



SHAPEness Metadata Editor

Getting Started

Title	SHAP <i>Ess</i> Metadata Editor – Getting Started
Abstract	This guide provides steps for installing SHAP <i>Ess</i> Metadata Editor and starting using it.
Date	24/11/2020
Version	1.0
Author	Rossana Paciello (rossana.paciello@ingv.it)
Contributors	Manuela Sbarra (manuela.sbarra@ingv.it) Valerio Vinciarelli (valerio.vinciarelli@ingv.it) Daniele Bailo (daniele.bailo@ingv.it)

Table of contents

1	Welcome to SHAPEness Metadata Editor	4
1.1	About SHAPEness Metadata Editor	4
1.2	Key Concepts and Terminology	4
DCAT-AP and its extensions		4
Resource Description Framework (RDF)		5
SHACL constraints		5
Turtle serialization		5
2	Downloading and installing SHAPEness Metadata Editor	6
2.1	Minimum System Requirements	6
2.2	Installing on Windows	6
2.3	Installing on Mac	6
2.4	Installing on Linux	6
3	Quick start	7
3.1	Graphical User Interface layout	7
3.2	Sample Data	13
4	Using SHAPEness Metadata Editor	14
4.1	Creating a new project	15
4.2	Opening an existing project	17
4.3	Working with nodes and relationships	18
Creating new node		18
Filling in node properties		20
Creating relationships between nodes		21
Handling errors and warnings		23
Managing Geospatial information		25

Filtering nodes on the graph	26
Managing nodes colours	27
4.4 Importing metadata from Datacite.....	28
4.5 Exporting Turtle File	29
Saving Turtle file	29
Pushing Turtle file to a GitLab repository	30
4.6 Troubleshooting.....	31
5 Getting support.....	39

1 Welcome to SHAP*Ess* Metadata Editor

Welcome to SHAP*Ess* Metadata Editor. This guideline helps users get started with SHAP*Ess* Metadata Editor. The next sections describe key concepts, introduce the graphical user interface and provide a tutorial that guides users performing the common operations.

1.1 About SHAP*Ess* Metadata Editor

The SHAP*Ess* Metadata Editor is a Java desktop application conceived to help users creating and updating RDF metadata descriptions. It provides a graph-based interface which allows users to easily populate metadata descriptions as data graphs, validate them against SHACL constraints, and serialize them in RDF/Turtle format.

The SHAP*Ess* Metadata Editor has been developed in the framework of the European Plate Observing System (EPOS) where an extension of DCAT-AP, called EPOS-DCAT-AP, was created (<https://github.com/epos-eu/EPOS-DCAT-AP>). However, as it is a SHACL driven Metadata Editor, it can be easily used by any application profile as long as accompanied with SHACL constraints.

1.2 Key Concepts and Terminology

Before to start working with SHAP*Ess* Metadata Editor, it is useful to understand the following concepts and terminology.

DCAT-AP and its extensions

The DCAT Application Profile for data portals in Europe (DCAT-AP) is a specification based on W3C Data Catalog Vocabulary (DCAT) for describing metadata of public sector datasets in Europe. The specification allows to extend it by the creation of Application Profiles. The extensions re-use terms from one or more standard vocabularies, and add more specificity by identifying mandatory, recommended and optional elements to be used for a domain-specific application.

Resource Description Framework (RDF)

DCAT data model is represented in Resource Description Framework (RDF). RDF data is commonly represented in the form of triples, comprising three components: subject, predicate, and object. The subject and object of a triple can be regarded as concepts (entities), or literals such as strings or numbers. The predicate can be regarded as the “label” of the edge, capturing the semantics of the expressed relationship. By considering triples as directed edges, an RDF dataset naturally becomes a directed graph where the subjects and the objects are nodes while the predicates are edges.

SHACL constraints

A W3C recommendation defines Shapes Constraint Language (SHACL) as language for validating RDF graphs against a set of constraints. These constraints are provided as shapes which are used to define classes together with constraints on their properties (e.g., cardinality, value type, allowed values, etc.).

Turtle serialization

RDF graphs can be serialized in different formats. Turtle, which stands for “Terse RDF Triple Language”, is one of them. It has textual syntax which allows RDF graphs to be completely written in a compact text form, with shorten for common usage patterns and datatypes. RDF files in Turtle serialization are usually given the file extension “.ttl”.

2 Downloading and installing SHAP*En*ess Metadata Editor

The SHAP*En*ess Metadata Editor is available for Windows, Mac and Linux operating systems. To get the latest version of the SHAP*En*ess Metadata Editor, visit the web page (<https://epos-eu.github.io/SHAPEness-Metadata-Editor/index.html>) and click on Download to access the download links for your platform.

2.1 Minimum System Requirements

- **Windows:** Windows 8
- **Mac:** OS X El Capitan 10.11.0
- **Linux:** Debian 8, Ubuntu 16.04, CentOS 7

2.2 Installing on Windows

- **Extract** the contents of the zip file to a folder on your computer.
- **Double-click** on the SHAP*En*ess Metadata Editor executable file to launch the application.

2.3 Installing on Mac

- **Double-click** on the PKG file to begin installing SHAP*En*ess Metadata Editor.
- Click **Change Install Location** if you want to choose another location.
- You may be asked to enter your Mac OS X password to complete the installation.
- Click **install** software to continue.
- Click **close** to close the setup wizard.
- The SHAP*En*ess Metadata Editor is added automatically to your Applications folder.

2.4 Installing on Linux

- **Extract** the contents of the zip file to a folder on your computer.
- **Double-click** on the SHAP*En*ess Metadata Editor executable file to launch the application.

3 Quick start

3.1 Graphical User Interface layout

The graphical user interface (GUI) has several views designed to cover different aspects of the SHAPENess Metadata Editor (Figure 1) such as creating graph, editing node properties, filtering nodes/relationships and displaying RDF/Turtle serialization. The GUI provides a menu bar and a tool bar at the top of the main window. It also provides a toolbar at top of some views (e.g., Graph, Palette), as well as context menus which are available by right-clicking on various objects in the views (e.g., nodes, relationships, shapes).

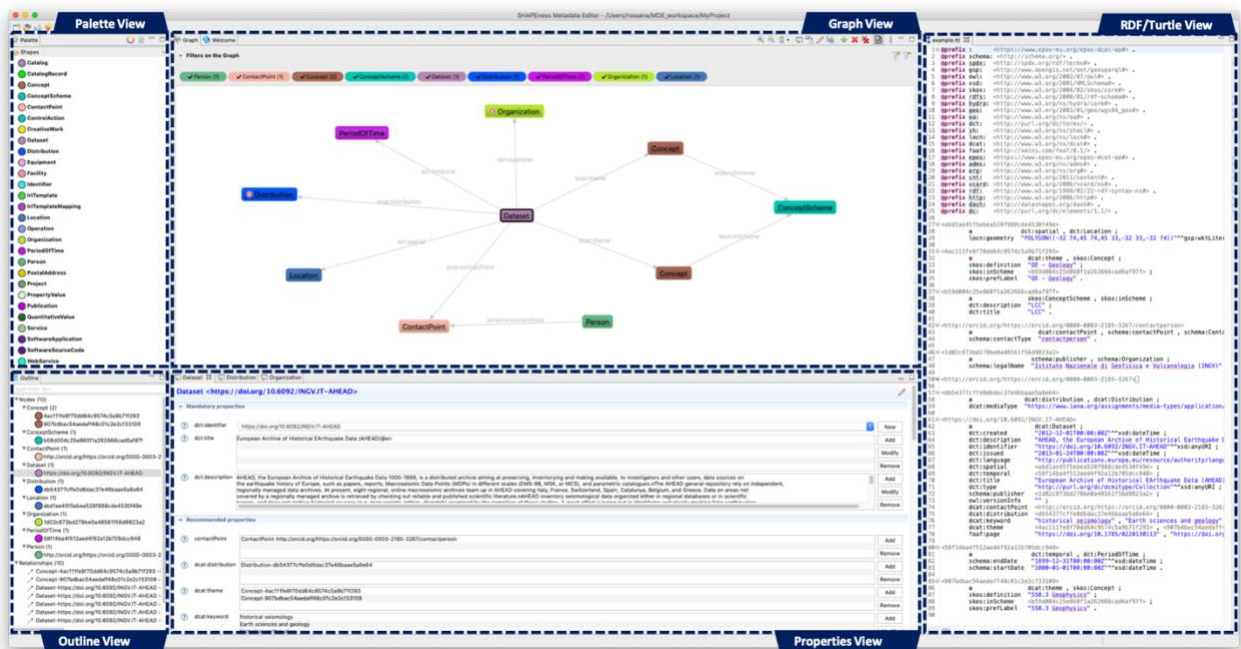


Figure 1 - SHAPENess Metadata Editor main window

Palette View - This view lists the shapes defined in the SHACL constraints (Figure 2). It allows users to create nodes by simply dragging the shapes from this view to the Graph View, or in alternative way, by right-clicking on the shapes.

The colours of the shapes affect nodes colours on the graph according to the shape type. It is possible to customize these colours at any time by using the palette toolbar or the context menu on the shapes.



Figure 2 - Palette View

Outline View - This view lists all nodes and relationships present on graph (Figure 3). The nodes are grouped according to the shape type and are displayed in a tree-like structure. The view is automatically updated according to changes made to the graph. Nodes and relationships can be searched by using a free-text search on top of this view.

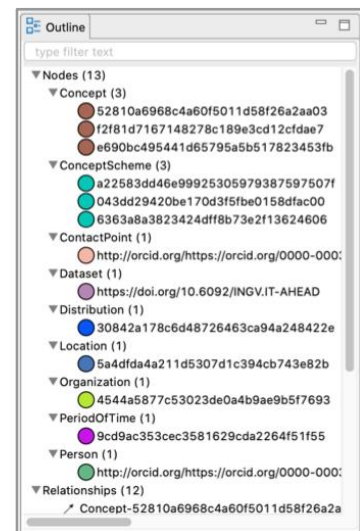


Figure 3 - Outline View

Graph View - This view is a graph representation of metadata entities (Figure 4). Nodes (oval shapes) represent entities connected together by edges (relationships), that are directed from one node to another. Nodes are labelled and coloured according to their shape type. Edges are labelled using relationships names. The graph is interactive, so nodes and edges can be moved around for better inspection and selection. Only one node or edge can be selected at a time and it will be highlighted with bold border line.

This view also includes:

- a toolbar which provides access to the common commands on the graph (e.g., *zoom in*, *zoom out*, *change graph layout*, *open node properties*, *rename node ID*, *save graph as image*, *add new nodes*, *remove node or connection*, *remove all nodes*, etc.);
- a filter panel in order to show or hide nodes on the graph;
- a context menu on nodes and edges which gives users a shortcut to frequently used commands.

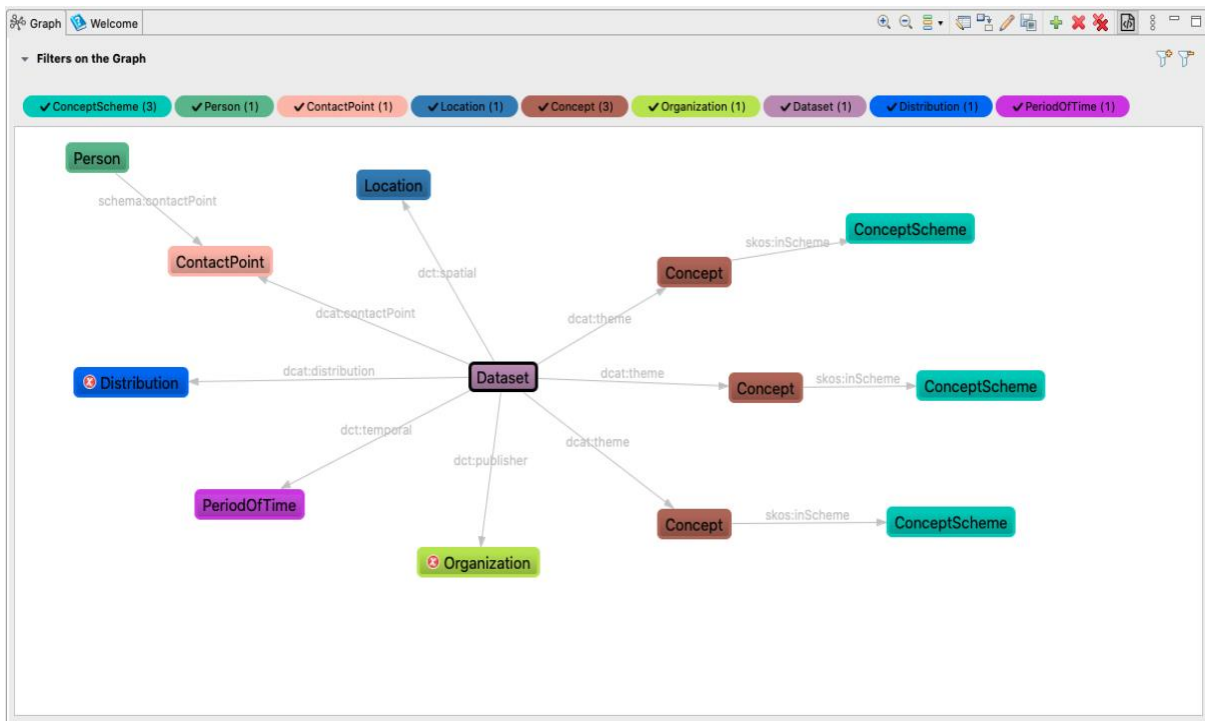


Figure 4 - Graph View

Properties View - This view is used to visualise and modify the properties of a currently selected node (Figure 5). It contains three sections in order to group the properties by mandatory, recommended and optional as they are defined in the SHACL constraints. Properties are labelled by their vocabulary term the meaning of which is provided by help button close to them. By default, all properties views, related to the nodes on the graph, are shown on the bottom of the Graph View.

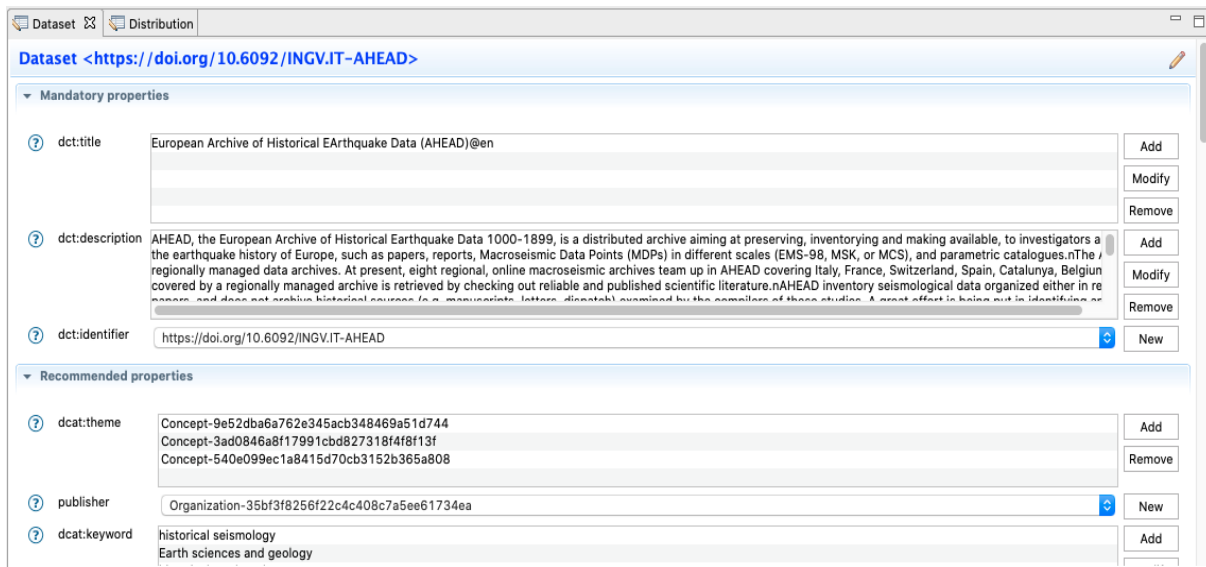


Figure 5 - Properties View

RDF/Turtle View - This view shows the data graph serialised as Turtle file (Figure 6). It features syntax highlighting, syntax validation, internal linking to descriptions, preview of resources, navigation in outline and quick outline, folding (prefixes, subject blocks, multiline literals). By default, this view is disabled. User can enable or disable it by using the command in the main menu or the icon on the graph toolbar. When the view is enabled, the application goes into a split-screen mode and shows the Turtle code dynamically generated according to the changes made to the Graph View or Properties View.

```
26 1# @prefix : <https://www.epos-eu.org/epos-dcat-ap#> .
27 27# <30842a178c6d48726463ca94a24842e>
28   a          dcat:distribution , dcat:Distribution ;
29   dcat:mediaType "https://www.iana.org/assignments/media-types/application/xml" .
30
31# <e690bc495441d65795a5b517823453fb>
32   a          dcat:theme , skos:Concept ;
33   skos:definition "QE - Geology" ;
34   skos:inScheme <3636a3a3823424dffb73e2f13624606> ;
35   skos:prefLabel "QE - Geology" .
36
37# <52818a6968c4a60f5011d58f26a2ae83>
38   a          dcat:theme , skos:Concept ;
39   skos:definition "550.3 Geophysics" ;
40   skos:inScheme <a22583d44e99925305979387597507f> ;
41   skos:prefLabel "550.3 Geophysics" .
42
43# <http://orcid.org/https://orcid.org/0000-0003-2185-3267/contactperson>
44   a          schema:contactPoint , dcat:contactPoint , schema:ContactPoint ;
45   schema:contactType "contactperson" .
46
47# <http://orcid.org/https://orcid.org/0000-0003-2185-3267>
48
49
50
51
52
53
54# <f2f81d7167148278c189e3cd12cfdae7>
55   a          dcat:theme , skos:Concept ;
56   skos:definition "551 Geology, hydrology, meteorology" ;
57   skos:inScheme <04363a8a3823424dffb73e2f13624606> ;
58   skos:prefLabel "551 Geology, hydrology, meteorology" .
59
60# <5a4dfda4a211d5307d1c394cb743e82b>
61   a          dct:spatial , dct:Location ;
62   locn:geometry "POLYGON((-32 74,45 74,45 33,-32 33,-32 74))"^^xsd:wktLiteral .
63
64# <9cd9ac353cec3581629cda2264f51f55>
65   a          dct:temporal , dct:PeriodOfTime ;
66   schema:endDate "1899-12-31T00:00:00Z"^^xsd:dateTime ;
67   schema:startDate "1000-01-01T00:00:00Z"^^xsd:dateTime .
68
69# <https://doi.org/10.6092/INGV.IT-AHEAD>
70   a          dcat:Dataset ;
71   dct:created "2012-12-01T00:00:00Z"^^xsd:dateTime ;
72   dct:description "AHEAD, the European Archive of Historical Earthquake Data 1000-1899,
73   dct:identifier "https://doi.org/10.6092/INGV.IT-AHEAD"^^xsd:anyURI ;
74   dct:issued "2013-01-24T00:00:00Z"^^xsd:dateTime ;
75   dct:language "https://publications.europa.eu/resource/authority/language/ENG" ;
76   dct:spatial <5a4dfda4a211d5307d1c394cb743e82b> ;
77   dct:temporal <9cd9ac353cec3581629cda2264f51f55> ;
78   dct:title "European Archive of Historical Earthquake Data (AHEAD)@en" ;
79   dct:type "http://purl.org/dc/dcmitype/Collection"^^xsd:anyURI ;
80   schema:publisher <4544a5877c53023de8a4b9ae9b5f7693> ;
81   owl:versionInfo "" ;
82   dcat:contactPoint <https://orcid.org/https://orcid.org/0000-0003-2185-3267/contactperson>
83   dcat:distribution <30842a178c6d48726463ca94a24842e> ;
84   dcat:keyword "historical seismology" , "Earth sciences and geology" , "historical e
85   dcat:theme <f2f81d7167148278c189e3cd12cfdae7> , <e690bc495441d65795a5b517823453fb>
86   foaf:page "https://doi.org/10.1785/0220130113" .
87
88# <4544a5877c53023de8a4b9ae9b5f7693>
89
90
91
92# <3636a3a3823424dffb73e2f13624606>
93   a          skos:ConceptScheme , skos:inScheme ;
94   dct:description "LCC" ;
95   dct:title "LCC" .
96
97# <04363a8a3823424dffb73e2f13624606>
98   a          skos:inScheme , skos:ConceptScheme ;
```

Figure 6 - RDF/Turtle View

Layout customization - All views described above can be resized and moved to any of the available positions in the GUI layout. It might be done by clicking and dragging the view tab to another region of the screen. The Figure 7 shows an example of a customized layout.

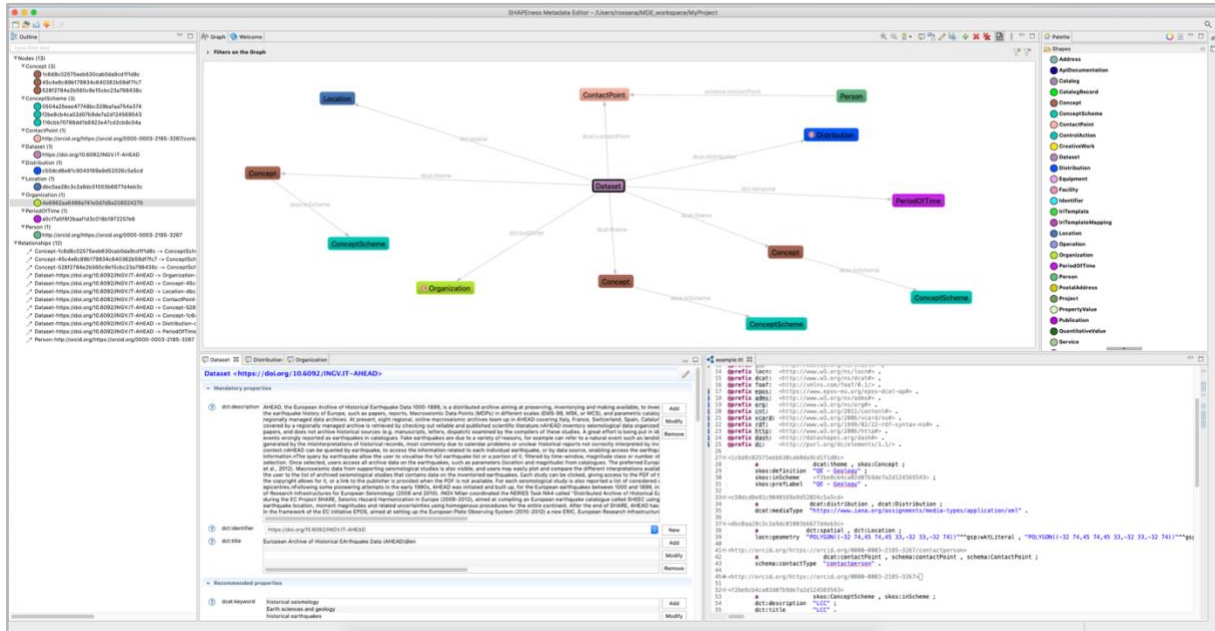


Figure 7 - Example of a customized layout

3.2 Sample Data

Sample data helps users to work with content that is similar to actual data that they would be working with at a later point. In particular, the EPOS-DCAT-AP SHACL file and an EPOS-DCAT-AP Turtle file are available online for downloading in order to provide a hands-on test experience and get familiarized with the SHAP*Eness* Metadata Editor.

SHACL file: https://raw.githubusercontent.com/epos-eu/EPOS-DCAT-AP/EPOS-DCAT-AP-shapes/epos-dcat-ap_shapes.ttl

Turtle file: https://raw.githubusercontent.com/epos-eu/SHAPEness-Metadata-Editor/master/files/sample-data/EPOS-DCAT-AP_example.ttl

4 Using SHAPEness Metadata Editor

The Welcome Page is shown every time SHAPEness Metadata Editor starts (Figure 8). You can follow the links to the information you are interested in, or click a link to perform the task that is most suitable for you.

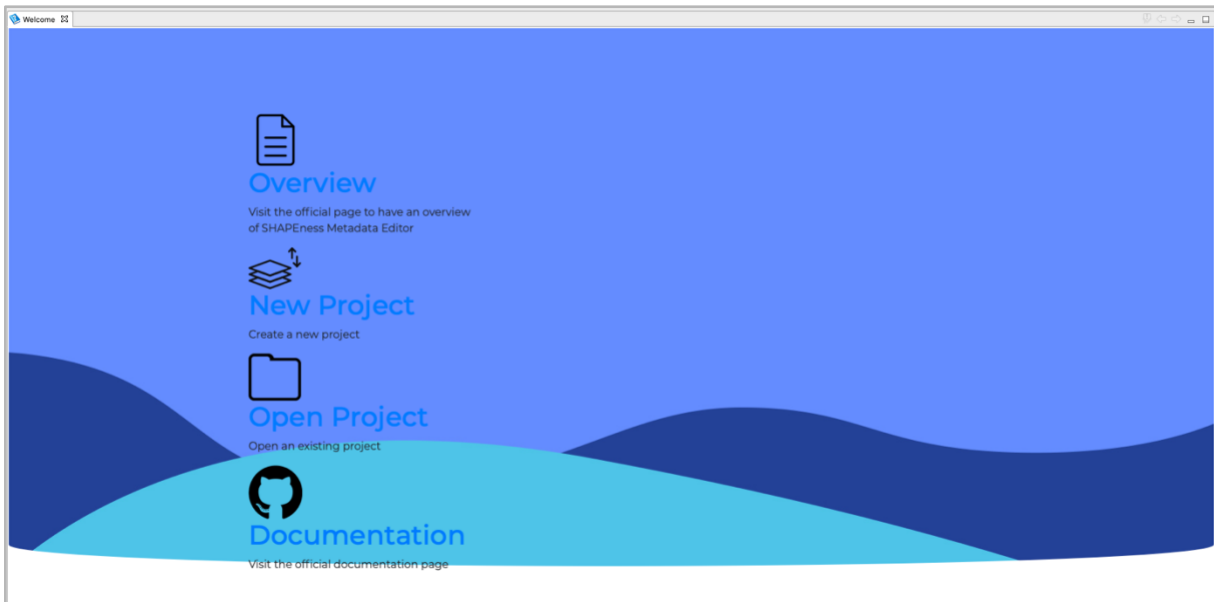


Figure 8 - Welcome page

4.1 Creating a new project

You can create a new standalone project on your computer in order to describe the metadata from scratch or import an existing metadata description serialized as Turtle format.

- Start **SHAPEness Metadata Editor**.
- In the **Welcome page**, click the **New Project** link (or select **Project > New project** on the menu if you already have the program running).
- In the setting for the **New Project wizard** (Figure 9), enter:
 - a project name;
 - URL or local path of SHACL constraints file;
 - if you choose **create a new RDF/Turtle file** option, enter a file name;
 - if you choose **import existing RDF/Turtle file** option, enter a URL or a local path of an existing file. In the latter case, if you import the file from a Git repository you should enter the link to the raw file.

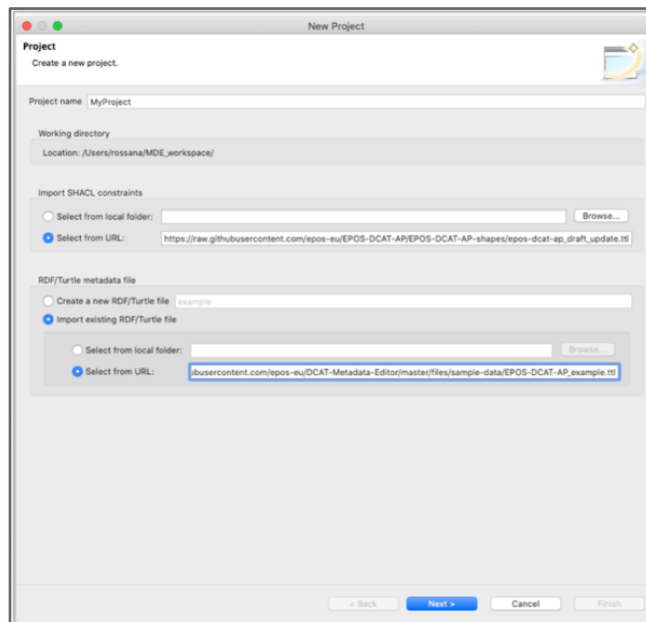


Figure 9 - Creating new project - Settings

- Click **Next**.
- This step uploads the SHACL file previously specified and lists all shapes defined by it. For each shape you can **change the colour** which will be used by the application to represent the nodes of that type (Figure 10).
- Click **Finish** when you are done.

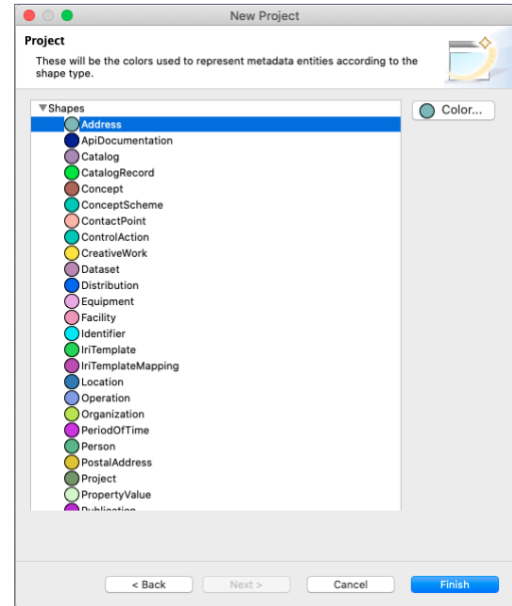


Figure 10 - Creating new project - Changing shapes color

Note: By default, projects are saved to `$User\MDE_workspace\`. For each project, it will be created a folder with the name of the project and two subfolders. **Shacl** folder will contain the SHACL file and all vocabularies needed for the validation. **Metadata** folder will contain the Turtle files generated by the application.

4.2 Opening an existing project

You can open a standalone project saved on your computer.

- Start **SHAPEness Metadata Editor**.
- In the **Welcome page**, click the **Open Project** link (or select **Project > Open Existing Project** on the menu if you already have the program running).
- In the **Open Project** dialog (Figure 11):
 - **browse** to the workspace folder (by default, projects are saved to \$User\MDE_workspace\);
 - **select the project folder** you want to open.
- Click **OK** to confirm and close the dialog.

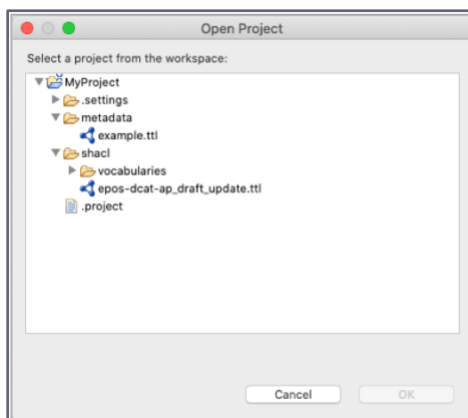


Figure 11 - Opening an existing project

4.3 Working with nodes and relationships

This section describes the common operations that you can perform on nodes and relationships.

Note: changes made to the Graph View and Properties View are reflected to the RDF/Turtle View, but not vice versa. Manual changes on the Turtle file have no effect on Graph View or Properties View.

Creating new node


- On the **Graph toolbar**, select  button.
- In the dialog, **select one or more shape types of nodes** you want to create (Figure 12).
- Click **OK** to confirm and close the dialog.
- For each node you should **confirm** the creation.
- The **new node** will appear at the top left corner of the Graph View represented through an **oval shape coloured** according to the shape type, as well as in the Outline View (Figure 13).



Figure 12 - Creating new nodes by selecting shape types

Tip: You can also create new nodes by double-clicking or right-clicking on the shape item in the Palette View, as well as by dragging-and-dropping shape item from the Palette View to the Graph View. In the latter case the node will appear at the mouse point when drag ended.

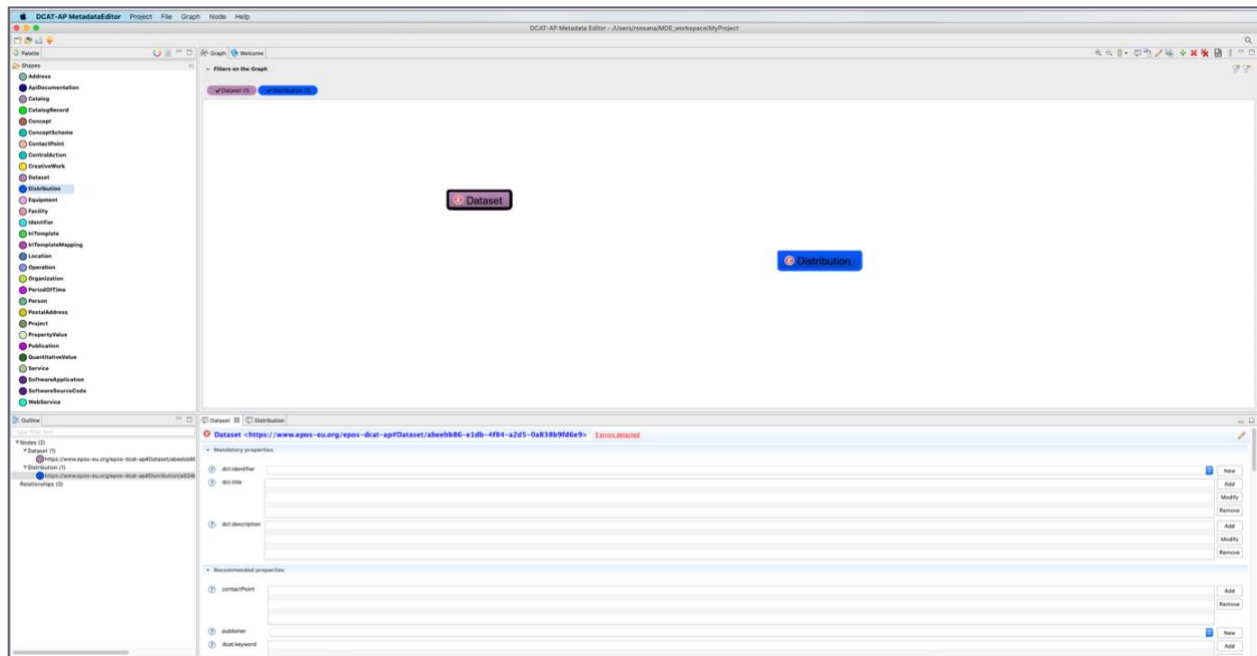
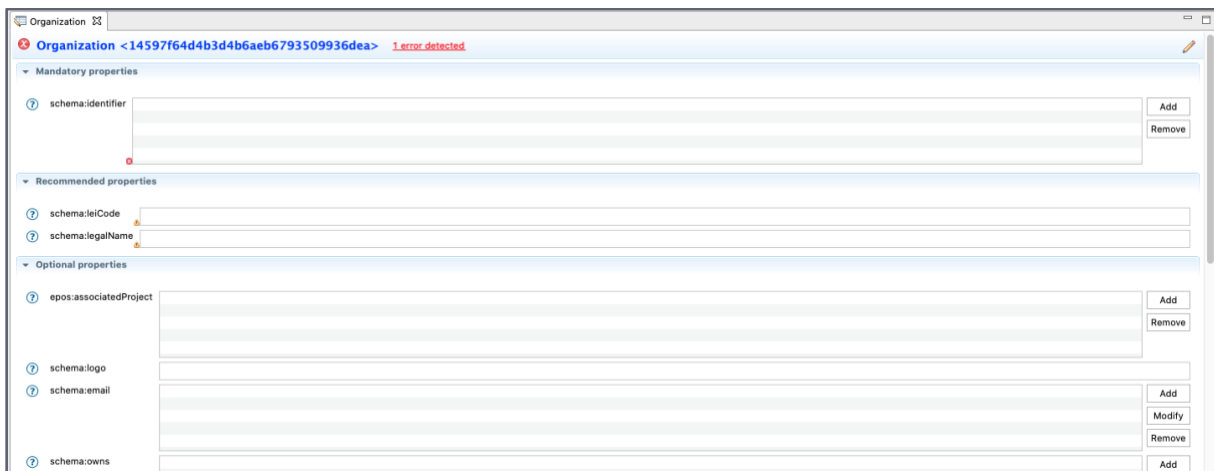


Figure 13 – New nodes on the Graph

Filling in node properties

- Double-click the **node** you are interested in on the **Graph View**.
- The node **Properties View** will be opened (or focused, if it is already open) at the bottom of the Graph View (Figure 14).
- Enter information about the properties which allow to specify a single value for them by using the text field (e.g. *schema:leiCode* property of *schema:Organization* entity).
- Enter information about the properties which allow to specify multiple values for them by clicking on the closer **Add** button (e.g. *schema:email* property of *schema:Organization* entity).
- **Mandatory** and **recommended** properties left empty will be notified with appropriate **error** or **warning** alert icons.

Note: The application manages nodes by unique identifiers (**IDs**). By default, each time a node is created, the application assigns an ID to that. It is shown on top of the node Properties View and it can be modified by using the pencil button on top right.



The screenshot shows a web application window titled 'Organization' with a unique ID and a '1 error detected' message. The interface is organized into three main sections for property management:

- Mandatory properties:** Contains a single text input field for 'schema:identifier'. It has a red error icon below the field and 'Add' and 'Remove' buttons to its right.
- Recommended properties:** Contains two text input fields for 'schema:leiCode' and 'schema:legalName'. Both have yellow warning icons below them.
- Optional properties:** Contains five text input fields for 'epos:associatedProject', 'schema:logo', 'schema:email', and 'schema:owns'. Each has a question mark icon below it. 'epos:associatedProject' has 'Add' and 'Remove' buttons. 'schema:email' has 'Add', 'Modify', and 'Remove' buttons. 'schema:owns' has an 'Add' button.

Figure 14 - Filling in the node properties

Creating relationships between nodes

Relationships between nodes can be created only by using the node Properties View.

- Open the **node Properties View** by double-clicking the node on the **Graph View** (or focused, if it is already open).
- Identify the **property** which represents the relationship you want to create (e.g. *dcat:distribution* property of *dcat:Dataset* entity).
- Click on the closer **Add** button.
- In the dialog you can **choose the type** of the node if you want to create a new one, or alternatively, you can **select an existing node** in the list (Figure 15).

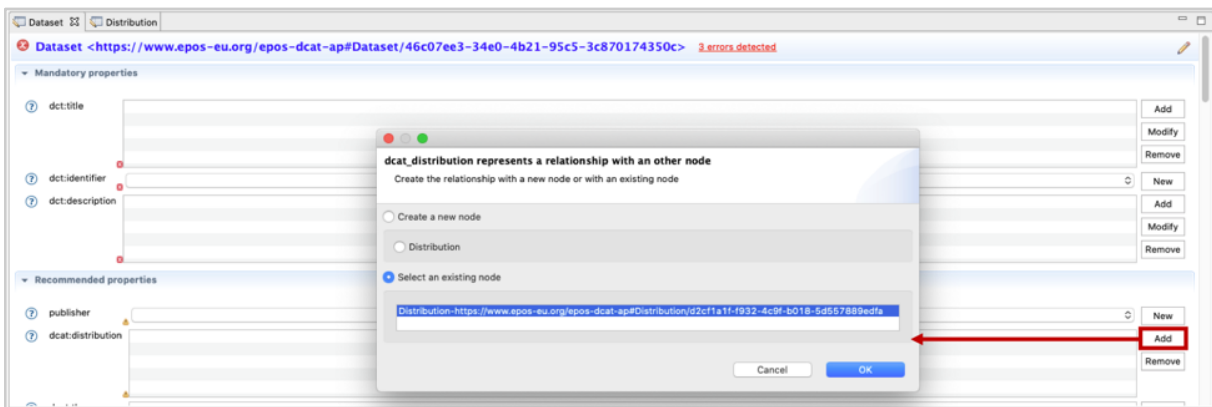


Figure 15 – Creating a relationship

- Click **OK** to close the dialog.
- The property field will contain the ID of the node to which it is linked.
- The relationship will be represented on the Graph View through an arrow, as well as in the Outline View (Figure 16).

Error! Use the Home tab to apply Titolo 1 to the text that you want to appear here.

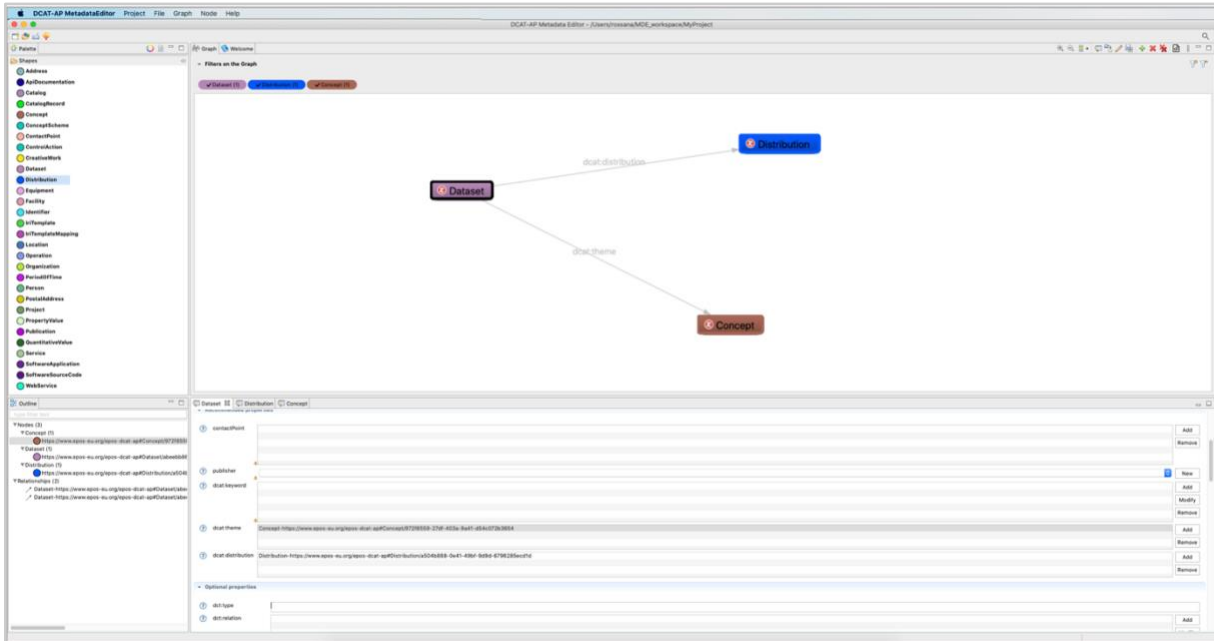




Figure 16 - New relationships between nodes

Handling errors and warnings

The SHAPe^{ness} Metadata Editor supports on-the-fly consistency check of data graph validating it against specified SHACL constraints. The problems found during such validation are of two types: **errors** and **warnings**.

 **Error** – critical problem, related to mandatory properties, that makes the metadata description not valid and should be necessarily fixed.

 **Warning** – not critical problem (an advice), related to recommended properties, that should be fixed in order to improve the quality of the metadata description.

Whenever the validation detects errors or warning, the application will notify them to you via two views:

- The **Graph View** adds **alert icons** inside the oval shapes for those nodes that contain properties for which the validation fails (Figure 17).

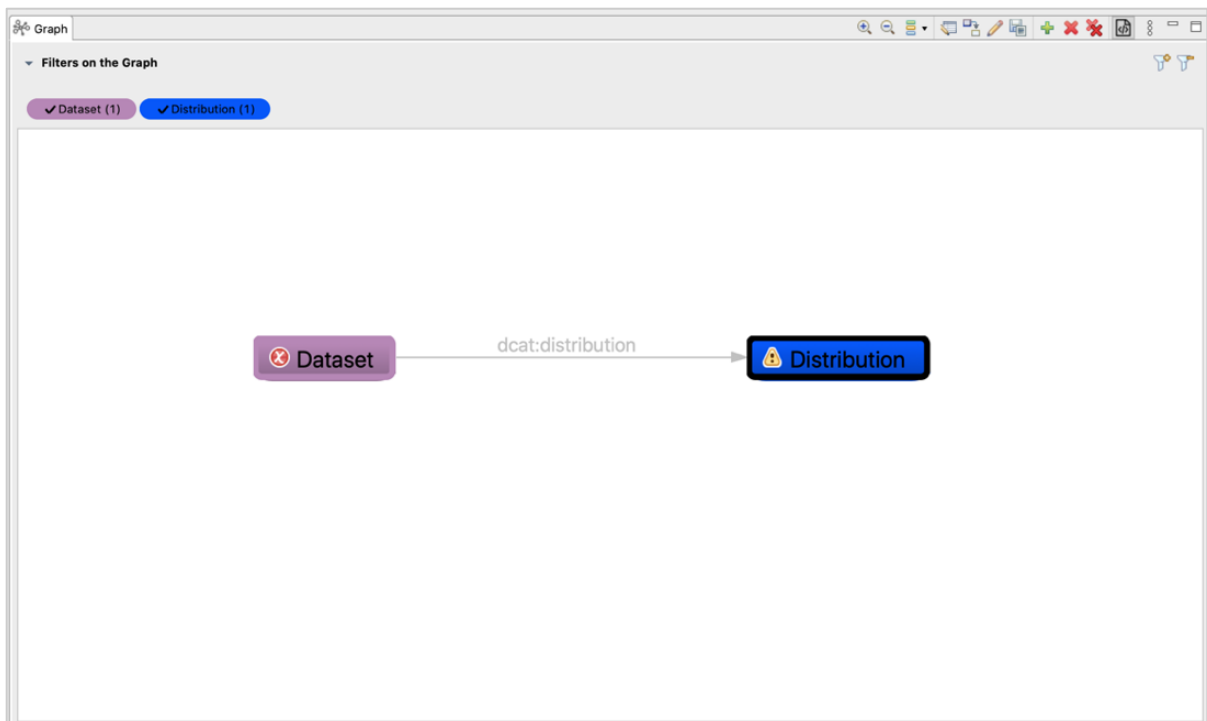


Figure 17 - Graph View with alert icons on the nodes

- The **Properties View** (Figure 18):
 - adds **alert icons** close to the property fields that fail the validation;
 - adds an **error message dialog** which shows the total number of detected problems and a brief description of them at top of the view.

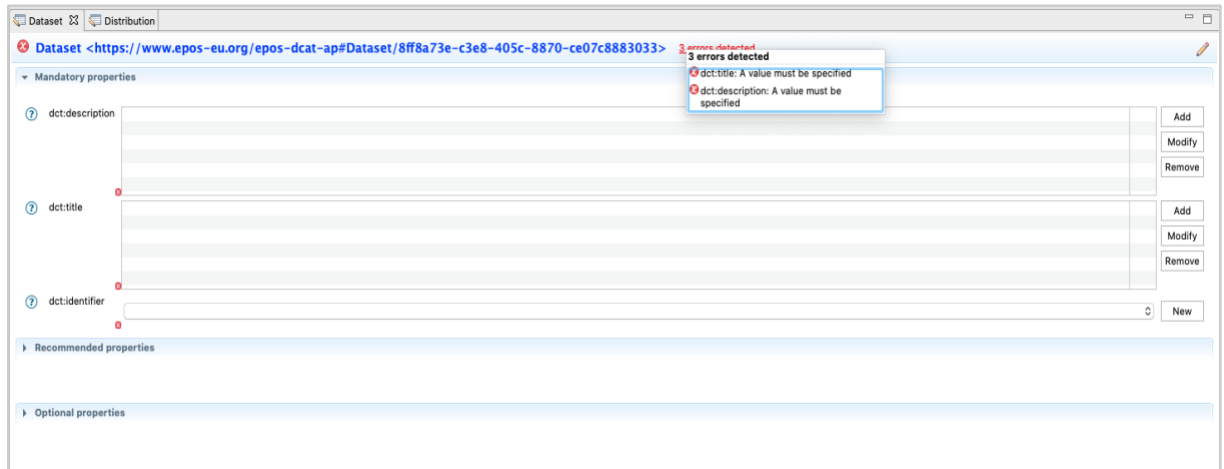


Figure 18 – Properties View with alert icons and error message dialog

Managing Geospatial information

To assist users with entering geospatial information (i.e. Points or Polygons) in some properties, the application provides a **GeoMap dialog**.

- Identify the property on the node Properties View which requires to enter geospatial information (e.g. *locn:geometry property of dct:Location entity*).
- Click on the closer **Add** button.
- In the **GeoMap dialog** (Figure 19), you can draw points or polygons by clicking on the map, or alternatively, you can manually enter geographic coordinates in the bottom text area.
- Click **Show on map** in order to update the map and check the information entered.
- Click **OK** to confirm and close the dialog.

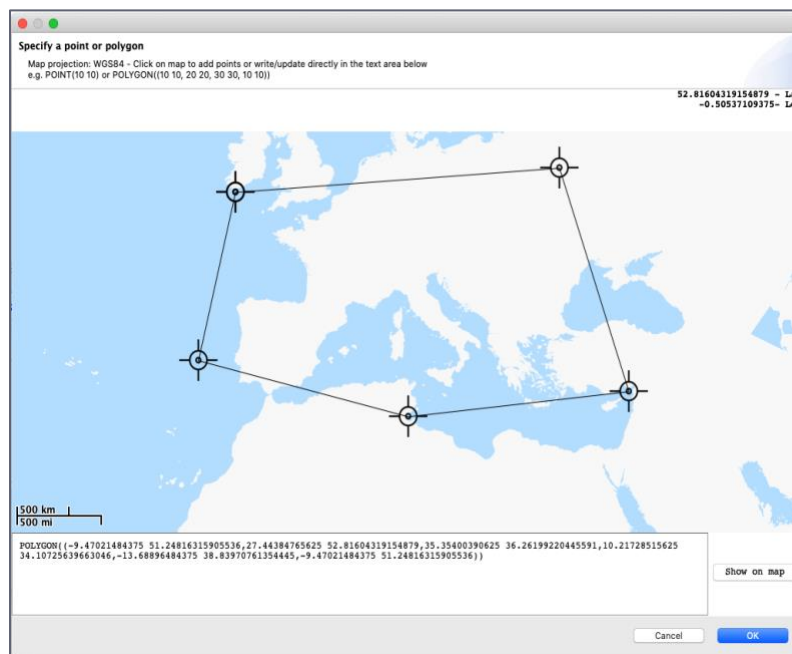




Figure 19 - GeoMap Dialog to enter geospatial information

Filtering nodes on the graph

When you have a lot of nodes on the Graph View, it might be useful to hide or show only some of them.

- Open the **Filter Panel** on top of the graph which let you filter (show/hide) nodes on the graph by shape type (Figure 20).
- Select or deselect the shape type to show or hide nodes of that type.
- Click the  button to show all nodes on the graph.
- Click the  button to hide all nodes on the graph.

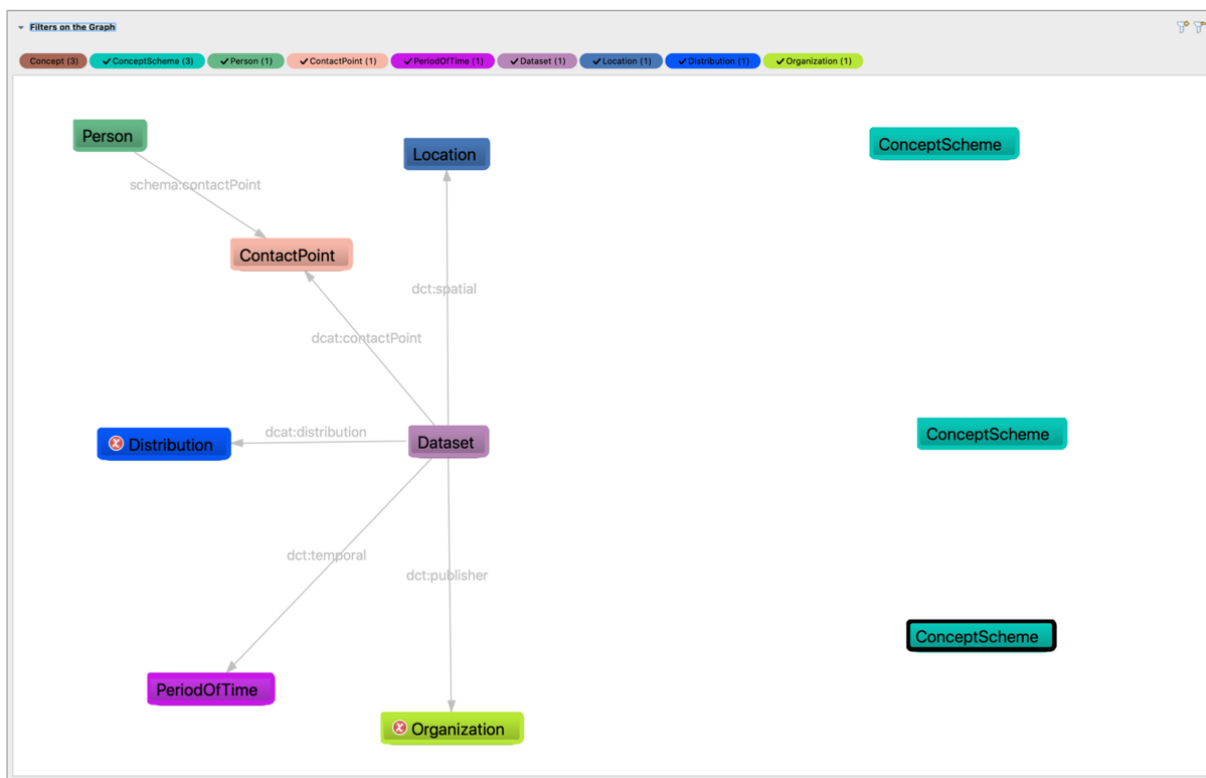



Figure 20 – Filtering nodes on the Graph View by hiding some of them

Managing nodes colours

Nodes on the graph are coloured according to their shape type and you are able to customize these colours.

- On the **Palette toolbar**, select  button.
- In the **Colour Preferences dialog**, change colour for one or more shapes (Figure 21).
- Click **OK** to confirm and close the dialog.
- All nodes on the graph will be coloured with the new colours.

Tip: You can also change the colour of the node by right-clicking on the shape item in the Palette View, as well as by right-clicking on the node in the Graph View.

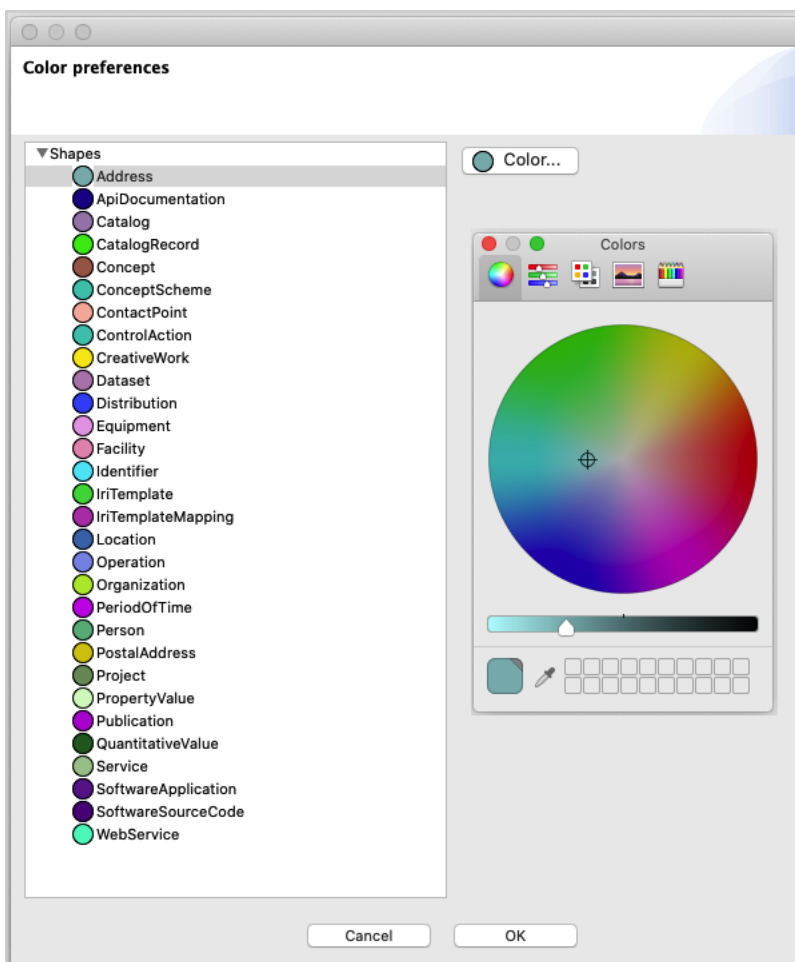



Figure 21 - Managing nodes colours

4.4 Importing metadata from Datacite

DataCite¹ is responsible for issuing persistent identifiers (in particular, DOIs) for datasets, and for registering dataset metadata. Such metadata are provided according to the DataCite metadata schema. You can use these DOIs to allow the SHAPEness Metadata Editor retrieving the associated metadata, creating nodes and relationships, and pre-compiling the nodes properties.

- On the **Graph toolbar**, select  button.
- In the import dialog (Figure 22), enter a **Dataset DOI** (e.g. *10.6092/INGV.IT-AHEAD*).
- Click **OK** to confirm and close the dialog.
- Nodes and relationship created by importing metadata information from DataCite will add on the Graph View, as well as node Properties Views will be pre-compiled.

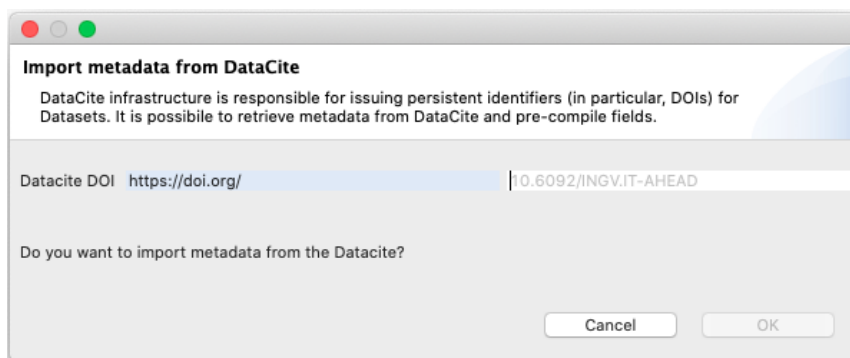


Figure 22 - Importing metadata from Datacite

¹ <https://datacite.org/>

4.5 Exporting Turtle File

By default, the current RDF/Turtle file is automatically saved to `$User\MDE_workspace\ProjectName\metadata\`. You can also save this file in another folder or push it to a Git repository.

Saving Turtle file

- Select **File > Export RDF/Turtle file** on the main menu.
- In the **Export RDF/Turtle file dialog** (Figure 23):
 - **browse** to a location where you want to save the file;
 - **insert** file name.
- Click **Save** button to save the file.

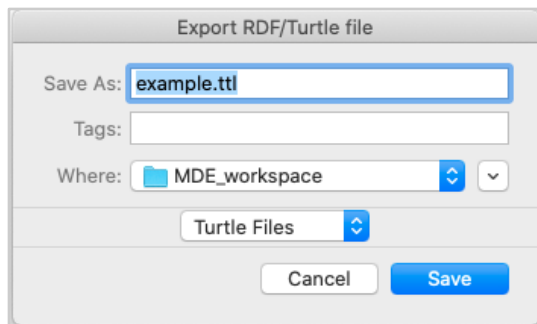


Figure 23 - Export RDF/Turtle file Dialog

Pushing Turtle file to a GitLab repository

- Select **File > Git push** on the main menu.
- In the **Git pushing** dialog (Figure 24), enter:
 - A **Git URL repository** where you want to push the current Turtle file;
 - A **branch name**;
 - A **commit message**;
 - **Username** and **password** which allow the application to authenticate you and perform the push.
 - You can also save these **preferences** that will be remembered later.
- Click **Push** button to confirm and close the dialog.
- A **confirmation** dialog will confirm you that the file has been pushed to the Git repository.

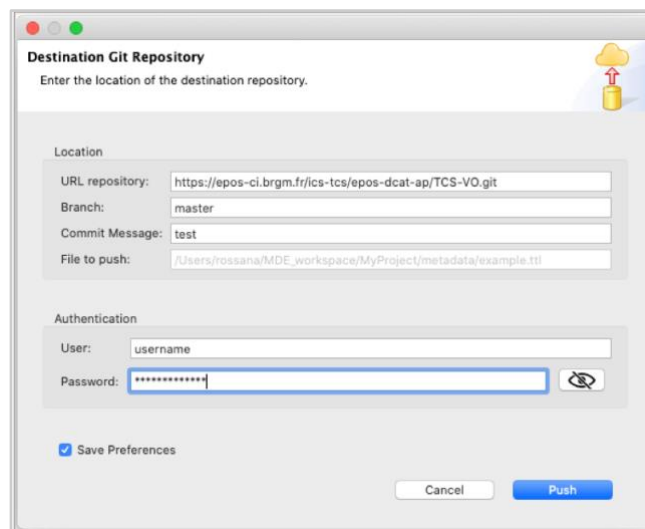


Figure 24 - Pushing Turtle file to a GitLab repository

4.6 Troubleshooting

This section is intended to provide information on how to resolve common problems you may encounter.

- **Problem: “A project with that name already exists in the workspace”**

The wizard dialog for creating a new project shows an error message about “**A project with that name already exists in the workspace**” in the title area (Figure 25).

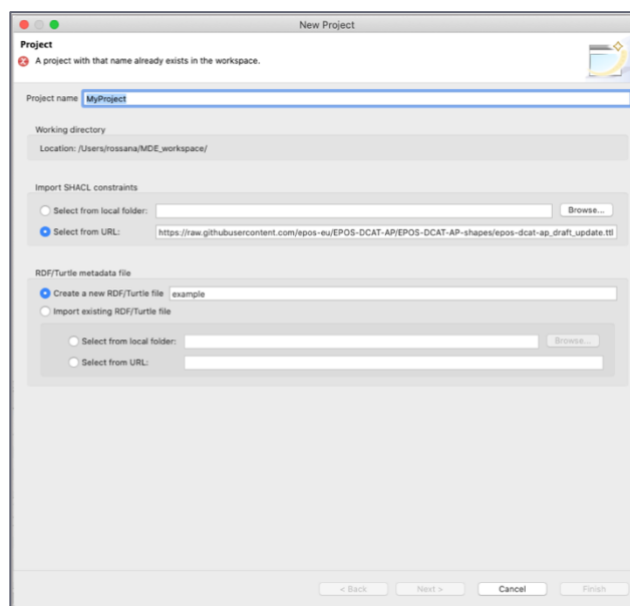


Figure 25 – Wizard error: A project with the name already exists in the workspace

Cause:

There is another project with the same name in your workspace folder located at \$User\MDE_workspace\.

Solution:

Enter a different name for the project name field in the wizard dialog for creating a new project.

- **Problem: “SHACL file not valid: InputStream is null”**

The wizard dialog for creating a new project shows an error message about “**InputStream is null**” in the title area (Figure 26).

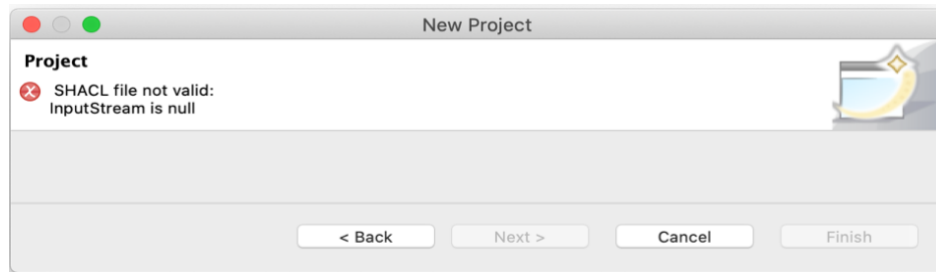


Figure 26 – Wizard error: SHACL file not valid - InputStream is null

Cause:

The SHACL URL you entered is not correct.

Solution:

Enter a correct SHACL URL.

- **Problem: “SHACL file not valid: wrongFile (No such file or directory)”**

The wizard dialog for creating a new project shows an error message about “**wrongFile (No such file or directory)**” in the title area (Figure 27).

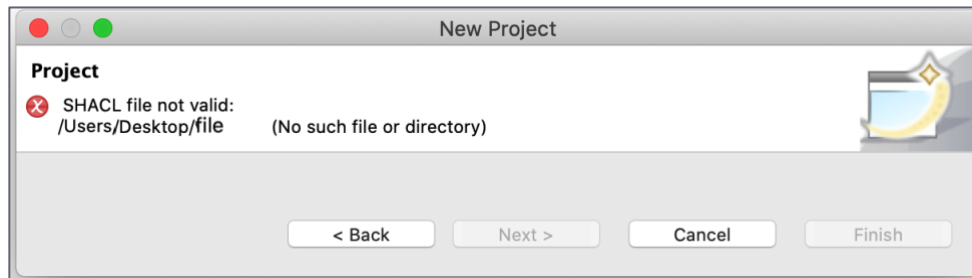


Figure 27 – Wizard error: SHACL file not valid - No such file or directory

Cause:

The local path for the SHACL file you entered does not exist.

Solution:

Enter an existing local path for the SHACL file.

- **Problem: “SHACL file not valid: specific syntax errors”**

The wizard dialog for creating a new project shows an error message about specific syntax errors in the title area (Figure 28).

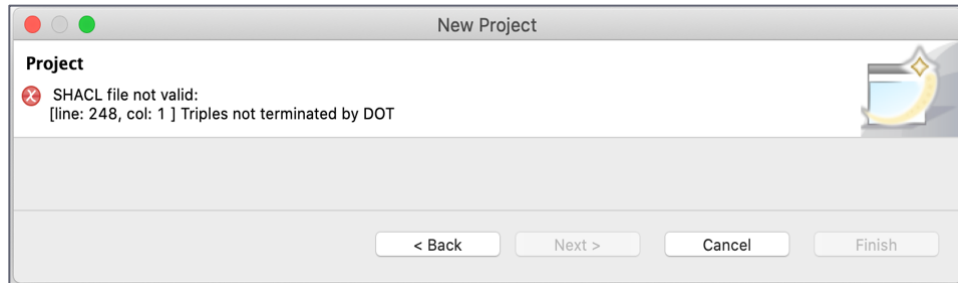


Figure 28 – Wizard error: SHACL file not valid: specific syntax errors

Cause:

There are syntax errors in the SHACL file.

Solution:

To detect syntax errors and fix them in the SHACL file you can use online validator tools like [https:// shacl.org/playground/](https://shacl.org/playground/).

- **Problem: “SHACL file not valid: Bad character in IRI (space): <!DOCTYPE[space]...>”**

The wizard dialog for creating a new project shows an error message about “**Bad character in IRI (space): <!DOCTYPE[space]...>**” in the title area (Figure 29).

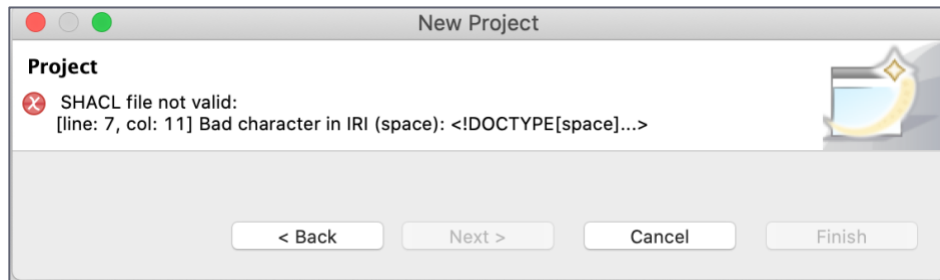


Figure 29 – Wizard error: SHACL file not valid: Bad character in IRI (space)

Cause:

The URL you entered for importing a SHACL file from a Git repository is not a link to the raw file.

Solution:

Enter a URL to import a SHACL file from a Git repository which is a link to the raw file.

- **Problem: RDF/Turtle file not valid: “fileName (No such file or directory)”**

The wizard dialog for creating a new project shows an error message about “**fileName (No such file or directory)**” in the title area (Figure 30).

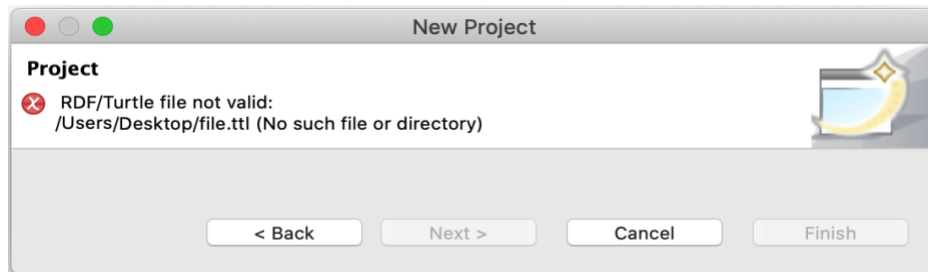


Figure 30 - Wizard error: RDF/Turtle file not valid: fileName (No such file or directory)

Cause:

The local path or URL you entered for the Turtle file does not exist.

Solution:

Enter an existing local path or a correct URL for the Turtle file.

- **Problem: “RDF/Turtle file not valid: specific syntax errors”**

The wizard dialog for creating a new project shows an error message about specific syntax errors in the title area (Figure 31).

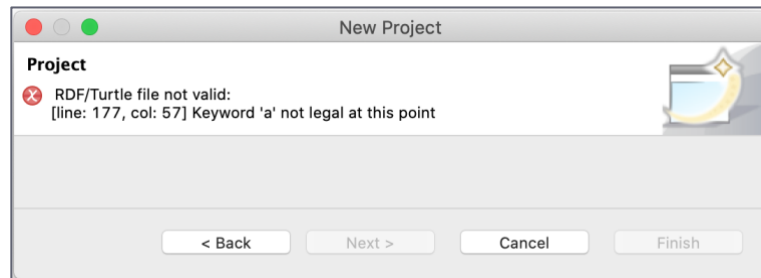


Figure 31 – Wizard error: RDF/Turtle file not valid: specific syntax errors

Cause:

There are syntax errors in the Turtle file.

Solution:

To detect syntax errors and fix them in the Turtle file you can use online validator tools like <https://shacl.org/playground/>.

- **Problem:** “RDF/Turtle file not valid: Bad character in IRI (space): <!DOCTYPE[space]...>”

The wizard dialog for creating a new project shows an error message about “**Bad character in IRI (space): <!DOCTYPE[space]...>**” in the title area (Figure 32).

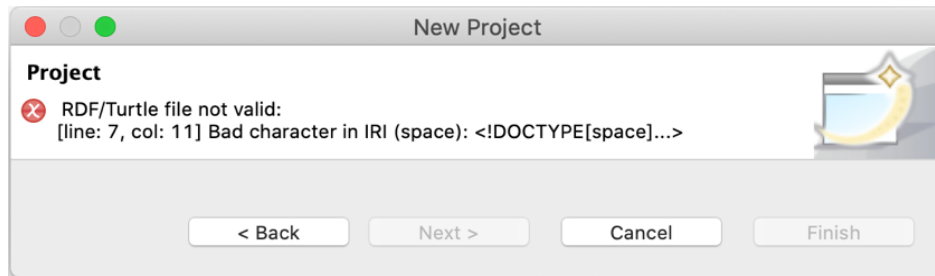


Figure 32 - Wizard error: RDF/Turtle file not valid: Bad character in IRI (space)

Cause:

The URL you entered for importing a Turtle file from a Git repository is not a link to the raw file.

Solution:

Enter a URL to import a Turtle file from a Git repository which is a link to the raw file.

5 Getting support

If you have an issue or question about specific functions in SHAP*Ess* Metadata Editor and are unable to resolve it using the documentation, you can visit the web page at <https://epos-eu.github.io/SHAPEss-Metadata-Editor/index.html> and open an issue on GitLab or GitHub.