

Data Catalog Vocabulary (DCAT)



Open Educational Resources for Spatial Data Infrastructures

In this tutorial, you will learn how important standardized metadata models and formats are for sharing data. You will learn about DCAT (Data Catalog Vocabulary), the concepts on which it is based and how it is used in practice.

James Ondieki, Albert Remke
Institute for Geoinformatics, University of Münster

August 2024

ein Kooperationsprojekt,
empfohlen durch:



gefördert durch:

Ministry of Culture and Science
of the State of
North Rhine-Westphalia



1. Overview.....	2
2. Background.....	3
2.1 Data Sharing and Metadata.....	3
2.2 Resource Description Framework - RDF.....	5
2.3 Data Catalog Vocabulary (DCAT).....	6
2.4 DCAT Profiles.....	8
3. Practical Exercise.....	10
3.1 Practical Example of DCAT Metadata.....	10
3.2 Creating, Editing and Visualizing DCAT metadata with RDForms.....	14
3.3 Validating DCAT Metadata.....	23
3.4 Running SPARQL Queries on DCAT metadata.....	25
3.4.1 Overview of Technologies.....	25
3.4.2 Workflow.....	26
3.4.3 Exercise.....	34
4. Summary and Discussion.....	35

1. Overview

In this tutorial, you will learn how important standardized metadata models and formats are for sharing data. You will learn about DCAT (Data Catalog Vocabulary), the concepts on which it is based and how it is used in practice.

In our practical exercises we will use:

- RDForms, an online RDF tool for creating DCAT metadata
- Apache Jena Fuseki Server, a triple store that enables the management and querying of DCAT metadata using the SPARQL query language
- Docker to manage our Apache Fuseki runtime environment

The tutorial is structured as follows:

1. Overview
2. Background
 - Data Sharing and Metadata
 - Resource Description Framework (RDF)
 - Data Catalog Vocabulary (DCAT)
 - DCAT Profiles
3. Practical Exercises
 - Practical Example of DCAT

- Creating, Editing and Visualizing DCAT metadata with RDForms
- Running SPARQL Queries on DCAT metadata.

4. Summary and Discussion

This tutorial is designed for students and professionals who want to improve their understanding of Spatial Information Infrastructures, and in particular, how DCAT promotes the reusability and interoperability of spatial data. We assume that you have basic knowledge about the concept of linked data, spatial data and spatial data services (for example Web Map Services (WMS) and Web Feature Services (WFS)). If you don't, you can still proceed with the module as we will not be going into the technical details or usage of the spatial data and data services, but just how they are described using DCAT. You will need about 60-90 minutes to work through this tutorial.

This tutorial has been developed in the context of the OER4SDI Project at the Institute for Geoinformatics, University of Münster. Authors are James Ondieki and Albert Remke.

You are free to use, alter and share the content of the tutorial under the terms of the [CC-BY-SA 4.0](#) license, unless explicitly stated otherwise for specific parts of the content. Any code provided with the tutorial can be used under the terms of the MIT license. Please see the full license terms: <https://github.com/oer4sdi/OER-DCAT/blob/main/LICENSE.md>.

The authors can be mentioned as follows: OER_DCAT (2024), [CC-BY-SA 4.0](#), James Ondieki, Albert Remke, www.uni-muenster.de

The OER4SDI project has been recommended by the Digital University NRW and is funded by the Ministry of Culture and Science NRW.

2. Background

2.1 Data Sharing and Metadata

Sharing of data and data services between different individuals, organizations, research institutions or governments requires the provision of metadata. Metadata refers to any information that describes an item or an information resource. Just like a library catalog card provides data about a book or a label on a can of fruit juice provides metadata about the fruit juice, a metadata description provides metadata about a dataset.

A book's metadata is useful when a reader wants to find out more about the book. When there are many books available, the reader can use a set of library catalog cards, to get an overview of the books available in the library, and search them by category or author. The set of library cards can be seen as aggregated metadata about the books.

The same concept applies to open data or information resources in general. Providers of data and data services create, manage and publish the metadata that accompanies the information resource or service they provide. The metadata can be aggregated so that interested users can get a good overview of existing data and services that can be used for various purposes.

Metadata of information resources can be categorized according to their purpose:

1. Supporting discovery and identification

This is metadata that supports the discovery and identification of information resources. It includes information such as title, abstract, author, keywords, publication date, language, ISBN, DOI, etc.

2. Supporting data understanding

This is metadata that supports a deeper understanding of the data and thus helps to decide on the usability of the data for a specific purpose. It includes information such as the conceptual data model, data elements and relations, reference systems, information on the data quality, links to comprehensive documentation and contacts for further questions.etc.

3. Supporting data access and use

This is metadata that supports data access and use. It includes information such as access points, terms for access and use (license), API definition, encoding, etc.

Library catalog cards have a standardized content structure. For example, they might contain the book title, the author's name, the category and the year in which the book was published. In the same way, descriptions of datasets and services from various providers must also be standardized in order to be published in a data portal. The use of standardized conceptual data models supports the interoperability of metadata.

To address the need for metadata standardization, a number of conceptual data models and encodings have been developed to promote metadata interoperability. These include:

1. ISO 19115

This is an international standard for describing metadata of geographic information. It defines a comprehensive set of metadata elements covering various aspects of geographic information including the identification, the geographical extent, the quality, the spatial and temporal aspects, the content, the spatial reference, distribution, and other properties of digital geographic data and services.

2. Dublin Core

It provides a simple and standardized set of metadata elements for describing digital resources. It includes elements such as title, creator, subject, description, publisher, date, format, and identifier.

3. Data Catalog Vocabulary (DCAT)

DCAT is a W3C recommendation for describing datasets on the web. DCAT can be used to describe datasets and data services in a catalog using standard terms that facilitate the consumption and aggregation of metadata from multiple catalogs.

These conceptual models play a role in facilitating the interoperability of metadata between different systems and applications. The interoperability of metadata helps to ensure that data can be discovered, shared, and understood across different domains.

The focus of this tutorial is on DCAT, a metadata standard that is based on RDF (Resource Description Framework). But before diving into DCAT, let us have a short overview of RDF and how it relates to DCAT.

2.2 Resource Description Framework - RDF

The Resource Description Framework (RDF) is a standard of the World Wide Web Consortium (W3C) for representing information about resources on the web (i.e. metadata).

RDF represents data as triples, consisting of a subject, a predicate, and an object, which together form a statement about resources:

- Subject is a resource being described by the triple.
- Predicate describes the relationship between the subject and the object.
- Object is a resource or a property value that is related to the Subject.

For example:



Fig. 1: An RDF graph containing one triple

The book “Pride and Prejudice” is a famous book authored by “Jane Austen”. As depicted in Figure 1, the book itself is the Subject of the triple statement while the author is the Object. The predicate is “isAuthoredBy” which shows the relationship between the subject and the object.

In this example,

- the subject may be represented by the URI '<http://example.com/books/PrideAndPrejudice>',
- the predicate may be represented by '<http://purl.org/dc/terms/creator>' and the
- object may be represented by '<http://example.com/authors/JaneAusten>'.

Sets of RDF triples form directed graphs. In our book example, you can imagine further triples describing further properties and resources related to the book and to Jane Austen.

RDF uses URIs (Uniform Resource Identifiers) to unambiguously represent the semantics of subjects, predicates and objects. Only for objects it is allowed to use literals, e.g. a text string that represents a title or an abstract. Using URIs ensures that data can be precisely referenced and interlinked across diverse datasets and applications.

RDF triples can be stored in so-called triple stores such as Apache Jena or Virtuoso. These are databases providing native support for managing and querying RDF data using the query language SPARQL.

The World Wide Web Consortium (W3C) maintains the standards for RDF, including specifications for the SPARQL query language and various RDF encodings such Turtle, RDF-XML, and JSON-LD.

RDF can be used to create so-called vocabularies, defining classes of objects, their properties and relationships as well as a set of terms and their meaning. The friends-of-a-friend vocabulary for example is about describing people, their relationships and activities. Dublin Core and DCAT are managed vocabularies that are used to describe information resources.

2.3 Data Catalog Vocabulary (DCAT)

DCAT was created to enable data publishers to describe datasets and services in a standardized manner. It is an RDF based vocabulary designed to support the interoperability between various data catalogs on the web.

Advantages of DCAT

1. It improves the discoverability of datasets and data services from different publishers.
2. It makes it possible to have a decentralized approach to publishing data catalogs. A federated search for datasets across different catalogs in different sites is possible using the same query mechanism and structure.

The namespace for DCAT is <http://www.w3.org/ns/dcat#>. DCAT defines just a minimal set of classes and properties of its own and extensively re-uses terms from existing vocabularies, particularly Dublin Core.

Figure 2 shows an overview on the classes and properties of the DCAT vocabulary.

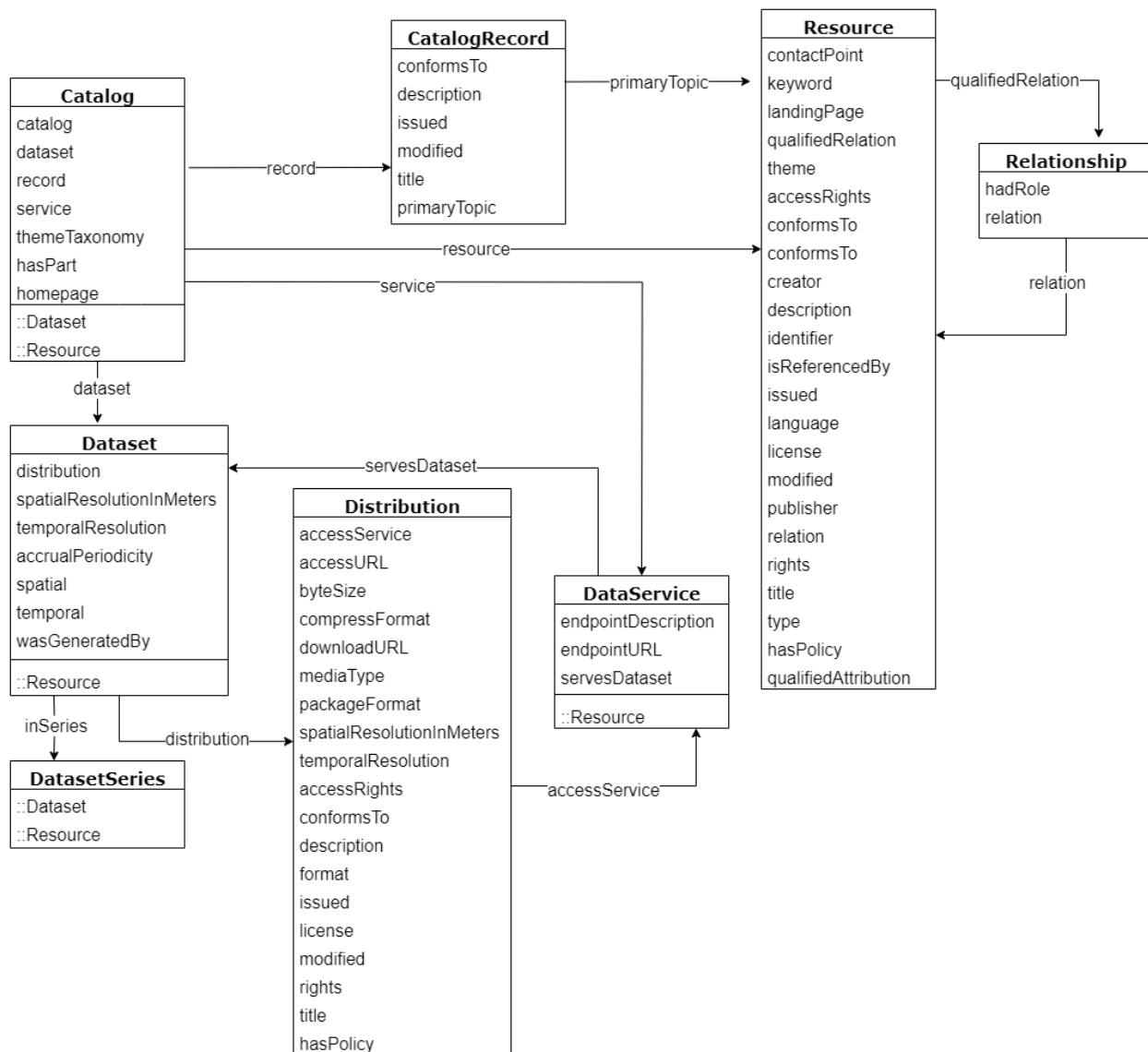


Fig. 2: Overview of DCAT model, showing the classes of resources that can be members of a Catalog, and the relationships between them.

The DCAT vocabulary defines the following classes:

- **dcat:Resource** - This class represents a dataset, a data service or any other resource that may be described by a metadata record in a catalog. The class is not intended to be used directly, but is the parent class of dcat:Dataset, dcat:DataService and dcat:Catalog.
- **dcat:Dataset** - This is the core class in DCAT, representing a dataset. It typically describes a collection of data that is published or curated by an organization or individual, and is available for download or access in one or more representations.
- **dcat:DatasetSeries** - This is a dataset that represents a collection of datasets that are published separately, but share some characteristics that group them.
- **dcat:Distribution** - Distribution refers to a specific representation of a dataset. This class represents an accessible form of a dataset, such as a downloadable file, an API endpoint, or a web service through which the dataset can be accessed.
- **dcat:DataService** - This class represents a service that provides access to data, such as a web service or API that allows querying or retrieving data from a dataset. A data service typically provides selection, extraction, combination, processing or transformation operations over datasets that might be hosted locally or remote to the service. The result of any request to a data service is a representation of a part or all of a dataset or catalog.
- **dcat:Catalog** - This class represents a catalog, which is a collection of metadata describing various resources such as datasets and services. It serves as a container for organizing and describing multiple datasets within a single catalog.

Each instance of the ‘dcat:Catalog’ class may contain multiple instances of ‘dcat:Dataset’ to represent the individual datasets contained within that catalog.

- **dcat:CatalogRecord** - This represents a record or entry in a catalog, describing the registration of a single ‘dcat:Resource’. This class is optional and it exists for catalogs where a distinction is made between metadata about a dataset or service and metadata about the entry in the catalog about the dataset or service. For example, the publication date property of the dataset reflects the date when the information was originally made available by the publishing agency, while the publication date of the catalog record is the date when the dataset was added to the catalog.

The properties shown in each class are both the unique properties of that particular class and those properties inherited or acquired from the “::super-class”. The classes “dcat:Dataset”, “dcat:DataService”, and “dcat:Catalog” all inherit from the “dcat:Resource” class. The class “dcat:DatasetSeries” inherits properties from both the “dcat:Resource” and “dcat:Dataset” classes.

2.4 DCAT Profiles

A number of user communities have developed DCAT Profiles to adapt DCAT to their specific needs. This is useful as soon as additional classes and properties are required to describe the resources of the community. A DCAT Profile is a specification of a data catalog that adds additional constraints to DCAT. A data catalog that conforms to the profile also conforms to DCAT.

Additional constraints in a profile may include classes and properties for additional metadata fields not covered in the DCAT vocabulary specification, sub-classes and sub-properties of the standard DCAT classes and properties or controlled vocabularies or IRI sets as acceptable values for the various properties. In some cases, a profile extends one of the DCAT profiles themselves, by adding classes and properties for metadata fields not covered in the reference DCAT profile.

DCAT Profiles maintain compatibility with DCAT by:

- a) adhering to mandatory DCAT elements
- b) adopting selected optional DCAT elements
- c) adding complementary elements

In the following, some DCAT profiles are listed that are particularly relevant in Europe and Germany:

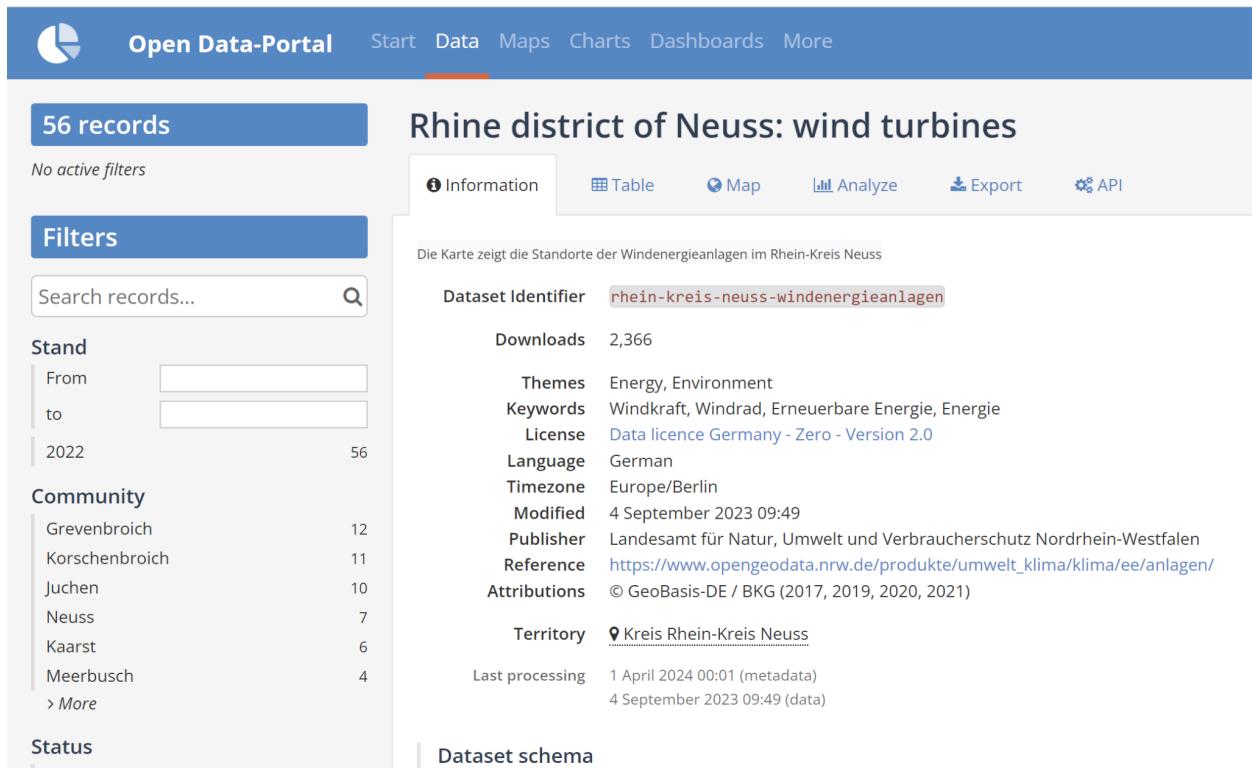
- **DCAT-AP**: This is a profile developed by the European Commission for data portals in Europe. It is the de facto standard of metadata interchange across European data portals.
- **GeoDCAT-AP**: Developed by the European Commission, this profile is a geospatial extension to the DCAT-AP profile. It closes the gaps in ISO19115. It defines a standard way of describing geospatial datasets, data series and services across data portals in Europe. The profile does not intend to replace the INSPIRE regulations on Metadata nor the technical guidelines but intends to support the data exchange between INSPIRE metadata catalogs in Europe
- **DCAT-AP.de** : This is the German National Profile of DCAT-AP. It is the official metadata exchange format for open data from public authorities in Germany. It adds some elements and constraints relevant to metadata exchange between the national GovData catalog and catalogs of various administrative levels. Other countries including Italy, Norway, Sweden, and Belgium have their own national profiles of the DCAT-AP profile.

3. Practical Exercise

3.1 Practical Example of DCAT Metadata

Having seen the importance of standardized metadata and how DCAT enables the interoperability of different data catalogs and the discoverability of datasets and data services, let's take a closer look at the DCAT metadata of a publicly accessible dataset. The Rhine district of Neuss maintains a dataset that shows the locations of various wind turbines in the region. The dataset is published by the State Office for Nature, Environment and Consumer Protection of North Rhine-Westphalia and is available in various formats.

The dataset can be found at the Open-Data Portal for Rhein-Kreis Neuss District, <https://opendata.rhein-kreis-neuss.de/explore/dataset/rhein-kreis-neuss-windenergieanlagen/information/>. Figure 3 shows the open data portal page for the Rhein-Kreis Neuss District presenting metadata on the wind turbines dataset.



The screenshot shows the Open Data Portal interface for the Rhine district of Neuss. The top navigation bar includes links for Start, Data, Maps, Charts, Dashboards, and More. A blue header bar indicates there are 56 records and no active filters. Below this is a search bar and a 'Filters' section. The 'Filters' section contains fields for 'Stand' (From and to dates), 'Community' (Grevenbroich, Korschenbroich, Juchen, Neuss, Kaarst, Meerbusch, and a 'More' link), and 'Status'. To the right, the dataset title 'Rhine district of Neuss: wind turbines' is displayed, along with a map showing the locations of wind turbines. Below the map, detailed dataset metadata is listed:

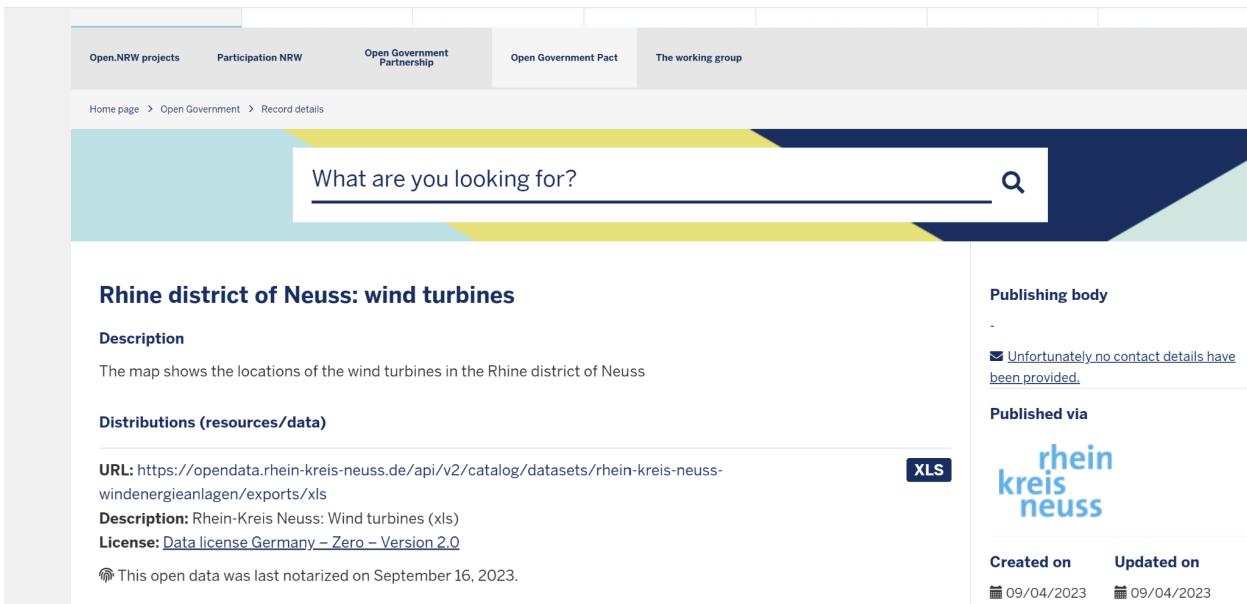
- Dataset Identifier:** rhein-kreis-neuss-windenergieanlagen
- Downloads:** 2,366
- Themes:** Energy, Environment
- Keywords:** Windkraft, Windrad, Erneuerbare Energie, Energie
- License:** Data licence Germany - Zero - Version 2.0
- Language:** German
- Timezone:** Europe/Berlin
- Modified:** 4 September 2023 09:49
- Publisher:** Landesamt für Natur, Umwelt und Verbraucherschutz Nordrhein-Westfalen
- Reference:** https://www.opengeodata.nrw.de/produkte/umwelt_klima/klima/ee/anlagen/
- Attributions:** © GeoBasis-DE / BKG (2017, 2019, 2020, 2021)
- Territory:** Kreis Rhein-Kreis Neuss
- Last processing:** 1 April 2024 00:01 (metadata)
4 September 2023 09:49 (data)

A 'Dataset schema' link is also present at the bottom of the metadata section.

Fig. 3: Open data portal page for the Rhein-Kreis Neuss District showing the wind turbines dataset

The various formats or distributions for the dataset can be viewed by clicking the “Export” tab at the top of the web page.

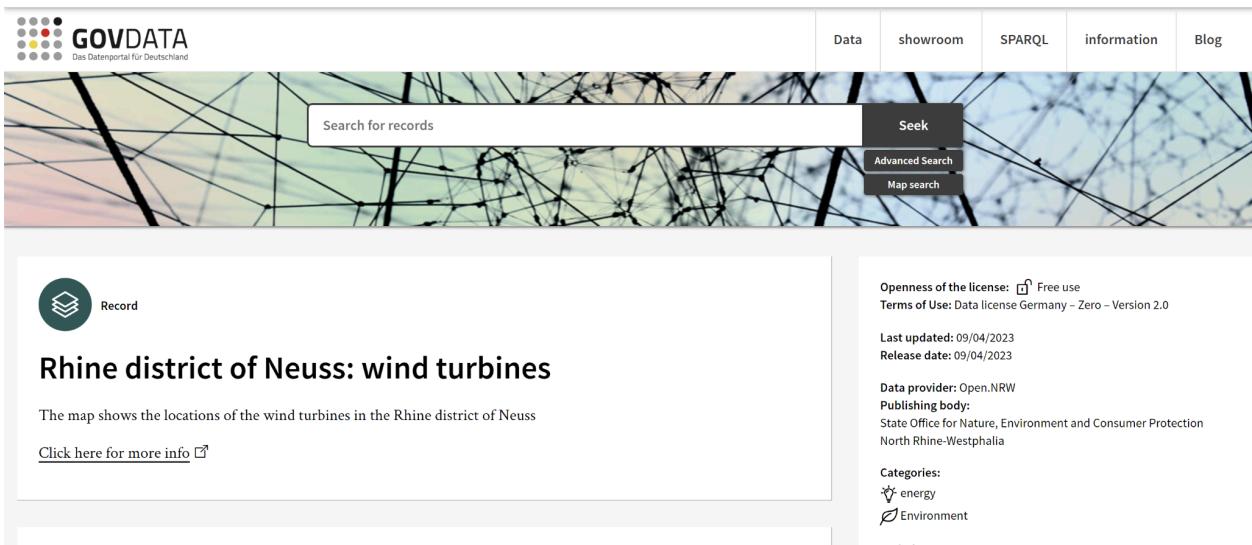
The state government of North Rhine-Westphalia operates and maintains another open data portal that provides users with access to open data from the public sector, <https://open.nrw/>. The open data portal aggregates and publishes metadata from the catalogs of lower administrative regions in North Rhine-Westphalia, making them visible and available on the open data portal of the state government. This includes the catalog from Rhein-Kreis Neuss district. Hence, the wind turbines dataset from the Rhein-Kreis Neuss District catalog is discoverable from the open data portal of North Rhine-Westphalia, <https://open.nrw/dataset/rhein-kreis-neuss-windenergieanlagen-ne>. The same is shown in Figure 4.



The screenshot shows a web page with a header navigation bar containing links for 'Open.NRW projects', 'Participation NRW', 'Open Government Partnership', 'Open Government Pact', and 'The working group'. Below the header is a breadcrumb trail: 'Home page > Open Government > Record details'. A search bar with the placeholder 'What are you looking for?' and a magnifying glass icon is present. The main content area features a title 'Rhine district of Neuss: wind turbines' and a 'Description' section stating 'The map shows the locations of the wind turbines in the Rhine district of Neuss'. A 'Distributions (resources/data)' section lists a URL (<https://opendata.rhein-kreis-neuss.de/api/v2/catalog/datasets/rhein-kreis-neuss-windenergieanlagen/exports/xls>) with an 'XLS' button, a 'Description' of 'Rhein-Kreis Neuss: Wind turbines (xls)', and a 'License' of 'Data license Germany – Zero – Version 2.0'. A note indicates the data was last notarized on September 16, 2023. To the right, there are sections for 'Publishing body' (with a note about no contact details), 'Published via' (with the 'rhein kreis neuss' logo), and 'Created on' and 'Updated on' both dated September 04, 2023.

Fig. 4: Open data portal page for the state government of North Rhine-Westphalia showing the wind turbines dataset from the Rhein-Kreis Neuss District.

The federal government of Germany operates and maintains an open data portal, <https://www.govdata.de/>. In addition to publishing the federal government's data, it aggregates and publishes metadata from the catalogs of the various federal states. As shown in Figure 5, the metadata of the wind turbines dataset of Rhein-Kreis Neuss district, can now be discovered from the open data portal of the federal government, <https://www.govdata.de/web/guest/suchen/-/details/rhein-kreis-neuss-windenergieanlagen>.



The screenshot shows the GOVDATA portal interface. At the top, there is a navigation bar with links for Data, showroom, SPARQL, information, and Blog. Below the navigation bar is a search bar with the placeholder "Search for records" and a "Seek" button. To the right of the search bar are links for Advanced Search and Map search. The main content area features a large image of a network graph. Below the image, on the left, is a "Record" icon and the title "Rhine district of Neuss: wind turbines". A brief description follows: "The map shows the locations of the wind turbines in the Rhine district of Neuss". A link "Click here for more info" is provided. On the right side, there is detailed metadata: "Openness of the license: Free use", "Terms of Use: Data license Germany – Zero – Version 2.0", "Last updated: 09/04/2023", "Release date: 09/04/2023", "Data provider: Open.NRW", "Publishing body: State Office for Nature, Environment and Consumer Protection North Rhine-Westphalia", and categories "energy" and "Environment".

Fig. 5: National open data portal page showing the wind turbines dataset from the Rhein-Kreis Neuss District.

We can download the DCAT metadata file for the wind turbines dataset using the link <https://www.govdata.de/ckan/dataset/rhein-kreis-neuss-windenergieanlagen.rdf>. Clicking the link will download the metadata file in RDF format. It is also available in the GitHub repository of this tutorial. The file can be opened and viewed using any code or text editor such as VS Code, Notepad, Notepad++, etc. Download the metadata file and open it using the text editor program of your choice. The wind turbines metadata is made up of various DCAT elements. This is shown in Figure 6.

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF
  xmlns:dcat="http://www.w3.org/ns/dcat#"
  xmlns:dcatape="http://data.europa.eu/r5r/"
  xmlns:foaf="http://xmlns.com/foaf/0.1/"
  xmlns:dct="http://purl.org/dc/terms/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dcatde="http://dcat-ap.de/def/dcatde/">
  <dcat:Dataset rdf:about="https://opendata.rhein-kreis-neuss.de/api/v2/catalog/datasets/rhein-kreis-neuss-windenergieanlagen">
    <dct:title>Rhein-Kreis Neuss: Windenergieanlagen</dct:title>
    <dct:description>Die Karte zeigt die Standorte der Windenergieanlagen im Rhein-Kreis Neuss</dct:description>
    <dcat:landingPage rdf:resource="https://opendata.rhein-kreis-neuss.de/explore/dataset/rhein-kreis-neuss-windenergieanlagen/">
    <dct:identifier>rhein-kreis-neuss-windenergieanlagen@rhein-kreis-neuss</dct:identifier>
    <dcat:keyword>energie</dcat:keyword>
    <dcat:keyword>erneuerbare-energie</dcat:keyword>
    <dcat:keyword>windkraft</dcat:keyword>
    <dcat:keyword>windrad</dcat:keyword>
    <dct:issued rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2023-09-04T07:49:36.309000+00:00</dct:issued>
    <dct:modified rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2023-09-04T07:49:36.309000+00:00</dct:modified>
    <dct:language rdf:resource="http://publications.europa.eu/resource/authority/language/DEU"/>
    <dcat:theme rdf:resource="http://publications.europa.eu/resource/authority/data-theme/ENV1"/>
    <dcat:theme rdf:resource="http://publications.europa.eu/resource/authority/data-theme/ENER"/>
    <dct:publisher>
      <foaf:Organization rdf:nodeID="N6d4decbb9ab14c6cb98259f0f3e8fe96">
        <foaf:name>Landesamt für Natur, Umwelt und Verbraucherschutz Nordrhein-Westfalen</foaf:name>
      </foaf:Organization>
    </dct:publisher>
    <dcat:distribution rdf:about="https://opendata.rhein-kreis-neuss.de/api/v2/catalog/datasets/rhein-kreis-neuss-windenergieanlagen-kml">
      <dct:title>kml</dct:title>
      <dct:description>Rhein-Kreis Neuss: Windenergieanlagen (kml)</dct:description>
      <dct:license rdf:resource="http://dcat-ap.de/def/licenses/dl-zero-de/2.0"/>
      <dcat:accessURL rdf:resource="https://opendata.rhein-kreis-neuss.de/api/v2/catalog/datasets/rhein-kreis-neuss-windenergieanlagen/expo">
      <dct:language rdf:resource="http://publications.europa.eu/resource/authority/language/DEU"/>
      <dct:format rdf:resource="http://publications.europa.eu/resource/authority/file-type/KML"/>
    </dcat:distribution>
  </dcat:Dataset>
</rdf:RDF>
```

Fig. 6: DCAT elements for wind turbines metadata.

At the beginning of the RDF file, you'll find the namespaces (prefixes) used for different vocabularies within the file. As earlier mentioned, DCAT reuses terms from other existing vocabularies, and defines a minimum set of classes and properties of its own. In this case, the metadata uses terms from:

- a) **xmlns:dcat="http://www.w3.org/ns/dcat#"** : This specifies the namespace for the DCAT (Data Catalog Vocabulary) standard, which is used for describing data catalogs, datasets, and distributions.
- b) **xmlns:dcatap="http://data.europa.eu/r5r/"** : This specifies the namespace for DCAT-AP (Application Profile), a European Union profile of the DCAT standard designed to facilitate interoperability and data sharing across EU member states.
- c) **xmlns:foaf="http://xmlns.com/foaf/0.1/"** : This specifies the namespace for the FOAF (Friend of a Friend) vocabulary, which is used for describing people, their relationships, and their activities.
- d) **xmlns:dct="http://purl.org/dc/terms/"** : This specifies the namespace for the Dublin Core Terms (DCT) vocabulary, a widely-used standard for describing metadata elements such as title, description, creator, and publisher.
- e) **xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"** : This specifies the namespace for RDF (Resource Description Framework), the standard model for data interchange on the web.
- f) **xmlns:dcatde="http://dcat-ap.de/def/dcatde/"** : This specifies the namespace for DCAT-AP.DE, a German application profile of the DCAT standard designed to facilitate interoperability and data sharing within Germany.

The **dcat:Dataset** element contains metadata properties and other elements that describe the wind turbines dataset. It includes various metadata such as title, description, landing page URL, identifier, keywords, issuing and modification dates, language, and details of the publisher.

The **dcat:distribution** element outlines the various distribution formats that are available for the dataset. Each distribution format has properties that describe its title, description, license, access URL, language, format, issuing date, modification dates and availability. The available distribution formats for this dataset are:

- a) KML (Keyhole Markup Language)
- b) JSON-LD (JavaScript Object Notation for Linked Data)
- c) GEOJSON (Geographic JSON)
- d) OV2 (TomTom POI format)
- e) GPX (GPS Exchange format)
- f) Parquet (columnar storage file format)
- g) XLS (Microsoft Excel Spreadsheet)
- h) SHP (Shapefile format)

- i) RDFXML (Resource Description Framework in XML format)
- j) JSONL (JSON Lines format)
- k) N3 (Notation3 format)

All of these distribution formats are available for download through the provided URLs, and the dataset is released under a license permitting open data use. As we can see, The dataset is easily accessible through various URLs hosted on the Open-Data Portal for the Rhein-Kreis Neuss District. The dataset's metadata plays a crucial role, as it is aggregated and shared with larger data portals such as [govdata.de](#). This federated approach to metadata search significantly enhances discoverability, enabling efficient and comprehensive data access across different platforms. Users can search for wind turbine data on the larger data portals, filter the results to the Rhein-Kreis Neuss District dataset and use the available access urls to access the data.

3.2 Creating, Editing and Visualizing DCAT metadata with RDForms

Sharing datasets or data services involves creating metadata that describes the datasets and data services. There are various softwares and tools that enable creating, editing and managing metadata, according to various metadata standards. These include ArcGIS Pro, ArcGIS Online, QGIS, Geoserver, GDAL, RDForms etc. However, not all of them have support for the DCAT standard. For this practical section, we're going to use RDForms as it offers both simplicity and has support for DCAT.

RDForms is an online tool specifically designed for creating and editing DCAT compliant metadata forms. With RDForms, you can create and edit metadata forms for datasets, data catalogs, and other data-related resources. The tool has an editor tab that allows editing of the descriptions of the various DCAT elements, a presenter tab that allows us to see the results and an RDF tab that shows the resulting metadata in both RDF and JSON-LD format. Any changes made in the editor tab are automatically reflected in the other tabs. The editor is driven by the RDForms library which uses templates to generate HTML forms to edit RDF. The RDForms DCAT editor can be accessed using the link <https://rdforms.com/editors/dcat/#editor>. Figure 7 shows the RDForms landing page.

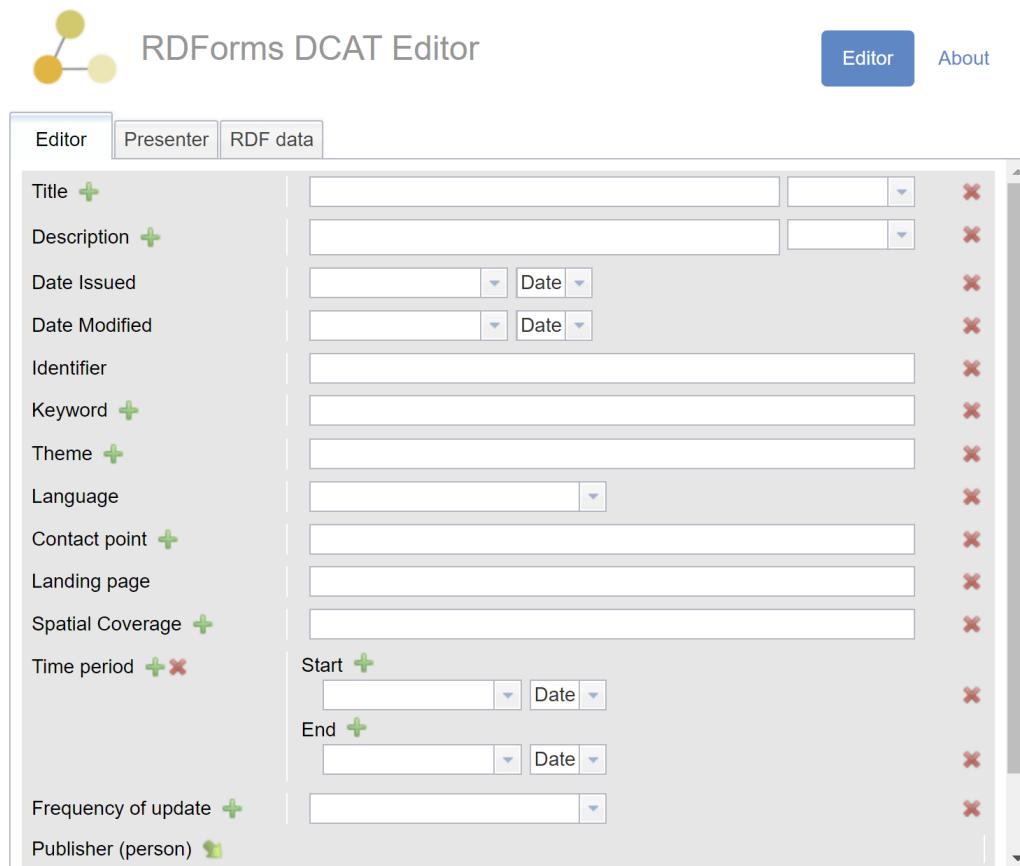


Fig. 7: RDForms landing page.

The online editor provides a simple and intuitive interface for creating and editing metadata, with a focus on simplicity and ease of use. It is divided into several sections, each with its own set of fields and controls.

The editor provides a range of fields and controls for creating and editing metadata including:

- a) Text fields for entering descriptive metadata, such as title, description, and keywords.
- b) Select boxes for selecting values from predefined lists, such as data formats, data types, and access rights.
- c) Date and time pickers for entering dates and times.

For this task, we are going to create metadata for a hypothetical dataset. Imagine that you work at the planning department of your city (Münster city). The department has collected data on buildings in the city and wants to make that data accessible and findable on other data portals. The dataset is available in different distribution forms, including shapefile download, GeoJSON and as a web service such as WMS. Your task is to create metadata for the buildings dataset.

We'll use RDForms for this task because of its simplicity and accessibility, knowing that it is hardly used in practice. In practice, publishers of datasets use standard software such as ArcGIS plus various specialized tools to create the metadata for the datasets, e.g. CKAN, Socrata, custom software.

A range of fields and controls are available on the editor tab RDForms. We are going to fill the form with various inputs to describe the buildings dataset.

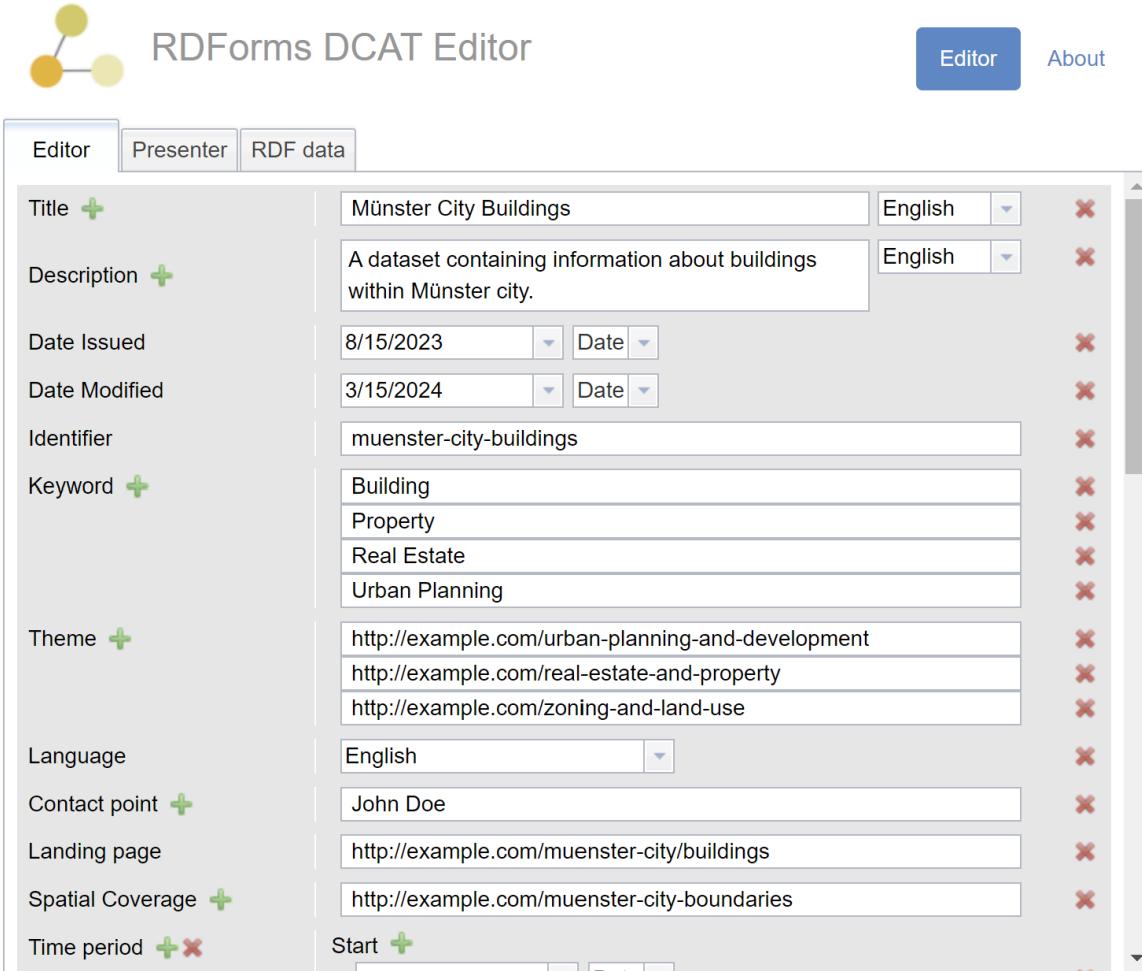
1. **Title** - This refers to the name given to the resource. In this case, this will be the title assigned to the building dataset. The dropdown field next to the title offers a selection for the language used in the title. This will be used by data catalogues that will aggregate our metadata, to categorize our metadata correctly based on the language used. The title can be repeated in a different language by clicking the green "+" sign next to the "Title" field, and selecting the appropriate language.

Choose an appropriate Title and corresponding Language selection for the Building dataset.

2. **Description** - This is text that offers more information about the resource. The dropdown field next to the description offers a selection for the language used in the description. The description can also be repeated in a different language by clicking the green "+" sign next to the "Description" and selecting the appropriate language. Enter a short description that describes the building, and choose the appropriate language.
3. **Date Issued** - This refers to the date when the resource was formally issued or made available, for example, through publication. The value of the date should be encoded using the ISO 8601 date and time format. However, RDForms provides a simple date selection input control, and implicitly converts the selected date to the required format. Select an arbitrary date that represents when the dataset was issued.
4. **Date Modified** - Datasets that are made available may undergo some changes, modifications and updates. The date modified refers to the most recent date when the dataset was updated or modified. The value may be null if the dataset has never been updated or if the last date of updating the database is not known. Just like the date issued property, RDForms provides a simple date selection input control, and implicitly converts the selected date to the required ISO 8601 date and time format. Select an arbitrary date when the dataset might have been modified. Ideally, the date modified should be more recent than the date issued.
5. **Identifier** - This refers to a text string that is assigned to the resource, that uniquely identifies the resource or dataset. The identifier is a text string that is assigned to the resource to provide an unambiguous reference within a particular context.
6. **Keyword** - A keyword is a specific term or phrase that describes a particular aspect of the dataset, such as a concept, entity, or attribute. Keywords are often used to facilitate the search and filtering of datasets. Multiple keywords can be assigned to the same

resource, by clicking the green “+” sign. For our buildings dataset, we could add the keywords “Building”, “Property”, “Real Estate”, “Urban Planning” etc. You can also consider adding keywords that are related to the attributes of the buildings dataset, for example: building type (residential, commercial, industrial), architectural style, accessibility features etc.

7. **Theme** - A theme is a broader category or topic that a dataset belongs to, which encompasses multiple related keywords. Themes provide a higher-level context for understanding the dataset's content and relevance. Multiple themes can be assigned to the resource being described by clicking the “+” sign. Choose an appropriate theme for the buildings dataset. Our buildings dataset could have the themes “Urban Planning and Development”, “Real Estate and Property”, “Zoning and Land Use” etc
8. **Language** - The language property is used to specify the language(s) used in the metadata, including the dataset's title, description, keywords, and other metadata elements. It also identifies the language used in the textual values of the dataset's distribution. This will help users search for datasets in their preferred language. RDForms provides a dropdown selection control for selecting the language used in the dataset. Choose an appropriate language that matches the language used in the dataset's metadata.
9. **Contact Point** - This property is used to provide contact information for the dataset's maintainer, curator, or owner. This information is essential for users who want to learn more about the dataset, request access, or report issues. Let us assume that you will be the contact point for the dataset. Input your name in the text box provided by RDForms.
10. **Landing Page** -This property is used to provide a URL that links to a page that provides more information about the dataset, such as a dataset description, documentation, or access instructions. Ideally, the web page should provide a brief summary of the dataset including its scope, purpose, content, information on how to access the dataset, including any necessary authentication or authorization procedures. The provided value should have the structure of a valid URL. Input a sample URL that will lead users to the landing page. You can use a dummy website URL such as ‘<http://example.com/muenster-buildings>’
11. **Spatial Coverage** - This property is used to describe the geographic boundaries or spatial coverage of the dataset. This information is essential for users who want to understand the dataset's spatial scope and relevance or find datasets that cover a particular region of interest. The spatial coverage can be provided as the Bounding Box (BBOX) coordinates that define the spatial extent of the dataset. It may also be indicated using an IRI reference (link) to a resource describing a location. For this task, we are going to provide a sample IRI link.



The screenshot shows the RDForms DCAT Editor interface. At the top, there is a logo consisting of three colored circles (green, yellow, and orange) connected by lines, followed by the text "RDForms DCAT Editor". To the right are two buttons: "Editor" (which is highlighted in blue) and "About". Below this is a navigation bar with three tabs: "Editor" (selected), "Presenter", and "RDF data".

The main area contains a table of dataset metadata fields:

Title +	Münster City Buildings	English	
Description +	A dataset containing information about buildings within Münster city.	English	
Date Issued	8/15/2023	Date	
Date Modified	3/15/2024	Date	
Identifier	muenster-city-buildings		
Keyword +	Building Property Real Estate Urban Planning		
Theme +	http://example.com/urban-planning-and-development http://example.com/real-estate-and-property http://example.com/zoning-and-land-use		
Language	English		
Contact point +	John Doe		
Landing page	http://example.com/muenster-city/buildings		
Spatial Coverage +	http://example.com/muenster-city-boundaries		
Time period +	Start +		

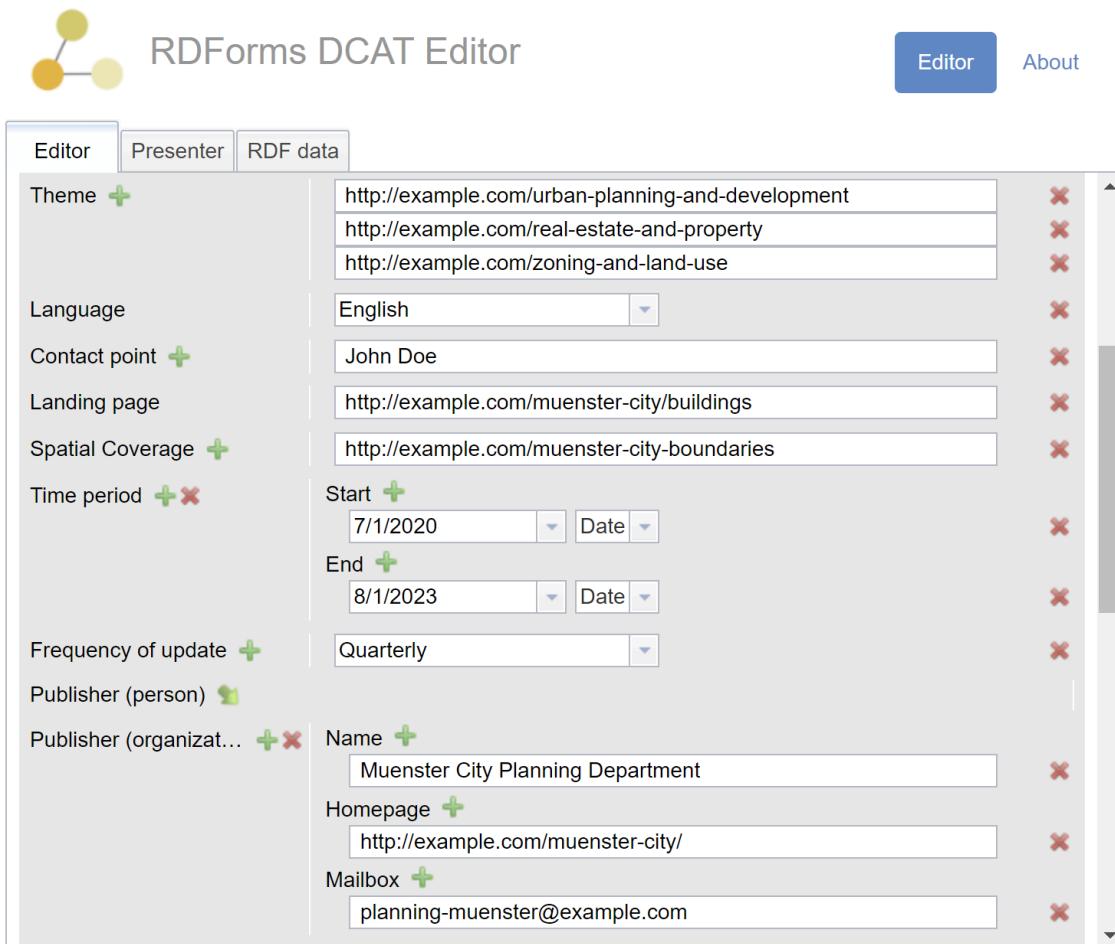
Fig. 8 : RDForms Editor with sample input values for the metadata of the building's dataset. Part 1.

12. **Time Period** - The time period property is the temporal period that the dataset covers and is represented by the start date and end date values. The start date represents the beginning of the time period that is covered by the dataset while the end date represents the end of the period that is covered by the dataset. This information can help users understand the dataset's content and relevance. RDForms provides simple date selection input controls for the start date and end date, and implicitly converts the selected dates to the required ISO 8601 date and time format. Choose arbitrary start and end dates for the Buildings dataset.

13. **Frequency of Update** - This represents the time interval in which the dataset is updated as a whole. RDForms provides a selection dropdown with various values for selection, for example daily, monthly, quarterly, annual, half-yearly etc. This information is important to users of the dataset, especially if the phenomenon represented by the dataset is

expected to change frequently. Choose an appropriate frequency of update from the dropdown.

14. **Publisher (Person)** - The publisher property is used to identify the entity responsible for publishing the dataset. This is typically used to provide attribution and credit to the entity responsible for creating and publishing the dataset. The entity in this scenario is a person. RDForms provides separate text boxes for inputting the details of the person, including the first name, family name, and personal mailbox details. These details can be used by users of the dataset to contact the publisher, incase of inquiries regarding the dataset. We are going to skip this section and instead provide the Publisher (Organization) details described in the next step.
15. **Publisher (Organization)** - This is used to identify the organization that is responsible for creating, maintaining and publishing the dataset. Such organizations include government agencies, research institutions, universities, or other organizations that produce and publish datasets. This information can be used to provide attribution and credit to the organization and it enhances findability of datasets published by the same organization. RDForms provide separate text boxes for inputting the name of the organization, a link to the homepage of the organization and mailbox details of the organization. These details can be used by users to contact the organization that is responsible for creating and publishing the dataset, incase of inquiries. Lets provide some arbitrary details of the planning department that made the dataset available.



The screenshot shows the RDForms DCAT Editor interface. At the top, there is a logo consisting of three yellow circles connected by lines, followed by the text "RDForms DCAT Editor". To the right are two buttons: "Editor" (which is highlighted in blue) and "About". Below this is a navigation bar with tabs: "Editor" (selected), "Presenter", and "RDF data".

The main area contains a list of dataset metadata fields with their corresponding input values:

- Theme**: http://example.com/urban-planning-and-development, http://example.com/real-estate-and-property, http://example.com/zoning-and-land-use
- Language**: English
- Contact point**: John Doe
- Landing page**: http://example.com/muenster-city/buildings
- Spatial Coverage**: http://example.com/muenster-city-boundaries
- Time period**: Start: 7/1/2020, End: 8/1/2023
- Frequency of update**: Quarterly
- Publisher (person)**: Muenster City Planning Department
- Publisher (organization)**: Name: Muenster City Planning Department, Homepage: http://example.com/muenster-city/, Mailbox: planning-muenster@example.com

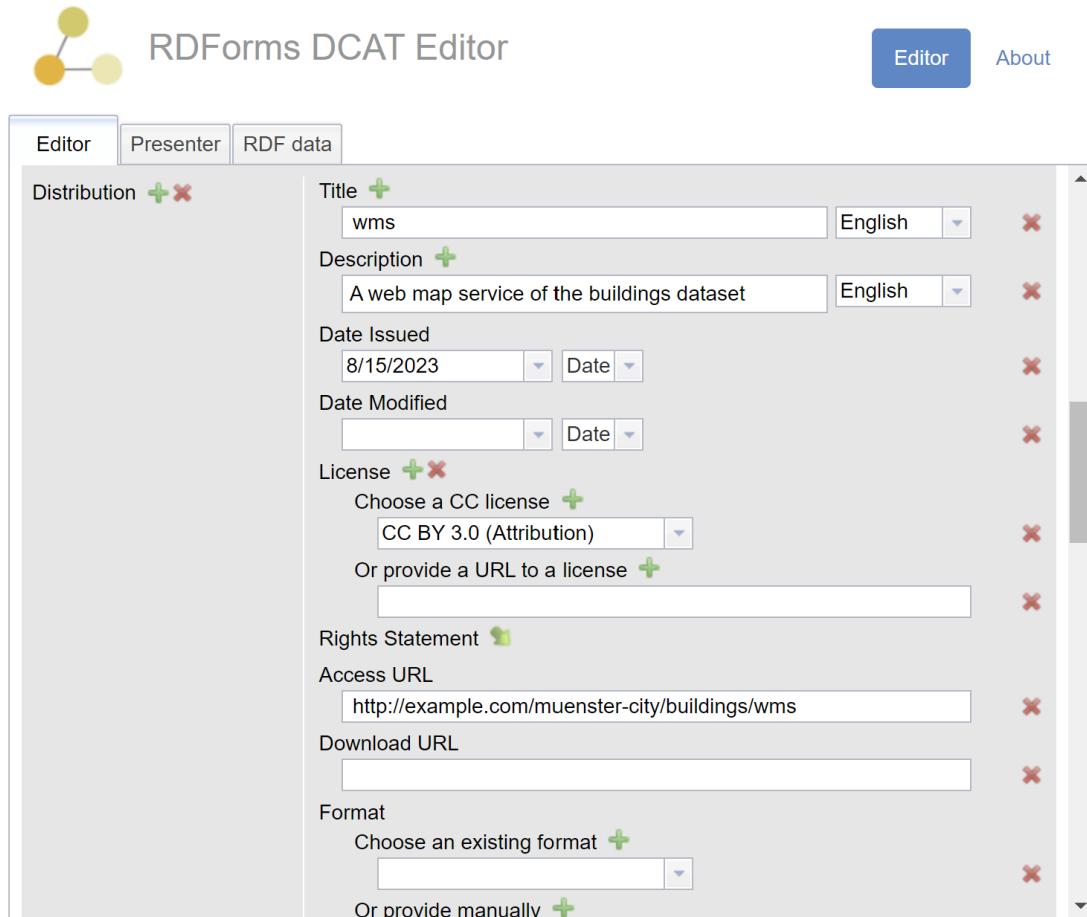
Fig. 9 : RDForms Editor with sample input values for the metadata of the building's dataset.
Part 2.

16. **Distribution** - The distribution property is used to provide information about how to access and retrieve the dataset, making it easier for users to find and use the dataset. RDForms provides various text boxes for inputting the distribution information. These include:

- Title of the distribution
- Description of the distribution
- The date that the distribution was issued
- The date that the distribution was modified
- The license information
- The rights statement
- Access URL of the dataset distribution

- Download URL of the dataset distribution
- The format of that distribution
- The size in bytes of the distribution.

A dataset can be available in multiple distribution formats, including CSV, shapefile, GeoJSON etc. Lets assume that our buildings dataset is available for download in shapefile and GeoJSON format, and can also be accessed as a Web Map Service (WMS). Input the details of the various distributions by clicking the “+” sign next to the “Distribution” property name.



The screenshot shows the RDForms DCAT Editor interface. At the top, there's a logo of three interconnected circles (yellow, orange, green) followed by the text "RDForms DCAT Editor". To the right are "Editor" and "About" buttons. Below the header is a navigation bar with tabs: "Editor" (which is selected), "Presenter", and "RDF data".

The main area contains a form for a "Distribution" resource. It includes fields for "Title" (wms), "Description" (A web map service of the buildings dataset), "Date Issued" (set to 8/15/2023), "Date Modified" (empty), "License" (choose CC BY 3.0 (Attribution)), "Access URL" (http://example.com/muenster-city/buildings/wms), "Download URL" (empty), and "Format" (choose an existing format or provide manually).

Fig. 10: RDForms Editor with sample input values for the metadata of the building's dataset.
Part 3.

Note that the WMS distribution has no download URL, as it is a web service that is meant to be accessed and used in other applications using the provided Access URL.

RDFForms DCAT Editor

Editor About

Distribution + ×

Choose an existing format	<input type="text"/>	×
Or provide manually	+	×
wms	<input type="text"/>	×
Size (in bytes)	<input type="text"/>	×
Title	+	×
GeoJSON	<input type="text"/>	×
Description	+	×
GeoJSON file containing the buildings information	<input type="text"/>	×
Date Issued	<input type="text"/> 8/15/2023	×
Date Modified	<input type="text"/> 3/15/2024	×
License	+ ×	×
Choose a CC license	+	×
CC BY 3.0 (Attribution)	<input type="text"/>	×
Or provide a URL to a license	+	×
Rights Statement	+	×
Access URL	<input type="text"/> http://example.com/muenster-city/buildings/geojson	×

Fig. 11: RDFForms Editor with sample input values for the metadata of the building's dataset.
Part 4.

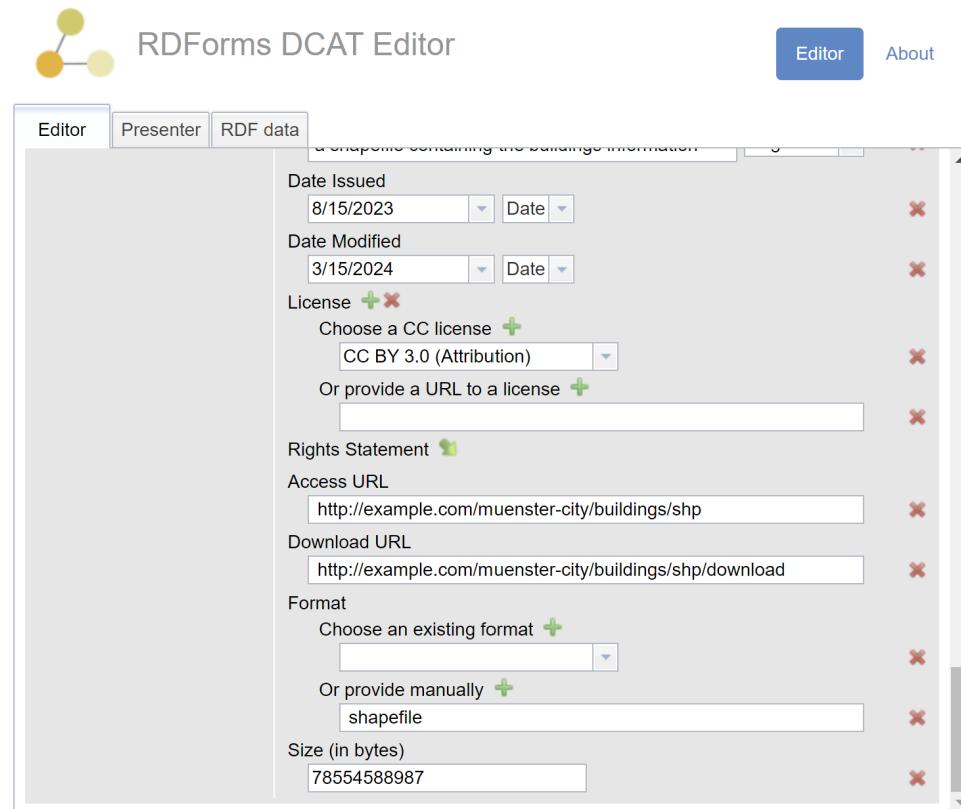
RDFForms DCAT Editor

Editor About

Distribution + ×

Access URL	<input type="text"/> http://example.com/muenster-city/buildings/geojson	×
Download URL	<input type="text"/> http://example.com/muenster-city/buildings/geojson/download	×
Format	Choose an existing format	+ ×
	<input type="text"/>	×
Or provide manually	+	×
application/json	<input type="text"/>	×
Size (in bytes)	<input type="text"/> 45589	×
Title	+	×
shapefile	<input type="text"/>	×
Description	+	×
a shapefile containing the buildings information	<input type="text"/>	×
Date Issued	<input type="text"/> 8/15/2023	×
Date Modified	<input type="text"/> 3/15/2024	×
License	+ ×	×
Choose a CC license	+	×
CC BY 3.0 (Attribution)	<input type="text"/>	×

Fig. 12: RDForms Editor with sample input values for the metadata of the building's dataset.
Part 5.



The screenshot shows the RDForms DCAT Editor interface. At the top, there is a logo consisting of three colored circles (green, orange, and yellow) connected by lines, followed by the text "RDForms DCAT Editor". To the right are "Editor" and "About" buttons. Below the header, there is a navigation bar with tabs: "Editor" (which is selected), "Presenter", and "RDF data". A title bar says "a dataset containing the buildings information". The main area contains the following fields:

- Date Issued:** 8/15/2023 (with a Date dropdown button)
- Date Modified:** 3/15/2024 (with a Date dropdown button)
- License:** CC BY 3.0 (Attribution) (with a dropdown menu and a red error icon)
- Rights Statement:** (with a green info icon)
- Access URL:** http://example.com/muenster-city/buildings/shp (with a red error icon)
- Download URL:** http://example.com/muenster-city/buildings/shp/download (with a red error icon)
- Format:** Choose an existing format (with a dropdown menu and a red error icon)
- Or provide manually:** shapefile (with a red error icon)
- Size (in bytes):** 78554588987 (with a red error icon)

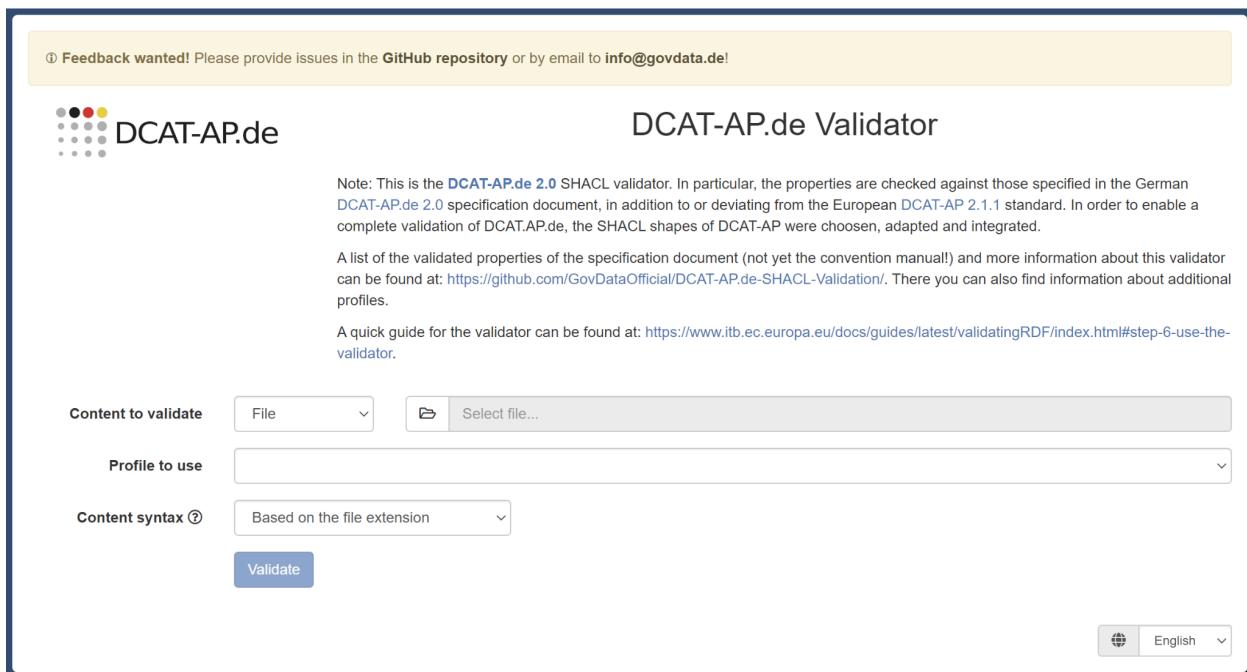
Fig. 13: RDForms Editor with sample input values for the metadata of the building's dataset.
Part 6.

Figure 8-13 shows the RDForms Editor with sample input values for the metadata of the building's dataset. The resulting DCAT metadata can be viewed by selecting the *RDF Data* tab. You can switch between XML and JSON-LD representation. We are going to use the JSON-LD format to validate our metadata.

3.3 Validating DCAT Metadata

Having created some DCAT metadata of the Buildings dataset, we are now going to validate it. This is to ensure that our metadata conforms to DCAT, and that it can be seamlessly integrated in other data portals. We are going to use the **DCAT-AP.de validator**, which is a tool developed to validate the DCAT metadata of the German Profile. Remember, metadata that conforms to a DCAT Profile, also conforms to the DCAT standard. The validator tool can be found at <https://www.itb.ec.europa.eu/shacl/dcat-ap.de/upload>.

Figure 14 shows the landing page of the validator tool.



① Feedback wanted! Please provide issues in the [GitHub repository](#) or by email to info@govdata.de!

DCAT-AP.de

DCAT-AP.de Validator

Note: This is the [DCAT-AP.de 2.0](#) SHACL validator. In particular, the properties are checked against those specified in the German DCAT-AP.de 2.0 specification document, in addition to or deviating from the European [DCAT-AP 2.1.1](#) standard. In order to enable a complete validation of DCAT-AP.de, the SHACL shapes of DCAT-AP were chosen, adapted and integrated.

A list of the validated properties of the specification document (not yet the convention manual!) and more information about this validator can be found at: <https://github.com/GovDataOfficial/DCAT-AP.de-SHACL-Validation/>. There you can also find information about additional profiles.

A quick guide for the validator can be found at: <https://www.itb.ec.europa.eu/docs/guides/latest/validatingRDF/index.html#step-6-use-the-validator>.

Content to validate File

Profile to use

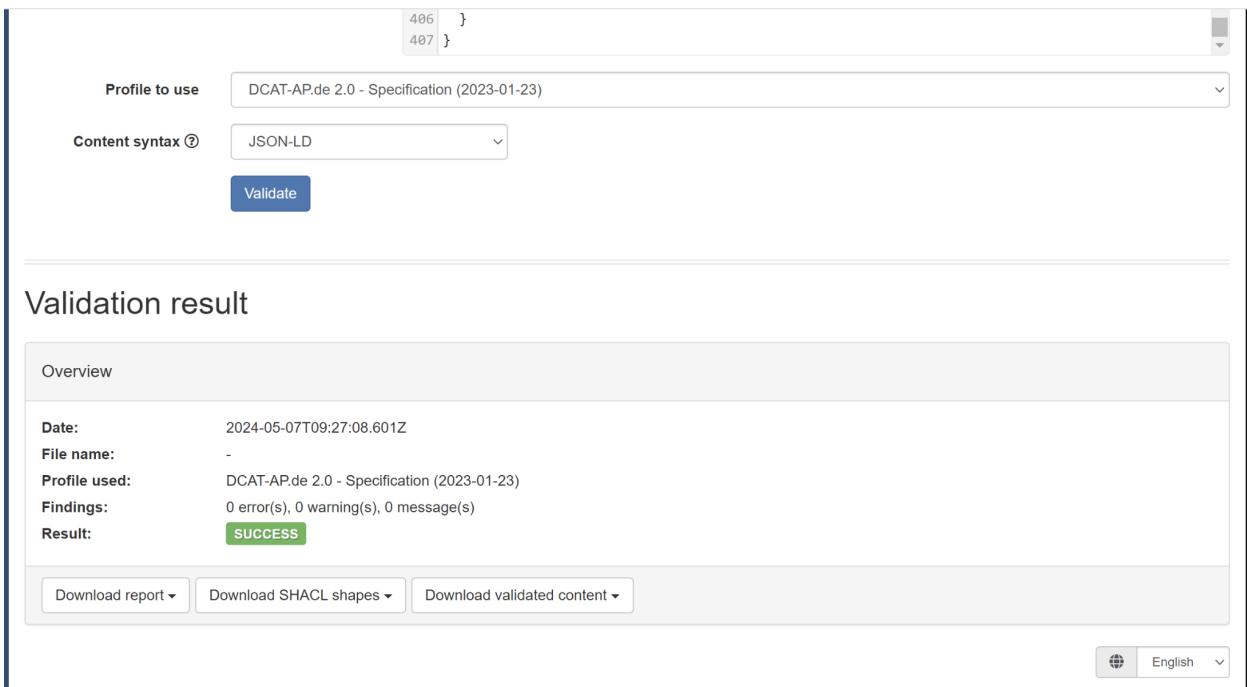
Content syntax

Validate

English

Fig. 14: Landing page of the DCAT-AP.de validator tool.

The language has been switched to English using the button on the bottom right corner. Change the *Content to validate* to “Direct Input” which means that we are going to paste the generated metadata from RDForms to the validator tool. The tool can validate different formats of DCAT metadata. For this task, we are going to use JSON-LD format. Select “JSON-LD” from the *Content syntax* and “DCAT-AP.de Specification (2023-01-23)” from the *Profile to use* section. Copy and paste the RDForms JSON-LD metadata into the *Content to validate* section and click *Validate*. Figure 15 shows the results of validating the metadata using the DCAT-AP.de validator tool.



The screenshot shows the DCAT-AP.de 2.0 - Specification (2023-01-23) validator interface. At the top, there's a code editor window containing JSON-LD code. Below it, the validation configuration includes 'Profile to use' set to 'DCAT-AP.de 2.0 - Specification (2023-01-23)' and 'Content syntax' set to 'JSON-LD'. A 'Validate' button is present. The main area is titled 'Validation result' and contains an 'Overview' section. The overview table includes the following data:

Date:	2024-05-07T09:27:08.601Z
File name:	-
Profile used:	DCAT-AP.de 2.0 - Specification (2023-01-23)
Findings:	0 error(s), 0 warning(s), 0 message(s)
Result:	SUCCESS

At the bottom of the validation result section are three download buttons: 'Download report', 'Download SHACL shapes', and 'Download validated content'. A language selection dropdown at the bottom right shows 'English'.

Fig. 15: Results of validating the RDForms metadata using the DCAT-AP.de validator tool.

3.4 Running SPARQL Queries on DCAT metadata.

In this section, you will learn how to:

1. Set up an RDF store using Apache Jena Fuseki with Docker.
2. Load DCAT metadata into the RDF store.
3. Execute SPARQL queries to retrieve and manipulate metadata.

By the end of this tutorial, you should have a functional understanding of DCAT, RDF, and SPARQL, and be able to apply these concepts to manage and query data catalogs.

3.4.1 Overview of Technologies

RDF is a standard model for data interchange on the web. It allows data to be expressed as triples (subject, predicate, object) and is the underlying format used for representing metadata in DCAT. Built on RDF, DCAT employs RDF schema to define classes and properties that describe datasets and their related entities. DCAT extends RDF to provide a structured vocabulary for cataloging datasets, making them discoverable and interoperable.

- **SPARQL (SPARQL Protocol and RDF Query Language)**

SPARQL is a powerful query language and protocol used to retrieve and manipulate data stored in RDF format. It allows users to write complex queries to access specific data points, filter data, and combine results from multiple datasets. SPARQL supports data querying and updating, making it a versatile tool for working with RDF data.

As a query language for RDF data, SPARQL allows users to retrieve and manipulate the metadata expressed in DCAT. It enables querying across RDF datasets to extract meaningful information, filter results, and perform various analytical operations.

- **Apache Jena Fuseki**

Apache Jena Fuseki is an open-source triple store for handling RDF data, providing support for reading, writing, and querying RDF using SPARQL. It offers a robust and scalable solution for managing semantic web data, supporting various RDF serialization formats and providing a convenient web-based interface for running SPARQL queries. Fuseki will be used as the RDF store and SPARQL endpoint where we will load DCAT metadata and run their SPARQL queries. It serves as the backend for storing and retrieving data, making it a central component of this tutorial.

- **Docker**

Docker is a platform that allows you to package applications and their dependencies into lightweight, portable containers that can run consistently across different computing environments. Docker simplifies the deployment process, ensuring that applications work seamlessly across various infrastructures.

Docker helps in setting up Apache Jena Fuseki in a standardized and reproducible manner. Using Docker makes the setup process straightforward, ensuring a consistent working environment regardless of the operating system or infrastructure being used.

3.4.2 Workflow

1. Install Docker

If you're completely new to Docker, don't worry! Follow these steps to get everything up and running. Visit the [Docker website](#) and download the Docker Desktop installer for your operating system. Run the installer which will guide you through the installation process. Once the installation is complete, please start up the Docker Desktop application and make sure that it is running.

2. Pull and Run Apache Jena Fuseki Docker Image

We are going to pull a docker image for the Apache Jena Fuseki Server. This will be used to instantiate a running container with the server.

Open a terminal or command prompt on your computer. On Windows, you can use PowerShell or Command Prompt. On Mac or Linux, open the Terminal application. In the terminal, run the following command to download and run the Apache Jena Fuseki Docker image:

```
docker run -p 3030:3030 -e ADMIN_PASSWORD=admin123 stain/jena-fuseki:5.1.0
```

Running the above command in the terminal or powershell will make docker pull or download the stain/jena-fuseki docker image if it is not locally available, before running it. The specific docker image is version 5.1.0 . We set the password for the admin interface to be 'admin123'. Running or executing a docker image results in docker container which can be accessed and used. After the process is complete, the docker container with Apache Jena Fuseki server will be available and accessible through <http://localhost:3030/>. Figure 16 shows the results of running the docker command on the terminal.

```
86aa3a46b511: Pull complete
8b18e2a7cd1d: Pull complete
75aa0f251c3c: Pull complete
e6b6992b25f2: Pull complete
1a9ecf1a9f58: Pull complete
423c2975d56e: Pull complete
dc3f2c91699f: Pull complete
0d3bea58fe16: Pull complete
a41b26252937: Pull complete
31b86156ff6b: Pull complete
10b0f3e014ec: Pull complete
Digest: sha256:b1d0c96f19adddef417520184471ac816cf36c6e72936a88ea02b29f4305929a
Status: Downloaded newer image for stain/jena-fuseki:5.1.0
#####
Initializing Apache Jena Fuseki

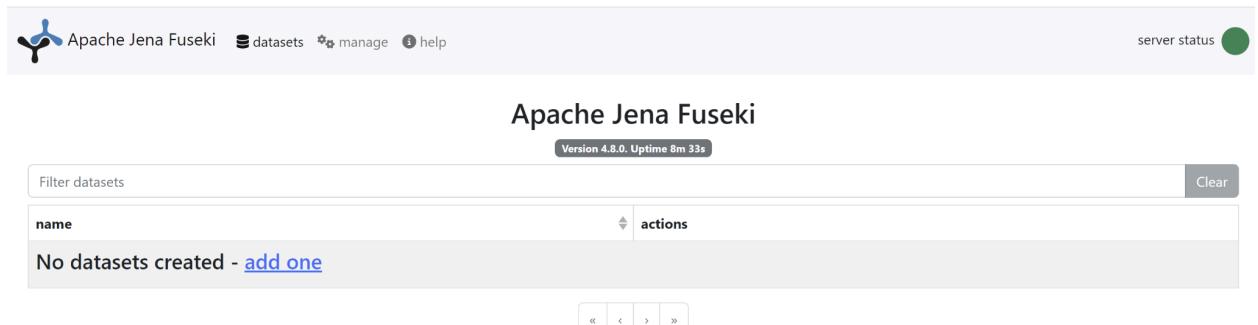
#####
Waiting for Fuseki to finish starting up...
22:08:55 INFO  Server      :: Apache Jena Fuseki 5.1.0
22:08:55 INFO  Config     :: FUSEKI_HOME=/jena-fuseki
22:08:55 INFO  Config     :: FUSEKI_BASE=/fuseki
22:08:55 INFO  Config     :: Shiro file: file:///fuseki/shiro.ini
22:08:55 INFO  Server      :: Memory: 4.0 GiB
22:08:55 INFO  Server      :: Java: 21.0.4
22:08:55 INFO  Server      :: OS: Linux 5.15.153.1-microsoft-standard-WSL2 amd64
22:08:55 INFO  Server      :: PID: 11
22:08:55 INFO  Server      :: Started 2024/08/22 22:08:55 GMT on port 3030
Fuseki is available :-)
```

Fig. 16: Running the docker command on the terminal.

3. Access the Fuseki Admin UI

Open any browser of your choice and go to <http://localhost:3030/>. This opens the Apache Jena Fuseki management interface. Log into the server using 'admin' as the

username and ‘admin123’ as the admin password. Figure 17 shows the Apache Jena Fuseki interface.



The screenshot shows the Apache Jena Fuseki web interface. At the top, there is a header with the Apache Jena Fuseki logo, navigation links for 'datasets', 'manage', and 'help', and a 'server status' indicator (green). Below the header, the title 'Apache Jena Fuseki' is displayed, followed by a sub-header 'Version 4.8.0. Uptime 8m 33s'. A search bar labeled 'Filter datasets' and a 'Clear' button are on the right. The main content area has a table with two columns: 'name' and 'actions'. A message 'No datasets created - [add one](#)' is shown. At the bottom, there are navigation icons for '«', '<', '>', and '»'.

Fig. 17: Apache Jena Fuseki interface.

At the moment, our server is empty and has no datasets or metadata that can be queried. We are going to load some metadata in the next step.

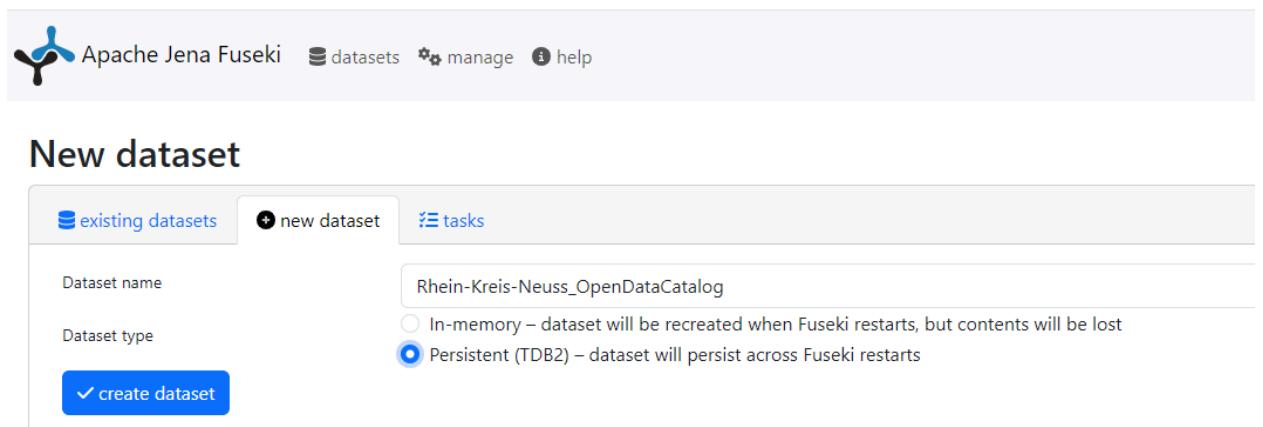
4. Load DCAT Metadata into Fuseki

We are going to use the open data catalogue of the Rhein-Kreis Neuss district for this section. The catalogue contains the description of various datasets that are owned or managed by the district. For simplicity, the metadata for the whole catalog can be found at

https://github.com/oer4sdi/OER-DCAT/blob/main/data/Rhein-Kreis-Neuss_OpenDataCatalog.rdf. The link will open the Github repository that contains the catalog metadata file. Click the download icon to download the file, which is in .rdf format.

To load DCAT metadata into our server, we need to create a dataset. Click the ‘**add one**’ option to open the page for adding new datasets. Enter ‘Rhein-Kreis-Neuss_OpenDataCatalog’ as the name of the dataset. Select ‘**Persistent**’ as the dataset type. This ensures that the dataset that we create is not lost when we stop or restart our docker container.

Click the ‘**Create Dataset**’ button to save the changes. Figure 18 is a visualization of this step.

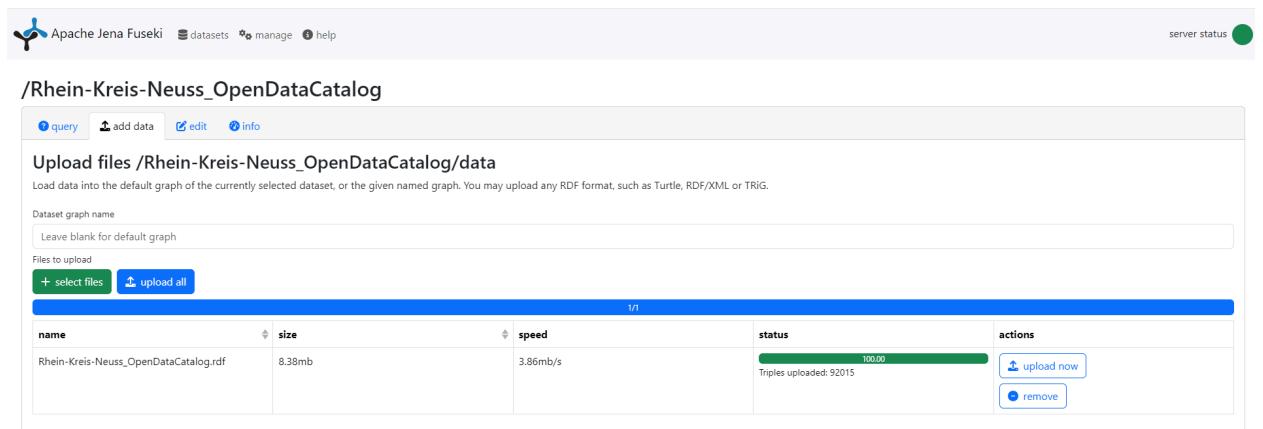


The screenshot shows the Apache Jena Fuseki interface for creating a new dataset. The top navigation bar includes links for 'existing datasets', 'new dataset', and 'tasks'. The 'new dataset' tab is active. The 'Dataset name' field contains 'Rhein-Kreis-Neuss_OpenDataCatalog'. The 'Dataset type' section has two options: 'In-memory – dataset will be recreated when Fuseki restarts, but contents will be lost' (unchecked) and 'Persistent (TDB2) – dataset will persist across Fuseki restarts' (checked). Below these fields is a blue 'create dataset' button with a checkmark icon.

Fig. 18: Adding the open data catalogue of the Rhein-Kreis Neuss district 0to the server

We can now add the DCAT metadata to the dataset that we created. In the Fuseki admin UI, choose the Rhein-Kreis-Neuss_OpenDataCatalog dataset that we have just created. Navigate to the "**Add data**" tab. Click the '**Select files**' option and choose the metadata file that we downloaded.

Click the '**upload now**' button under the '**actions**' tab to finish the process. Figure 19 shows the server interface after uploading the metadata. The status button turns to Green, which indicates that our metadata has been uploaded successfully. As can be seen, 92015 triples have been uploaded.



The screenshot shows the Apache Jena Fuseki interface for adding data to the '/Rhein-Kreis-Neuss_OpenDataCatalog' dataset. The top navigation bar includes links for 'query', 'add data', 'edit', and 'info'. The 'add data' tab is active. The 'Upload files' section shows a single file named 'Rhein-Kreis-Neuss_OpenDataCatalog.rdf' with a size of '8.38mb'. The 'Actions' column for this file shows a progress bar at '100.00%' completion, indicating 'Triples uploaded: 92015'. Below the table are buttons for 'upload now' and 'remove'.

Fig. 19: Uploading the metadata to the server.

5. Query the DCAT Metadata using SPARQL

We are now going to see some sample SPARQL queries that we can use to interact with and extract some useful information from the DCAT metadata that we just uploaded. You don't need any knowledge of SPARQL as the code will be provided to you together with its explanation.

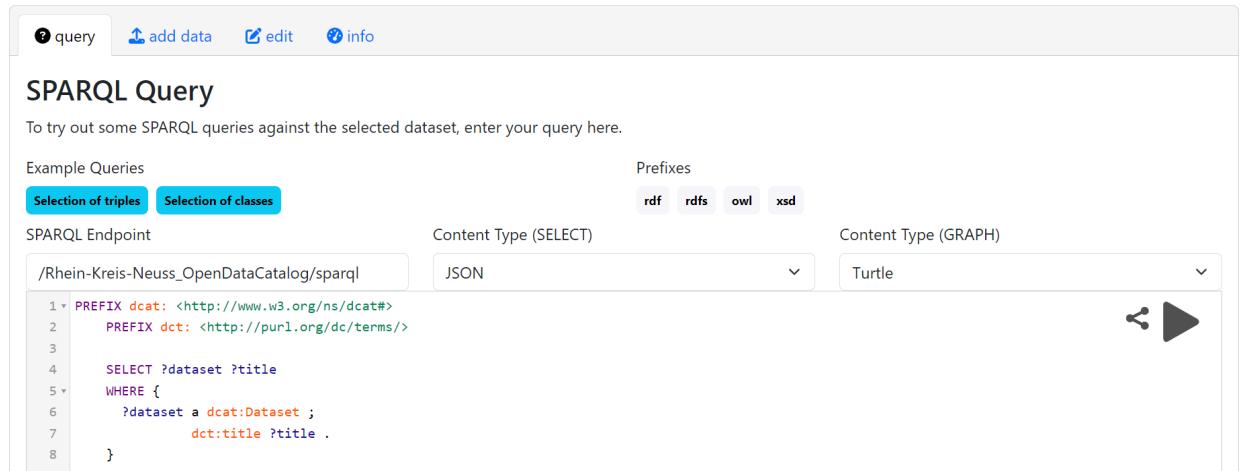
We will start with a basic query to retrieve all datasets and their titles from the metadata.

```
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?dataset ?title
WHERE {
  ?dataset a dcat:Dataset ;
    dct:title ?title .
}
```

Copy the highlighted text above and paste it into the Query text area of Apache Jena Fuseki server as shown in Figure 20.

/Rhein-Kreis-Neuss_OpenDataCatalog



The screenshot shows the Apache Jena Fuseki interface for the dataset '/Rhein-Kreis-Neuss_OpenDataCatalog'. The top navigation bar includes links for 'query', 'add data', 'edit', and 'info'. Below the navigation is a 'SPARQL Query' section. It contains a text input field for entering SPARQL queries, which currently displays the query from the previous code block. To the right of the input field are tabs for 'Content Type (SELECT)' (set to 'JSON') and 'Content Type (GRAPH)' (set to 'Turtle'). Above the input field are sections for 'Example Queries' (with tabs for 'Selection of triples' and 'Selection of classes', the latter being selected) and 'Prefixes' (with buttons for 'rdf', 'rdfs', 'owl', and 'xsd'). On the far right of the interface are two icons: a left arrow and a right arrow.

Fig. 20: Adding a SPARQL query to the Query text area of the Fuseki server.

This basic query retrieves all datasets and their titles from the DCAT metadata. The query uses the *dcat:Dataset* class to identify datasets and the *dct:title* property to retrieve the titles.

PREFIX: These lines define the namespaces for the DCAT and Dublin Core terms vocabularies, allowing us to use short forms like `dcat:` and `dct:`.

SELECT: This specifies the variables we want to retrieve, in this case, the dataset (`?dataset`) and its title (`?title`).

WHERE: This part of the query specifies the pattern to match in the RDF data. Here, we're looking for triples where the subject is a dcat:Dataset and has a dct:title property.

Figure 21 shows the some results of executing the query.

Table		Response	301 results in 0.241 seconds	Simple view	Ellipsis	Filter query results	Page size: 50	Download	?
		dataset			title				
1	<https://opendata.rhein-kreis-neuss.de/explore/dataset/dormagen-integrationsratswahl-2014-Stimmbezirk@de>		"Dormagen: Integrationsratswahl-2014-Stimmbezirk"@de						
2	<https://opendata.rhein-kreis-neuss.de/explore/dataset/daten/>		"FTP Daten"@de						
3	<https://opendata.rhein-kreis-neuss.de/explore/dataset/dormagen-wahllokale-integrati...>		"Dormagen: Wahllokale-Integrationsratswahl-2020"@de						
4	<https://opendata.rhein-kreis-neuss.de/explore/dataset/dormagen-landratsstichwahl-2...>		"Dormagen: Landratsstichwahl-2020-Gemeindeergebnis"@de						
5	<https://opendata.rhein-kreis-neuss.de/explore/dataset/dormagen-buergerentscheid-2...>		"Dormagen: Bürgerentscheid-2012-11-25-Kommunalwahlbezirk"@de						
6	<https://opendata.rhein-kreis-neuss.de/explore/dataset/rhein-kreis-neuss-aufbauorgan...>		"Rhein-Kreis Neuss: Aufbauorganisation"@de						
7	<https://opendata.rhein-kreis-neuss.de/explore/dataset/dormagen-bundestagswahl-20...>		"Dormagen: Bundestagswahl-2013-Kommunalwahlbezirk"@de						
8	<https://opendata.rhein-kreis-neuss.de/explore/dataset/dormagen-kreistagswahl-2014...>		"Dormagen: Kreistagswahl-2014-Stadt-Dormagen"@de						
9	<https://opendata.rhein-kreis-neuss.de/explore/dataset/dormagen-landratswahl-2020...>		"Dormagen: Landratswahl-2020-Gemeindeergebnis"@de						
10	<https://opendata.rhein-kreis-neuss.de/explore/dataset/dormagen-landratsstichwahl-2...>		"Dormagen: Landratsstichwahl-2020-Wahlbezirke"@de						
11	<https://opendata.rhein-kreis-neuss.de/explore/dataset/stadt-neuss-statistischer-bezir...>		"Stadt Neuss: Statistischer Bezirk 08 - Uedesheim"@de						
12	<https://opendata.rhein-kreis-neuss.de/explore/dataset/schuelerstatistik-oktober/>		"Dormagen: Schülerstatistik Oktober"@de						

Fig. 21: Results from the SPARQL query showing all datasets in the uploaded metadata.

Executing the query returns 301 datasets. Since the datasets are too many to fit in one view, the results table is paginated. These datasets represent or describe the various actual datasets that are available through the open data portal of Rhein-Kreis-Neuss district. We are now going to look deeper for some individual datasets or datasets that meet a particular criteria.

In the second query, we are going to retrieve dataset(s) with a specific keyword, in this case, datasets with the “windkraft” keyword. Remember, a keyword is a specific term or phrase that describes a particular aspect of the dataset and is used to facilitate searching and filtering datasets.

```
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?dataset ?title ?keyword
WHERE {
?dataset a dcat:Dataset ;
```

```

dct:title ?title ;
dcat:keyword ?keyword .
FILTER (CONTAINS(LCASE(?keyword), "windkraft"))
}

```

This query retrieves dataset(s) that contain a specific keyword, "windkraft", along with their titles and keywords.

FILTER: The FILTER function is used to apply a condition to the results. CONTAINS checks if the keyword includes the term "windkraft", while LCASE ensures that the filtering is case-insensitive.

Executing the above query produces the result shown in Figure 22.



The screenshot shows a SPARQL query results interface. The query is:

```

1+ PREFIX dcat: <http://www.w3.org/ns/dcat#>
2+ PREFIX dct: <http://purl.org/dc/terms/>
3+
4+ SELECT ?dataset ?title ?keyword
5+ WHERE {
6+   ?dataset a dcat:Dataset ;
7+     dct:title ?title ;
8+     dcat:keyword ?keyword .
9+   FILTER (CONTAINS(LCASE(?keyword), "windkraft"))
10+
}

```

The results table has columns: dataset, title, and keyword. One row is shown:

dataset	title	keyword
< https://opendata.rhein-kreis-neuss.de/api/v2/catalog/datasets/rhein-kreis-neuss-windenergieanlagen	Rhein-Kreis Neuss: Windenergieanlagen	windkraft

Showing 1 to 1 of 1 entries

Fig. 22: Results from the SPARQL query showing all datasets with the keyword 'windkraft'

The results show that *Rhein-Kreis Neuss: Windenergieanlagen* is the only dataset that meets the criteria. Such a query is important for searching through metadata that has multiple datasets, for example, the metadata catalog of an entire data portal.

Task

Open the *Rhein-Kreis-Neuss_OpenDataCatalog* metadata that we uploaded to the server on a text editor, for example Notepad. Copy one of the keywords in the windturbines metadata and replace it in SPARQL query above. Execute the query and view the results.

In the third query, we are going to extract dataset(s) and their distributions. Remember, distribution refers to a specific representation of a dataset. The information about the distribution helps the user to

```
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?dataset ?title ?distribution
WHERE {
  ?dataset a dcat:Dataset ;
    dct:title ?title ;
    dcat:distribution ?distribution .
}
```

This query retrieves datasets along with their titles and distributions.

dcat:distribution: This property links datasets to their distributions. The query retrieves the URI of the distribution, which can provide further details like the format and access URL.

Executing the query produces the results shown in Figure 23.

Table		Response	3870 results in 0.423 seconds	Simple view	<input checked="" type="checkbox"/> Ellipse	<input type="checkbox"/> Filter query results	Page size:	50		
		dataset	title			distribution				
1<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
2<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
3<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
4<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
5<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
6<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
7<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
8<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
9<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
10<	https://opendata.rhein-kreis-neuss.de/explore/ > "Dormagen: Integrationsratswahl-2014-S... < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									
11<	https://opendata.rhein-kreis-neuss.de/explore/ > "FTP Daten"@de < https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/d...									

Fig. 23: Results from the SPARQL query that shows all distributions of all datasets

From the results, there are 3870 distributions belonging to the various datasets that are managed by the open data catalogue. However, these results are not very useful when we are interested in a particular dataset. Let's assume we are interested in the

Rhein-Kreis Neuss: Windenergieanlagen dataset that we initially filtered using the keyword. We want to know what distributions are available for that dataset. We can do this by:

```
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?dataset ?title ?distribution
WHERE {
  ?dataset a dcat:Dataset ;
    dct:title ?title ;
    dcat:keyword ?keyword ;
    dcat:distribution ?distribution .
  FILTER (CONTAINS(LCASE(?keyword), "windkraft"))
}
```

The above query retrieves all distributions that are available for the *Rhein-Kreis Neuss: Windenergieanlagen* dataset.

dcat:distribution property is used to get the distribution details of the filtered dataset(s).

Running the query produces the results shown in figure 24.

dataset	title	distribution
1< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
2< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
3< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
4< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
5< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
6< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
7< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
8< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
9< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	
10< https://opendata.rhein-kreis-neuss.de/explore/data... >	"Rhein-Kreis Neuss: Winde..."< https://opendata.rhein-kreis-neuss.de/api/explore/v2.1/catalog/datasets/rhei... >	

Fig. 24: Results from the SPARQL query that shows all distributions from the *Rhein-Kreis Neuss: Windenergieanlagen* dataset

The *Rhein-Kreis Neuss: Windenergieanlagen* dataset has 15 different distributions. However, this information is not very useful to a user who wants to use the dataset, as there is no mention of the distribution names and the associated information.

We can go further and get more information about the distribution formats, including their names and access URLs.

```

PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX dct: <http://purl.org/dc/terms/>

SELECT ?dataset ?datasetTitle ?distribution ?distributionTitle ?accessURL
WHERE {
    ?dataset a dcat:Dataset ;
        dct:title ?datasetTitle ;
        dcat:distribution ?distribution ;
        dcat:keyword ?keyword .

    OPTIONAL { ?distribution dct:title ?distributionTitle . }
    ?distribution dcat:accessURL ?accessURL .

    FILTER (CONTAINS(LCASE(?keyword), "windkraft"))
}

```

This advanced query retrieves dataset(s) of interest, their titles, the titles of their distributions (if available), and the access URLs for those distributions. It first gets the details of all the distributions for all the datasets. It then filters them down to only retain the distributions of those datasets that have the keyword *windkraft*.

OPTIONAL: This clause is used to include optional information. In this case, it retrieves the title of the distribution if it exists. If a distribution does not have a title, it will still be included in the results without discarding the entire result.

dcat:accessURL: This property gives the URL where the distribution can be accessed or downloaded.

Executing the query produces the results shown in Figure 25.

dataset	datasetTitle	distribution	distributionTitle	accessURL
1< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
2< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
3< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
4< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
5< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
6< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
7< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
8< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
9< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
10< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...
11< https://opendata.rhein-kr...	"Rhein-Kreis Neuss: ...	< https://opendata.rhein-kreis-neuss.de/...	"rhein-kreis-neuss-winden...	< https://opendata.rhein-kreis-neuss...

Fig. 25: Results from the SPARQL query that shows all distributions of the dataset, including their titles and access URLs.

We now see the titles of the various distributions and their access URLs.

4. Summary and Discussion

In this tutorial, we explored DCAT and its significance in enhancing the discoverability, accessibility, and interoperability of datasets through standardized metadata descriptions. DCAT provides a uniform way of cataloging and describing datasets, making it easier for users to find and use data across different platforms.

Throughout the module, we delved into the core elements of DCAT, including mandatory elements such as titles, descriptions, and distribution details, as well as optional elements like spatial and temporal coverage. By adhering to these elements, data providers can ensure that their metadata is comprehensive and useful for a wide range of applications.

The practical exercise of creating, editing, validating and querying DCAT metadata using tools such as RDForms and SPARQL demonstrated the hands-on process of metadata management. RDForms simplifies the creation and maintenance of DCAT-compliant metadata, allowing users to focus on the content rather than the technicalities of metadata standards. Validation tools such as the DCAT-AP.de validator further ensure that the metadata conforms to established standards, promoting consistency and reliability across data portals. SPARQL — the powerful query language designed for RDF data — plays a crucial role by facilitating advanced queries on DCAT metadata. This allows users to extract and manipulate the data effectively. Apache Jena Fuseki, an open-source server for managing and querying RDF data, further enhances this capability by providing a robust platform for executing SPARQL queries.

The case study of the wind turbines dataset from the Rhein-Kreis Neuss district illustrated how DCAT can be applied in real-world scenarios. The dataset's metadata, available in multiple formats, is accessible through various URLs and is integrated into larger data portals like [govdata.de](#). This federated approach enhances data discoverability and access, demonstrating the practical benefits of adopting DCAT standards.

The following resources may help you to further dive into the topic and deepen your understanding.

- 1) W3C official documentation for DCAT.
<https://www.w3.org/TR/vocab-dcat-3/>
- 2) Linked data
<https://www.w3.org/wiki/LinkedData>
- 3) Getting started with Docker
<https://docker-curriculum.com/>
- 4) Resource Description Framework (RDF)
<https://www.w3.org/TR/rdf12-concepts/>