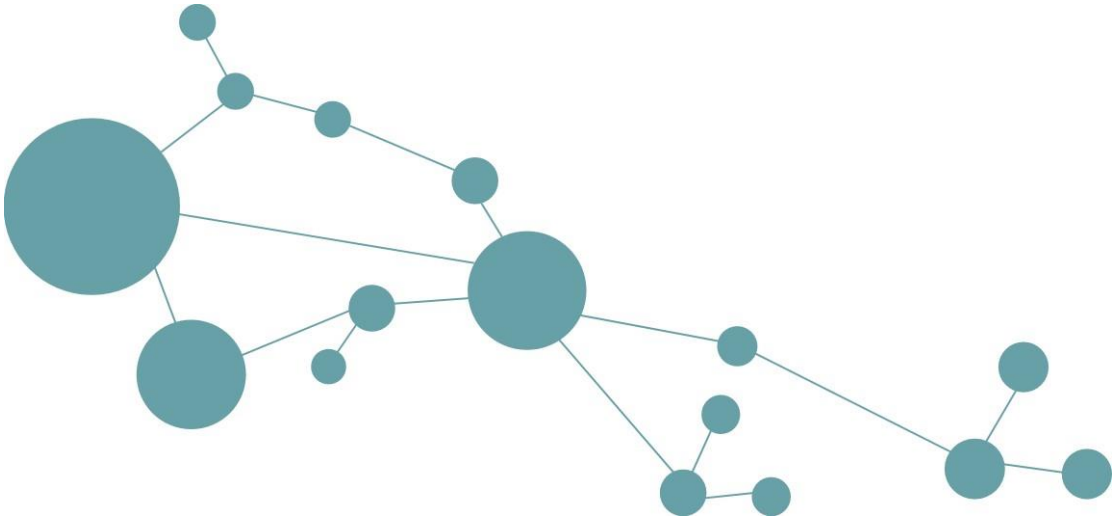


i-views content 1.4

i·views

Contents

1 Users' Manual

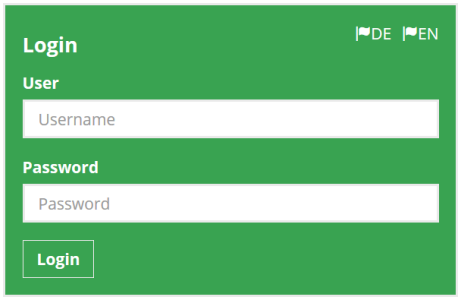
This user manual gives you an overview on how to use the functionalities provided within the web frontend of the content delivery portal i-views content 1.4.

In the web frontend, we can query content by means of keywords, put together individual collections of relevant and often required chapters and make a download of it in forms of a PDF document. Metadata structures open the possibility to discover content without having to know certain search terms. Furthermore, we can post feedback about single chapters.

You need functionalities for document- and user management? This is no problem, since the subject administrator access within the web frontend allows you creating new user accounts and uploading document bundles in a simple way. Due to the patented rights management, authorization can be controlled in a fine granular manner.

1.1 Login

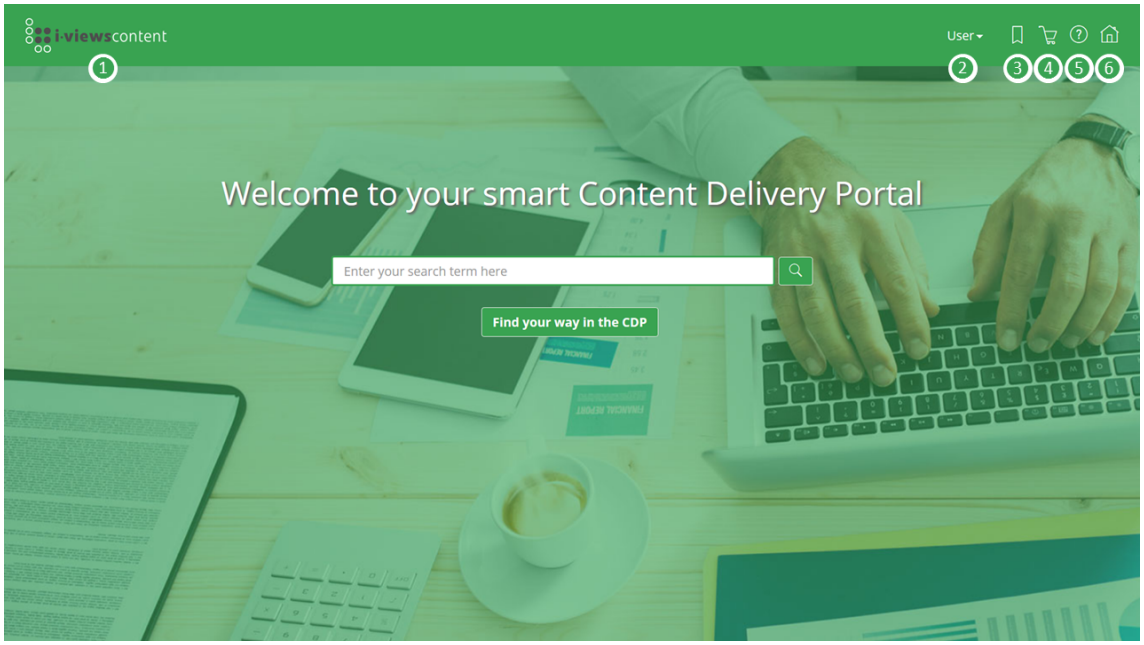
The login for the i-views content web frontend is automatically shown in the display language of the browser. Additionally, we can switch between English or German user interface.



1.2 Start page and navigation

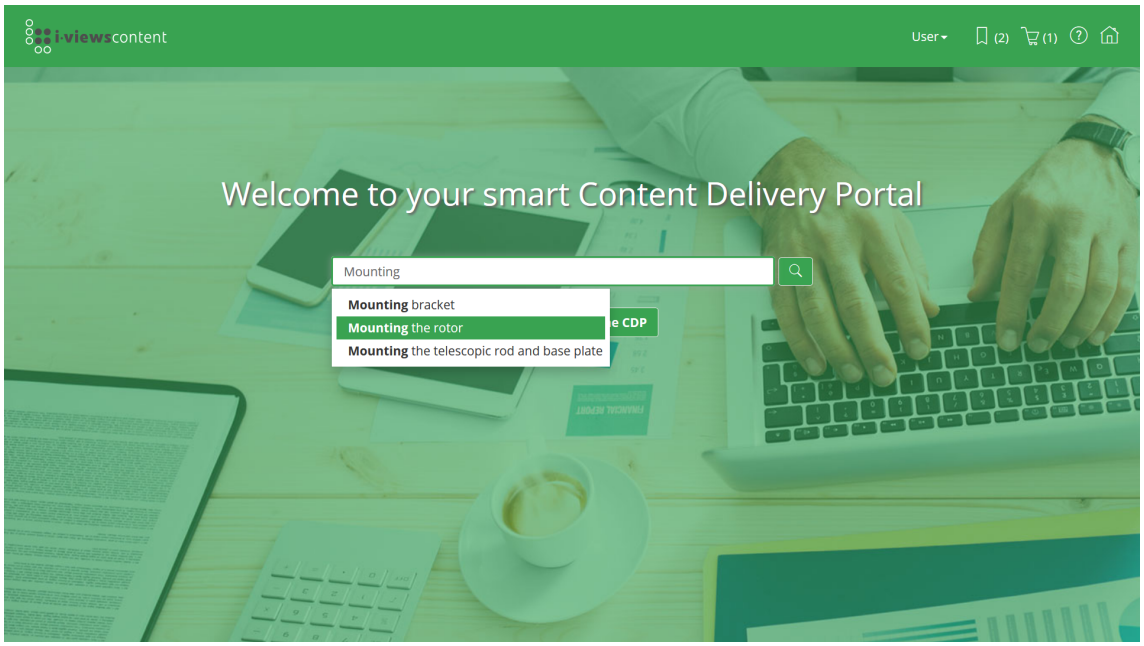
The start page offers following functionalities:

- Search input field for full text indexed query
- Persistent navbar:
  Returning to the start page by clicking on the logo
  Switching display language and logout
  Displaying chapter bookmarks
  Displaying the content cart
  Online help with quick overview
  Home button for returning to start page

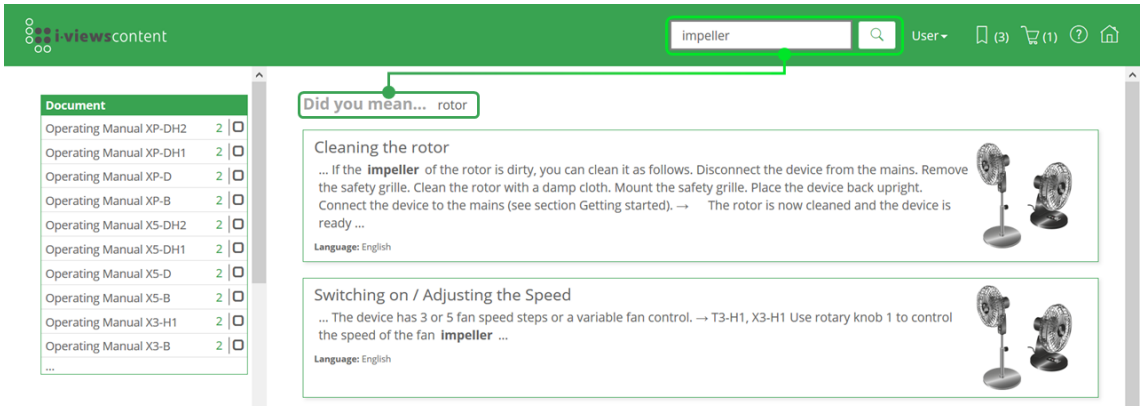


1.2.1 Full text query

The full text query allows us searching for content of individual chapters concerning their full text, their metadata and their tags. When entering the search term, a type-ahead function proposes possible terms for using as query input. By clicking on the type-ahead proposal, the proposed search term will be used without further ado.

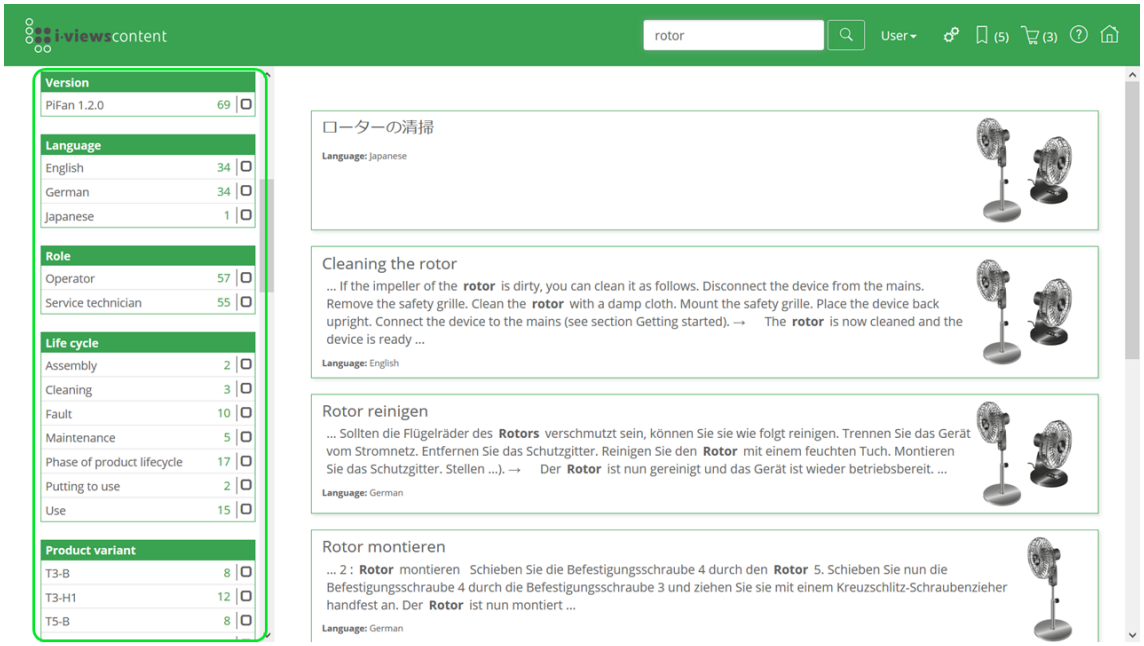


Furthermore, the terminology feature of i-views content offers the preferred search terms: When a user enters an acronym or a familiar term, the portal will suggest a preferred search term as a link additionally to the alternative search results:



Within the search result view, facet terms are shown for filtering and reducing the amount of search results according to following preconfigured topic types:

- Document type, Document
- Version
- Language
- Manually assigned metadata
- Further individual or domain specific topics
- iiRDS-specific types: role, life cycle, product variant
- Subtypes of iiRDS administrative metadata, documentation metadata and information type



The search results list shows the title of the found chapters, including the text snippet underneath that contains the highlighted search term.

The foot notes of each search result entry show the context of the content. A click on the title

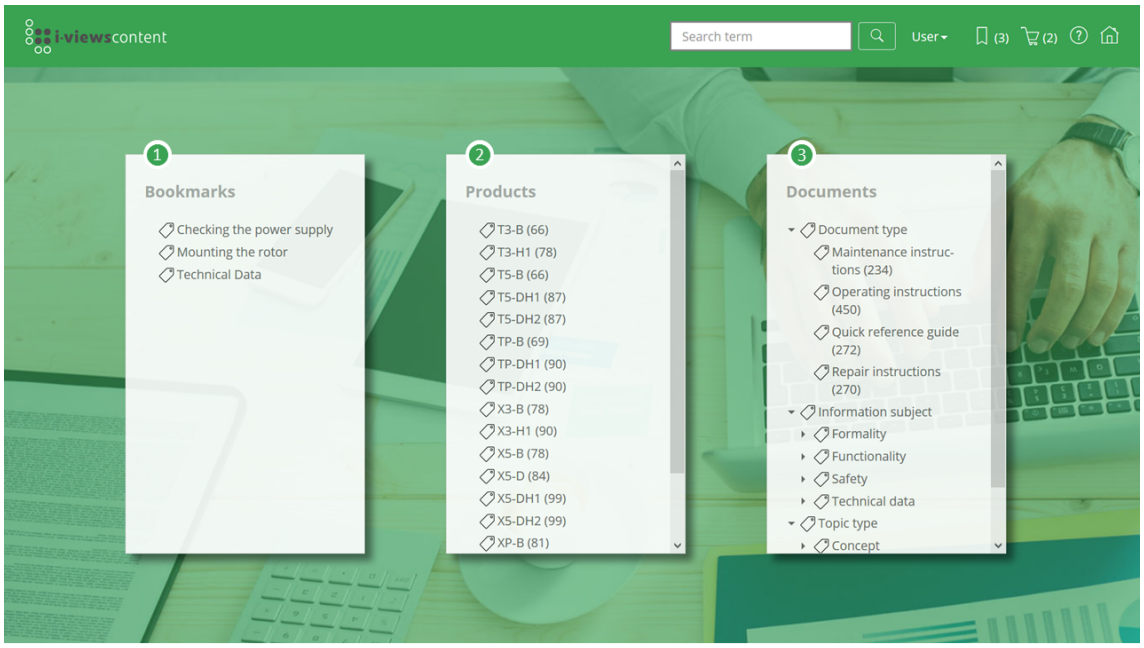
link immediately leads to the position within the document.

When a more specific focus on metadata is needed, the search result view can be enhanced to display metadata, too. The detailed view following on selecting metadata results can be adjusted to the use case accordingly.

1.2.2 Metadata navigation

As an alternative to using the common full text query we also have the chance to explore the content due to its semantically enriched metadata. To do so, we click on the link “Find your way in the CDP” which leads to the navigation page. The navigation page offers contextual access to documentation by means of following preset metadata:

Bookmarks: user specific bookmarks
Products: iiRDS product specification (component / phase of product life cycle)
Documents and topics: iiRDS metadata of documents (information type and document type)



Clicking on the metadata either leads us to a detailed view with the location of the metadata within the Knowledge Graph types and the related iiRDS document links or to a chapter of the bookmark or to the version-oriented document overview.

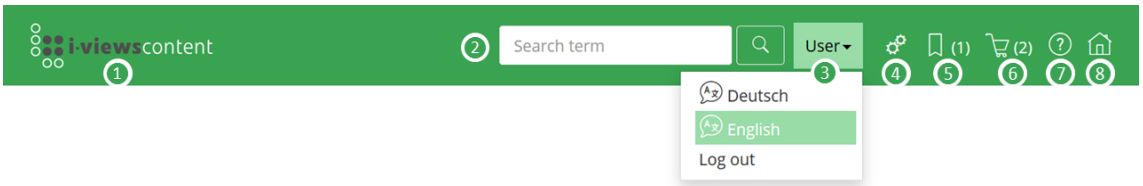
The detailed view for iiRDS metadata shows a structured tree for selection and a listing of the content for which the metadata is about.

1.2.3 The navbar

The navbar always is available in the header of the website, offering following features:

Home button with logo: Always leads back to the start page
Search input field: When the start page is not apparent, we can put in our search term for full text query here
User drop-down menu: Switching display language or logging out

Link to the administration section: Only available for the subject administrator; for further information see chapter “Administration section”
Favorites link: Selection dialog containing the bookmarks made by the user
Content cart: Selection dialog of the cart containing the listing of its chapters for download as a single PDF
Online help: Quick overview about the user interface
Home button: For returning to the start page



1.3 Document features

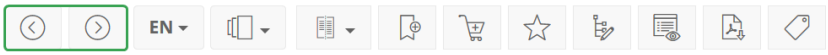
i-views content 1.4 comes with two kinds of views:

- document-oriented view: the topic is shown together with its hierarchical document structure; besides metadata, topic-related actions and document-related actions are available, such as flipping through the topics one by another and displaying the whole document text
- topic-oriented view: only the topic itself of the modular documentation is shown, with topic-related actions and metadata

1.3.1 Full text view

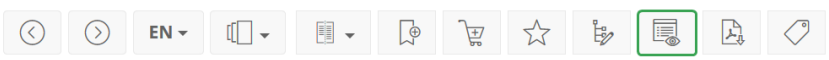
In the Content Delivery Portal, structured content is shown in separate chapters. In the document view, there are actions available as follows:

- The arrow buttons allow scrolling through the individual chapters one by one.



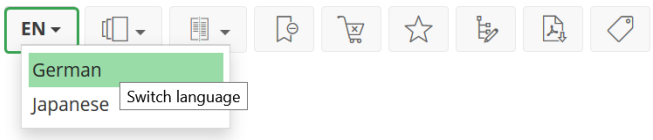
Clicking on the structured chapter tree causes the related chapter to be shown in the text area accordingly.

- By clicking on “Show full document”, the chapters of the document will be shown as continuous text.



1.3.2 Switching the document language

When several language versions of a topic exist, the topic can be shown in the desired language by selecting the language in the drop-down menu.

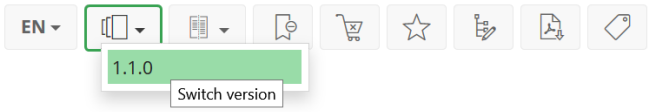


Please keep in mind that this feature only changes the display language for the document and its metadata to be shown, whereas the global language settings determine the display language of the whole user interface and the *initial* display language of documents.

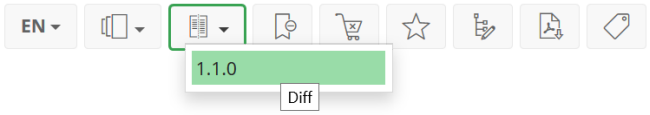
Note: Manually assigned metadata is only shown for a topic when the metadata has the same language as the display language of the topic.

1.3.3 Switching the document version

When a topic exists in terms of several document versions, a drop-down menu allows displaying the related topic in another version. If the document structure of the other version is identical, the equivalent chapter will be shown right after switching.

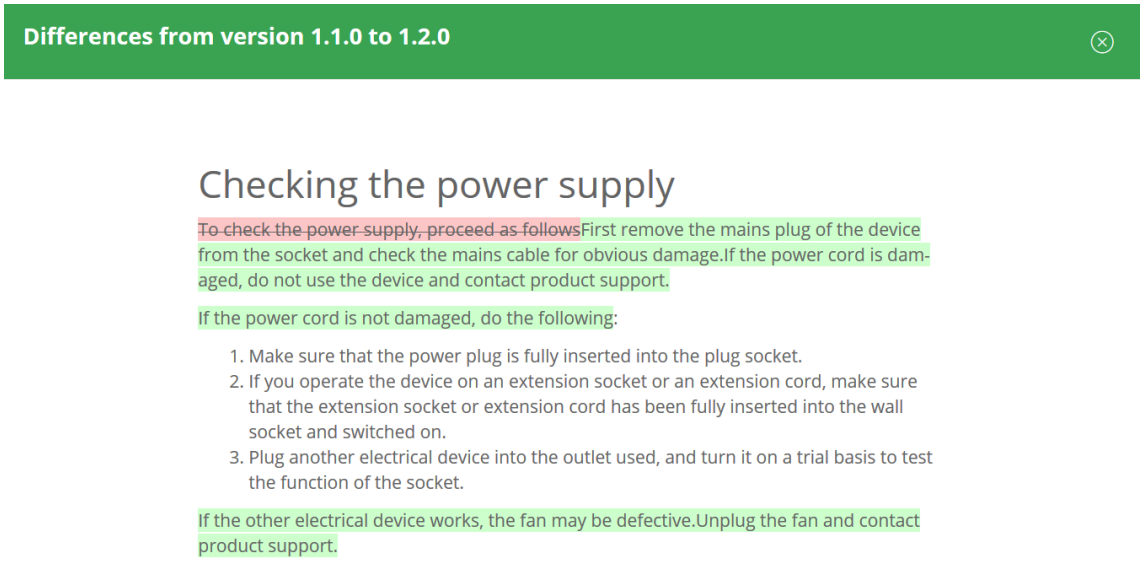


1.3.4 Diff-function - Displaying version differences



By clicking onto the button “Differences”, a drop-down menu shows a selection of versions by which the currently shown topic can be compared against. Choosing one of the versions reveals us the differences from one document version of the topic to another:

- **Red highlighting** emphasizes crossed out content that is existent in the predecessor version but not in the subsequent version.
- **Green highlighting** indicates content that came along within the subsequent version.
- No highlighting indicates content that is identical in both versions.



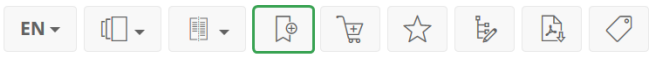
The diff functionality therefore can be used to indicate changes amongst different document version combinations of topics.

Note: The diff functionality only works on textual changes; images are excluded from the comparison.

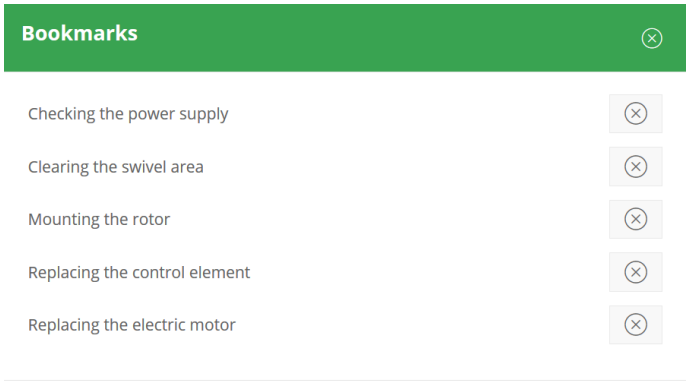
1.3.5 Bookmarks

Adding bookmarks

For every topic we can add a user-specific bookmark by clicking on the button “Add bookmark”:



All bookmarks can be accessed in the bookmark container via the button “Bookmarks” in the navbar:



Additionally, all bookmarks are presented as metadata links on the navigation page.

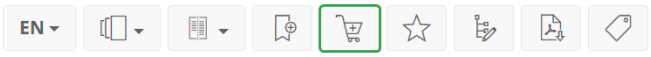
Removing bookmarks

After the bookmark has been added to the topic, the "Add bookmark" button changes to "Remove bookmark". When clicking on the button again, the bookmark is going to be removed. Bookmarks also can be removed when accessing the bookmark container and clicking on the button.

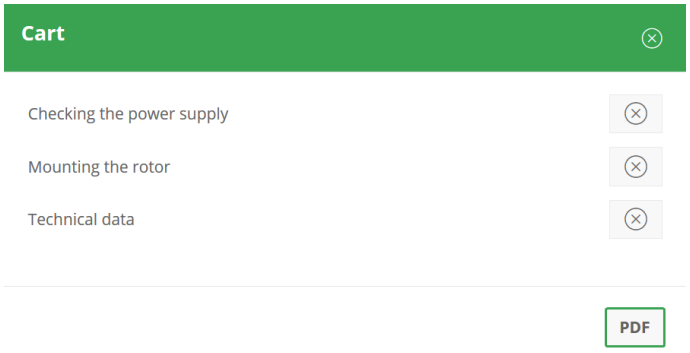
1.3.6 Content Cart and Download function

Download per content cart

In case that content from different topics of the same document or from different documents is needed as offline content in one piece, we simply add the concerning chapters to the content cart by clicking on the button “Add to cart”:



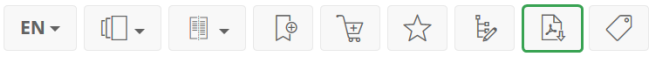
Afterwards, we can download the content as a single PDF file by opening the content cart and clicking on the PDF button.



In both the content cart and the view of the chapter we can remove the chapter from the cart again, either by clicking on the button “Remove from cart” or by clicking on the cross symbol at the cart entry.

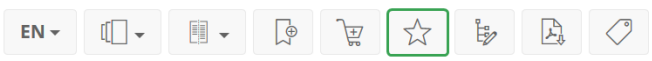
Direct download

The currently shown content can be provided as offline content by means of clicking on the button “Download as PDF”.

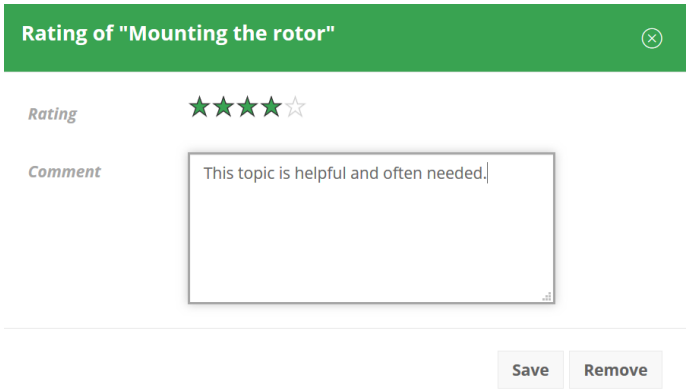


1.3.7 Rating chapter content

For the topic of a document, we can give a feedback which serves for the content provider and for other users as well. To do so, we click on the button “Rate this topic”.



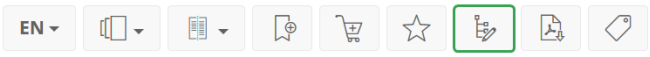
In the rating dialog, a comment can be stated along with the rating:



After being reviewed and released by the subject administrator, the rating including the comment is visible to all users at the end of the topic.

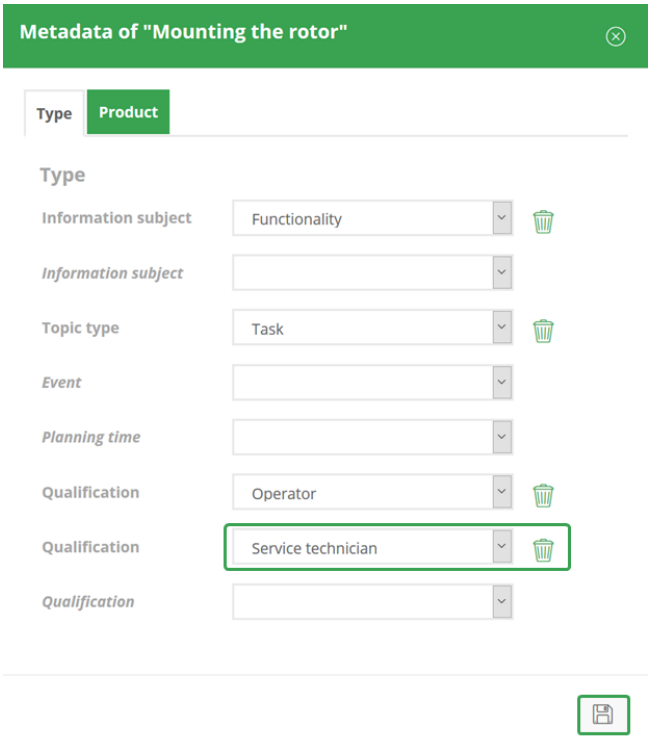
1.3.8 Managing metadata

If we want to add metadata to the currently shown topic, we can do this manually by clicking on the button “edit metadata”.

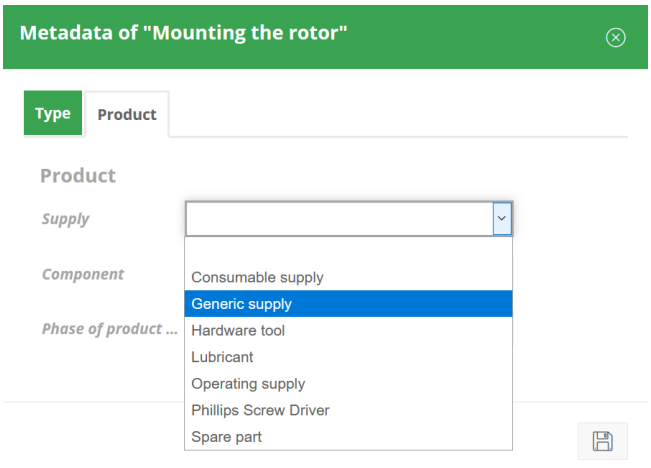


Depending on the kind of document (iiRDS or DITA-XML), we can assign metadata either by selecting predefined metadata or by creating new metadata:

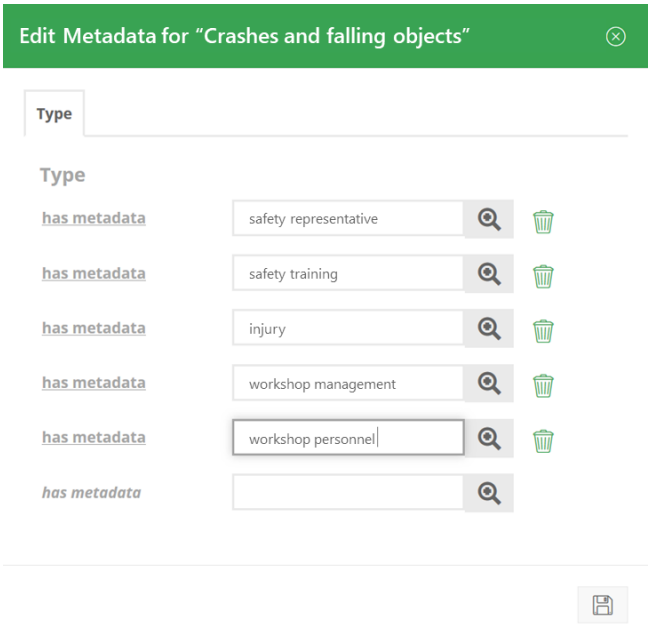
- For iiRDS documents, we can assign predefined metadata for the topic type or the related product by selecting a drop-down list entry. A click on the dustbin removes entries again. After having edited the entries, a click on the save button saves the changes:



The editing of product metadata relating to the topic-related product is done analogously:



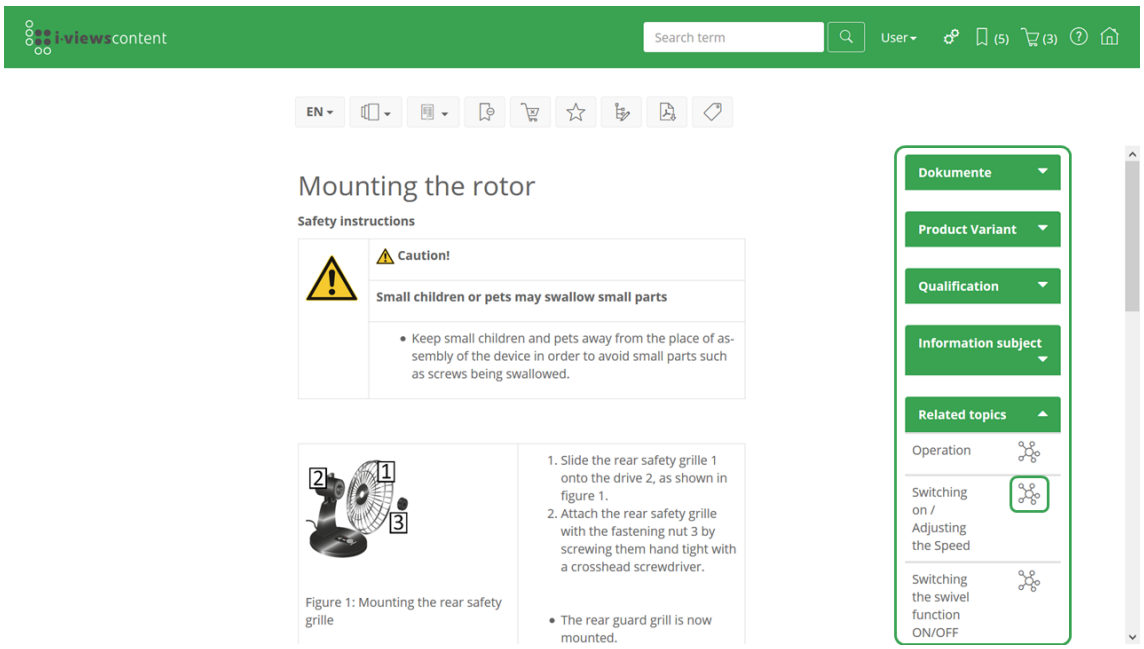
- For DITA-XML documents, we can assign predefined metadata by clicking on the magnifier button or typing in the metadata term. Additionally, we can create new metadata:



If the metadata to be assigned is already existing in the Knowledge Graph, it will be proposed in forms of a type-ahead list entry when we type in the metadata term. If the metadata term is unknown, the system offers us to create it as a new metadata by selecting the metadata type in the type-ahead list.

The assignment of a metadata can also be cancelled by clicking the dustbin button.
Note: To make changes permanent, we must click onto the “Save” button instead of the “x” button.

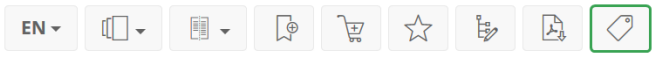
By means of assigning metadata individually, user can contribute to improvement of content evaluation in terms of domain specific and professional aspects. Metadata can be shown for all users in context boxes aside the content area for invoking detailed information or for linking to similar chapters. Another gain is the possible improvement of search result precision.



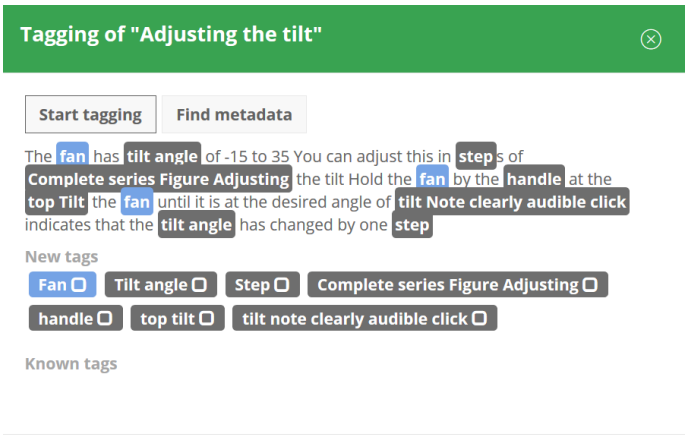
Note: The assignment of manually created metadata is language sensitive; this means when a metadata is created for the English version of a topic, a context box containing the metadata will be shown for the English topic exclusively but not for the German topic.

1.3.9 Automated metadata assignment (Tagging)

For automated assignment, a click on the button “Tagging” opens a dialog containing the full text of the topic.



To start the tagging process, we click on the button “Start tagging”. When the tagging process is finished, all identified terms are highlighted as tag proposal within the text; different highlighting colors indicate different tag types.



In the area “New tags”, found terms are shown for assignment. To commit a tag, we click on the checkmark box at the tag. As soon as the term is committed, it appears in the area “Known tags”. In this area tags are also shown which already have been assigned in a previous tagging process. A click on “Find metadata” shows which further iiRDS metadata can be found according to the terms, if available.

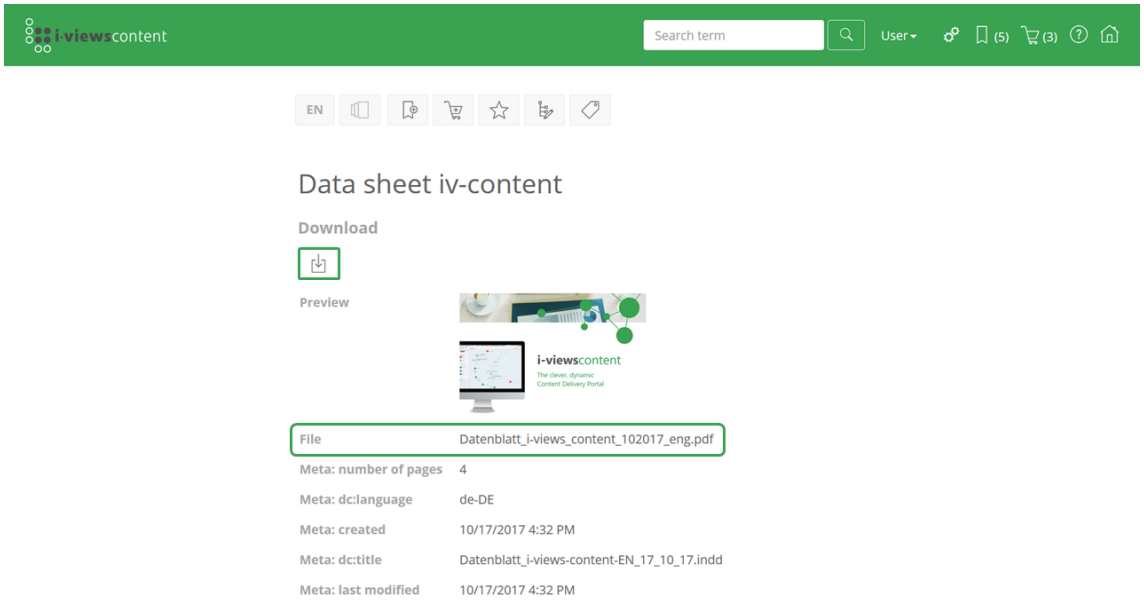
By closing the dialog, changes are automatically being saved.

1.4 Supplementary features

Single File Documents

Besides publishing structured content, there is also the possibility to provide so-called “single file documents” - or BLOB data - for download as offline content. If published for the user, single file documents can be accessed by selecting the document within the search results list.

Clicking on the link shows the detailed view about the single file document, including the link for download:



Customized web frontend

The i-views content delivery portal comes with further display features provided by the view-config mapper engine:

- Net-Navigator
- Render modes for further display modes like calendar, diagrams etc.
- Plugins for integration of components like maps etc.

i-views content Offline App

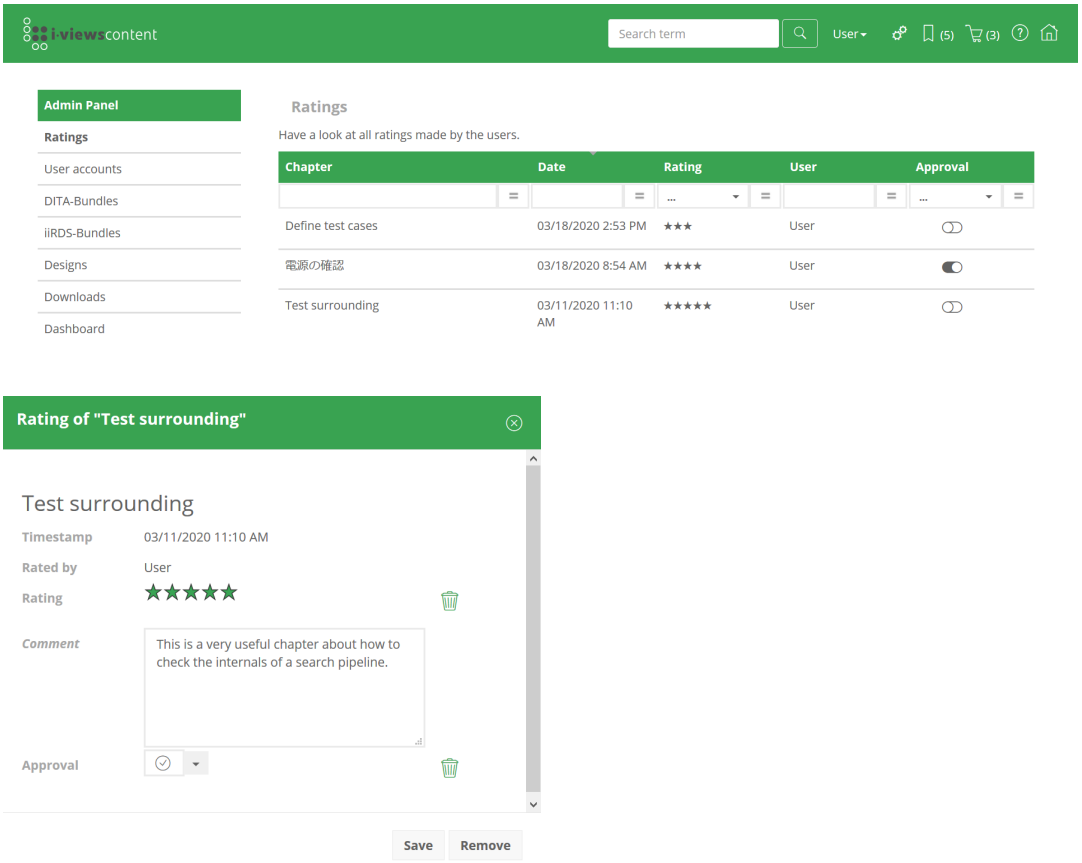
For special user scenarios in which portable content delivery is needed (e. g. on construction sites with lack of internet access), the new i-views content Offline App is available now. It allows the download of partial content and provides searching functionalities for the offline content on your mobile device. For more information, send an email to: contact@i-views.com.

1.5 Administration section

For the subject administrator, following functionalities are available by standard:

Rating

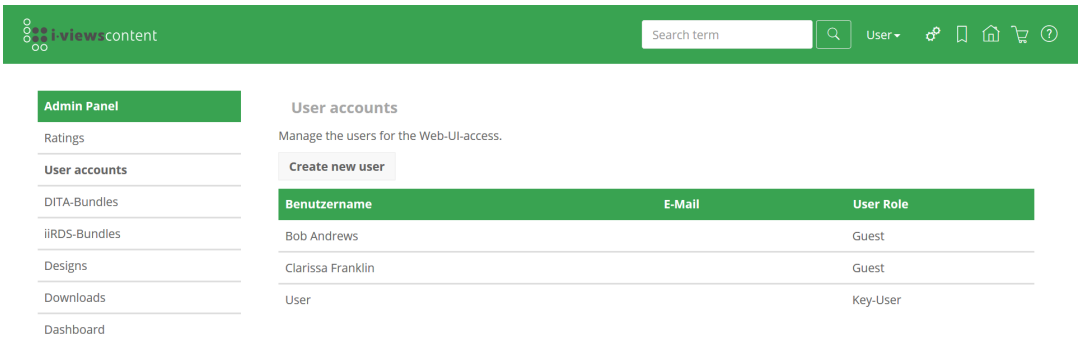
- Listing of all ratings made by users: The administration section provides access for review and release control. Every rating can be reviewed by clicking on the respective entry.



- Release control flow for publishing ratings: After a rating (incl. comment) has been made by a user, it is still visible to the user only. The toggle switch in the approval column releases the rating to all user by standard.

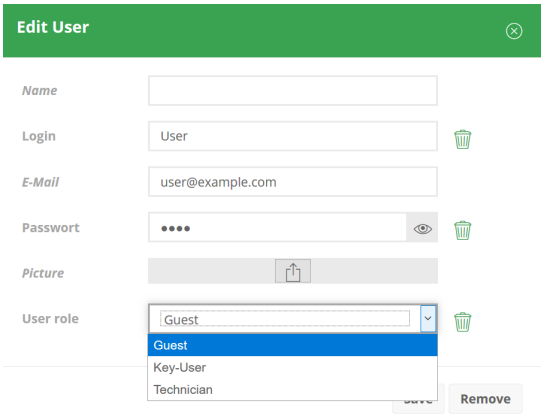
User Account

- Administration of web frontend user accounts: Here we can add or remove user and change their roles and detail information.



The minimum information for creating a user account are login name, password and a user role which determines the access rights of the user. Further data can be added as

needed.

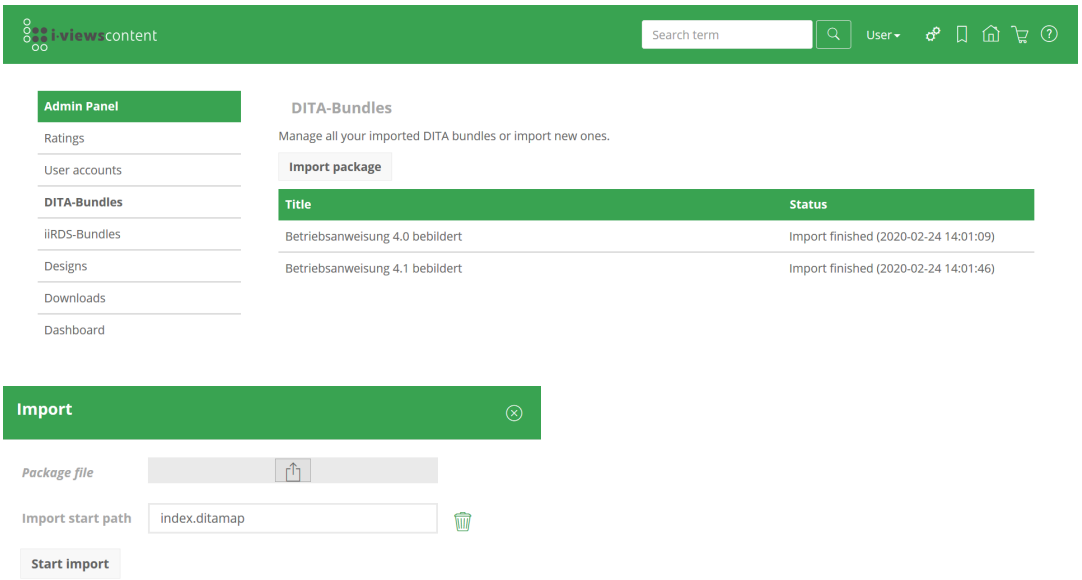


For each user account, a separate bookmark container and a separate content cart is available with read/write access. Ratings can be posted at topics for subsequent release by the subject administrator.

The user roles ("Guest", "Key-User" and "Technician") are preset values:

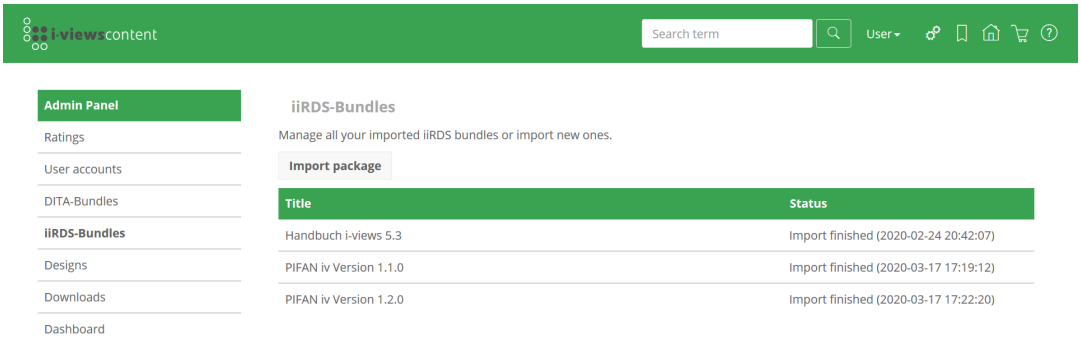
DITA-XML-Bundle

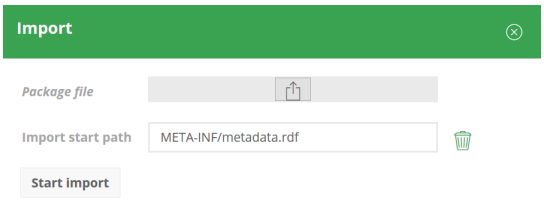
- User interface for simplified upload of DITA-XML document bundles by uploading the bundle zip file and by defining the ditamap start path



iiRDS-Bundle

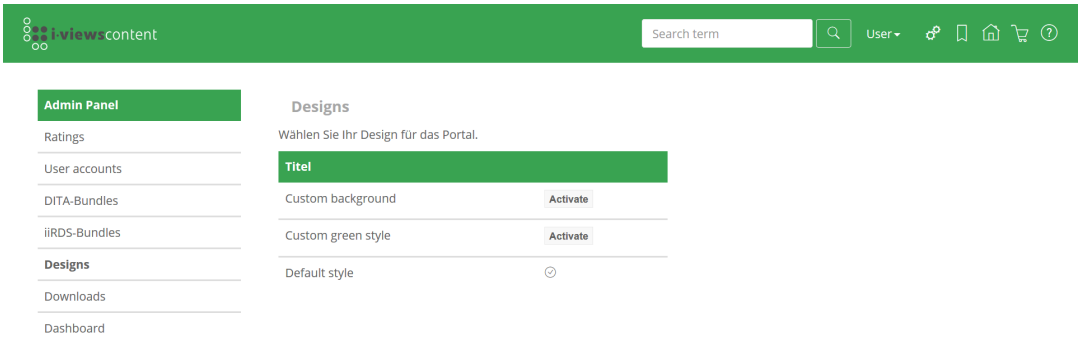
- User interface for simplified upload of iiRDS document bundles by uploading the bundle zip file and by defining the start path





CSS style

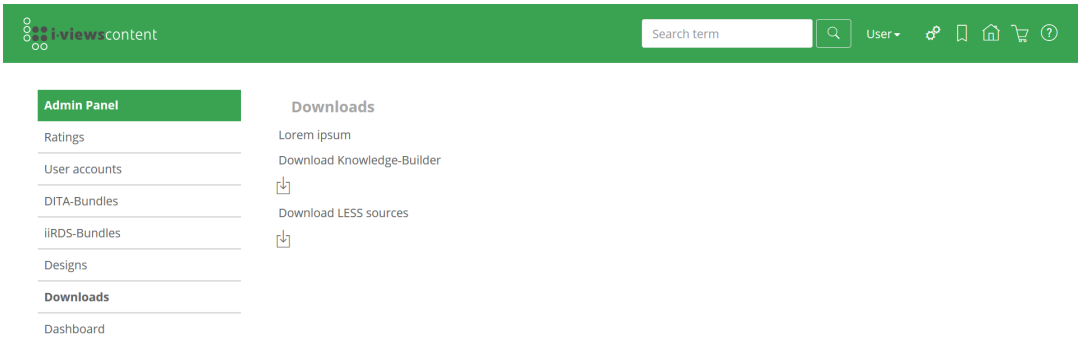
- Activating predefined CSS styles (implementation per LESS, configuration via backend)



Download

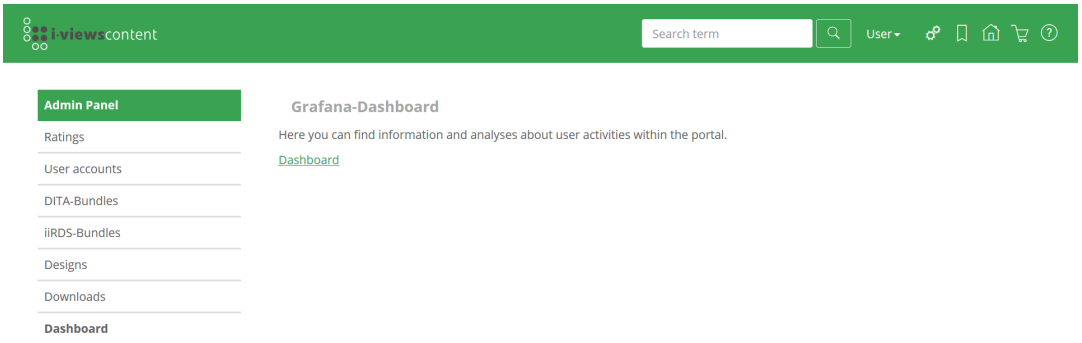
The download area offers following options:

- Download of the i-views content Knowledge-Builder for backend configuration
- Download of LESS sources for customizing the i-views content web frontend according to corporate design



Dashboard

- The dashboard section provides access to the Grafana monitoring tool, allowing to analyze the anonymized search history of the content delivery portal.



2 Administrators' Handbook

2.1 General principles

i-views content is a database system for web based technical documentation. It comprises the backend with the database and the frontend with a user interface. As database technology the i-views Knowledge Graph is being used which allows storage and networking of content according the Linked Data principle. In the web-based user interface, content modules of technical documentation can be presented and - thanks to networked data structures - queried in a semantic manner.

For the administering i-views content by means of this documentation, following skills are required:

- Defining object types, relation types and attribute types
- Creating objects, relations and attributes
- Defining and creating meta attributes

- Creating import mappings
- Importing data per import mapping
- Deleting data

These basic principles can be read in the general documentation of i-views:
http://documentation.i-views.com/5.3/.

2.2 Release notes i-views content

2.2.1 Release notes version 1.4.4

Neue Features:

New
New

- Suche ist toleranter bei Tippfehlern (1)
- Verbesserte Darstellung von Suchtreffern, die durch Metadaten gefunden wurden (1)
- Favicon kann konfiguriert werden (1)

Bugfixes - Frontend:

- Kontextboxen scrollen mit Inhalt, lange Boxen werden nicht mehr abgeschnitten
- Aufruf eines kopierten Links scrollt wieder zur richtigen Position
- Content Security Policy restriktiver, keine Warnungen beim Start
- Performance bei Dokumenten mit vielen Kapiteln verbessert (1)
- Wenn bei einer Hierarchie die Detailainsicht ausgeblendet wird, wird auch der aktive Knoten nicht mehr hervorgehoben
- XSS-Anfälligkeit durch Benachrichtigungen behoben

Bugfixes - iiRDS:

- Renditions von Dokumenten (iirds:Document) werden nun angezeigt (bisher nur von Topics) (2)
- Es werden alle Renditions ohne ID angezeigt (2)

(1) Änderung in der Viewkonfiguration
(2) Erfordet erneuten Import des betroffenen Bundles

2.2.2 Release notes version 1.4.3

iiRDS

- Improved display of iiRDS fragments
- Support of iiRDS 1.1

Web frontend

- Improved rating functionality
- Access of CDP without login (optional)
- Click on topics opens them within their document context, if unique

Knowledge Builder

- Improved detail view of bundles within the Knowledge Builder

2.2.3 Release notes version 1.4.2

- Integrated display of PDF documents
- Toggle switch for topics with several alternative display options (renditions of iiRDS documents)
- Foldable navigation area within web frontend
- Controlling of display language by means of URL query parameter 'lang'

2.2.4 Release notes version 1.4

- Bookmarking web pages for direct linkage and for using browser history
- Web frontend plugins and render modes

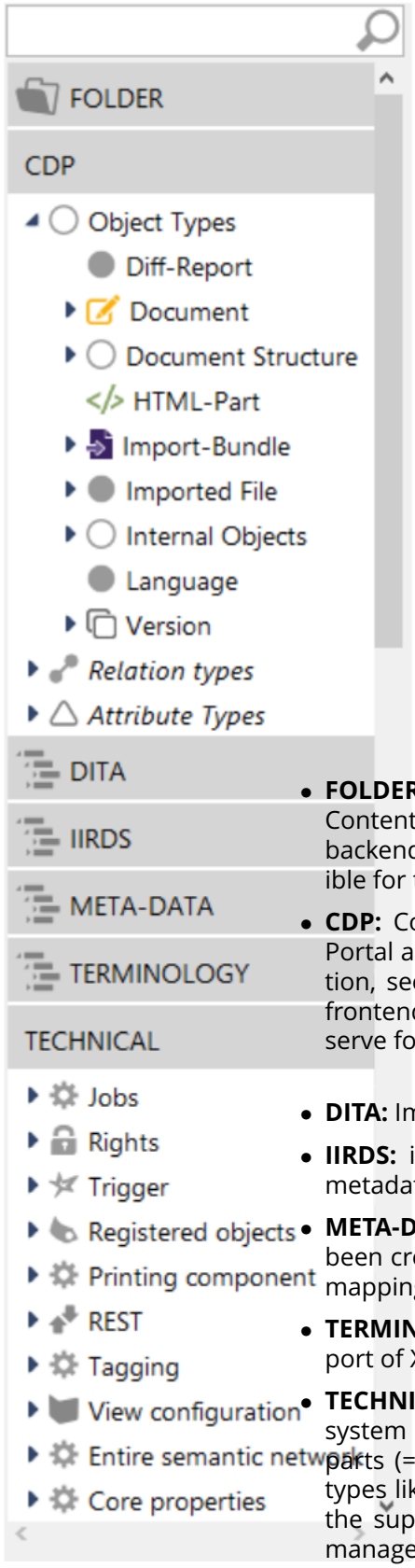
2.2.5 Release notes version 1.3

- Tagging of content
- Sharing ratings and comments with other users
- Metadata management
- Administration section:
  Activation of webpage styling using LESS interface
  Upload of iiRDS- and DITA-XML bundles
  Type-ahead suggestions for full text search
  Switching display language

2.3 Backend configuration and taxonomy of the CDP

2.3.1 Top Types

The backend of the content delivery portal is built up as a Knowledge Graph which can be administered using the Knowldge-Builder (KB). The Knowledge Graph contains following main types:



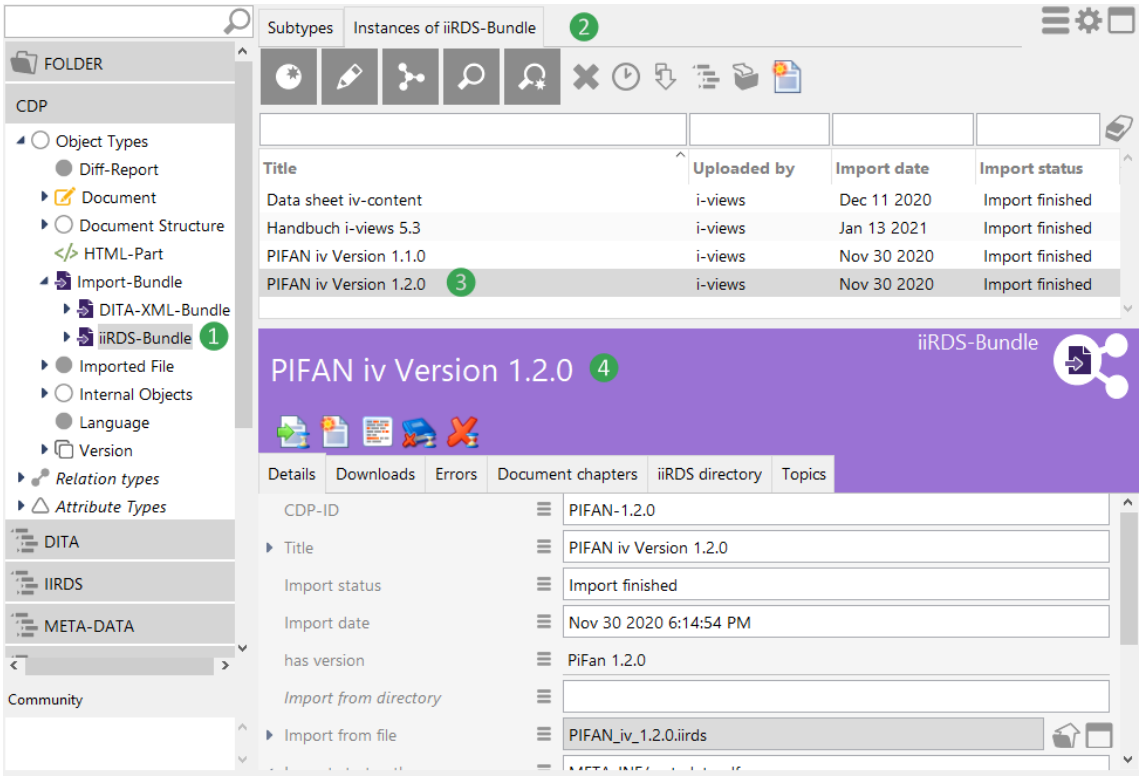
- **FOLDER:** Includes the working folder and the private folder. Content within the working folder can be accessed by every backend user, whereas content within the private folder is visible for the respective backend user only.
- **CDP:** Contains the import bundles of the Content Delivery Portal and content specific system data - for further information, see chapter 2.3.2. The structured content for the web frontend is organized in so-called "Import-Bundles" which serve for importing and publishing the content.
- **DITA:** Imported DITA-XML data
- **IIRDS:** iiRDS specific data including the standardized iiRDS metadata schema
- **META-DATA:** Taxonomy for domain specific metadata that has been created manually or that has been imported via import mapping; includes the demo-data
- **TERMINOLOGY:** Partial Knowledge Graph for automated import of XML based tbx terminology data
- **TECHNICAL:** Elementary object/relation/attribute types of the system that serve as template types for all Knowledg Graph parts (= main types and their subtypes). Provides attribute types like primary name for objects of the top-level type and the super types for view configuration, rest services, rights management, tagging configuration and trigger mechanisms. Furthermore, all registered scripts and queries can be invoked here.

2.3.2 Important object types of the top type "CDP"

- **Document:** Identifying element for the content; is needed for version assignment and diff functionality; the CDP-IP is obligatory
- **Version / Document version:** Serves for version control of documents and their content; the diff function uses the version string for comparing different document versions **Note:** If documents of a certain version have to be hidden in the web frontend, the Boolean attribute value “hide in navigation” can be set for the version instance
- **Import-bundle:** The content of the CDP is organized by means of "Import-Bundles". An import bundle is a container for the (structured) content to be imported, either based on the iirds standard 1.0.1 or based on the DITA standard 1.3. The import bundle is linked to a CDP document and to a version.

  By means of the CDP document and the version linked to an import bundle, the Content Delivery Portal identifies which bundles belong together redarding their version (revision) or their language variant. If several bundles - which semantically belong together due to their version or language variants - are linked to a single CDP document, the web frontend offers dropdown menus for switching between the versions and languages.

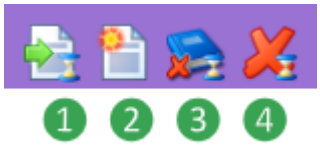
2.3.3 Import bundles



The import bundle can be found within the type hierarchy via CDP > Import-Bundle > DITA-XML-Bundle or iiRDS-Bundle . If Content is to be imported, an instance of the respective im-

port bundle type needs to be used or created . Dependent on the selected bundle instance , the detail editor shows the details and possible actions of the bundle.

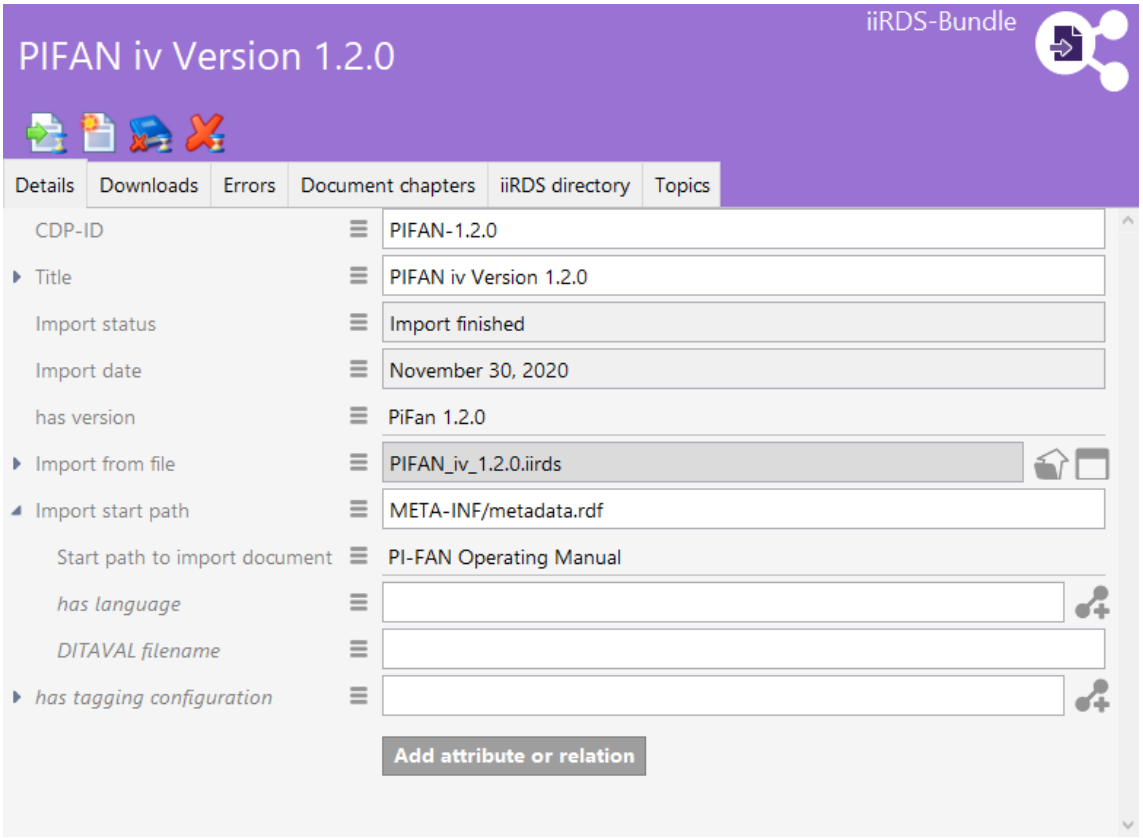
The follwing actions are available:



1. **Import bundle:** The import is processed by means of the job client (which is provided for every CDP demo instance). During import, "Import status" will first show "Import job added", which indicates that the import is waiting for a free job client. Once the import begins, "Import status" changes to "Importing bundle". When the import has finished, it will either show "Import finished" or "Import finished with errors". The latter indicates that errors occurred during import, which can be seen on the "Errors" tab.
2. **Upload new version:** This action allows to creat a new version of an already imported bundle by creating a new bundle instance automatically.
3. **Delete content:** This action deletes all imported content, comprising directory structures, topics and their HTML pendant. The source data of the import bundle is not deleted by this action.
4. **Delete bundle and content**: Deletes the content and the bundle including the source data as well

2.3.3.1 Details

Within the tab "Details", basic data of the bundle can be edited



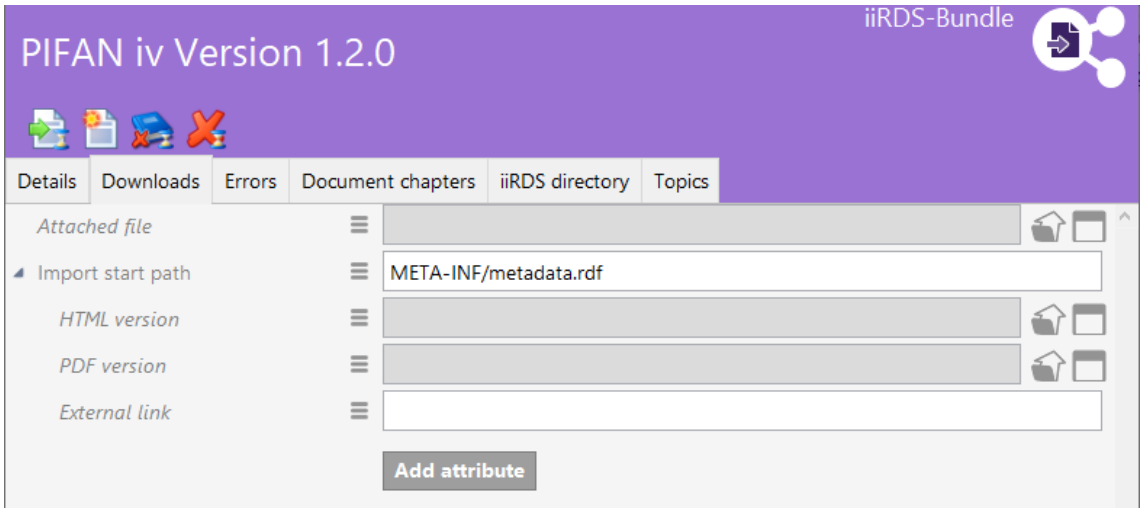
The individual values are as follows:

CDP-ID	Unique ID ot the bundle which can be defined freely anyways.
Title	Title of the bundle which will be displayed within the administrative area. User of the CDP will not see this title.
Import status	Current import status (read-only).
Import date	Time stamp of import (read-only).
has version	The version of the bundle.
Import from file	Source file (iiRDS package, ZIP archive with DITA files)
Import start path	Path of the document to be imported, relative to the achive file (iiRDS metadata, DITA map).
... Start path to import document	Linked object of type "Document", which aggregates language variants or versions of the same document.
... has language	Language of the imported document (optional). Required if the source data does not contain any language information.
... DITAVAL filename	Optional, only for DITA: DITAVAL file for conditional processing of DITA files.

has tagging configuration	Optional: configuration to be used for tagging.

2.3.3.2 Downloads

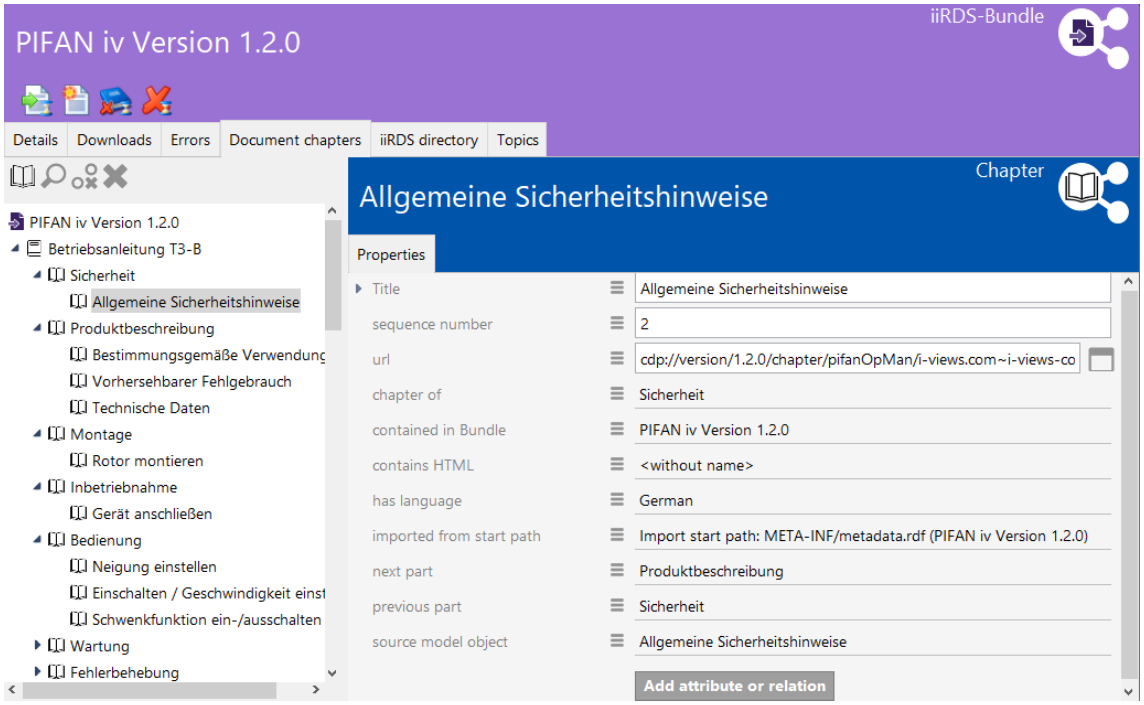
Within the tab "Downloads", additional documents can be specified which are accessible in the CDP for the user for opening or for download:



Attached file	Arbitrary attached file
HTML version	Alternative HTML version
PDF version	Preprocessed PDF
External link	External HTTP link

2.3.3.3 Document chapters

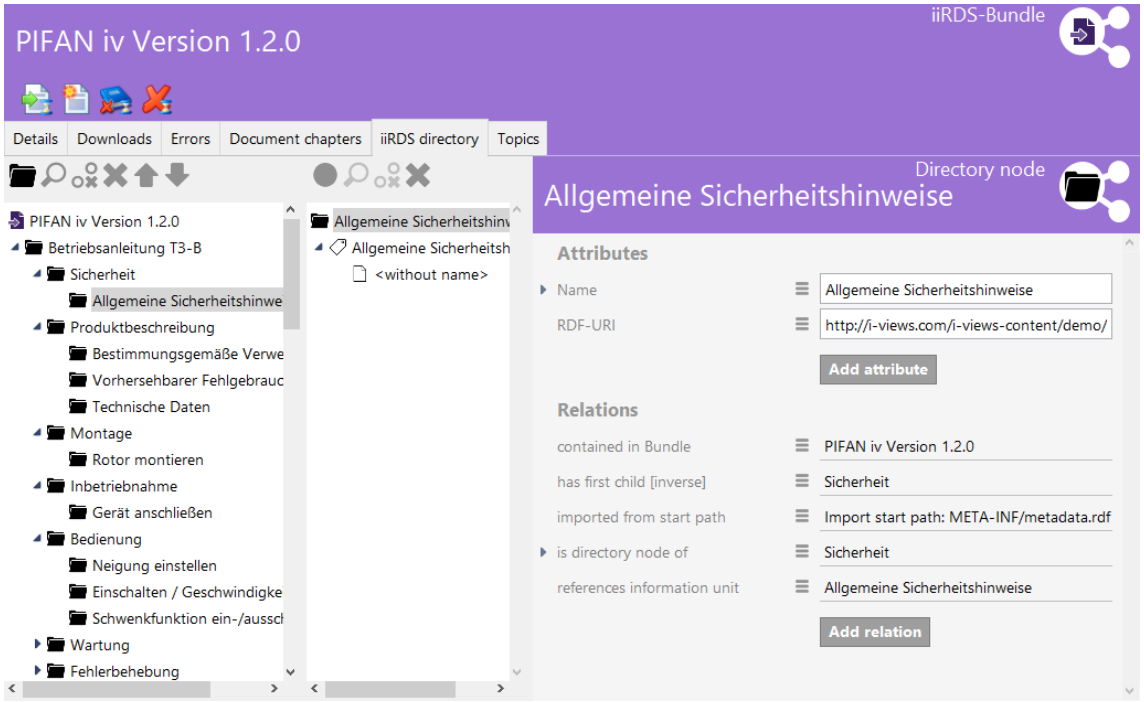
The tab "Document chapters" shows the imported chapter structure on the left side. When clicking onto a chapter, technical details of the selected chapter are shown on the right side.



2.3.3.4 iiRDS directory / DITA Map

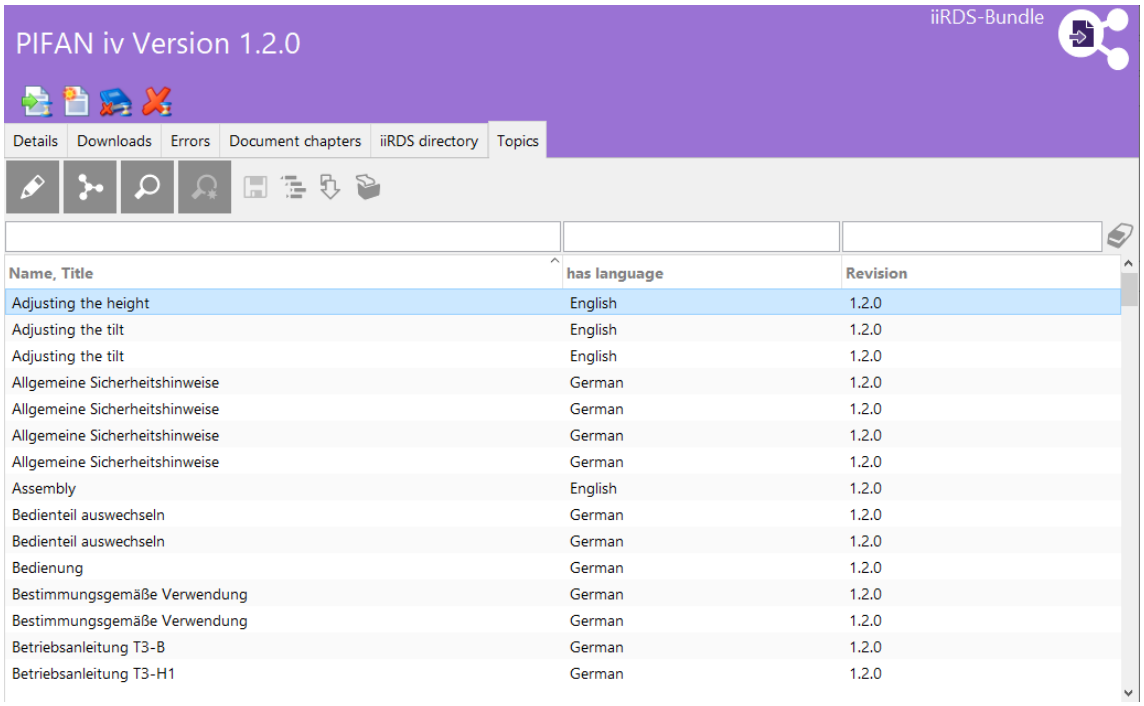
The tab "iiRDS directory" (for iiRDS documents) or "DITA map" (for DITA documents) shows the directory structure or the map structure of the source file on the left side.

When clicking onto an element, the details are shown on the right side. In case of an iiRDS directory, the structure of the selected directory node, its linked topics and their renditions are displayed in forms of a hierarchical structure in the center section accordingly.



2.3.3.5 Topics

The tab "Topics" shows all imported topics tabularly.



2.4 Manage documentation

2.4.1 Publishing new documentation

As central aspect of the content delivery portal is the immediate provision of structured and unstructured content to an end user in a unified manner with personalized interface. Following chapters explain how to provide structured content like DITA-XML, HTML and iiRDS.

2.4.1.1 Publishing unstructured content

Unstructured content comprises all document files that have no explicit machine-readable information about their inherent structure. Mostly, these are documents without any markup language or metadata.

As a solution, i-views content offers the C-REX converting service in order to convert unstructured content into inherent structured DITA content so that a modularized (chapter-based) presentation of previously monolithic documentation can be realized.

Regarding the C-REX converter, Word files with style sheets offer the best prerequisites for a successful conversion into separated topics, whereas PDF files often vary in their construction.

2.4.1.1.1 Converting Documents using C-REX

Prerequisite	*Suitable objects created for:* • Document • Version • DITA-XML-Bundle
Note	PDF files that are intented for conversion must not be saved by a password.
Step 1	*Import document for conversion* Attribute • Import start path Action • Insert start path of ditamap (per default "/input.ditamap").
Step 2	*Import document for conversion* Attribute • Convert and import from file Action 1. Select document file for upload and confirm 2. Ensure that the document file has been converted proberly: If a file withe the name [Date]_Demo.zip has been created in the input field "Import from zip file", the conversion has been performed succesfully. If not, an error message has been created in the tab "Errors".
Step 3	*Import DITA zip file created by conversion* Attribute: • Start path to import document • (has language) Action 1. Select Document element for which the bundle is going to be published (attribute „Start path to import document“). 2. Select the language of the documents (relation target "has language"). 3. Start import by clicking on the button "Start import job" or "Start direct import".

Step 4	*Check import result* Action • In the tab "Errors", check if an error message has been created (recently).
Result	*Document has been converted into DITA content and it has been published by importing the bundle. After update of HTML, the document is available in the web frontend.*

2.4.1.2 Publishing structured content

Structured content means that the metastructure of a document is available in an evaluable and/or machine readable form.

Markup languages allow structured content to be readable by human and by machine. Typically, the concept of structured content comprises XML based documents and the iiRDS standard.

2.4.1.2.1 Publishing a new iiRDS bundle

Prerequisites	*Suitable objects created for:* • Document • Version
Step 1	*Creating new iiRDS bundle* • Create a new object of the type "iiRDS-Bundle" and name it with a self-explanatory name.
Step 2	*Define or create the version of the bundle* Relationen: • has version Aktion • Create a relation from the bundle to the version.

Step 3	*Specify dokument type of the documents* **Note:** The entry "META-INF/metadata.rdf" in the field "Import start path" will be created automatically. Meta relations: • Start path to import document Action • Connect the attribute to the document type by using a meta relation.
Step 4	*Upload* • Start import by clicking on one of the buttons „Start import job“ or „Start direct import“.
Result	*Document is imported; after HTML update the document is published and avialable in the web frontend.*

2.4.1.2.2 Publishing a new DITA document

Prerequisites	*Suitable objects created for:* • Document • Version • (DTD)
Note	All documentation of a bundle must be of the same version.
Step 1	*Creating a new bundle* • Create a new object of the type DITA-XML-Bundle and name it with a self-explanatory name.
Step 2	*Click the details tab*

Step 3	*Specify the file path of the documentation* Attributes: • Import from ZIP file • Import from Directory • Import start path • (DTD catalog path) Actions 1. Upload *.zip file including all its topics and its ditamap file (attribute „Import from ZIP file“). 2. Specify relative file path of the *.ditamap file (attribute „ Import from Directory“). 3. Specify the ditamap file path for each language instance of the document, relative to the root path of the import zip file (attribute „Import start path“) and link it to the document („Start path to import document“). 4. If the ditamap files are missing the language attributes "xml_lang", specify the language of each document instance within the bundle in the field „has language“. **Note:** Pay attention for the "xml_lang" attributes being unique. Duplicate language attributes as well as "xml_lang" attributes concurring with the manual specified language leads to inconsistent language selection in the web frontend. 5. Optional: Specify path and name of the XML catalog file, relative to the root path of the import zip file (attribute „DTD catalog path“).
Step 4	*Specify version of the document* Relations: • has version Action • Linkt the bundle to the related version.
Step 5	*Specify the type of the document* Attribute: • Import start path Meta relation: • Start path to import document Action • Link the attribute to the document by using the meta relation.
Step 6	*Upload* • Start the import by clicking on one of the buttons „Start import job“ or „Start direct import“.

Results	*The document is imported. After succesful HTML update the document is published and available in the web frontend.*

2.4.1.2.3 Publishing content under new version

Note	To ensure that the diff function works correctly for comparing documents of different version, the documents intended to be compared against each other must be related to the same document element (object of the type with the internal name "cdp.document") via the relation "Start path to import document". In case of a DITA-XML import, the topics intended for comparison must have the same DITA URL (= ID attribute of the topic).
Prerequisites	Bundle is already imported. Suitable objects created for: • Document • Version • (DTD)
Step 1	*Create new import bundle*
Step 2	*Assign import bundle to its new version and to the bundle of the same document against which it is intended to compare with* Attribute: • Import from ZIP file Relations: • has version • Start path to import document Actions 1. Specify new document version in "has version" 2. Upload *.zip file of import bundle (in "Import from ZIP file"). 3. Specify relativen path of ditamap file (in "Import start path"). 4. In "Start path to import document", specify the document element containing the other document bundle for which the current document bundle is intended to be compared against. 5. Start import by clicking on one of the buttons "Start import job" or "Start direct import".

Result	*New document version is published in the web frontend. In the content page of the document, the "Difference" button is enabled. It can be invoked for showing the differences between the currently shown chapter and the equivalent chapter of another document version.*

2.4.1.2.4 Uploading a DTD file

Note	The DTD file can be uploaded at two locations: • At the object type DITA-XML-Bundle • Right at the bundle containing the documentation The DTD files must be addressable via the XML catalogs file.
Action	*Click on the tab "Overview"*
Option 1	*Upload the DTD files at the object type DITA-XML-Bundle* Attributes: • DTD ZIP file • DTD catalog path Actions 1. Upload the *.zip file containing the XML catalog file and the DTD files. The XML catalog must be on the top level of the folder path and it must be named "catalog.xml", or ... 2. Specify file path and file name of the XML catalog file.
Option 2	*Upload the DTD files altogether with the documentation* • Upload new DITA document as described in previous chapter
Result	*DTD is stored in the Knowledge Graph.*

2.4.1.3 Providing documents for download

Prerequisites	*Suitable object created for:* • DITA-XML-Bundle

Note	The bundle must be set up fully but it is not mandatory for the document being uploaded until now.
Step 1	*Click on the tab "Downloads"*
Step 2	*Upload document in another file format* Attribute: • Import start path Meta attributes: • HTML version • PDF version Action 1. Check if the ditamap file path is already specified. 2. Select the required meta attribute. 3. Upload the file of the required document file format into the meta attribute.
Result	*In the web frontend, the reader now can access the document in forms of an alternative format.*

2.4.1.4 Uploading supplementary information

Prerequisites	*Suitable objects created for:* • DITA-XML-Bundle
Note	The bundle must be completely specified but the upload doesn't have to be performed until now.
Step 1	*Click on the tab "Downloads"*
Step 2	*Upload the document in another file format* Attributes: • HTML file • PDF file • XLSX file Actions 1. Add the needed attribute values. 2. Upload the files.

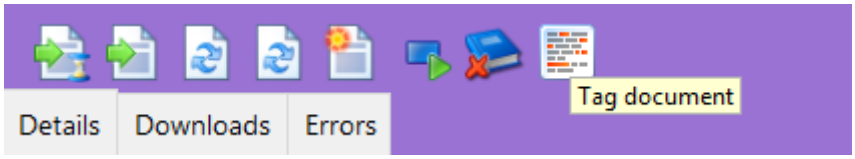
Result	*In the web frontend, the reader now can access additional information about a document.*

2.4.1.5 Checking imports

Prerequisites	*Import of the bundle to be checked is done.*
Step 1	*Click on the tab "Errors"*
Option 1	*Check import errors* Attribute: • Import errors Actions 1. If available, analyse the error log file. If no error log file exists, the import didn't cause any error. 2. If errors are evident, take counter measures accordingly; see section "Import errors" for more details. 3. Restart import
Option 2	*Check for conversion errors* Attribute: • HTML conversion errors Actions 1. Analyse the error log messages 2. Take counter measures accordingly; see section "HTML conversion errors" for more details. 3. Restart HTML update
Result	*Correct import of the document bundle. Document can be accessed in the web frontend.*

2.4.2 Tagging of documentation

Import bundles of recently imported documentation can be tagged by means of the tagging service. The tagging service allows automatic generation of metadata which can be used for networking content.



The tags generated by the tagging are proposed as objects of the type "tag proposal" and linked to the content via relations of the type "tagged test term".

The tagging of further documents leads to the matching of the potential tags with already existing tags.

When a match of a potential tag with exact one existing tag has been identified, the existing tag will be reused via a relation (the potential tag will be discarded therefore) - leading to networking of the content. If several homonymous tags of the type "tag proposal" exist, no additional networking take effect due to the lack of uniqueness, resulting into the tag being created as a separate new tag proposal.

The newly created tags automatically will be shown in a context box in the web frontend when the related chapter is invoked.

For tagging a document via the backend, proceed as follows:

Prerequisite	*Imported bundle*
Step 1	*Start tagging* Element: • Objects of DITA-XML-Bundle • Objects of HTML-Bundle • Objects of iiRDS-Bundle Action 1. Click on the button „Tag Document“. 2. Wait until the tagging process is done. 3. Check if objects of the type "tag proposal" (main type "META-DATA") have been created.
Step 2	Manage tags Actions 1. Check the quality of the newly created tags regarding domain specific usability. 2. Delete tags which are not needed.
Result	*The import bundle is linked to the tags and the tags will be shown as metadata for the relevant chapter in the web frontend.*

2.4.3 Deleting documentation

Note: For deleting content, always use the button “delete content imported by this bundle” before deleting the bundle object itself. This ensures data being kept consistent and prevents the creation of unused content fragments.

Prerequisites	*Suitable objects created for:* • DITA-XML-Bundle
Step 1	*Deleting* Action • At the import bundle instance, click on the button „Delete content imported by this bundle“.
Result	*All content of the document is removed. Information and configuration of the tab "Downloads" and the import bundle itself remain existant. As long as the import file entry hasn’t been removed, the content can be imported again.* *If the bundle, its version and its document element is not used anymore (not even be another bundle), it can be removed by deleting objects of the regarding type.*

2.4.4 Monitoring imports

Prerequisite	*Import startet by job client*
Step 1	*Main type "TECHNICAL" > "Registered objects" > "Scripts"* Action • Open the part of the "Scripts" in the organizer (structured tree) of the Knowledge-Bduiler
Step 2	*Monitoring import process* Object types: • Jobs by script Action • Processing and progess of the import can be monitored within the "Scripts" part.
Result (positive)	*In case of a succeeded import, all process steps have been done and the documentation is imported.*

Result (negative)	*In case of a failed import, a new object type "Failed jobs" arised as a subtype of "Script", containing the causes of the termination.*

2.4.5 Versioning

2.4.5.1 Creating a new version cycle

Step 1	*Create object type* Actions 1. Creat a new object type as subtype of "Version". 2. Specify a self-explanatory name for the new type.
Step 2	*Alternative name for web frontend* Attribute: • Title Action • Specify the alternative title of the version which will be shown in the web frontend.
Step 3	*Sequence of the version cycles in the web frontend* Attribute: • Sequence Action • Specify the sequence number of the version.
Result	*By creating a new object type for version, an new version cycle is created. For the web frontend, this version type can obtain a specific name. The version cycles in the web frontend will be listed in the specified sequence order.*

2.4.5.2 Creating a new version

Prerequiste	*Suitable version cycle is already created*

Schritt 1	*Create a new version object* Object type: • Version Action 1. Create a new object for the relevant version type. 2. Specify a self-explanatory name for the object (which will be shown in the web frontend).
Schritt 2	*Specify version number components separately* Attributes: • Version number component • (Version suffix) Action 1. Specify each single version number component of the version number in forms of an attribute value. 2. Optional: Specify version suffix.
Ergebnis	*A new version object is created. The version number string is created out of the version number components (and its optional suffix).*

2.4.5.3 Creating a new document

Note	The document type is a version independent identifier for a document.
Step 1	*Create document type* Object type: • Document Action • Create a new object of the type "Document" and specify a self-explanatory name for it.
Step 2	*Specify ID* Attribute: • CDP-ID Action • Specify an object identifying character string without whitespaces.

Result	The document type is a bundling element for version-numbered documents. This results into different documents being able to have the same version number while being able to be identified as different document elements. The ID is needed for generating the URL.

2.4.5.4 Hiding version instances

Prerequisite	*Suitable obejcts created for:* • Version
Note	During the publishing process it can be helpful to hide the content until the import of all content has been succeeded. This comes into account if, by example, different language versions of on document have to be imported.
Step 1	*Hide version* Attribute: • Hide in navigation Actions 1. If not apparent, add the attribute "Hide in navigation" by cklicking on "Add attribute" for the relevant version. 2. Set the boolen attribute (by setting the checkmark in the checkbox).
Result	By means of hiding the version, all content of the version will not be displayable or searchable in the web frontend. When the checkmark is removed again, all content of the version is visible again in the web frontend.

2.5 Metadata

2.5.1 Import über DITA subject scheme map

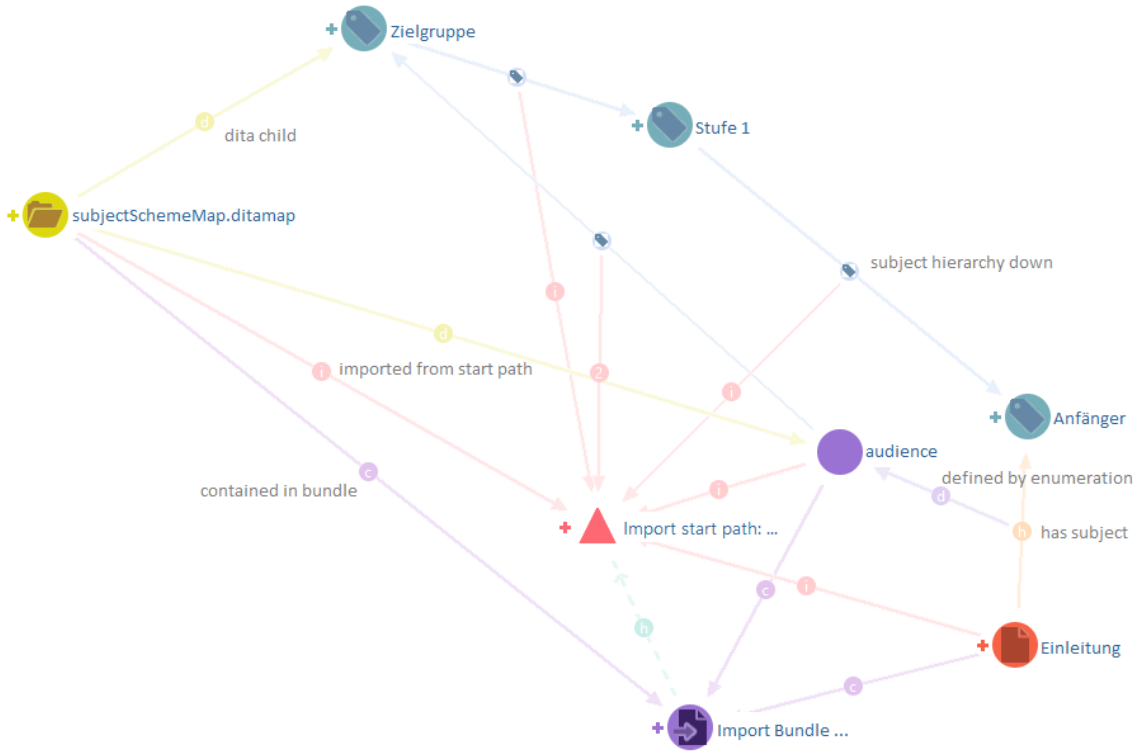
By integrating a subject scheme map in the documentation, interrelations between metadata also can be provided for the reader.

Note: The import of metadata relations by means of <reltable> elements is not supported until now. As a substitute, use subject scheme maps or separate table imports for META-DATA main type.

2.5.2 Reworking metatada in the Knowledge Graph

To identify the context of a metadata exactly, metadata should be equipped with some relations and attribute values therefore. By attaching a DITA subject scheme map, these relations and attributes are set automatically.

The following example shows the structure of a metadata hierarchy after DITA import:



2.5.2.1 Creating an enumeration

Prerequisite	*Suitable objects and attributes create for:* • Import-Bundle, Import start path • topic

Step 1	*Create new metadata class* Object: • enumeration Attributes: • attributedef • title Action 1. Create a new obejct. 2. Specify a self-explanatory name and titel for the object.
Step 2	*Link metadata class with other objects* Objects: • subject • Import-Bundle Relations: • imported from start path Attributes: • Import start path Action • Draw relation to relevant import start path.
Result	*A metadata class that can be assigned exactly to a document.*

2.5.2.2 Creating an networking subjects

Prerequisites	*Suitable obejcts and attributes created for:* • Import-Bundle, Import start path • topic
Note	To be able to assign metadata a specific context (for identifying uniquely), subjects have to be created in a way as described in the following. By making some changes at the frontend configuration, the data model (attribute/object/relation types) can be adopted to individual concepts.

Step 1	*Create new metadata* Object: • subject Attributes: • key • title • url Actions 1. Create a new object. 2. Specify a self-explanatory name for the object. 3. Specify a unqiue key for the object. 4. Specify a URL for the object according to the following pattern: cdp://subject/key-[key of the object].
Step 2	*Link metadata to other metadata* Objects: • subject • Import-Bundle Relations: • subject hierarchy down or • subject hierarchy up • imported from start path Attribute: • Import start path Action 1. Draw relation to another metadata. 2. (If the metadata has no hierarchical predecessor, draw a relation of the type "subject hierarchy up" to the relevant object of "enumeration".) 3. Draw a meta relation from this relation to the relevant Import start path of the document.

Step 3	*Link a topic to the metadata* Objects: • subject • topic Relations: • has subject • defined by enumeration Actions 1. Link metadata with relevant content section via the relation. 2. Draw a meta relation from this relation to the relevant enumeration.
Result	*A metadata which relates its hierarchy position to a certain context (document and metadata type).*

2.6 User roles

Im Bereich Internal Objects > User Role werden die Nutzerrollen administriert. Initial angelegt gibt es die Rollen „Gast“, „Key-User“, „Service technician“ und „Operator“. Beschreibungen der Rollen sind im Anwenderhandbuch zu finden.

Es können aber auch neue Rollen angelegt werden, die dann vom Administrator im Administratorbereich für Benutzer den Benutzern zugewiesen werden können.

Mit Nutzerrollen kann gesteuert werden welche Funktionen und Ansichten dem Nutzer zur Verfügung stehen und welche Inhalte er sehen darf. Diese sind in den folgenden Unterkapiteln beschrieben.

2.6.1 User roles define what view configurations a user sees

Nutzerrollen bestimmen, welche View-Konfigurationen einem Nutzer angezeigt werden. Um dies zu demonstrieren gibt es initial eine eigene Facettenkonfiguration für die Nutzerrollen „Key-User“ und „Gast“. Den Nutzern der Rollen werden teils unterschiedliche Facetten auf der Suchergebnisseite angezeigt. Die Relation „User Role nimmt View-Konfig“ am Typ der Nutzerrollen bestimmt welche View-Konfigurationen für die Rolle angezeigt werden sollen. Im Falle der Facettenkonfiguration gibt es zusätzlich zu den spezifischen Ansichten eine Fallback-Ansicht, die genommen wird, wenn ein Nutzer keine der beiden Rollen hat. Rollenspezifische alternative Ansichten müssen in ein flexibles Panel eingebettet sein.

2.6.2 User roles define what content a user sees

Nutzerrollen bestimmen, welche Topics dem Nutzer angezeigt werden. Bislang ist dies für iiRDS Rollen umgesetzt. Die Voraussetzung hierfür ist, dass das Topic über die Relation „bezieht sich auf Qualifikation“ mit der Rolle im iiRDS Datenmodell verknüpft sein muss und diese Rolle wiederum mit der Relation „iiRDS Rolle entspricht User role“ mit der User Rolle.

So können Nutzer, die die Rolle „service technician“ haben auch nur Topics sehen, die „service technician“ als Metadatum haben. Topics, die keine iiRDS Rolle als Metadatum haben, sind immer sichtbar.

2.6.3 Special role key-user

Die Nutzerrolle „key-user“ bestimmt, dass der Nutzer im Frontend den Administrator-Bereich sehen kann.

2.6.4 Possibility to define further authorizations

Aufgrund des patentierten feingranularen Rechtesystems können aber auch weitere unterschiedliche Berechtigungen definiert werden, theoretisch bis hin zur Art des Zugriffes auf einen einzelnen Eigenschaftswertes eines semantischen Elements. Dies wäre dann aber Teil eines kundenspezifischen Projekts, d.h. im Einzelfall muss geschaut werden, wie solche zusätzlichen Anpassungen von Upgrades und Updates berücksichtigt werden können.

2.7 Translation Layering

i-views content supports usage of different languages by means of translation layering. This means that the identifying name attributes of all objects and types in i-views in principle can be specified with translations of that name. For the content delivery portal, the following attributes can be provided with translations:

Attribute	Domain of definition	Usage in the CDP
Name (Primary name)	Objects of Top-Level-Type	Metadata
Title (Primary name)	Objects of CDP Types of Version	Titles of content units like Chapter, Document, Import-Bundle, Version etc.
title	Objects of DITA	topics
Label	Objects of several types of the view configuration	Title of views in the web frontend (like tables, menu labels etc.)

The translation layering is needed when it is intended to display the content in different languages in the web frontend and - besides the content language - also metadata should be found when searched in different languages.

When translation layering is needed, the translations for the relevant attributes must be activated first if not initially configured.

If tranlations for an attribute are existing, the Knowledge-Builder shows the attribute values in forms of the application language. Starting the Knowledge-Builder in another language is described in the standard i-views documentation:

http://documentation.i-views.com/5.3/.

2.7.1 Configuring translation layering

Prerequisite	*Knowledge about the location of detail configurations.*
Caution	Risk of information loss Removing a language from the translation of an attribute type leads to deletion of **all** existing attibute values in this language. When removing a language, it must be ensured that this form of information loss is acceptable.
Note	If all translations are going to be deactivated, one remaining language variant is asked to be selected as substitute for the untranslated form of the attribute values. If the translation is activated for the values of an attribute, all attribute values are assigned to the language selected at first.
Step 1	*Open the detail configuration of the attribute* Attribute: • Name (Primary name) Actions 1. Open the context menu of the name attribute ba clicking on the context menu button . 2. Select the menu entry "Schema". 3. Ensure that it is the correct attribute (by looking on "defined for"). 4. Select the tab "Details", then the category "Translations".
Step 2	*Add translation language* Actions 1. If the attribute value translation is not already activated, activate the checkbox "Attribute values have translations" and add the language to the translations. 2. Add further languages by clicking on "Add".
Result	*Knowledge Graph elements with translation layering.*

2.7.2 Adding translated terms

Prerequisites	*Knowledge about following details:* • *Location of detail configurations* • *Creating import mappings*
Note	When translation layering is activated, attention must be paid on attribute values being mapped to the correct language of the attribute value. Per default, the application language of the Knowledge-Builder is used as preselection ("Current language").
Step 1	*Import translated values* Attributes: • Name • Title Actions 1. Create import mapping for the translated attribute values to be imported. 2. At the attribute in the import mapping, select the tab "Language" and specify the language for the attribute values of the attribute. 3. Start the import.
Step 2	Specify translations manually Attributes: • Name • Title • Label Actions 1. For semantic objects: Select the object in the objects list in order to display the detailed view of the object. For view configuration elements: Open the view configuration. 2. Open the drop-down menu of the name/title/label attribute and type in the translation.
Result	*Knowledge Graph objects with translated attributes.*

2.8 Terminology

In i-views content, terminolgy lists can be used for automated searching for synonyms. For this purpose, the object/relation/attribute types in the main type TERMINOLOGY can be used and enhanced.

To build up the terminology base an XML-, XLS(X)- or CSV file can be imported. The content must be mapped onto the object types and the available relations. For more information about the import of a file, see: http://documentation.i-views.com/5.3/.

When a search term is entered in the web frontend, the search term is compared to the terminology base. If a more preferred term can be found, the web frontend suggests the preferred term on top of the search results list. The related sarch pipeline for this purpose is "cdp.didYouMean.term" which will be invoked by the script "cdp.vc.search.didYouMean" and can be adjusted if required.

2.8.1 Constructing the terminology base manually

When the terminology base is intended to be set up in the Knowledge Graph from scratch, do the steps as follows:

Prerequisite	*Defined required object-, relation- and attribute types in the main type TERMINOLOGY (if standard types are not sufficient).*
Step 1	*Create objects* Objects: • preferred term • admitted term • deprecated term Attributes: • Name • Term name Actions 1. Create object for the relevant object type and specify the term name. 2. If required, create domain specific attributes or translations.

Step 2	*Network terms* Objects: • preferred term • admitted term • deprecated term Relation: • has less preferred term or • has preferred term Action • Link the terms via the relaions "has less preferred term" or "has preferred term".
Result	*Preferred, admitted and deprecated terms are interrelated to each other. In the web frontend, admitted terms will be propesed as substitute search terms when the typed in search term is equal to one of the deprecated or addmitted terms.*

2.8.2 Importing *.tbx-files

The terminology import in forms of *.tbx files is performed by means of an XML import mapping with XPath expressions. An example is shown as follows:

```
<?xml version='1.0'?>
<!--<!DOCTYPE martif SYSTEM "TBXBasiccoreStructV02.dtd">-->

<martif type="TBX-Basic" xml:lang="en-US">
    <text>
        <body>
            <termEntry id="c8">
                <descripGrp>
                    <!--switched--><definition metaType="descrip">Maßverkörperung zum Erfassen der Außen- oder Innenabmaße von
                    Werkstücken</definition>
                    <!--switched--><source metaType="admin"></source>
                </descripGrp>
                <langSet xml:lang="de">
                    <tig>
                        <term>Messschieber</term>
                        <!--switched--><partOfSpeech metaType="termNote">noun</partOfSpeech>
                        <!--switched--><administrativeStatus metaType="termNote">preferredTerm-admn-sts</administrativeStatus>
                    </tig>
                    <tig>
                        <term>Schieblehre</term>
                        <!--switched--><partOfSpeech metaType="termNote">noun</partOfSpeech>
                        <!--switched--><termType metaType="termNote">fullForm</termType>
                        <!--switched--><administrativeStatus metaType="termNote">admittedTerm-admn-sts</administrativeStatus>
                    </tig>
                    <tig>
                        <term>Schublehre</term>
                        <!--switched--><partOfSpeech metaType="termNote">noun</partOfSpeech>
                        <!--switched--><termType metaType="termNote">acronym</termType>
                        <!--switched--><administrativeStatus metaType="termNote">deprecatedTerm-admn-sts</administrativeStatus>
                    </tig>
                </langSet>
            </termEntry>
        </body>
    </text>
</martif>
```

XPath: /martif/text/body/termEntry/langSet/tig/administrativeStatus[text() = preferredTerm-admn-sts]

This entry identifies the first content of the preferred term according to the administrativeStatus entry and it consists of an absolute path pattern of the (opening) tag in the *.tbx file.

All further path patterns can be defined relative to the preferred term entry:

XPath: ../../../descripGrp/definition
= Definition of a term

XPath: ../../tig/administrativeStatus[text() = admittedTerm-admn-sts]
= Value of the admitted term

XPath: ../../tig/administrativeStatus[text() = deprecatedTerm-admn-sts]
= Value of the deprecated term

XPath: ../term
= The value of the actual term

XPath: ancestor::termEntry/attribute::id
= ID of the term entry which is valid for the whole group of preferred/admitted/deprecated terms. This XPath expression is a pattern for the path nodes.

In principle, the XPath identification works as follows:

XPath: ../../xyz[text() = abc] = All entries within the identifier xyz and the text abc, relative to the abstract path of the preferredTerm tag.

XPath: ../../xyz = All entries within the identifier xyz, relative to the absolute path of the preferredTerm tag.

Note: Be aware that XPath expressions are case-sensitive and whitespace-invariant. This means that lower case letters and upper case letters and whitespaces must be written correctly, otherwise the expression cannot be interpreted correctly.

2.9 Simple frontend configuration

When adjustments are made to the data model (e. g. networking metadata or terminology), also the standard configuration of the web frontend needs to be adjusted if needed in order to let the reader benefit from the new information structures.

2.9.1 Configuration names and Registry keys

For reuse of viewconfiguration elements like panels (layout panel, dialog panel) and views (group, graph, menu, edit, properties, table etc.), unique **configuration names** must be specified.

Note: Define one consistent way for the wording of the configuration names an keep this practice. This makes it easier for you to find elements again for reuse in backend and frontend.

For configuration names, a self-explanatory, classifying name convention has proven its worth. Example: "cdp.panel.document.context" tells that this is a panel for CDP elements

which contains context information for documents.

The assignment of **registry keys** for scripts and queries work on the same principle: By setting classifying names, elements are made identifiable and referenceable. The punctuation of the names automatically generates a hierarchy within the registered types of the Knowledge-Builder.

Example: "cdp.context.metadata" is the registry key of a structure query that is used for context boxes for obejcts of the type "META-DATA". By means of this wording, the structured query is automatically assorted to *TECHNICAL* > *Registered objects* > *Queries* > *cdp* > *context*.

All other queries which have the same prefix "cdp.context.[...]" in their registry key can be found within the same folder on the same type hierarchy level in the Knowledge-Builder.

2.9.2 Location of the configuration

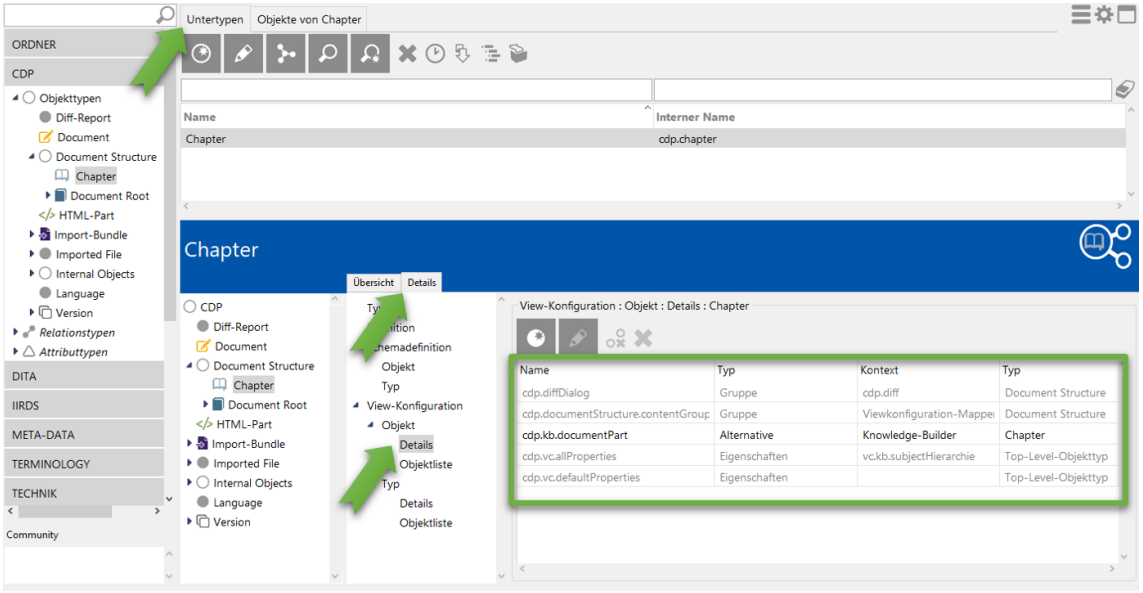
The frontend configuration is similar to the backend configuration. However, there are difference between the *object configuration* defined for the regarding obejct type and the *view configuration* in the viewconfiguration mapper.

2.9.2.1 Object configuration

The object configuration is part of the view configuration and it is applicable for objects of the respective object type only. For the object type "Chapter", there are following web frontend settings for instance:

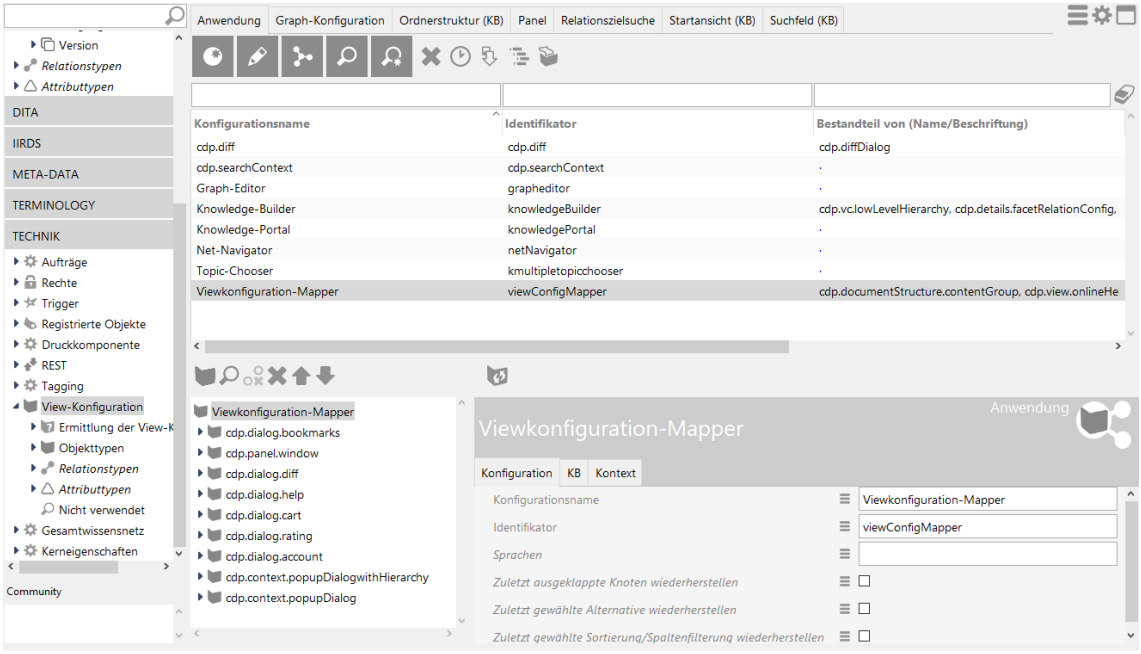
- cdp.diffDialog: Diff dialog for comparing chapters
- cdp.documentStructure.contentGroup: Configuration for document structure

The object configuration can be found within the tab "Detail":



2.9.2.2 View configuration

The viewconfiguration mapper is an application of the Knowledge-Builder and it can be accessed via TECHNICAL > View configuration, including the tab "Application":



The elements of the view configuration are used within the viewconfiguration mapper. The viewconfiguration mapper defines layout and style of the content in the web frontend. For further information, see http://documentation.i-views.com/5.3/.

2.9.3 Panel configuration

Since version 1.3, i-views content offers a new operating concept using the Viewconfig-mapper: Configuration of the web frontend by means of panels that contain the view on the data. This means that the inner construction of the website is built up in a hierarchical structure, enabling layout and styling to be customized at no time, either for individual information needs or for including corporate design.

The new i-views content 1.4 release offers now more features - such as styling the web frontend using the LESS standard, render modes and plugins.

As a part of the Viewconfig-mapper, mechanisms and components like panel influencing, menu actions and responsive design are provided for the user interface of the content delivery portal.

More information on how to configure the Viewconfig-mapper elements can be found in chapter 3 of the general documentation for i-views 5.3: http://documentation.i-views.com/5.3/.

The i-views content 1.4 web frontend is divided into following areas:

- Login page (multilingual):
- Start page (new since version 1.4)

- Navigation page (since version 1.3):
- Content area (CDP page):
- Navbar

Construction of the content area (configuration name of panel written in brackets):

- Navbar with logo, persistent menu and search entry field (cdp.panel.navbar)
- Depending on the shown content, either a search result configuration (cdp.panel.searchResults) with facets for search results filtering (cdp.panel.sidebar.facets) or a configuration containing a structured tree for content selection (cdp.panel.sidebar.[...])

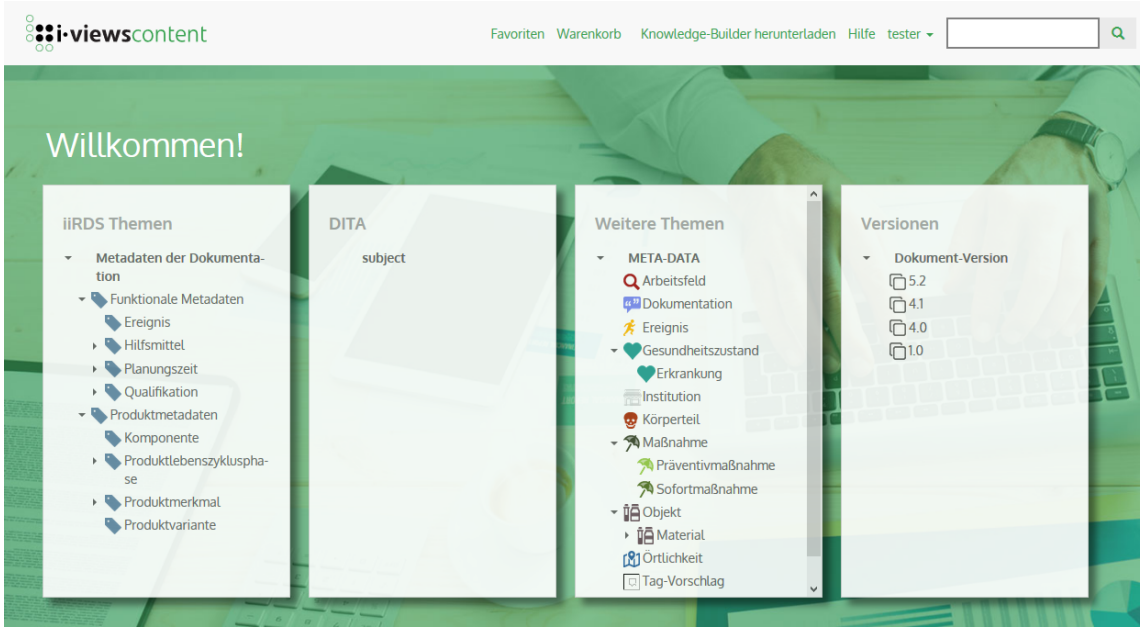
Standard features

- Navigation page for individual content access via metadata, without the need for entering search terms
- Usage of the tagging service for automated keywording and metadata assignment
- Import and administration of iiRDS bundles
- Conversion of documents (Word, PDF) into DITA-XML by means of the C-REX converter

User specific features

- Content cart for bundled PDF download of content
- Rating chapters
- Bookmarking chapters for quick access

2.9.4 Navigation page



The navigation page of the web frontend is realised by the content of the document structure and its metadata structure. Per default, views for following types are configured:

- IIRDS
- META-DATA
- Document versions

The navigation page is configured by means of the layout panel "cdp.panel.startPage". For the views of each structured tree, a sub-configuration is defined which contains a group view including a tree view.

2.9.4.1 Configuring the navigation page

Prerequisites	*Existing Metadata* *Knowledge about view configuration in i-views*
Note	Hierarchies can be defined by means of ascending or descending relations or by means of structured queries.
Step 1	*Create new panel for view* Configuration object: • cdp.panel.startPage.topics Actions 1. Select panel "cdp.panel.startPage.topics" in the structured tree of the viewconfiguration mapper: *TECHNICAL > View configuration > Viewconfiguration mapper > cdp.panel.window > cdp.panel.main > cdp.panel.startPage > cdp.panel.startPage.topics* 2. Create sub panel with fixed view. 3. Within the tab "Layout", specify the class "cdp-start-page-box". 4. Create sub configuration „Group“ and name it.
Step 2	*Create hierarchy view* Object configuration: • Group Action 1. In the group view, create an object configuration of the type "hierarchy". 2. Within the tab "Style" of the hierarchy, search for the style "cdp.navTree" and assign it.

Option 1	*Define a query for the hierarchy* Object configuration: • Hierarchy Actions 1. Add a structured query for "Structured query (up)". 2. Define the structured query, beginning with the object type to be shown as root node. From there on, continue with the schema relations down to the object types for the subordinate nodes and add the identifying access parameter "Accessed element". 3. Add a structured query for "Structured query (down)". 4. Define the structured query, beginning with the object type to be shown as sub node. From there on, continue with the schema relations up to the object types for the superordinate nodes and add the identifying access parameter "Accessed element". 5. Select the option "Do not show detail view". 6. Refresh view configuration and browser.
Option 2	*Select relations for the hierarchy* Object configuration: • Hierarchy Actions 1. Select a relation type for "Relation (up)" for the ascending brach of the tree. 2. Select a relation type for "Relation (down)" for the descending brach of the tree. 3. Select the option "Do not show detail view". 4. Refresh view configuration and browser.
Result	*Navigation page with individual navigation.*

2.9.5 Configuring new context boxes

Prerequisites	*Existing metadata* *Knowledge about queries in i-views* *Knowledge about table view configuration*

Note	The context boxes can be configured and adjusted as needed. For this purpose, basic knowledge about defining queries and table view configuration is required; for more information see http://documentation.i-views.com/5.3/. In general, context boxes are search result tables with one column.
Step 1	*Open the frontend configuration for the context box* View configuration: • cdp.panel.document.context Actions 1. Select the configuration in the structured tree of the viewconfiguration mapper. 2. Open the group "cdp.documentStructure.contextSearches" (line "Sub-Konfiguration").
Step 2	*Add new context box* Object configuration: • Query Action • Create a new object configuration (of type "Query") within the group.
Step 3	*Define query for the context box* Object configuration: • Query Action 1. Add a new query for the created query configuration. 2. Select query type "Structured query". 3. Define the structured query.
Step 4	*Create table for presentation of query restults* Object configuration: • Table Action • Create a table view for the query configuration (self-explanatory name).

Step 5	*Configure table* Object configuration: • Column • Column element • Structured query Attribute: • Label • Property Menu actions: • Action for invoking dialog panels: Create action of action type "Selection", newly created standard script for "Script" and related dialog panel for "Show result in panel" • Action for linking to related chapters: Create action of action type "Selection" and related panel "cdp.panel.document.content" for "Show result in panel" Actions 1. Label the table (for web frontend). 2. Assign the selection action for the table configuration in "Action on a row". 3. Create a column configuration within the table configuration. 4. Create a column element for the column. 5. Define the result list entries to be shown by means of a property or the a structured query which specifies the property to be shown for the search result element (which is the output from the query described in step 3)
Step 6	*Style the table* Table style: • cdp.cdpContextBox Action 1. Select the tab "Table" within the table configuration. 2. Select the tab "Styles". 3. Add the existing style "cdp.cdpContextBox".
Step 7	*Refres view configuration* Action • Refresh the view configuration and the browser.

Result	*In the web frontend, the new context box with the assigned style and its label is shown. Every hit will be shown in an individual row in the table. In the row the values will be shown according to the configured attribute value or structured query of the column element.*

2.9.6 Configuring metadata details for context boxes

Prerequsites	*Set up context boxes* *Existing metadata* *Knowledge about queries in i-views* *Knowledge about view configuration in the view configuration mapper*
Note	The detailed view for metadata is shown in a dialog as soon as the reader clicks on a hit in the context box. For determination of the view content, queries need to be defined. For more information about defining queries, see http://documentation.i-views.com/5.3/. The view configuration of the dialog panel either can be defined via an object detail configuration of the related metadata object type (option 1) or view an independent view configuration within the panel configuration of the viewconfiguration mapper (option 2).
Option 1	*Create object detail configuration for dialog panel: Variant for reuse of the dialog pane "cdp.contex.popupDialog" (**flexible view**).* Note: The object oriented view configuration is invoked by the action type "Selection" which is oriented on the accessed element and is suitable for views onto exactly one object type. Actions 1. Open the detail view of the metadata object by selecting the tab "Detail". 2. Open the object configuration for objects of the type by selecting the category "Details" in *View configuration* > *Object* > *Details*. 3. Create a new object configuration (type "Group") by clicking onto the button "New" and name it with a self-explanatory name. 4. Succeed with Step 1.

Option 2	*Create new view configuration for dialog panel:* *Variant with independent view configuration in panel configuration of the viewconfiguration mapper (**fixed view**).* Note: The object independent view configuration is invoked by means of the action type "Show" and it can be located easier due to its location within the panel structure of the viewconfiguration mapper. Because of the independence of the accessed object, views with mixed object types can be realized easier. Action 1. In the viewconfiguration mapper, create a dialog panel (panel type: "linear layout") and define the layout orientation. 2. Define self-explaning name and a configration name for the dialog panel. 3. Create a window title panel for the dialog panel. 4. For the dialog panel having a label and a close-button, either create a label configuration or reuse the existing label configuration "cdp.context.popup.label". 5. Create sub panel with fixed view. 6. Create sub configuration for metadata overview. 7. Succeed with Step 1.
Step 1	*Open frontend configuration for context box* View configuration: • cdp.panel.document.context Actions 1. Select the configuration in the structured tree in the viewconfiguration mapper. 2. Open the group view "cdp.documentStructure.contextSearches" (line "Sub configuration").
Step 2	*Open the table of the context box* View configuration: • cdp.documentStructure.contextSearches Actions 1. Select the object configuration (Query) of the related context box within the group "cdp.documentStructure.contextSearches". 2. Open the search results table by selecting the entry "Table".

Step 3	*Add an action for invocation of the dialog* View configuration: • Table of the context box query Action: • demo.openNewLayer Actions 1. In the table configuration, select the tab "Configuration". 2. For the entry "Action on a row", add the action "demo.openNewLayer" by typing in the configuration name or by clicking on "Select relatin target". Note: The action "demo.openNewLayer" relys on the dialog panel "cdp.context.popupDialog" which has a flexible view. If separate panel configurations need to be used (Option 2), a new action must be created here fore which links to the relevant dialog panel (by defining "Show result in panel").
Step 4	*Refresh view configuration* Action • Refresh view configuration and browser view.
Result	*In the web frontend, a dialog will be shown when clicking on a list item of the context box.*

2.9.7 Adjust global search

The global search of i-views content (which performs the fulltext search when a search term is entered in the web frontend) is predefined for querying dedicated object types. In the case the taxonomy has been changed or enhanced, the global search might need to be adjusted as well.

The global search can be found in forms of the search pipeline "cdp.vc.search" of the search field view "cdp.view.navbar.searchField", contained in the panel "cdp.panel.navbar.search".

The search result view contains a menu "cdp.menu.searchField" for starting the query, containing the actin "cdp.action.search" and "cdp.action.autocomplete". These actions determine in which panels the search result will be shown. The display format of the search result depends on the kind of view in the respective target panel.

Action	Show result in panel	Configuration to be adjusted
cdp.action.search	cdp.panel.navbar	(none)

	cdp.panel.searchResults.didYouMean	Only when changes of the terminology took place (see chapter "Terminology")
	cdp.panel.searchResults.list	Table cdp.table.searchResult in view cdp.vc.searchResults
	cdp.panel.sidebar.facets	Facet view cdp.facets
	cdp.panel.sidebarWithContent	(none)
	cdp.panel.updateUrl	(none)
cdp.actio	cdp.view.navbar.searchField	Query cdp.vc.autocomplete.nameQuery within script cdp.actionResponse.autocomplete

2.9.7.1 Adjusting query configuration

Prerequisites	*Knowledge about queries in i-views* *Knowledge about table configuration*
Note	The global search is configured in forms of a search pipeline. For further unformation about configuration of search pipelines and other query types, see http://documentation.i-views.com/5.3/.
Step 1	*Open frontend configuration for global search* View configuration (Search result view): • cdp.view.navbar.searchField Action • Open the configuration.
Step 2	*Open and configure the search pipeline:* • cdp.vc.search Actions 1. Open the query of the query configuration 2. Rework the query according to individual needs 3. After rework and testing (within the test environment), close the query.

Step 3	*Rework the table for the search result view (according to the changes made in the search pipeline)* Object configuration: • cdp.table.searchResult (search result table) • cdp.facets (facets) Actions 1. Open and adjust the table configuration of the query configuration. If required, adjsut the search result amount (page size of the table). 2. If needed, open the facet configuration and adjust it to the new properties.
Step 4	*Adjust the configuration for the type-ahead function* Object configuration: • cdp.actionResponse.autocomplete (type-ahead) Action • Adjust the query for the type-ahead function.
Step 5	*Refresh the view configuration* Action • Refresh the view configuration and the browser view.
Result	In the web frontend, search results will be displayed according to the criteria deifined in the search pipeline. Note: Missing indices on the attributes for queried elements can lead to lack of query performance. For more information about indices see http://documentation.i-views.com/5.3/.

2.9.7.2 Adding query facets

Prerequisites	*Knowledge about queries in i-views* *Knowledge about facet configuration*
Note	The search facets use the search results of the global search for filtering the displayed results. For further information, see http://documentation.i-views.com/5.3/.

Step 1	*Open the frontend configuration of the global search* View configuration: • cdp.view.navbar.searchField Actions 1. Open the configuration. 2. In the menu "cdp.menu.searchField", open the action "cdp.action.search". In the action configuration, open the panel for the facets "cdp.sidebar.facets". 3. Open the panel "cdp.facets".
Step 2	*Add new facet confguration* Attribute: • Label Object configuration: • cdp.facets • Facet • Query for term detection • (Query for parent term detection) Actions 1. Open the facet configuration. 2. Add a new facet to the facet view. 3. Define the query for the facet, starting from the search result amount of the global search. 4. Add a label for the facet.
Step 3	*Refresh view configuration* Action • Refresh the view configuration and the browser view.
Result	*In the web frontend, the newly created facet is visible and can be used for the search results amount.* *Note: Missing indices for object attributes within the queried objects can lead to lack of performance. For more information on how to configure indices, see http://documentation.i-views.com/5.3/.*

2.9.7.3 Adjusting synonym proposals

Prerequisites	*Knowledge about queries in i-views*
Note	The query for synonyms is done by a search pipeline. For further information, see http://documentation.i-views.com/5.3/. The query "cdp.didYouMean.terms" is invoked within the global search "cdp.action.search" by means of the script "cdp.vc.search.didYouMean".
Step 1	*Open query* Main type TECHNICAL: • Registered objects > Queries > cdp Query: • cdp.didYouMean.terms Action • Within the regiestered objects, look up for the search pipeline an open it.
Step 2	*Rework the search pipeline* Actions 1. Rework the query according to individual needs. 2. Test the query in the testing environment and close the query.
Result	*In the web frontend, synonymous terms (of the main type TERMINOLOGY) will be searched and proposed additionally according to the criteria of the search pipeline.*

2.9.8 Style refinements (LESS/CSS)

i-views content 1.4 offers the possibility of refining the web frontend style by means of CSS or LESS. For instance, typical use-cases are changing the background image for the start page and refinement of colors according to corporate design principles.

Switching styles

To switch between different style, do as follows:

1. Within the main type "CDP", change to subtype "CSS Style".
2. Select the tab "Instances" and select the style which is going to be used instead of the currently active style.
3. Click on the button "Activate CSS": A Boolean attribute indicates "Style is active".
4. Refresh the browser view.

   Result: Web frontend with redefined style.

Using custom background for start page

The background of the start page can be adjusted as follows:

1. Change to the style object "Custom background".
2. At the entry "CSS source", download the zip file by clicking on the window button ("Open").
3. Extract zip file.
4. In the folder "gfx", replace the background image.
5. In the CSS file, replace the file URL of the class definition "background-image".
6. Pack both the CSS file and the image folder into a new zip file.
7. In line "CSS source" of the style object "Custom background", upload the zip file by clicking on the button "Choose file".
8. Activate the style by clicking on the button "Activate CSS".
9. Refresh the browser view.

   Result: Web frontend with custom background on the start page.

Creating custom styles

In the administration section of the web frontend, a LESS file called "Less-source" can be downloaded as style template.

When compiling LESS to CSS and/or activating the CSS Style, all variables will be evaluated by the system and translated for application to the respective elements.

It is recommended to enhance existing styles rather than defining styles from scratch since it is easier to implement. For this purpose, the option "Import default LESS sources when compiling" can be selected for the entry "Default less style" when compiling the customized LESS file.

Creating a style instance

For applying a new style, a style instance must be created:

1. Within the main type "CDP", change to the subtype "CSS Style".
2. Select the tab "Instances" and create an instance.
3. Name the instance with a self-explanatory name.

Note: The title of the style object is used for display in the administration section of the web frontend.

Uploading and converting a LESS file

1. Select the style object in the list of instances.
2. For the "LESS archive" entry in the detailed view, upload the new LESS archive in forms of a zip file.
3. For "Path to LESS file", specify the file name of the *.less file.
4. Optional: Specify "Default LESS source".
5. In order to create the CSS style, click on the button "Compile LESS to CSS". Check for succeeded compiling according the entry "LESS compiling status".

6. To activate the style, click on the button "Activate CSS": The Boolean attribute "Style is active" will be displayed.
7. Refresh the browser view.

   Result: Web frontend with refined style.

Uploading a CSS file

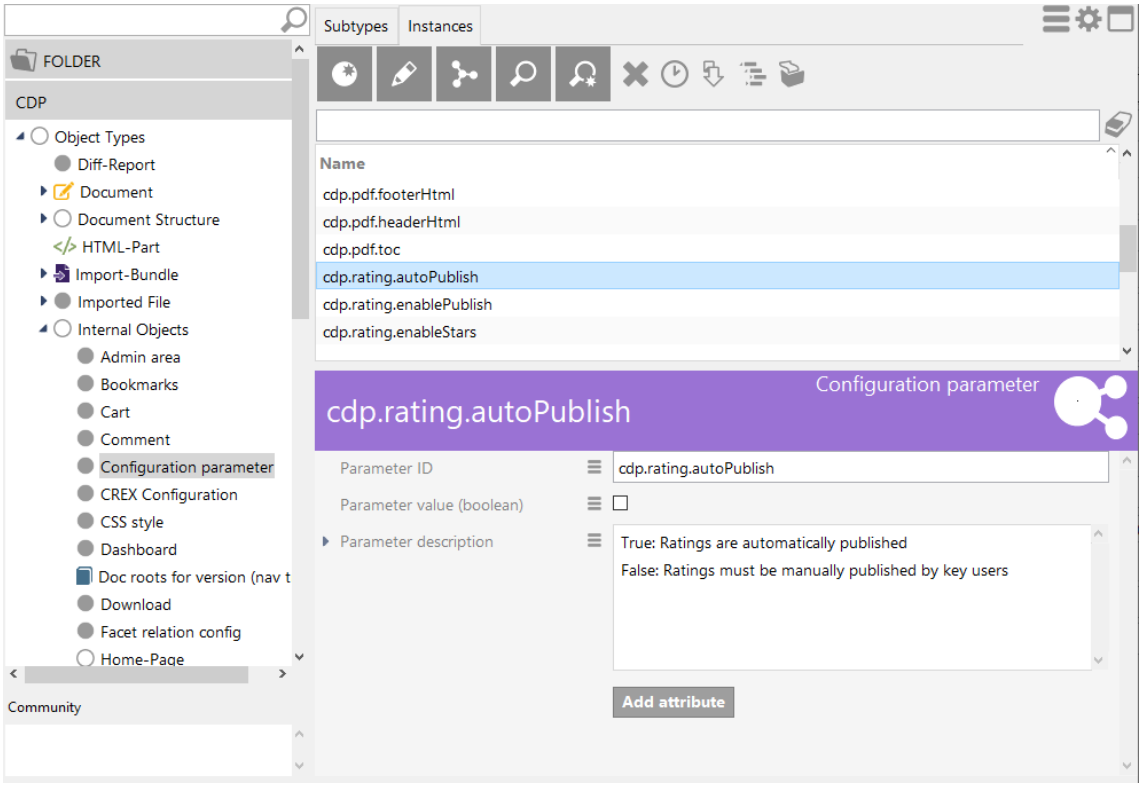
1. Select the style object in the list of instances.
2. For the "CSS archive" entry in the detailed view, upload the new CSS archive in forms of a zip file.
3. For "Path to CSS file", specify the file name of the *.css file.
4. To activate the style, click on the button "Activate CSS": The Boolean attribute "Style is active" will be displayed.
5. Refresh the browser view.

   Result: Web frontend with refined style.

2.10 Configuration parameters

Das Aussehen und Verhalten des CDP kann zum Teil mit Konfigurationsparametern beeinflusst werden. Diese finden sich unter "CDP > Internal Objects > Configuration parameter".

Durch Änderung des Werts ("Parameter value") wird die Konfiguration beeinflusst. "Parameter description" beschreibt die Auswirkung des Parameters.

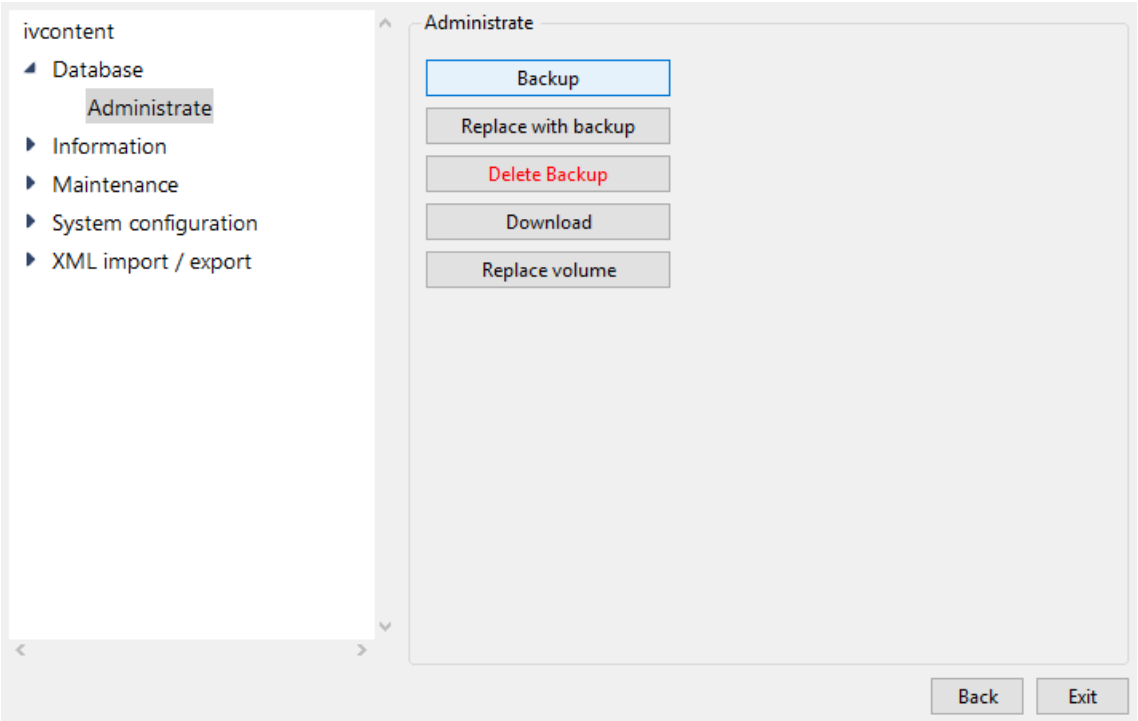


2.11 Backup

Automated backups

Automated backups can be configured and provided by means of the Admin tool.
For more information, see http://documentation.i-views.com/5.3/.

Manual Backups



Backups also can be set up in the Admin tool manually. To do so, open the admin tool for the network and click on "Start". Then go to "Database" > "Administrate". There you can click on "Backup" in order to create a backup of the current Knowledge Graph.

To restore a backup, click on "Replace with backup" and select the snapshot you want to take for replacement of the current Knowledge Graph.

Local copy of the network

To secure the Knowledge Graph in forms of a local copy, the Admin tool can be used here fore. To do so, open the admin tool for the graph and click on "Start". Then go to "Database" > "Administrate". There you can click on "Download" in order to get a local copy of the Knowledge Graph.
Note: Per default, downloaded volumes can be found in the "volumes" folder which is located in the same folder as the admin tool.

Replacement of a Knowledge Graph by a local copy

Caution: When replacing a Knowledge Graph by a local copy, the Knowledge Graph on the server will be overwritten.

Analogous to the backup, local copies can be uploaded from the "volumes" folder by means of the "Replace volume" function in order to replace the Knowledge Graph on the server.

2.12 Creat a new database administrator

The Admin tool is also used for creating new backend users. For more information, see: http://documentation.i-views.com/5.3/.

2.13 Installation on Windows

2.13.1 Java Installation

I-views content requires a 64-bit Java 8 runtime environment (JRE), which is not included in the scope of delivery. Since there are numerous commercial (e.g. Oracle Java) and non-commercial versions (e.g. Adopt OpenJDK) of Java 8, each of which is offered with different additional services and update plans, it is up to the respective server operator to select a suitable runtime environment.

The i-views content services read the path to the Java runtime environment upon start from the Windows registry (path "HKEY_LOCAL_MACHINE \ SOFTWARE \ JavaSoft \ Java Runtime Environment"). When installing the Java runtime environment with an installer (MSI or EXE file), these registry entries are usually created automatically. However, if the Java runtime environment can only be provided without an installer, then the registry entries may still have to be created manually.

I-views content first reads the exact version designation of the latest installed Java version from the registry key "CurrentVersion" in the path "HKEY_LOCAL_MACHINE \ SOFTWARE \ JavaSoft \ Java Runtime Environment" from the Windows registry (String value; e.g. "1.8. 0_292 ").

Then the path to the DLL file of the respective Java Virtual Machine is read from the key "RuntimeLib" in the path "HKEY_LOCAL_MACHINE\SOFTWARE\JavaSoft\Java Runtime Environment\[Version]", where "[Version]" will be replaced by the value from "CurrentVersion" (e.g."HKEY_LOCAL_MACHINE \ SOFTWARE \ JavaSoft \ Java Runtime Environment \ 1.8.0_292"). The DLL file specified there is then loaded by the i-views content process and the virtual machine is executed "in process". Since the i-views content services use the 64-bit architecture of Windows, this also requires a Java runtime environment, which is also designed for the 64-bit architecture. If a 32-bit Java runtime environment is installed on the system, i-views content can possibly find the DLL file, but it cannot be loaded in the 64-bit process. Therefore, in the event of starting problems with the i-views content services, please check whether the above registry entries are set correctly and refer to a 64-bit Java runtime environment DLL file. As an example, the MSI installation package for the 64-bit Adopt OpenJDK 8 runtime environment can be found at the following address:

```
https://adoptopenjdk.net/installation.html?variant=openjdk8&jvmVariant=hotspot#x64_win-jre
```

2.13.2 MSI installation package

An MSI installation package with the name "Installer-i-views-content-1.4.X" is available for installation under Windows, with a sequential patch version number instead of the "X".

The MSI installer can either be run in GUI mode (double-click on the MSI file in the file explorer) or from a command prompt with the Windows utility "msiexec". Administrator rights are required for both the GUI installation and the installation from the command prompt. When running in GUI mode, the target directory for the installation and the components

(features) to be installed are selected in dialog windows. When installing on a server, all features should be selected for installation.

2.13.3 MSI-Features

The following features are contained in the MSI installation package:

Name	MSI Feature-ID	Description
Server components	ServerFeature	Installs the Knowledge Graph server service (i-views Mediator), a service for performing searches and background tasks (i-views job client) and a service that provides the HTTP endpoints for the web UI and the REST API (i-views Bridge).
Volume	VolumeFeature	Installs a data set with demo data (Knowledge Graph "ivcontent-demo-master").
Web server (nginx)	WebServerFeature	Installs an NGINX web server service that acts as a reverse proxy and forwards incoming HTTP requests to various HTTP endpoints (i-views Bridge, Grafana) based on path patterns. The NGINX service also acts as a load balancer for forwarding to the i-views bridges.
Grafana	GrafanaFeature	Installs the Grafana and InfluxDB services for collecting and evaluating performance and usage statistics.
Client tools	ClientFeature	Installs the Knowledge Graph modeling and administration tools (i-views Knowledge Builder and Admin Tool). These should be installed on the server, but can also be installed individually (without the other features) on remote workstations from which system or content administrators want to administer the i-views Knowledge Graph.

2.13.4 Installation using "msiexec"

The following command should be used to invoke the installation using "msiexec" ("X" must be replaced by the respective number from the specific MSI file name):

```
msiexec.exe /i Installer-i-views-content-1.4.X.msi /lvx* install-ivc.log /passive
```

The parameter "/lvx* install-ivc.log" creates an installation log file named "install-ivc.log" in the current working directory (requires write access in the working directory).

The "/passive" parameter ensures that a minimal GUI with a progress indicator is displayed

during the installation and that a dialog box with an error message is also displayed in case of installation errors. If the installation fails with an error message, please send the log file "install-ivc.log" to your contact at Empolis.

The installation command described above can be supplemented with additional arguments (MSI properties) in order to adapt certain parameters of the installation. As a rule, properties must be appended in the form PROPERTYNAME="PROPERTYVALUE" at the end of the "msiexec" command. Multiple property arguments must be separated by spaces.

Details on the MSI properties are described in the documentation for "msiexec" at: https://docs.microsoft.com/w
server/administration/windows-commands/msiexec

2.13.5 MSI-Properties

The following MSI properties are relevant for the i-views content installation:

Name	Description
ADDLOCAL	Comma-separated list of the MSI feature IDs to be installed. The possible feature IDs are named in the MSI feature list above. All features are installed by default. As an example, adding the following argument to the end of the "msiexec" command would install only the client tools: `ADDLOCAL="ClientFeature"`
APP_PROFILE	Name of a configuration profile that is to be used to initialize the configuration files during installation. Possible values are "prod" or "demo". The "demo" profile is used by default. Details on the configuration profiles are described in the chapter of the same name.
BASE_URL	Base URL for access to the i-views content web server, which will be stored in the client tools configuration files as standard server address (standard value: "http://localhost:31020").
DATADIR	Absolute path to the directory in which the i-views content data should be stored (default value: "C:\Program Files\i-views content\data").
INSTALLDIR	Absolute path to the installation directory of the application (default value: "C:\Program Files\i-views content").
LOGDIR	Absolute path to the directory in which the log files of i-views content are to be saved (default value: "C:\Program Files\i-views content\logs").
STARTSERVICE	Specifies whether the generated Windows services should be started immediately during installation ("1") or not ("0"). The default value is "1".

2.13.6 Configuration profiles

The MSI installation package is supplied with two predefined configuration profiles that can be selected during installation via an MSI property:

- Profile "demo": This profile is used for demo operation and ensures that i-views content on the host system does not consume as much memory and CPU resources. However, simultaneous requests to the web UI or the REST API of i-views content are processed sequentially in this configuration, which can lead to higher response times.

- Profile "prod": This profile is used for installation for productive operation on a system with 4 CPU cores. An installation with this profile requires approx. 2-3 times more memory than an installation with the demo profile. In this configuration, the system can process 3 requests to the Web-UI or REST-API at the same time and, if necessary, use all CPU cores to the maximum.

2.13.7 Installed services

The following Windows services are created during installation depending on the selected MSI features:

MSI-Feature-ID	Names of created Windows Services
ServerFeature	i-views content-Mediator i-views content-Bridge i-views content-Jobclient
WebServerFeature	i-views content-Nginx
GrafanaFeature	i-views content-InfluxDB i-views content-Grafana

The start-up type "Automatic" is selected for the services created during installation. In addition it is configured that in case of a crash the services will be restarted after 1 minute waiting time. This will happen for a maximum of three consecutive crashes. After crashing three times a service will remain stopped.

2.13.8 Uninstallation

The uninstallation of i-views content can either be carried out via the Windows settings ("Apps and Features" or "Programs and Features" in older Windows versions) or using the Windows "msiexec" utility.

If the MSI file belonging to the installation is still available at the time of the uninstallation then the following call can be used to uninstall the product:

```
msiexec.exe /x Installer-i-views content-1.4.X.msi /lvx * uninstall-ivc.log /passive
```

Alternatively, the product GUID can be used for uninstallation:

```
msiexec.exe /x {7df091f0-676e-4dcc-987f-384860168440} /lvx* uninstall-ivc.log /passive
```

The "/lvx* uninstall-ivc.log" parameter is used to generate a log file of the uninstallation. If the uninstallation fails with an errror then the exact reason can be determined from the content of the file "uninstall-ivc.log". The files contained in the i-views content data and log directories as well as the configuration files of i-views content (*.ini) are not removed during the uninstallation. These files and the corresponding directories may have to be deleted or archived manually after installation.

2.14 Appendix

2.14.1 Glossary

Term	Explanation
admitted term	Admitted synonym for a terminology term (medium priority)
CDP	Content Delivery Portal; tool for providing personalized content
deprecated term	Prohibited synonym for a terminology term (lowest priority)
iiRDS	Intelligent Information Request and Delivery Standard
i-views	Abbreviation for für intelligent views gmbh; specifies the main product "Knowledge-Builder"/"KB" of intelligent views gmbh (in combination with software context or a version number)
i-views content	Content Delivery Portal; comprises the Knowledge-Builder including web frontend configuration
KB	Knowledge-Builder
Panel	Layout place holder for the viewconfiguration mapper web frontend, in which views of the data are presented
preferred term	Preferred synonym for a terminology term (high priority)
REST	REpresentational State Transfer; interface for data transder between applications
Translation Layering	Provision of multilingual attribute values by means of meta attributes

VCM	Viewconfiguration mapper; applicataion within the i-views content Knowledge-Builder for displaying content in the web frontend
View	View on data. Represents content of data in a dedicated form of interpretation: text, table, edit, property, graph etc.

2.14.2 Import errors

Error message	Meaning
Unable to import non-existing binary file, File not found	External file source could not be loaded (e. g. a picture reference)
Error: SAX parser warning while parsing file [...]. The DTD does not permit text here	DITA elements have been used for the DITA-XML import bundle which are not declared for usage within i-views content. See section "proposed standards".
"b" is not permitted at this point in the "title"	Invalid inline format elements are evident in a content bundle file. Possible reason: Imported word file that has been converted from PDF.
"body" is not permitted at this point in the "title" node	Invalid markup structure within a content bundle file. Possible reason: Imported word file that has been converted from PDF.

2.14.3 HTML conversion errors

Error message	Meaning
Could not find image for href [...] while generating HTML for a CDP.ImageConverter	Referenced picture could not be loaded. Possible reason: Picture is not contained in the bundle *.zip file.
Warning: Missing HTML converters for DITA classes: topic/object	

2.14.4 Suggested standards

Standard/purpo	Implementation	
	–	++
DITA / representation of a metadata taxonomy	reltable	subject scheme map
DITA / glossary entries	glossentry, glossref	topic, topicref
DITA / IDs	Identical IDs within the same language instance of a document in a DITA bundle	Use unique IDs for topics within the same language instance of the document in the DITA bundle
DITA / diff funktion	Different/no IDs for topics of different bundles but of the same document element which are intended to be compared	Use the same ID for topics of different bundles but of the same document element for comparison using the diff functionality