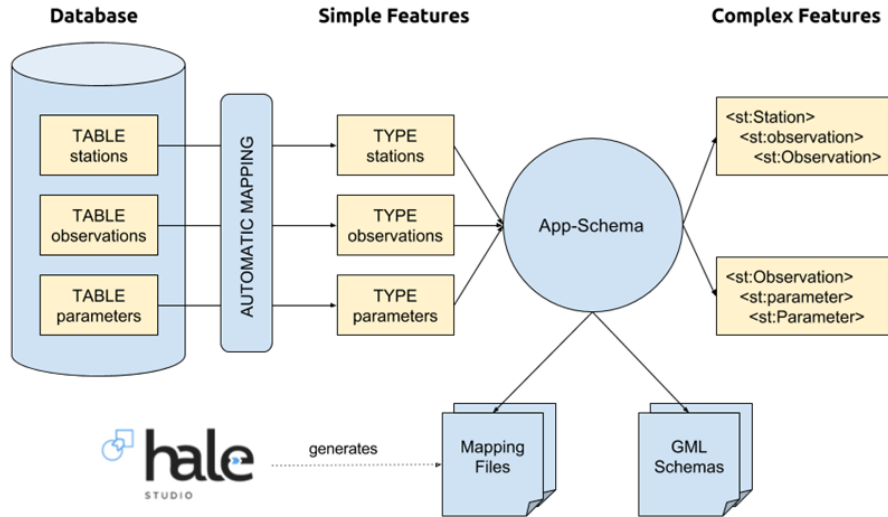
GeoServer Extensions

- One mapping between the data and a target schema
    ➡️ **Multiple output formats**!

How to get rid of the mapping between **the data** and **target schema**?

GeoServer Extensions

Why not use
templating
instead of
mappings?

↓

**What You
See Is What
You Get!**

```
"geometry": {
        "@type": "Point",
        "wkt":
"$${strConcat('<http://www.opengis.net/def/crs/OGC/1.3/CRS84>',toWKT(xpath('eposb:bholeHe
adworks/gwml-wellconstruction:BoreCollar/gwml-wellconstruction:collarLocation')))}"
        },
        "sam:sampledFeature": {
          "@id": "https://sweet.jpl.nasa.gov/2.3/realmEarthReference.owl#EarthLithosphere",
          "name": "Lithosphere"
        },
```

```
"eposb:lifeCycleInfo": {
        "$source": "eposb:lifeCycleInfo/eposb:LifeCycleInfo",
        "@type": "LifeCycleInfo",
        "eposb:updateDate": {
          "@type": "time:Instant",
          "time:inXSDDateTime": "${eposb:updateDate/gml:TimeInstant/gml:timePosition}"
        },
        "status": {
          "@id": "${eposb:status/@xlink:href}",
          "name": "Validé"
        },
        "eposb:creationDate": {
          "@type": "time:Instant",
          "time:inXSDDateTime": "${eposb:creationDate/gml:TimeInstant/gml:timePosition}"
        }
      },
      "eposb:locatedOnAdminUnit": {
        "@id": "${eposb:locatedOnAdminUnit/@xlink:href}",
        "name": "SANGATTE"
      },
```

GeoServer Extensions

```
▼ geometry: {
    "@type": "Point",
    wkt: "<http://www.opengis.net/def/crs/OGC/1.3/CRS84>POINT (1.35559927237556 51.1020179771059)"
},
▼ "sam:sampledFeature": {
    "@id": https://sweet.jpl.nasa.gov/2.3/realmEarthReference.owl#EarthLithosphere,
    name: "Lithosphere"
},
```

Filtering using CQL is
supported:

http://.../geoserver/ogc/features/coll
ections/eposb:Borehole/items?f=ap
plication/ld+json&limit=50&filter=**fea
tures.gsmlp:boreholeLength_m.o
m:amount>83**&filter-lang=cql-text

```
▼ "eposb:lifeCycleInfo": {
    "@type": "LifeCycleInfo",
    ▼ "eposb:updateDate": {
        "@type": "time:Instant",
        "time:inXSDDateTime": "2008/12/357 05:41:34"
    },
    ▼ status: {
        "@id": http://id.eaufrance.fr/nsa/390#XXX,
        name: "Validé"
    },
    ▼ "eposb:creationDate": {
        "@type": "time:Instant",
        "time:inXSDDateTime": "1998/03/70 11:55:17"
    }
},
▼ "eposb:locatedOnAdminUnit": {
    "@id": "6262774",
    name: "SANGATTE"
},
```
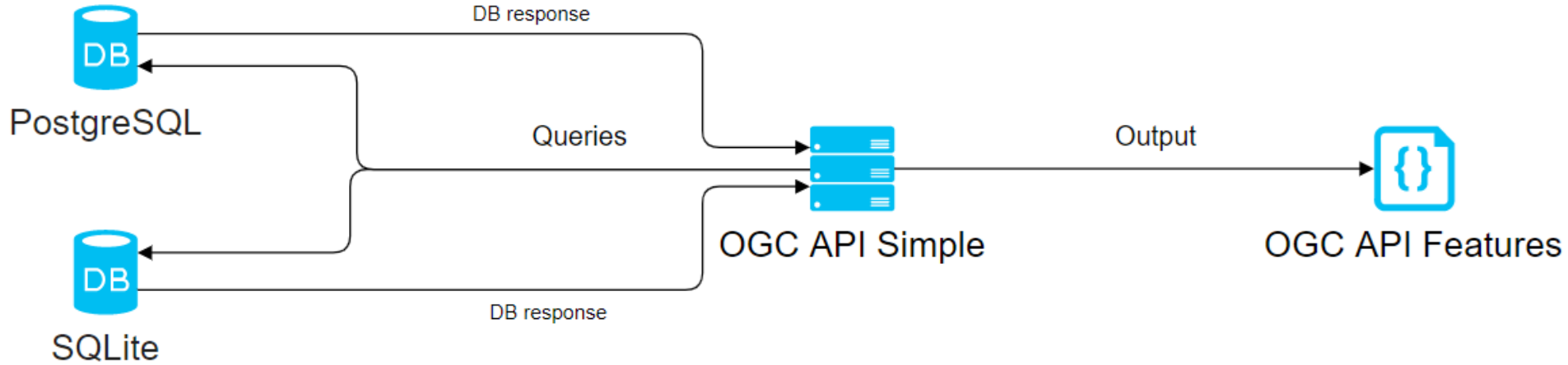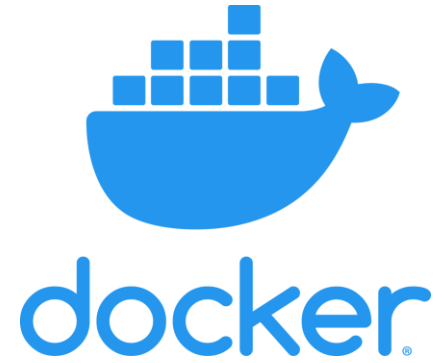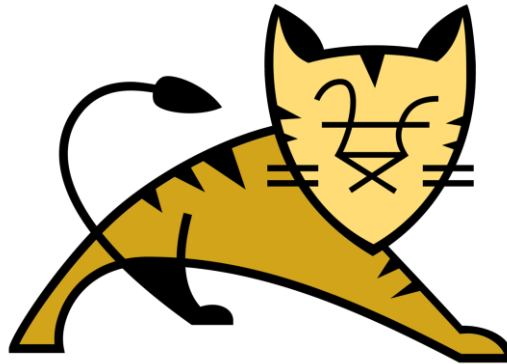
GeoServer Extensions
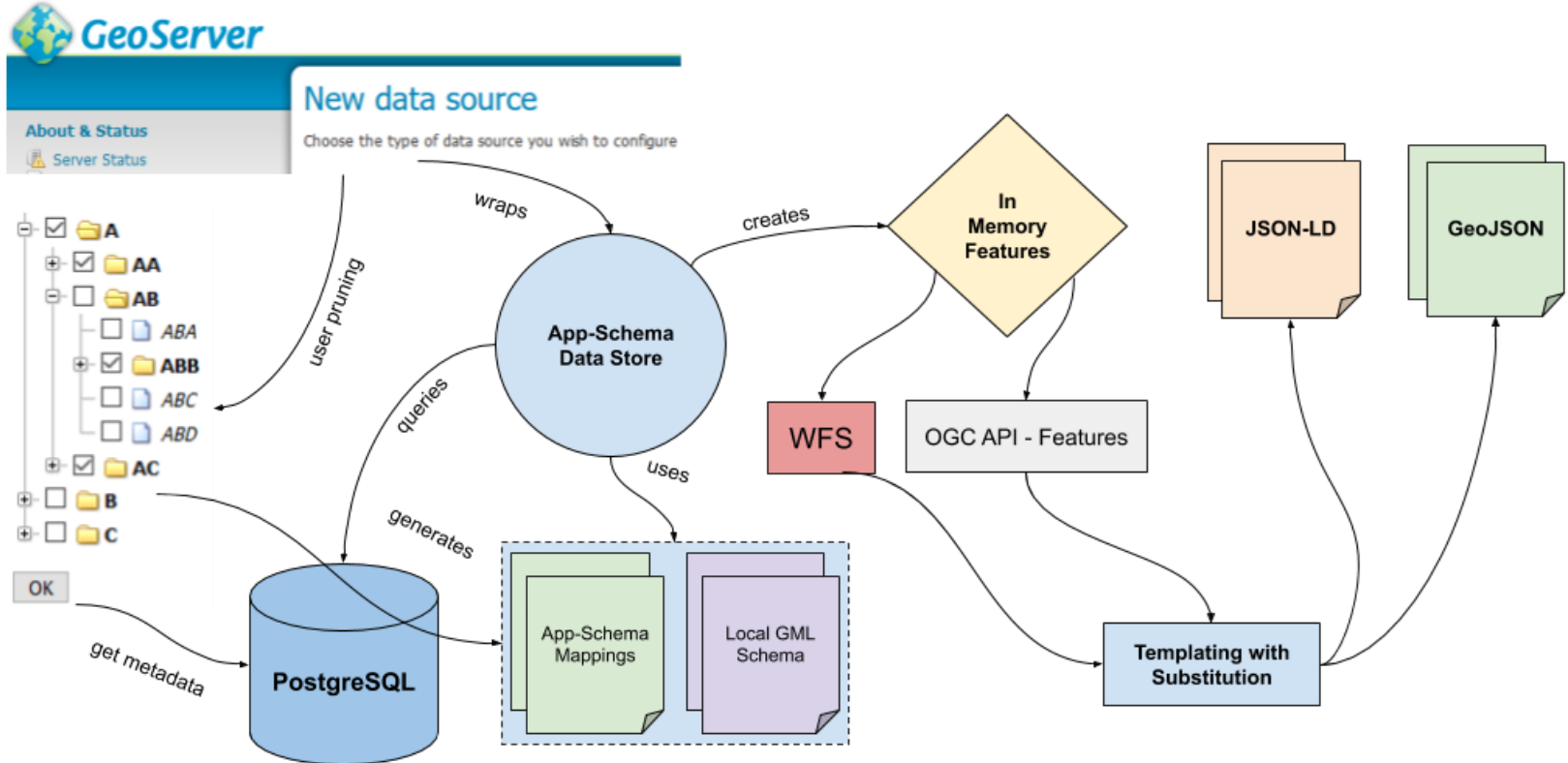
OGC API - Simple

- Who are we?
- Order:
  - implement the OGC API for simple features
  - according to the OGC API Core specification from October 2019
- Tasks:
  - REST API
  - JSON + GeoJSON
  - Admin Panel

OGC API - Simple

OGC API - Simple

OGC API Simple

Home

Login

OGC API Simple

Conformance declaration	Feature Collections

API definition

The API definition provides a machine processable description of this service API conformant to OpenAPI 3.

It is available as application/json and text/html
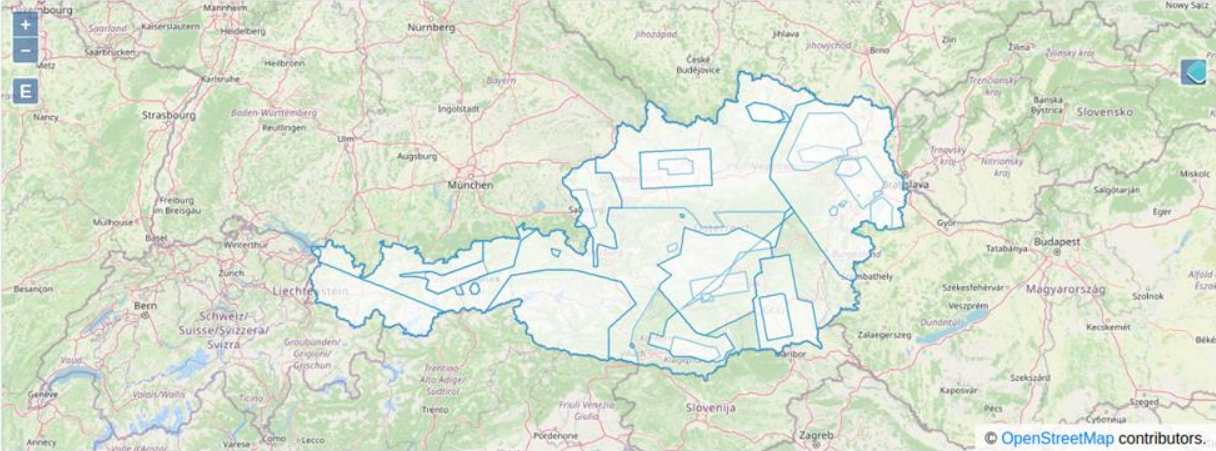
INSPIRE

With supporting text below as a natural lead-in to additional content.

INSPIRE Website

What is the OGC API?

The OGC API is a REST API in which environmental data is provided.

OGC API Specifications

#	Collection	Show in map
2	insp_airspacearea	☑
4	insp_designatedpoint	☐
5	insp_runwayarea	☐

© OpenStreetMap contributors.

OGC API Simple

localhost:8080/ogcapisimple/dashboard

/api   Home   Options ▾   Logout

Dashboard

INSP

Connector properties

Connector management

#		Exclude	Collection name
1	insp_aerodrometype	☑	insp_aerodrometype
2	insp_airspacearea	☑	insp_airspacearea
3	insp_conditionoffairfacility	☑	insp_conditionoffairfacility
4	insp_designatedpoint	☑	insp_designatedpoint
5	insp_runwayarea	☑	insp_runwayarea
6	insp_tn_air_acg_features	☑	insp_tn_air_acg_features

Change admin password

#	Column name	Exclude	Property name
1	statename	☐	
2	geodatenstelle	☐	
3	ds_md_fileidentifier	☐	
4	annex_theme	☐	
5	featuretype	☐	
6	localid	☐	

New collection name

New collection name

Assign collection name

New property name

New column name

Assign name

Write SQL

Collection name

Write and execute sql here

OGC API Simple

localhost:8080/ogcapisimple/dashboard

/api   Home   Options ▼   Logout

INSP

#	Table name	Exclude	Collection name
1	insp_aerodrometype	☐	insp_aerodrometype
2	insp_airspacearea	☐	insp_airspacearea
3	insp_conditionofairfacility	☐	insp_conditionofairfacility
4	insp_designatedpoint	☐	insp_designatedpoint
5	insp_runwayarea	☐	insp_runwayarea
6	insp_tn_air_acg_features	☐	insp_tn_air_acg_features

#	Column name	Exclude	Property name
1	statename	☐	
2	geodatenstelle	☐	
3	ds_md_fileidentifier	☐	
4	annex_theme	☐	
5	featuretype	☐	
6	localid	☐	

New collection name

New collection name

Assign collection name

New property name

New column name

Assign name

Write SQL

Collection name

Write and execute sql here

/api   Home   Options ▾   Logout

Feature Collections

- This document as JSON

1: insp_aerodrometype

Links:

- The feature collection as JSON
- The items as JSON
- The items as HTML

2: insp_airspacearea

Links:

- The feature collection as JSON
- The items as JSON
- The items as HTML

3: insp_conditionofairfacility

Links:

- The feature collection as JSON
- The items as JSON
- The items as HTML

4: insp_designatedpoint

Links:

- The feature collection as JSON
- The items as JSON
- The items as HTML

5: insp_runwayarea

Links:

- The feature collection as JSON
- The items as JSON
- The items as HTML

6: insp_tn_air_acg_features

JSON Raw Data Headers

Save Copy Collapse All Expand All ▼ Filter JSON

```
▼ collections:
  ▼ 0:
      id:          "insp_aerodrometype"
    ▼ links:
      ▼ 0:
          href:    "http://localhost:8080/ogcapisimple/collections/insp_aerodrometype"
          rel:     "self"
          type:    "application/json"
          title:   "this document"
      ▼ 1:
          href:    "http://localhost:8080/ogcapisimple/collections/insp_aerodrometype"
          rel:     "alternate"
          type:    "text/html"
          title:   "this document as html"
      ▼ 2:
          href:    "http://localhost:8080/ogcapisimple/collections/insp_aerodrometype/items"
          rel:     "items"
          type:    "application/json"
          title:   "this document as html"
  ▼ 1:
    ▼ extent:
      ▼ spatial:
        ▼ bbox:
          ▼ 0:
              0:   9.530749083
              1:   46.372300139
              2:   17.160778722
              3:   49.020714833
      id:          "insp_airspacearea"
    ▼ links:
      ▼ 0:
          href:    "http://localhost:8080/ogcapisimple/collections/insp_airspacearea"
          rel:     "self"
          type:    "application/json"
          title:   "this document"
      ▼ 1:
          href:    "http://localhost:8080/ogcapisimple/collections/insp_airspacearea"
          rel:     "alternate"
          type:    "text/html"
          title:   "this document as html"
      ▼ 2:
          href:    "http://localhost:8080/ogcapisimple/collections/insp_airspacearea/items"
          rel:     "items"
          type:    "application/json"
          title:   "this document as html"
  ▼ 2:
      id:          "insp_conditionofairfacility"
```

OGC SensorThings API

The successor to SOS

- REST + JSON + Full Editing
- Sensible Data Model
  - Extendible properties
- Powerful Queries
  - Across the entire data model
  - Composable response data
- Scalable
  - Thousands of stations
  - Millions of observations
- Understandable
  - Follow the links to *all* data
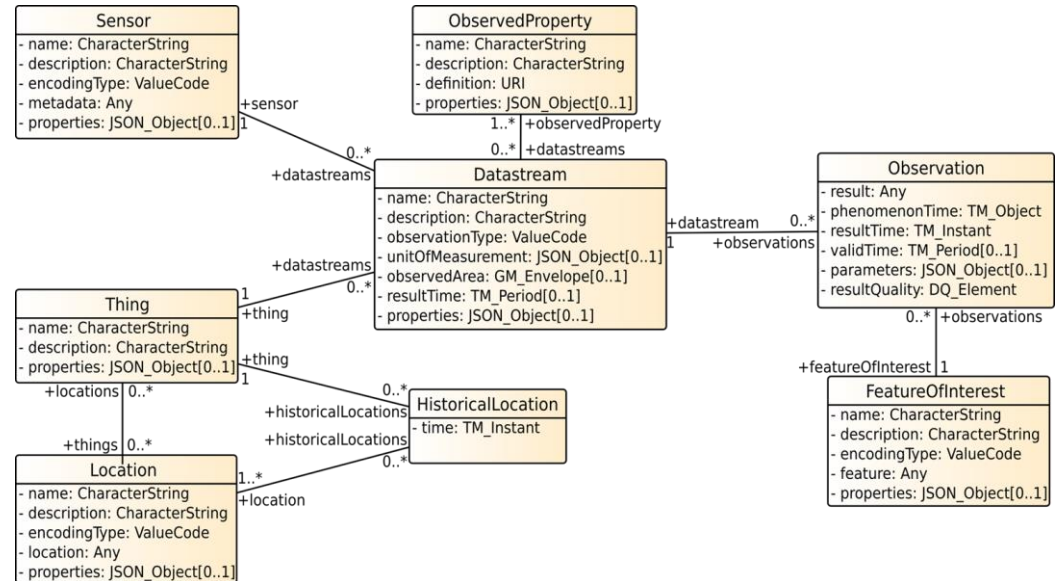
OGC STA – GETting data 1

- Index document describing the server and its extensions
        https://server.de/FROST-Server/v1.1
- The list of "Things" (or Sensors, Locations, Observations, etc...)
        v1.1/Things
- A single Thing
        v1.1/Things(42)
- The list of Datastreams for Thing 42
        v1.1/Things(42)/Datastreams
- The Thing for Datastream 123
        v1.1/Datastreams(123)/Thing
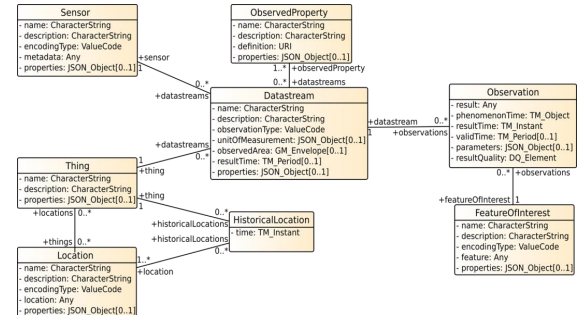
OGC STA - GETting data 2

- On every list:
  - Pagination with $top and $skip
    - v1.1/Things?$top=100&$skip=200
  - Selecting which fields to return with $select
    - v1.1/Things?$select=id,name
  - Filtering the entities with $filter
    - v1.1/Observations?$filter=result gt 5
  - Requesting the total number (given the filter)
    - v1.1/Things?$count=true
  - Adding related entities with $expand
    - v1.1/Things?$expand=Datastreams($expand=Observations)
    - Can be nested & combined with the other options
- More on https://datacoveeu.github.io/API4INSPIRE/

OGC STA - Editing data

- Create: POST on an Entity Set
  - POST v1.1/Things
- Update: PATCH or PUT on an Entity
  - PATCH v1.1/Things(42)
- Delete: DELETE on an Entity
  - DELETE v1.1/Things(42)
  - Also deletes dependent entities
    - Things → Datastreams
    - Datastreams → Observations
    - Things → HistoricalLocations

FROST-Server

A complete, feature-rich implementation

- Version 1.0 & 1.1
- Extensive querying capabilities
- Extensions for CSV & GeoJSON output

- Open Source (LGPL)
  https://github.com/FraunhoferIOSB/FROST-Server
- Java / Tomcat & Docker
- Horizontal scalability

FROST-Server Database

- **FROST-Managed**
  - Best when no database exists yet
  - FROST creates and maintains the database
  - All updates through the SensorThings API
- **Using Database Views**
  - existing PostgreSQL database with data
  - database views to mimic the FROST tables
  - updates directly in the database

FROST-Server Demo Servers

- Free-For-All Demo server (Read & Write)
  https://ogctest.docker01.ilt-dmz.iosb.fraunhofer.de/v1.1/
- European Air Quality Data
  https://airquality-frost.docker01.ilt-dmz.iosb.fraunhofer.de/v1.1
  - 4382 Stations
  - 17960 Datastreams
  - 260 million Observations
  - Viewer: https://api4inspire.docker01.ilt-dmz.iosb.fraunhofer.de/servlet/is/113/
- European Demography Data
  http://service.datacove.eu/DemographyThings/v1.1
  - Total population and population density
  - NUTS levels 0 to 3

More at https://datacoveeu.github.io/API4INSPIRE/

Data Nests

Sets of colocated and complementary data sources exposed by the APIs under evaluation:

- **Airy Austria:** Air Transport information complemented by meteorological data
- **Urban Data Platform Hamburg:** Smart City Sensors together with road transport networks
- **Franco-Germanic Flow:** Cross-border water data, surface and ground water, quality and quantity, flood zones
- **Covid ad-hoc:** Realtime air quality, Covid-19 case data complemented by a background demography layer

Airy Austria

Air Transport information complemented by meteorological data. This nest consists of the following data providers and end points:

- **Austro Control** is the air navigation services provider that controls Austrian airspace
  WFS2: https://sdigeo-free.austrocontrol.at/geoserver/tn-a/wfs?service=WFS&version=2.0.0&request=GetCapabilities
  OGC API: https://inspire.austrocontrol.at/ogcapi/ogc/features
- **Austrian Meteorological Agency (ZAMG)** has a wide range of expertise at its disposal pertaining to all aspects of meteorological data management and provision

Urban Data Platform Hamburg

The **City of Hamburg** has long seen the potential of Smart City technology, and been involved in diverse Smart City initiatives, successively extending their smart sensor infrastructure to an ever widening usage area.

Endpoint: https://iot.hamburg.de/v1.0

- Charging stations for electric cars
- Bike sharing stations from StadtRad
- Data from the Energy Campus of the Hamburg University of Applied Sciences

Future plans: A Lot!

- Traffic lights, traffic density, etc...

Franco-Germanic Flow

- The French Geological Survey (**BRGM**) has long been involved in pushing the envelope pertaining to the possibilities of environmental data provision.
- Along with the French Office for Biodiversity (**OFB**) and their joint research center on information system (**INSIDE**), they provide access to (linked) datasets from various French Information Systems on Water, Underground Risk using
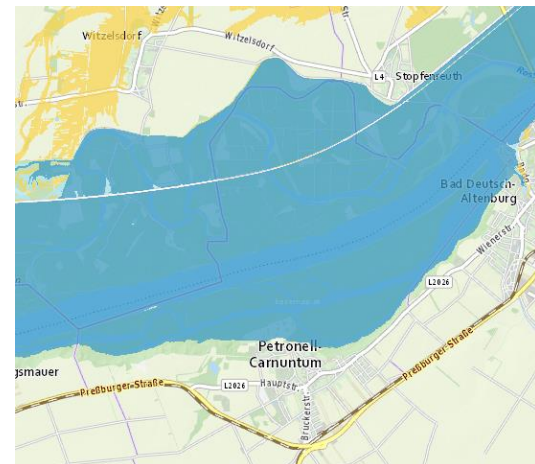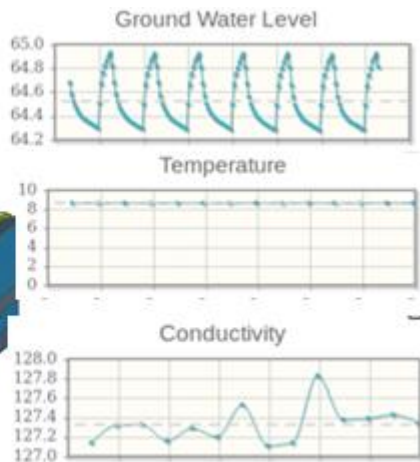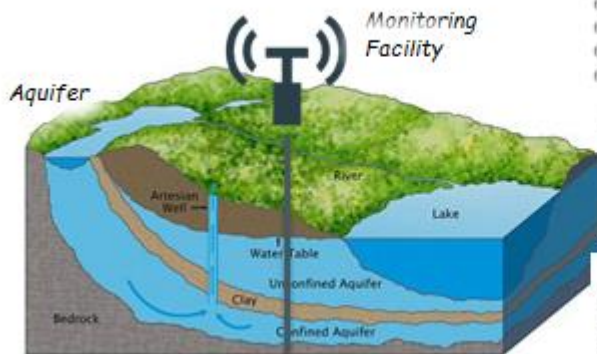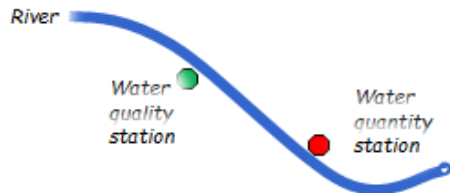

- The Environment Agency Baden-Württemberg – **LUBW** provides diverse water resources within Germany.
  STA: https://lubw-frost.docker01.ilt-dmz.iosb.fraunhofer.de/v1.1
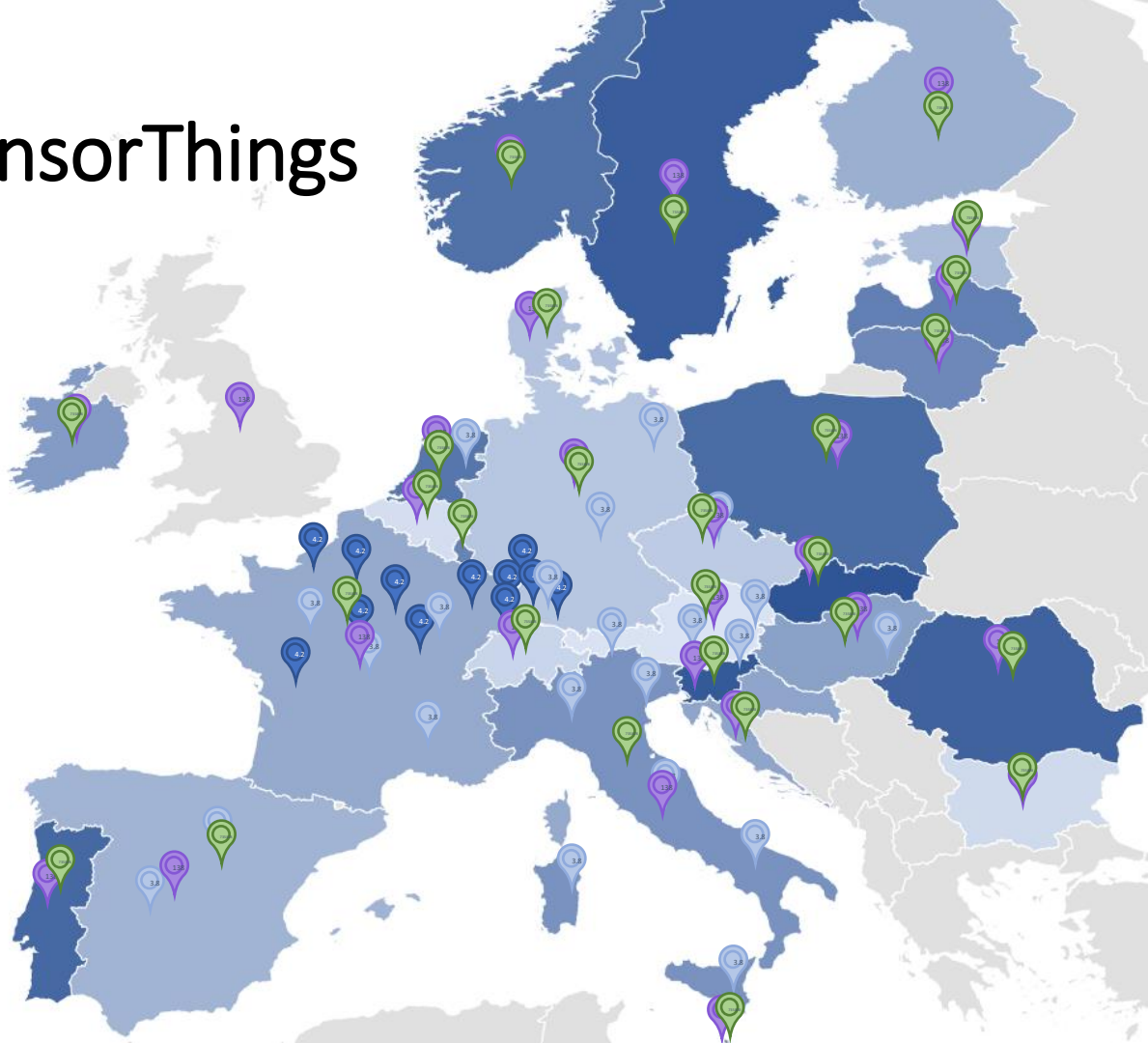
Franco-Germanic Flow

API4INSPIRE - SensorThings

- HydroThings
- AirThings
- CovidThings
- DemographyThings

Ad-Hoc Flows

AirThings

- Data from Umweltbundesamt (AT) and EEA (EU)
- https://airquality-frost.docker01.ilt-dmz.iosb.fraunhofer.de/v1.1
- https://api4inspire.docker01.ilt-dmz.iosb.fraunhofer.de/servlet/is/113/

CovidThings

- http://covidsta.hft-stuttgart.de/server/

DemographyThings

- Data from Eurostat DB
- http://service.datacove.eu/DemographyThings/v1.1

USE!

https://github.com/DataCoveEU/API4INSPIRE

Thanks

In order of appearance
- Alexander Kotsev – JRC – Alexander.KOTSEV@ec.europa.eu
- Kathi Schleidt – DataCove e.U. – Kathi@DataCove.eu
- Sylvain Grellet – BRGM – S.Grellet@brgm.fr
- Nuno Oliveira – GeoSolutions – nuno.oliveira@geo-solutions.it
- Lukas Gäbler – School – lukas.gaebler@gmail.com
- Hylke van der Schaaf – Fraunhofer IOSB - hylke.vanderschaaf@iosb.fraunhofer.de
- Marco Minghini – JRC – Marco.MINGHINI@ec.europa.eu

Conclusions?

We do not provide conclusions!

Further discussion: https://inspire.ec.europa.eu/forum/groups/profile/215867

Further Info: https://datacoveeu.github.io/API4INSPIRE/

Further Events: TBD

- OGC API - GeoServer
- STA - FROST
- OGC API - Simple?
- Usage of APIs