SLUB
Wir führen Wissen.

d:swarm - Hilfe zum Alpha-Release

Seitenverzeichnis

Schritte:

	Was ist d:swarm?

SLUB
Wir führen Wissen.

Was ist d:swarm?

d:swarm ist ein dem Sharing-Gedanken verpflichtetes **Datenintegrations- und modellierungswerkzeug** zur verlustfreien Überführung von Daten aus heterogenen Datenquellen in ein flexibles (elastisches), graphenbasiertes Datenmodell, welches sich explizit in den (Linked) Open Data Webgraphen einbettet. Alle designten Logiken können in zukünftigen Versionen der Software der Anwender-Community zur Nachnutzung zur Verfügung gestellt werden.

d:swarm ist eine **Middleware-Lösung**, auf deren Grundlage sämtliche bibliothekarischen Datenverarbeitungsprozesse durchgeführt werden können und die sich zwischen **bestehende Datenmanagementsysteme** und **bestehende Frontend-Applikationen** legen lässt.

d:swarm sorgt für die **Verbesserung** der Datenqualität, für die Deduplizierung, das Merging und die FRBRisierung bibliographischer Daten sowie ihre semantische Anreicherung und Publikation als LOD.

d:swarm kanalisiert alle individuellen **bibliothekssystemische Datenflüsse** und erzeugt zu jeder bibliothekarischen Ressource einen den Bedürfnissen der jeweiligen Institution entsprechenden Datensatz im Sinne einer Single Version of the Truth (SVOT).

d:swarm ist als Web-Anwendung konzipiert und läuft in allen modernen Webbrowsern.

Schritte:

d:swarm - Hilfe zum Alpha-Release	Funktionsumfang der Alpha-Version

SLUB
Wir führen Wissen.

Funktionsumfang der Alpha-Version

Die Alpha-Version von d:swarm dient zu Demonstrations- und Testzwecken, um das Arbeitsprinzip der webbasierten grafischen Anwendungsoberfläche (GUI) zu validieren.

Modellierungsumfang

Zum Zeitpunkt des Alpha-Release kann bereits der folgende ETL-Prozess modelliert werden :

1. Testdaten (von geringer Größe ~ 1.000 Records) im CSV- oder XML-Format können in d:swarm importiert werden.
2. Das Schema der importierten Testdaten kann konfiguriert werden.
3. Es können Projekte angelegt werden, in denen alle Logiken für die Überführung einer importierten Daten-Ressource in ein graphenbasiertes Datenmodell zusammengefasst werden.
4. Das konfigurierte Input-Schema einer Testdaten-Ressource kann auf ein Output-Schema gemappt (Schema Mapping) werden.
5. Für jedes Mapping können Datentransformationen mit den zur Verfügung stehenden Funktionen erstellt werden.
6. Es können Filter definiert werden, unter welchen Bedingungen eine Datentransformation durchgeführt werden soll.
7. Die modellierten Mappings - inklusive der Datentransformationen - können während der Modellierung ausgeführt werden, um die Arbeitsergebnisse direkt zu betrachten und in den zentralen Datenhub in einem Graphdatenmodell zu speichern.
8. Alle Ergebnisse können in der graphischen Anwendungsoberfläche der Graphdatenbank exploriert werden.
9. Alle Daten des zentralen Datenhubs können zur weiteren Verwendung außerhalb von d:swarm in RDF exportiert werden.

Einschränkungen Alpha-Version

- Die aktuellen d:swarm Designansätze der GUI sind noch nicht umgesetzt.
- Es gibt noch kein Nutzer- und Rechtemodell in d:swarm. Dies hat zur Konsequenz, dass jeder alles darf. Jeder Nutzer hat beispielsweise Zugriff für die Bearbeitung aller in d:swarm importierten Daten-Ressourcen und kann Projekte anderer Nutzer einsehen und manipulieren (überschreiben). Daher ist es empfehlenswert für die Tests ein eigenes Projekt anzulegen. Die Daten im zentralen Datenhub sind in Konsequenz die Ergebnisse aller Tester.
- Darüber hinaus steht eine große Menge an Funktionalitäten in der Alpha-Version noch nicht zur Verfügung. Der Roadmap ist zu entnehmen, welche Funktionalitäten in den zukünftigen Releases bereitgestellt werden.

Schritte:

Was ist d:swarm?	Import von Testdaten

SLUB
Wir führen Wissen.

Import von Testdaten

Um Mappings und Datentransformationslogiken definieren zu können, müssen Testdaten ins d:swarm-System importiert werden.

> **Hinweis**
> In der Alpha-Version erfolgt dies über Datei-Uploads. Zukünftig wird d:swarm auch das Laden von Testdaten über Schnittstellen wie OAI-PMH und dergleichen erlauben.

Import-Workflow

Importieren Sie Daten mit der Import-Funktion:

- Klicken Sie auf den Tab "Import" (1).
- Öffnen Sie den Dialog zum Auswählen einer Datei, indem Sie auf "Durchsuchen" bzw. "Datei auswählen" (abhängig vom verwendeten Browser) klicken (2).
- Wählen Sie die gewünschte Datei auf Ihrem Computer aus (es kann pro Upload nur eine Datei verarbeitet werden).
  - ***Hinweis:*** *Obwohl beliebige Dateiformate importiert werden können, stehen aktuell nur Schema-Konfiguratoren für CSV und XML Formate zur Verfügung. Es erfolgt keine Validierung des Dateiformats.*
- Mit dem Klick auf den Button "Öffnen" beenden Sie den Dateiauswahl-Dialog.
- Versehen Sie die zu importierenden Daten-Ressource mit einem sprechenden Namen (obligatorisch) und wählen Sie eine geeignete kurze Beschreibung für den Datensatz (obligatorisch) (3).
- Führen Sie den Import-Vorgang mit einem Klick auf "Submit" aus (4).

Nach einem erfolgreichen Import befindet sich die Daten-Ressource im Tab "Data" innerhalb des Bereiches "Available for configuration".

Schritte:

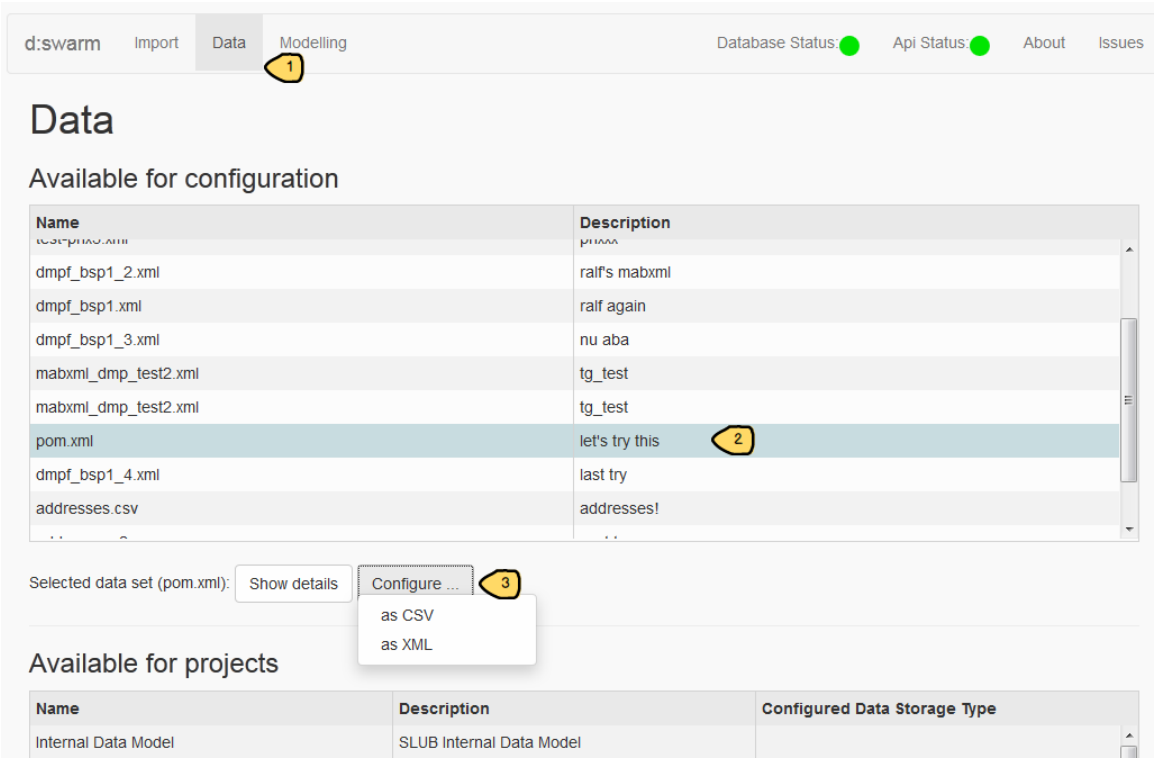
Funktionsumfang der Alpha-Version	Daten zur Schema-Konfiguration auswählen

SLUB
Wir führen Wissen.

Daten zur Schema-Konfiguration auswählen

Nach dem Import von Testdaten muss das Schema der Daten-Ressource konfiguriert werden. Hierfür stehen im Alpha-Release von d:swarm Konfiguratoren für die Formate CSV und XML zur Verfügung.

Auswahl-Workflow



- Klicken Sie den Tab "Data" an (1)
- Markieren Sie den gewünschten Daten-Ressource im Bereich "Available for configuration" (2)
- "Show Details" zeigt Informationen zur importierten Daten-Ressource.
- Klicken Sie auf "Edit Configuration" und wählen sie aus dem Pop-up-Menü eine der beiden Möglichkeiten "... as CSV" o der "...as XML" (3)

Schritte:

Import von Testdaten	CSV-Daten-Ressource konfigurieren XML-Daten-Ressourcen konfigurieren

SLUB
Wir führen Wissen.

CSV-Daten-Ressource konfigurieren

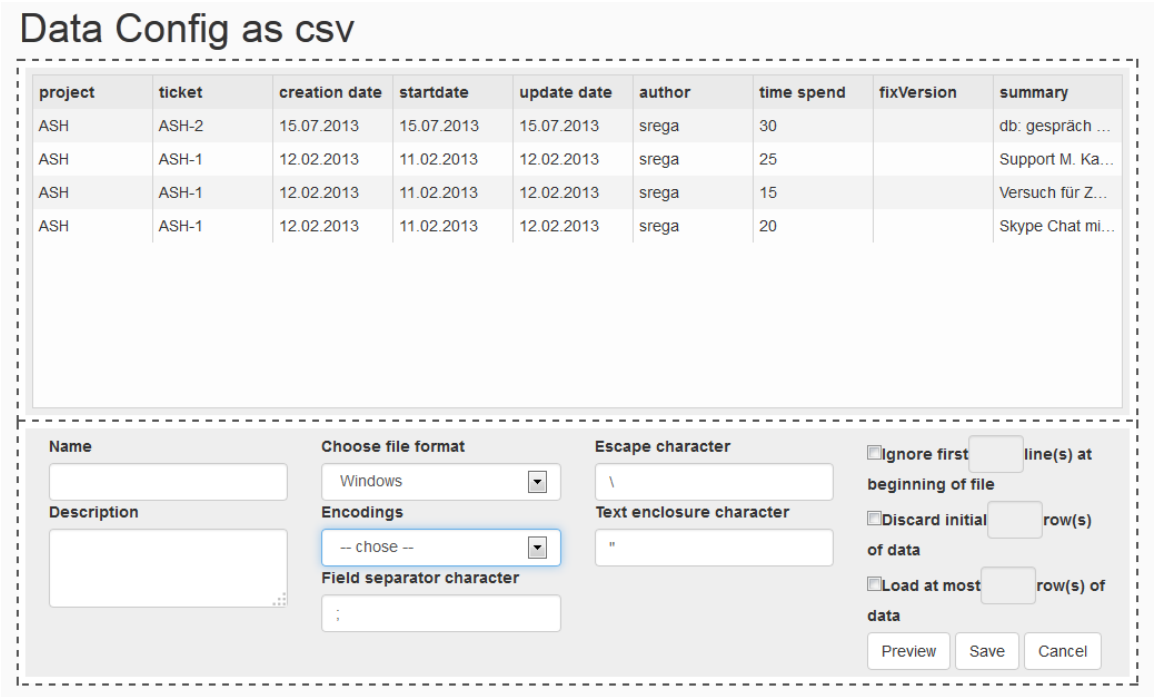
Nach der Selektion eines zu konfigurierenden Daten-Ressource kann durch Klicken auf den Menüpunkt "... as CSV" das Schema eines CSV-Daten-Ressource konfiguriert werden. Um die Parametrisierung zu erleichtern, wird eine Vorschau auf die ersten Werte der hochgeladenen Datei generiert.

Konfigurationsmöglichkeiten

Für CSV-Dateien stehen folgende Konfigurationsmöglichkeiten zur Verfügung:

Parameter	Beschreibung
Name	Benennung der Konfiguration
Description	Beschreibung der Konfiguration
Choose file format	Zeilenende (Unix / Windows)
Encoding	verwendete Zeichenkodierung
Field separator character	verwendeter Feldtrenner (Besonderheit : Tabulartor = \t)
Escape character	verwendete Escape-Sequenz zum Maskieren von Zeichen mit Steuerzeichenfunktion
Text enclosure character	verwendetes Textbegrenzungszeichen
Ignore first ... lines at beginning of file (Checkbox)	definiert die Anzahl von Zeilen am Anfang der Daten-Ressource, die nicht ausgewertet werden sollen (z.B. weil sie leer sind oder eine abweichende Struktur haben)
Discard initial ... rows of data (Checkbox)	definiert die Zeilenanzahl, die am Beginn der Datenstruktur, die die Datensätze enthält, übergangen wird
load at most ... rows of data (Checkbox)	definiert die Anzahl von Zeilen der Datenstruktur, die als Testdaten geladen werden sollen

Konfigurations-Workflow



- Konfigurieren Sie das Schema entsprechend des Dateiaufbaus mit den zur Verfügung stehenden Konfigurationsfunktionalitäten.
- Falls sich die Vorschau auf die ersten Werte des importierten Datasatzes nicht dynamisch aktualisiert, kann sie durch Klick auf "Preview" aktualisiert werden.
- Per Klick auf den Button "Save" werden Ihre vorgenommenen Konfiguration gespeichert.

- Per Klick auf den Button "Cancel" werden Ihre Eingaben verworfen und Sie werden auf den Tab "Data" zurückgeleitet.

Schritte:

Daten zur Schema-Konfiguration auswählen	Projekte anlegen und bearbeiten

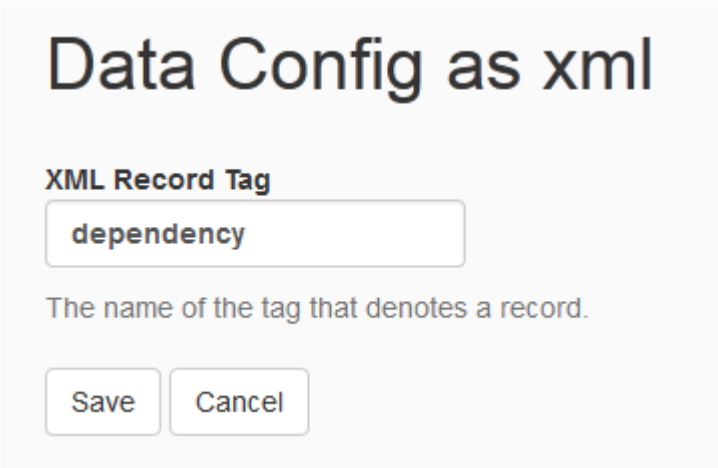
SLUB
Wir führen Wissen.

XML-Daten-Ressourcen konfigurieren

Nach der Selektion eines zu konfigurierenden Daten-Ressource kann durch Klicken auf den Menüpunkt "... as XML" das Schema einer XML-Daten-Ressource konfiguriert werden. Aus der XML-Struktur der importierten Daten-Ressource wird das grundlegende XML-Schema automatisiert abgeleitet.

Hinweis
Aktuell werden XSD-Schema-Definitionen der XML-Dateien noch nicht automatisch extrahiert.

Konfigurations-Workflow



- Geben Sie hierfür im Textfeld "XML Record Tag" den Namen des Elementes an, mit welchem Records in der Daten-Ressource typisiert sind.
- Per Klick auf den Button "Save" werden Ihre vorgenommenen Konfiguration gespeichert.
- Per Klick auf den Button "Cancel" werden Ihre Eingaben verworfen und Sie werden auf den Tab "Data" zurückgeleitet.

Hinweis
Wenn Sie einen anderen Bezeichner nehmen oder einen Bezeichner der ggf. gar nicht in der XML-Daten-Ressource auftaucht, dann wird die Daten-Ressource falsch eingelesen. Sie bekommen diesbezüglichen keinen Hinweis und können nur mit Hilfe der Graphexploration überprüfen, ob Ihre Daten-Ressource richtig eingelesen wurde.

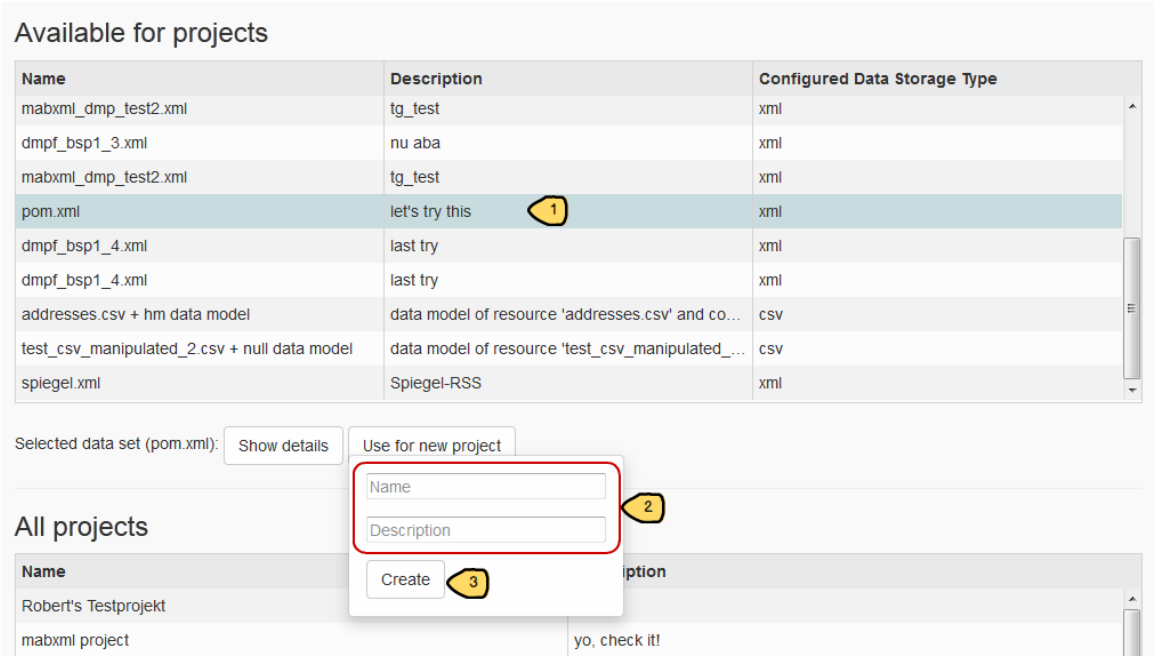
Schritte:

Daten zur Schema-Konfiguration auswählen	Projekte anlegen und bearbeiten

Projekte anlegen und bearbeiten

Nach der Schema-Konfiguration muss ein Projekt angelegt werden, in welchem Mappings und Datentransformationen modelliert werden können.

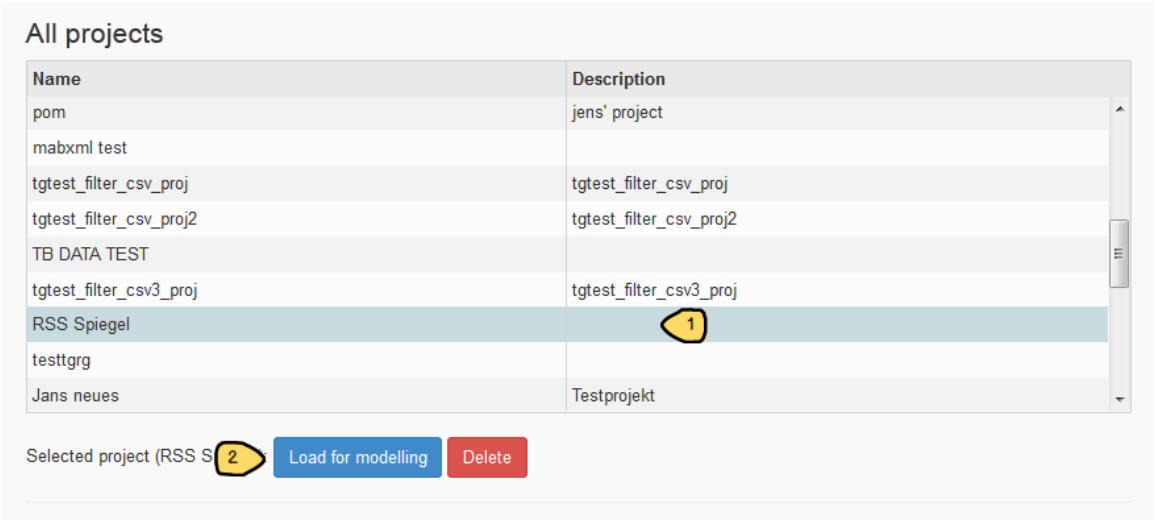
Projekt anlegen - Workflow



- Klicken Sie auf den Tab "Data".
- Wählen Sie im Bereich "Available for projects" eine Daten-Ressource aus (sie enthält bereits ein konfiguriertes Schema) (1).
- Es werden die Optionen "Show details" und "Use for new project" angeboten.
- Der Button "Show details" ermöglicht eine erneute Konfiguration des jeweiligen Datensatzes.
- Der Button "Use for new project" zeigt ein Pop-up-Menü an. Benennen Sie das neue Projekt und geben Sie optional eine kurze Beschreibung ein (2) und klicken Sie auf "Create" (3).
- Das neu angelegte Projekt erscheint in der Liste der Projekte "All projects".

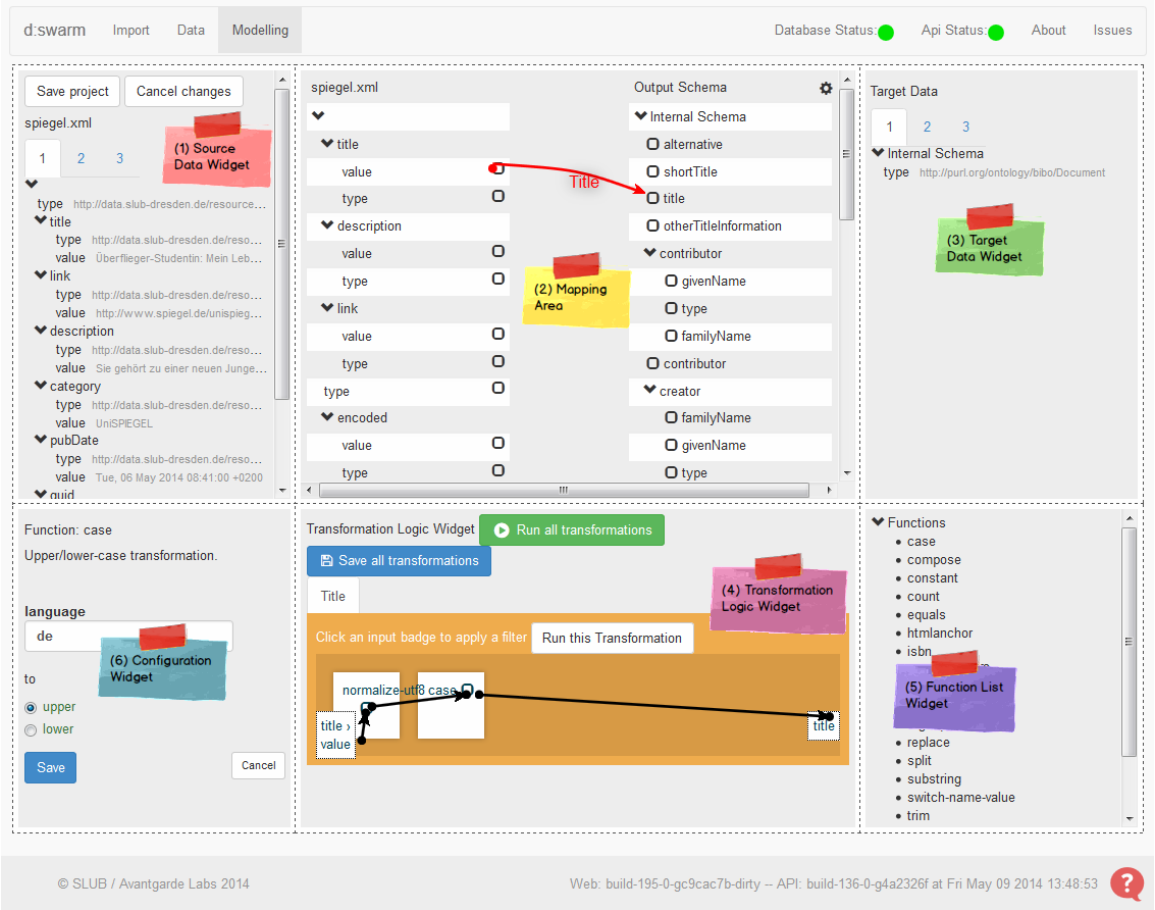
Projekt bearbeiten - die Modellierungsperspektive

Um ein Projekt in der Modellierungsperspektive bearbeiten zu können, gehen Sie wie folgt vor:



- Markieren Sie im "Data" Tab das gewünschte Projekt (1) im Bereich "All projects".
- Nach dem Klick auf den Button "Load for modelling" (2) wird das Projekt automatisch in die Modellierungsperspektive geladen.

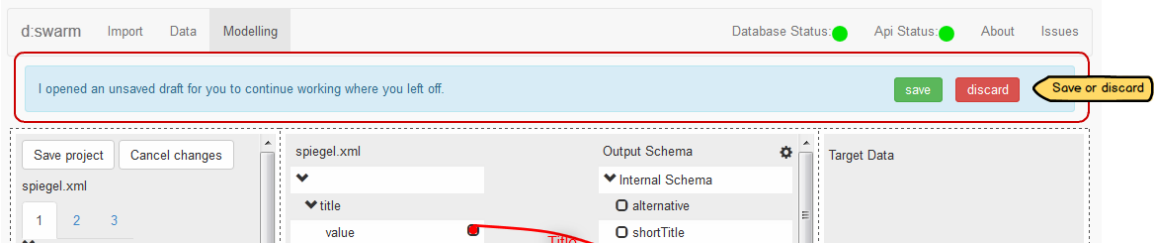
Die Modellierungsperspektive ist das zentrale Werkzeug von d:swarm, mit dem Schema-Mappings und Datentransformationen modelliert werden können. Die Modellierungsperspektive ist in sechs Bereiche udn eine Lightbox untergliedert.



Bereich	Anordnung	Beschreibung / Verwendungszweck
Quelldaten Records	Links Oben (1)	• Für die Unterstützung bei der Modelierung von Mappings und Datentransformationen werden Beispiel-Records aus der konfigurierten Daten-Ressource dargestellt. • Falls mehrere Records vorhanden sind, werden die ersten Records unter verschiedenen Tabs angezeigt. • ***Hinweis:*** *Im Alpha-Release können spezifische Records noch nicht selektiert werden.*
Schema-Mapping	Mitte Oben (2)	• Auf der linken Seite wird das Input-Schema, auf der rechten Seite das selektierte Output-Schema dargestellt. • Durch die Verbindung eines Input- mit einem Output-Attributpfad kann schrittweise eine komplettes Schema-Mapping erzeugt werden.

Ergebnis-Records	Rechts Oben (3)	• Einzelne oder alle Transformationen eines Projektes können auf die Beispiel-Datensätze angewendet werden. Die Ergebnisse können somit schon während der Modellierung - "auf Knopfdruck" - auf ihre Korrektheit evaluiert werden.
Transformation	Mitte Unten (4)	• Nach der Definition eines Mappings kann in diesem Bereich ein Transformations-Workflow mit den bereitgestellten Funktionen m odelliert werden.
Funktionen	Rechts Unten (5)	• Eine Liste von Funktionen steht für den Aufbau eines Transformations-Workflow zur Verfügung. • ***Hinweis:*** *Im Alpha-Release stehen nur recordzentristische Funktionen zur Verfügung. Zukünftig werden recordübergreifende Funktionen (z.B. Look-Ups) oder eine rekursive Verschachtelung von Transformationen ermöglicht.*
Funktionsparametrisierung	Links Unten (6)	• Parametrisierungen für selektierte Transformationsfunktionen können hier vorgenommen werden.
Filter	Lightbox	• Filter ermöglichen die Einschränkung von Transformationen auf bestimmte Ausprägungen eines Input-Attribut pfades, so dass komplexe Bedingungen für die Ausführung der Transformationslogik konstruiert werden können.

Sie können Ihre Arbeit jederzeit durch Klick auf den Button "Save project" (oben links) speichern. Alle Änderungen werden aber auch automatisch gespeichert. Falls Sie die Modellierungsperspektive ohne manuelle Speicherung verlassen, wird nach dem Neuladen des Projektes ein Dialog angezeigt, der Sie zum Speichern oder Verwerfen der zwischenzeitlichen Änderungen auffordert.



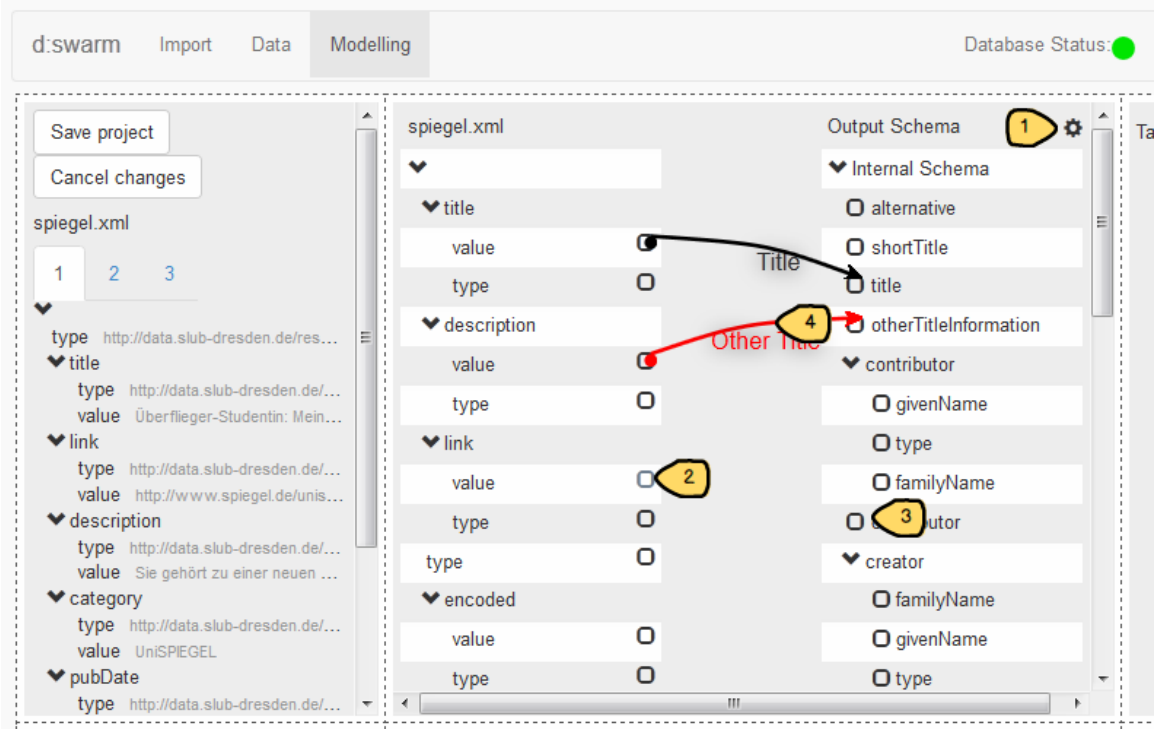
Schritte:

CSV-Daten-Ressource konfigurieren Configure XML Data Resources	Mappings definieren

Mappings definieren

Mapping-Workflow

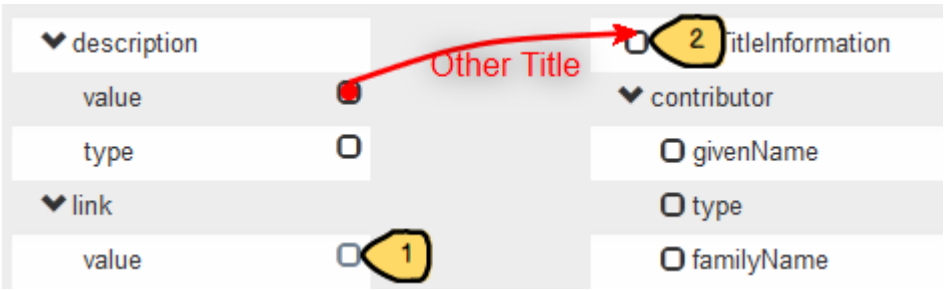
Um Mappings zu definieren, gehen Sie wie folgt vor:



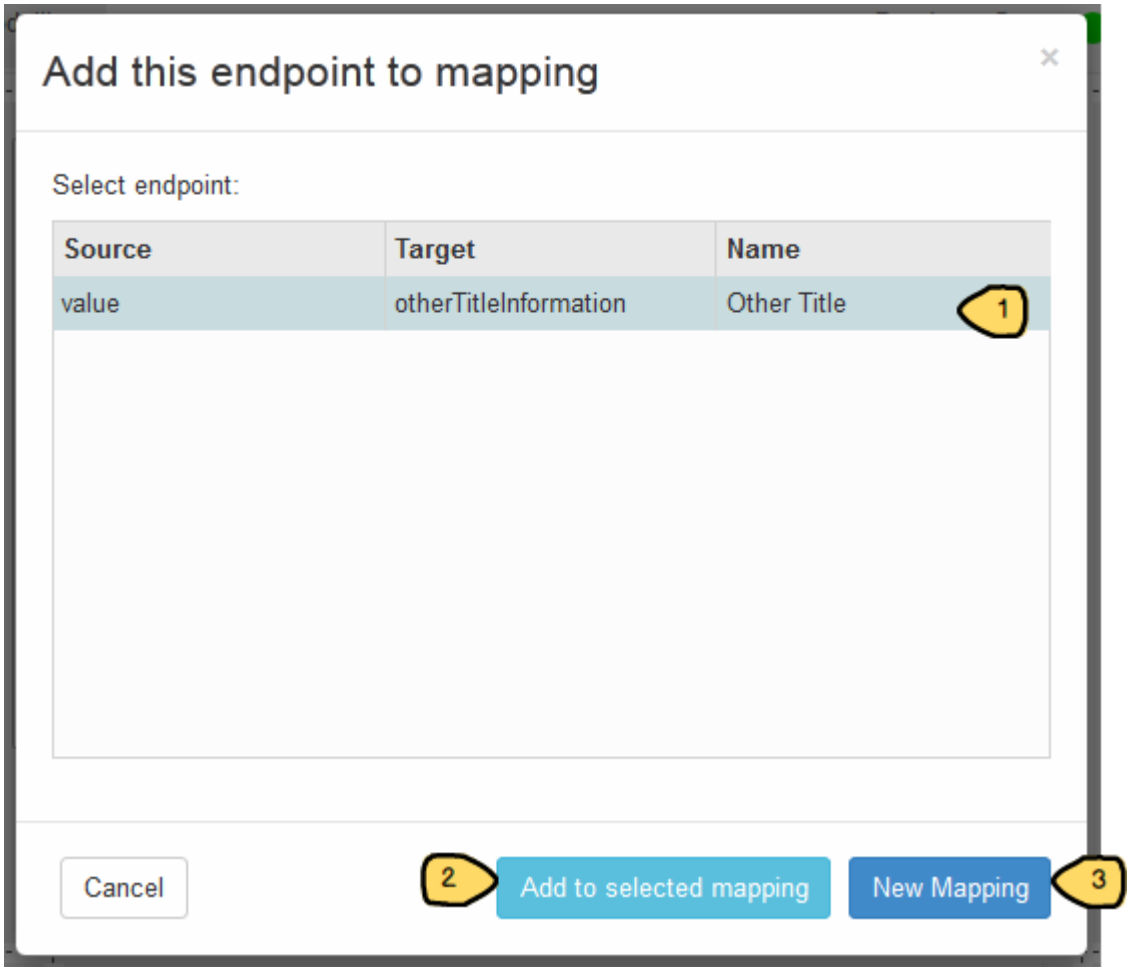
- Wählen Sie zunächst das gewünschte Output-Schema per Klick auf den Button "Select output schema" bzw. ändern Sie das Output-Schema nachträglich durch Klick auf das ⚙-Symbol rechts oben im Mapping-Bereich (1).
- Ein Mapping erfolgt per Klick auf ein ▢-Symbol innerhalb des Input-Schemas (2) und einem folgenden Klick auf das gleiche Symbol innerhalb des Output-Schemas (3).
- Geben Sie im nun angezeigten Dialog eine Bezeichnung für die vorzunehmende Verknüpfung ein (mindestens fünf Zeichen). Bestätigen Sie Diese durch einen Klick auf den Button "Name that mapping".
- Das erstellte Mapping wird mit einem roten Pfeil visualisiert (4).
- Wiederholen Sie diesen Schritt für alle zu definierenden Mappings.

> **Hinweis**
> Im Alpha-Release kann nur auf Attribute der obersten Ebene gemappt werden, d.h. im Beispiel oben ist ein Mapping etwa auf "contributor/givenName" momentan noch nicht möglich, wohl aber auf Attribute der obersten Ebene wie "title", "shortTitle".

Komplexe Mappings



Komplexe Mappings - wie die Zusammenführung von zwei Input-Attributpfaden auf einen Output-Attributpfad (beispielhafter Anwendungsfall: Vorname + Nachname) - kann folgendermaßen umgesetzt werden, indem Sie ein Mapping (1) auf einen bereits schon verwendeten Output-Attributpfad (2) erzeugen. Eine Lightbox öffnet sich:



- Wählen Sie eine bestehenden Output-Attributpfad durch Klick aus der Tabelle (1).
- Der Button "Add to selected mapping" wird eingeblendet (2).
- Mit einem Klick auf "Add to selected mapping" wird ein integriertes Mapping definiert.
- Sollen mehrere Input-Attributpfade **nicht** zu einen gemeinsamen Output-Attributpfad zusammengeführt werden, kann dies durch einen Klick auf "New Mapping" (3) erreicht werden. In der Konsequenz entstehen zwei voneinander unabhängige Mappings.

Schritte:

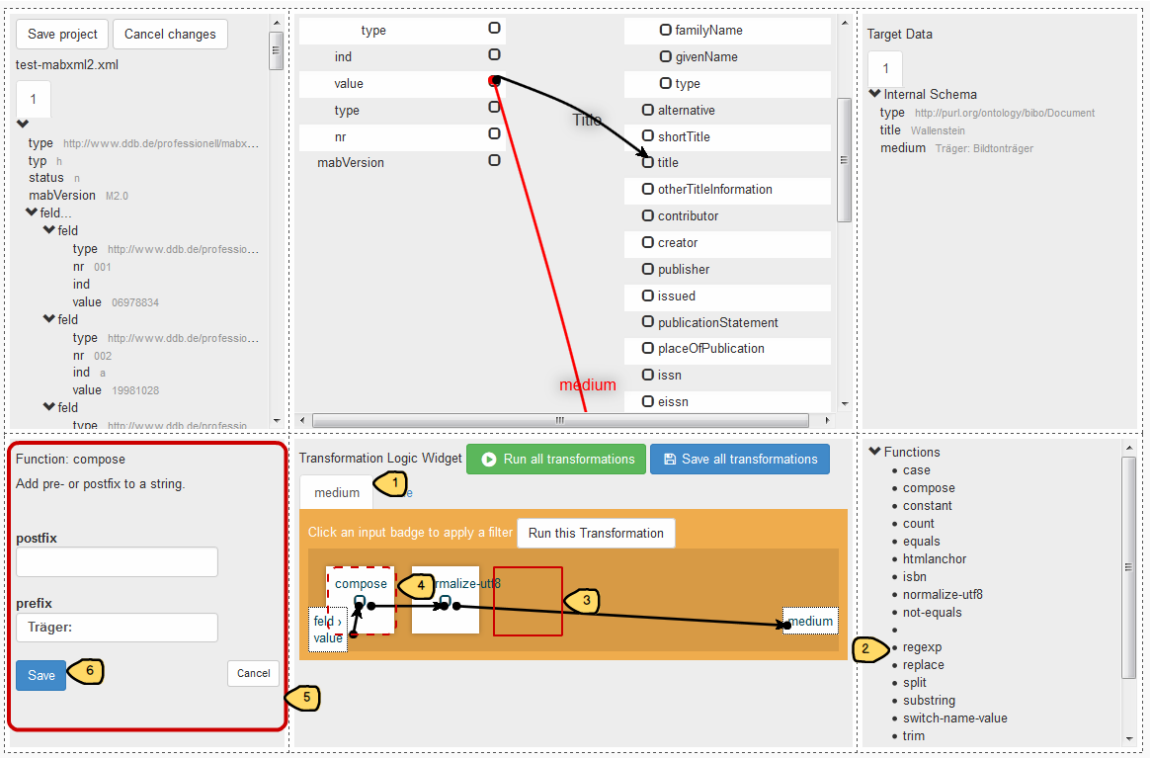
Projekte anlegen und bearbeiten	Transformationen designen und ausführen

Transformationen designen und ausführen

Datentransformationen können nur für existierende Mappings erzeugt werden. Für jedes Mapping innerhalb eines Projektes wird ein eigener Tab im unteren mittleren Bereich eingeblendet. Das Mapping, dessen Transformation gerade bearbeitet wird, ist im Mapping-Bereich jeweils rot markiert.

Design eines Transformations-Workflow

Einfache Transformationen



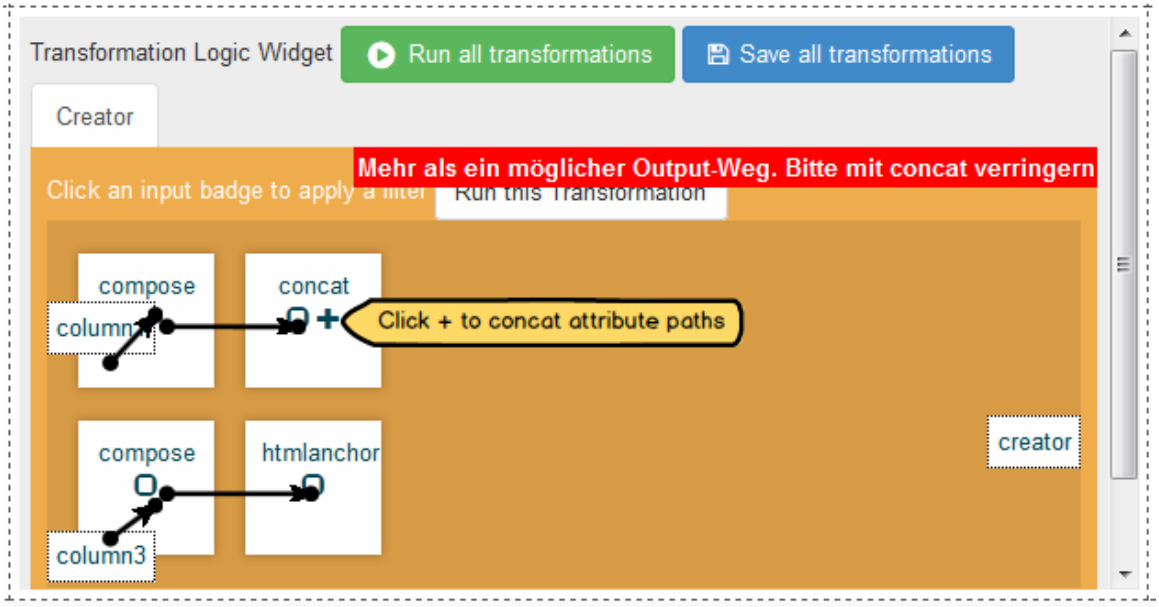
Eine Datentransformationslogik kann wie folgt modelliert werden :

- Wählen Sie das Mapping, dessen Transformation Sie designen wollen, durch Klick auf den entsprechenden Tab im Transforamtionsbereich (1).
- Anfangs besteht jedes Mapping aus einer leeren Transformation, in welcher Input- mit Output Attributpfade direkt miteinander verbunden sind
- Um eine Transformationslogik abzubilden, ziehen Sie mit der Maus eine Funktion (2) vom rechten unteren Funktions-Bereich auf den Transformations-Bereich und lassen sie Box an einer gewünschten Stelle im Grid fallen (3).
  - Eine Liste der im Alpha-Release verfügbaren Funktionen ist hier verfügbar.
  - Die Abfolge des Transformationslogik wird durch einen Pfeil visualisiert.
  - Ziehen Sie ggf. weitere Funktionen an die gewünschte Stelle im visualisierten Prozess.
  - Eine Funktion kann jederzeit innerhalb des Grids des Transformationsprozesses verschoben werden.
- Per Klick auf eine "Funktions-Box" innerhalb des Transformationsprozesses (4) wird eine Konfigurationsmöglichkeit unten links (5) zur Verfügung gestellt.
- Geben Sie die erforderlichen Parameter für die Funktion ein und speichern Sie die Konfiguration mit dem Button "Save" (6).
- Wiederholen Sie diese Schritte für alle definierten Mappings.

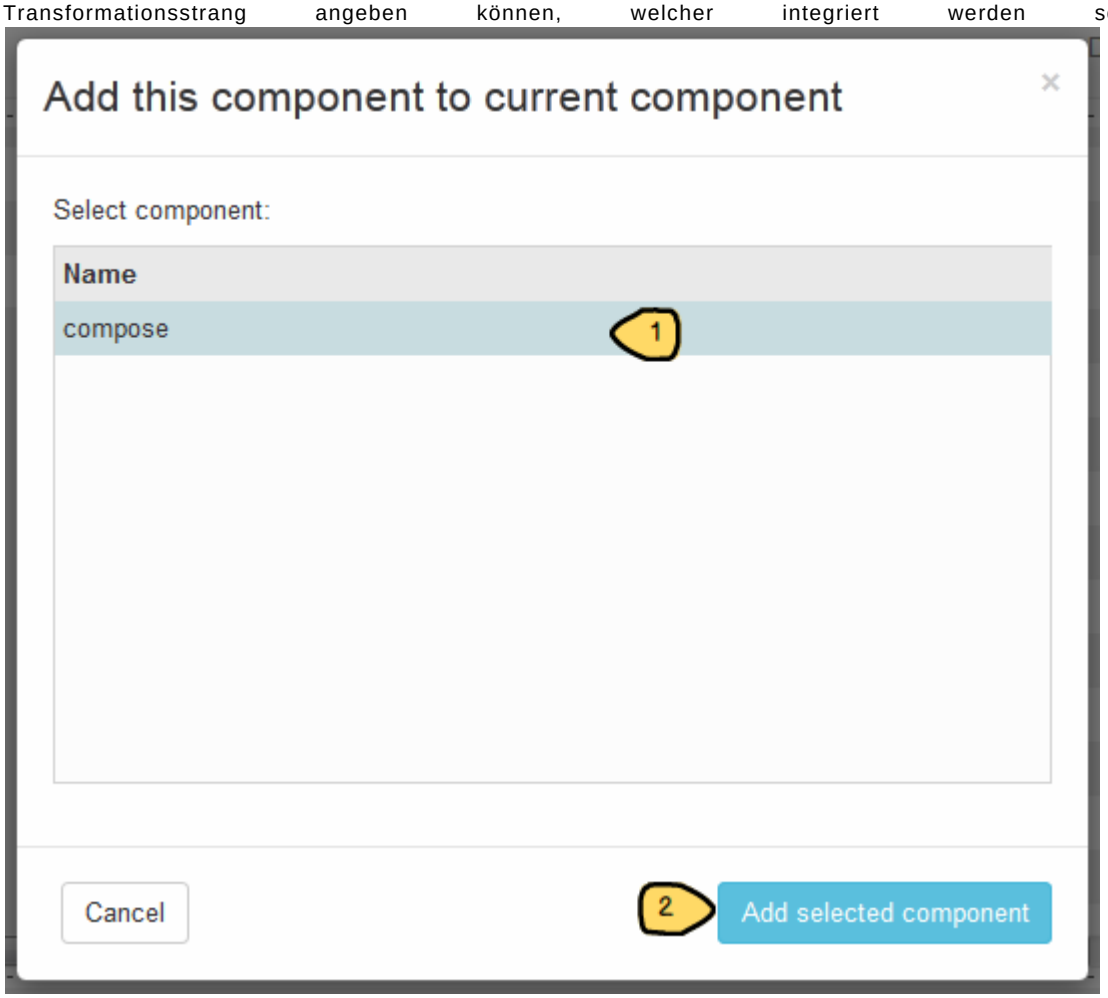
> **Hinweis**
> Innerhalb des Alpha-Releases besteht nicht die Möglichkeit, eine Funktion aus dem Transformationsprozesses zu löschen.

Komplexe Transformationen

Zusätzlich besteht innerhalb des Alpha-Releases bereits die Möglichkeit, eine komplexe Transformation - hier die Konkatenierung von zwei individuellen Input-Attributpfade auf einen Output-Attributpfad - vorzunehmen. Dabei stehen innerhalb des Transformationsprozesses zwei Quellen zur Verfügung, für die individuell Funktionen gemäß Business-Logik verwendet werden können. Beide individuellen Transformationsprozesse müssen durch die Konkatenierungs-Logik integriert werden:



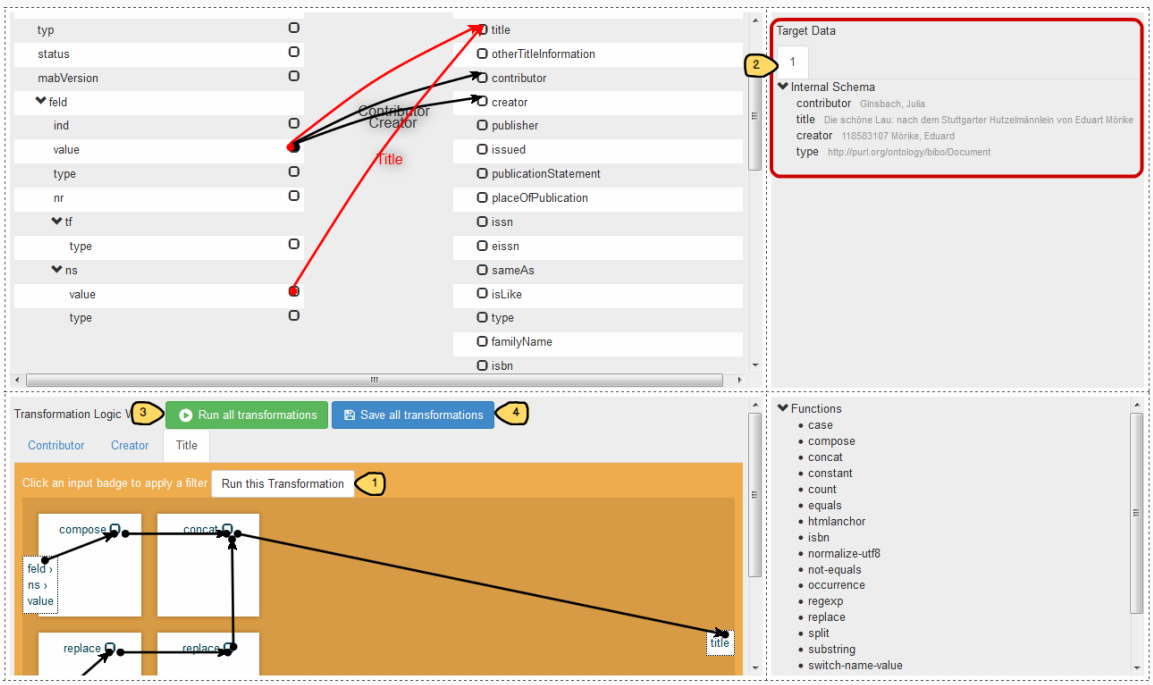
- Ziehen Sie hierfür die Concat-Funktion aus der Funktionsliste (unten rechts) in den benachbarten Transformationsprozess (unten Mitte).
- Durch Klick auf das "+"-Symbol innerhalb der Concat-Funktion öffnet sich ein Dialog, in dem Sie einen zweiten Transformationsstrang angeben können, welcher integriert werden soll:



- Wählen Sie die Komponente, die Sie zum aktuellen Attributpfad hinzufügen wollen, indem Sie sie aus der Liste der verfügbaren Komponenten auswählen (1).
- Klicken sie auf den Button "Add selected component", um den dialog zu verlassen un dide beiden Komponenten zu einem Output-Attribuitpfad zu vereinigen (2).

Transformationen ausführen

Jede Transformation ist jederzeit ausführbar.



- Ein Klick auf den Button "Run this transformation" (1) führt die Transformationslogik auf dem aktuell aktiven Mapping für die Beispieldatensätze aus und präsentiert die Ergebnisse oben rechts (2).
- Per Klick auf den Button "Run all Transformations" (3) werden für alle bereits existierenden Mappings die jeweiligen Transformationen innerhalb des Projektes ausgeführt.
- Durch den Klick auf "Save all transformations" (4) wird das Projekt, d.h. alle Mappings und Transformationen für die gesamte Daten-Ressource ausgeführt und die Ergebnisse werden im Datahub gespeichert. Der Zugriff auf die Ergebnisse kann über die Oberfläche der Datenbank wie in Graphdatenmodell beschrieben.

Schritte:

Mappings definieren	Verfügbare Transformationsfunktionen Filter definieren

SLUB
Wir führen Wissen.

Verfügbare Transformationsfunktionen

Für die Datentransformation verwendet d:swarm Metamorph-Funktionen. Weitere Informationen zu diesen Funktionen finden Sie unter https://github.com/culturegraph/metafacture-core/wiki/Metamorph-functions.

Transformation	Erläuterung	Parameter	Erläuterung	Beispiele
case	Im aktuell gemappten Wert werden alle Buchstaben zu Groß- bzw. Kleinbuchstaben transformiert. Zu mappender Wert: "SLUB Dresden"	upper	Minuskel wird zu Majuskel	SLUB DRESDEN
		lower	Majuskel wird zu Minuskel	slub dresden
		language	Locale	de (für Deutsch)
compose	Verknüpft den aktuell gemappten Wert mit einem Präfix oder Suffix. Zu mappender Wert: "swarm" prefix: "d:" Ergebnis: "d:swarm"	prefix	Präfix-String	d:
		postfix	Suffix-String	
concat	Fasst die Werte mehrerer Attribute in einem Element zusammen, ergänzt anschließend die optionalen Prefix- und Suffix-Strings und übergibt das Ergebnis an den Output. Beispiel: Zu mappender Wert 1: "SLUB" Zu mappender Wert 2: "Dresden" delimiter: "-" prefix: "Pre" postfix: "Post" Ergebnis: "PreSLUB-DresdenPost"	delimiter	Trennzeichen zwischen den konkatenierten Werten.	
		prefix	Prefix-String	
		postfix	Suffix-String	
constant	Ersetzt den aktuell gemappten Wert durch den fixen, parametrisierten String.	value	Ersetzungswert	
count	Gibt die Anzahl der Vorkommen eines Attributes am Output, z.B. Teilmengen bei der split-Funktion.	parameterlos		

equals	Vergleicht den aktuell gemappten Wert mit dem im Parameter angegebenen String und gibt im Falle der Übereinstimmung den Wert an den Output.	string	Vergleichswert	
htmlanchor	Erzeugt aus dem aktuell gemappten Wert einen HTML Verweis nach folgendem Muster: title Beispiel zu mappender Wert: "slub-dresden" Ergebnis (ohne "+"): Homepage SLUB Dresden	prefix	Präfix-String	http://www.
		postfix	Suffix-String	.de
		title	angezeigter Verweis	Homepage SLUB Dresden
isbn	Wandelt 10 und 13-Stellige ISBNs ineinander um. Nicht-Zahlzeichen, die der ISBN vorangestellt/nachgestellt sind, können aus dem gemappten Wert entfernt werden. Ebenfalls "." und "-" innerhalb der ISBN. Der Wert kann daraufhin validiert werden.	isbn13	10 nach 13-Stellig	
		isbn10	13 nach 10-Stellig	
		clean	Entfernung Nicht-Zahlzeichen	
		verifyCheckDigit	Validierung	
normalize-utf8	Wandelt den Unicode String des aktuell gemappten Wertes in eine Normalisierte Form.	parameterlos		
not-equals	Vergleicht den aktuell gemappten Wert mit dem parametrisierten String. Stimmen beide nicht überein, wird der gemappte Wert an den Output übergeben.	string		
occurence	Filtert den aktuell gemappten Wert in Abhängigkeit des Vorkommens am Output. Bei der split-Funktion kann	only	Position des Elementes	moreThen 2 3 lessThen

	somit ein Teilstring separiert werden. Zu mappende Werte (z.B. als Ergebnis von split): "SLUB" "Dresden" "d:swarm" "Datenmanagement" only: "moreThen 2" sameEntity: "True" Ergebnis: "d:swarm" "Datenmanagement"	sameEntity	Prüft auf das Vorkommen innerhalb der Entität	
regexp	Vergleicht den aktuell gemappten Wert mit einem parametrisierten Regulären Ausdruck, gibt im Falle des matches den Wert aus.	match	Regex Pattern	^isbn\d\d\-(\d{10,13})
		format	Reihenfolge der verwendeten Capturing Groups	${1}
replace	Ersetzt im aktuell gemappten Wert, den auf einen Regulären Ausdruck matchenden Teil durch einen parametrisierten Ersetzungsstring.	pattern	Regex Pattern	^isbn\d\d\-(\d{10,13})
		with	Ersetzungsstring	
split	Teilt den aktuell gemappten Wert an der auf den parametrisierten Regulären Ausdruck matchenden Stelle. Beispiel zu mappender Wert: "SLUB-Dresden" delimiter: "-" Es werden 2 Strings "SLUB" und "Dresden" als mapping zurückgegeben	delimiter	Regex Pattern	
substring	Extrahiert einen Teilstring aus dem aktuell gemappten Wert. Beispiel zu mappender Wert: "SLUB Dresden" start=0, end=7 gibt "SLUB Dr" zurück	start	Indexstelle des ersten Zeichens, beginnend bei 0.	
		end	Indexstelle des letzten Zeichens, exklusiv! D.h. es wird der Teilstring bis zu diesem Zeichen zurückgegeben.	

trim	Entfernt alle Leerzeichen am Anfang und Ende des aktuell gemappten Wertes. Entfernt auch Tabs und andere nicht-druckbare Zeichen.	parameterlos		
urlencode	Wandelt alle Zeichen des aktuell gemappten Wertes, die nicht in einer URL interpretiert werden können, in eine kompatible Zeichenfolge um.	parameterlos		

Schritte:

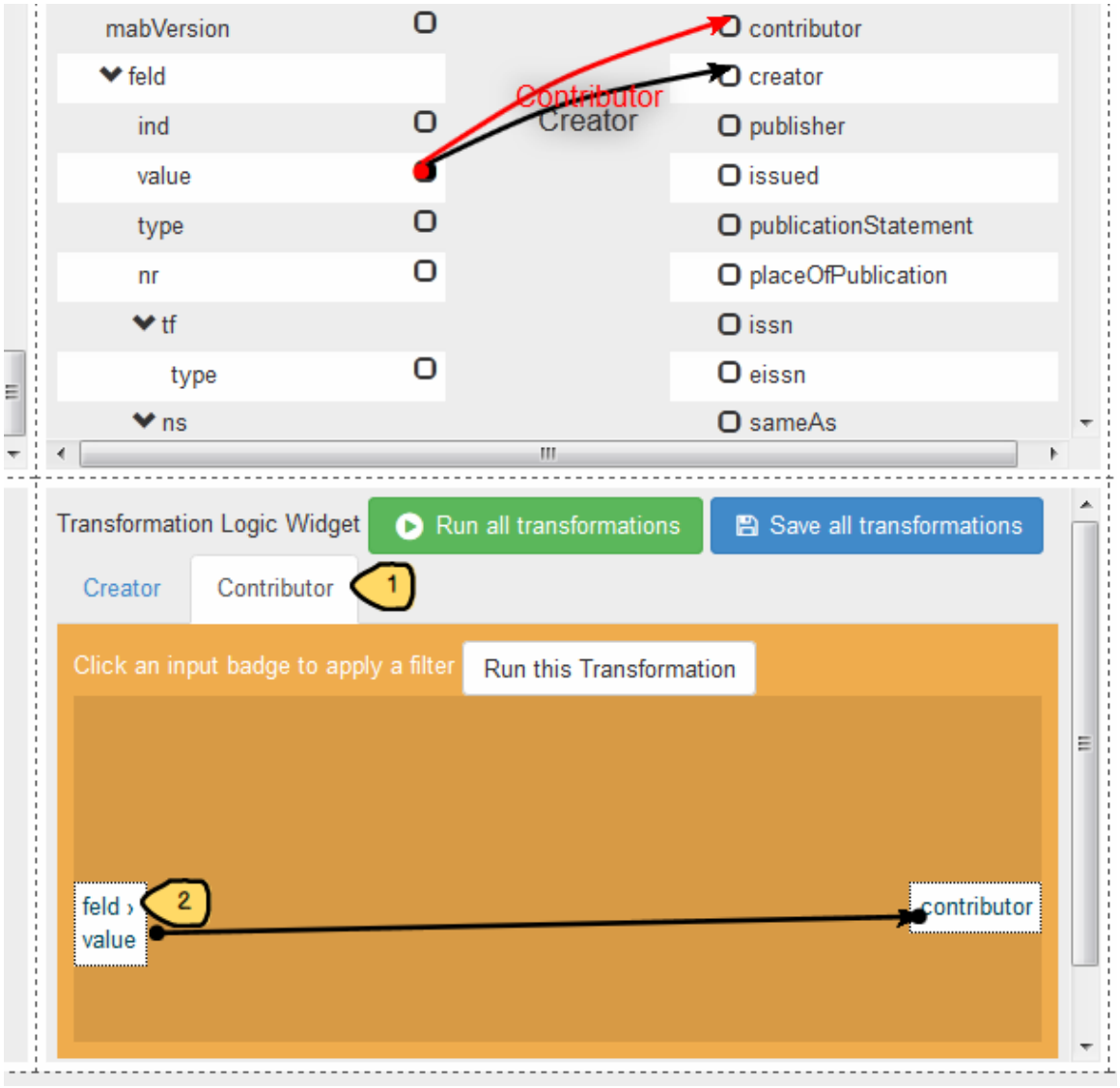
Transformationen designen und ausführen Filter definieren	Graphdatenmodell

Filter definieren

Filter können benutzt werden, um zu spezifizieren, unter welchen (komplexen) Bedingungen eine Transformation für ein Mapping angewandt werden soll.

Workflow der Filter-Definition

Voraussetzung für die Erstellung eines Filters ist das Vorhandensein eines Mappings, das im Transformations-Bereich im mittleren unteren Bereich ausgewählt sein muss.



- Um einen Filter zu erstellen, wählen Sie das Mapping aus, für das Sie einen Filter definieren wollen (1).
- Klicken Sie auf den (linken) Input-Attributpfad innerhalb der Transformations-Pipeline (2).

Es öffnet sich eine Lightbox:

- In der Lightbox befindet sich links ein Beispieldatensatz an Hand dessen die Filterregeln direkt überprüft werden.
- Zur Erstellung einer Filterregel klicken Sie oben links auf "add filter" (1). Es öffnet sich ein Filter-Formular in dem Sie eine Filterregel definieren können.
- Jede Eingabe (2) führt direkt zu einer Überprüfung ob der Filter auf den Beispieldatensatz angewandt werden kann. Treffer werden hierbei im Beispieldatensatz rot markiert (3).
- Durch erneuten Klick auf "add filter" können mehrere Filter miteinander kombiniert werden.
- Die einzelnen Regeln innerhalb eines Filters und filterübergreifend werden mit einem logischen UND verknüpft.
- Mit Klick auf "OK" wird der Filter übernommen (5).

Hinweis
Folgende Einschränkungen existieren im Alpha-Release:

- Filter sind nur auf der Ebene des Feldes anwendbar, auf das gemappt wurde. In unserem Beispiel können die Attribute *type*, *nr* und *ind* gemappt werden (alles direkte Attribute unterhalb des Attributpfades *feld*), nicht aber das Unterattribut *feld->ns->value* (4).
- Einzelne Filterregeln können geändert, aber nicht gelöscht werden. Hier hilft das Leeren des Filters.
- Der angezeigte Beispieldatensatz ist nicht selektierbar.

Schritte:

Transformationen designen und ausführen	Verfügbare Transformationsfunktionen

SLUB
Wir führen Wissen.

Graphdatenmodell

Ein Graphdatenmodell besteht grundsätzlich aus Knoten und Kanten. Aussagen werden innerhalb des Graphenmodells als gerichtete, binären Graphen repräsentiert - sie bestehen immer aus **Knoten** (Subjekt) - **Kante** (Prädikat) - **Knoten** (Objekt).

Knoten

Knoten stellen entweder

1. eine Ressource bzw. den Einstiegspunkt zu einer Ressource dar oder
2. sind Teil dieser Ressource (BNode), d.h. ein Unterelement einer hierarchischen Beschreibung einer Ressource - z.B. ein Feld eines MABXML-Datensatzes (Records) oder
3. können auch einfache Wert-Knoten, d.h., Literale sein.

Attribute an Knoten

Attribut	Definition
__NODETYPE__	Knoten haben immer einen Knotentyp (**__NODETYPE__**). Knotentypen dienen zur Orientierung im Graphdatenmodell und können folgende Ausprägungen haben: • __RESOURCE__ • __BNODE__ • __LITERAL__ • __TYPE_RESOURCE__ • __TYPE_BNODE__
__URI__	falls __NODETYPE__ == __RESOURCE__
__VALUE__	falls __NODETYPE__ == __LITERAL__
__DATATYPE__	falls __NODETYPE__ == __LITERAL__ und das Literal ein typisiertes Literal ist
__PROVENANCE__	falls __NODETYPE__ == __RESOURCE__ und der Knoten als Knotentyp nicht __TYPE_RESOURCE__ ist
__RESOURCE__	zum schnellen Erkennen zu welche Ressource dieser Knoten gehört (wenn es ein Literal- oder BNode-Knoten ist)

Beispielhafte Erklärung

Ein Record ist immer eine Ressource. Deshalb ist der Einstiegsknoten zu diesem Record immer vom Knotentyp **__RESOURCE__**. Ressourcen oder Teile von Ressourcen können typisiert sein. Diese Typen tauchen als Knotenbezeichner (Labels) im Graphen auf:

- ein MABXML-Record für sich ist beispielsweise vom Typ 'Datensatz' (`http://www.ddb.de/professionell/mabxml/mabxml-1.xsd#datensatzType`) und
- ein Feld innerhalb eines MABXML-Datensatzes ist vom Typ 'Feld' (`http://www.ddb.de/professionell/mabxml/mabxml-1.xsd#feldType`).

Als Bezeichner für Ressource-Identifier, Typen und Attribute (Prädikate) von Aussagen werden URIs verwendet. Alle Kanten und Ressource-Knoten haben ein Herkunftsattribute (**__PROVENANCE__**). Dieses verweist auf das Datenmodell, zu dem die Aussage oder die Ressource gehört. Ressource-Identifier hängen an Ressource-Knoten über das Attribut **__URI__**. Werte von Literalen hängen an Literal-Knoten über das Attribut **__VALUE__**.

Kanten

Kanten sind immer Attribute von Ressourcen oder Teil-Ressourcen - beispielsweise mabxml:nr (`http://www.ddb.de/professionell/mabxml/mabxml-1.xsd#nr`).

Attribute an Kanten

Attribut	Definition

__PROVENANCE__	
__RESOURCE__	zum schnellen Erkennen zu welche Ressource dieser Knoten gehört (z.B. wenn es ein Literal- oder BNode-Knoten ist)
__ORDER__	zum Identifizieren der "In-Elemente"-Ordnung von Werten einer bestimmten Eigenschaft, z.B. wird diese bei "mixed XML elements" genutzt
__INDEX__	zum Identifizieren eines ressource-weiten Statement-Index (d.h. der ursprünglichen "Elemente"-Ordnung, z.B., wie sie in einer XML-Datei vorkommen)

Schritte:

Filter definieren Verfügbare Transformationsfunktionen	Graphexploration

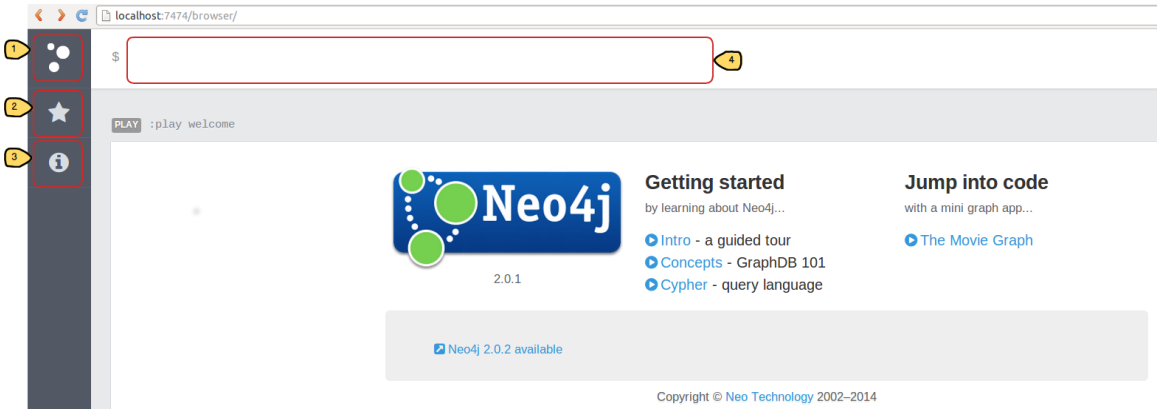
Graphexploration

Für den zentralen Datahub von d:swarm wurde die Graphdatenbanklösung von Neo4J gewählt. Mit Hilfe des webbasierten Neo4j-Graph-Browser werden die Inhalte des Datahubs graphisch explorierbar gemacht. Die Anwendung kann über folgende URL http://sdvdswarm01.slub-dresden.de/neo/ erreicht werden.

Neo4j-Graph-Browser

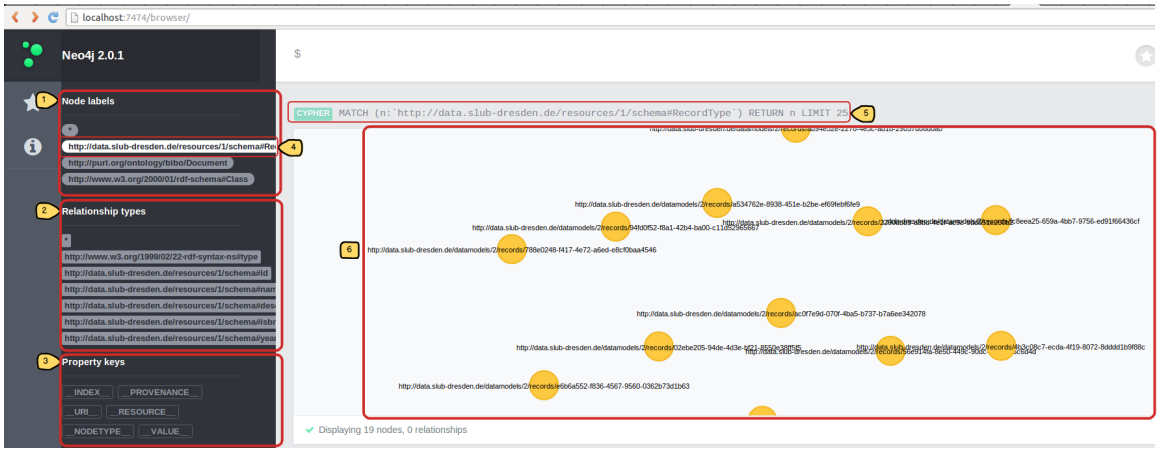
Übersicht

Folgende Grundfunktionalitäten sind von der Startoberfläche direkt erreichbar:



- (1) Übersicht der Knotenbezeichner (Node labels), Kantentypen (Relationship types) und Attribute (Property keys), die als Einstieg in den Graphen genutzt werden können.
- (2) Ermöglicht das Manipulieren des Graphen; enthält ein Einstiegsanfragen auf den Graphen und zeigt Details über die Graphdatenbankkonfiguration an.
- (3) Führt zur Dokumentation, Hilfe, Tutorials usw. über Neo4j.
- (4) Eingabemaske für Cypher-Anfragen. Eine Übersicht und Kurzanleitung zur Anwendung von Cypher befindet sich hier.

Graphexploration



- (1) Eine Übersicht der im Graphen vorkommenden Knotenbezeichner, d.h., Ressource-Typen (siehe Graphdatenmodell).
  - ***Hinweis**: Knotenbezeichner können nicht aus der Graphdatenbank gelöscht werden, d.h., auch Knotenbezeichner von bereits gelöschten Knoten tauchen immer noch in dieser Übersicht auf.*
- (2) Eine Übersicht der im Graphen vorkommenden Kantentypen, d.h. Attribute oder Prädikate.
  - ***Hinweis**: Kantentypen können nicht aus der Graphdatenbank gelöscht werden, d.h., auch Kantentypen von bereits gelöschten Kanten tauchen immer noch in dieser Übersicht auf.*
- (3) Eine Übersicht der im Graphen vorkommenden Attribute (an Knoten und Kanten).
- (4) Ein ausgewählter Knotenbezeichner (`http://data.slub-dresden.de/resources/1/schema#RecordType`). Hierdurch wird eine Cypher-Anfrage an den Graphen gesendet; das Ergebnis wird entweder visuell oder tabellarisch dargestellt.
- (5) Beispielhafte Cypher-Anfrage zum Auffinden von Knoten mit dem Knotenbezeichner '`http://data.slub-dresden.de/resources/1/schema#RecordType`' limitiert auf maximal 25 Ergebnisse:

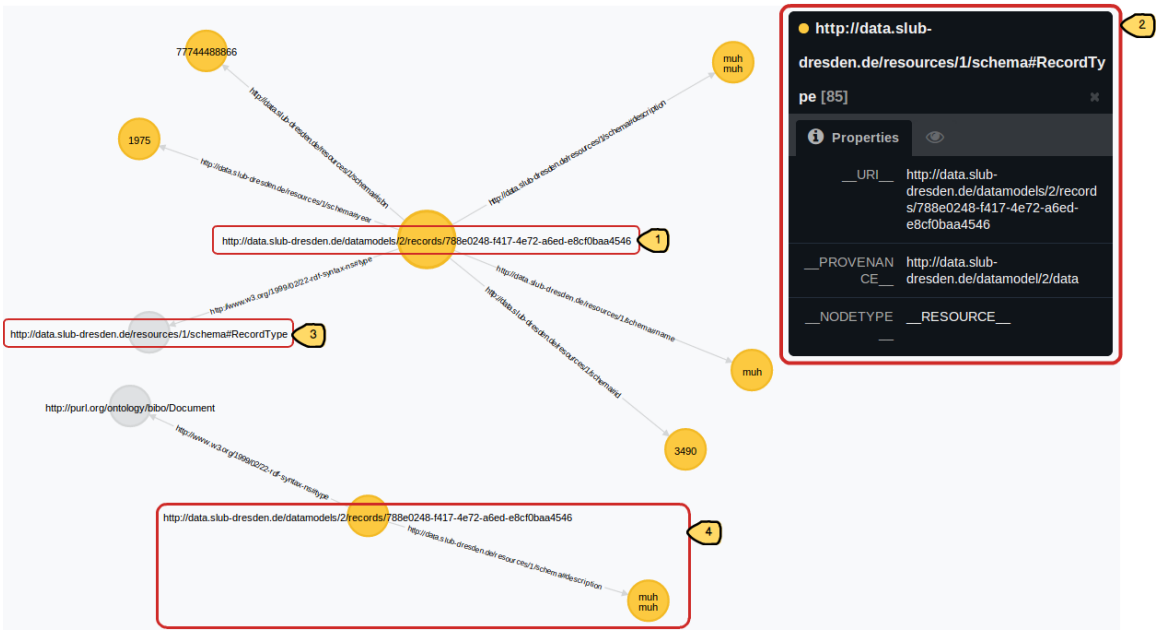
```
MATCH (n:`http://data.slub-dresden.de/resources/1/schema#RecordType`)
RETURN n LIMIT 25
```

- (6) Die visuelle Repräsentation der Ergebnismenge. Dargestellt werden Knoten mit ihrem Identifier (z.B. `http://data.slub-dresden.de/datamodels/2/records/788e0248-f417-4e72-a6ed-e8cf0baa4546`).
  - ***Hinweis**: Per Doppelklick auf einen Knoten werden alle eingehenden und ausgehenden Kanten (inkl. End- bzw. Anfangsknoten) dieses Knotens angezeigt.*

Beispiel

Beispiel eines aus einer CSV-Datei importierten Records inkl. Mapping kann mit folgender Cypher-Query abgefragt werden:

```
MATCH (n)-[r]->(m) WHERE n.__URI__ =
"http://data.slub-dresden.de/datamodels/2/records/788e0248-f417-4e72-a6ed-e
8cf0baa4546" RETURN n, r, m;
```



- (1) Der im Eingangsdatenmodell selektierte Record (Ressource-Knoten) `http://data.slub-dresden.de/datamodels/2/records/788e0248-f417-4e72-a6ed-e8cf0baa4546`
- (2) Die Detailansicht des selektierten Ressource-Knotens mit folgenden Knoten-Attributen:
  - __URI__ = `http://data.slub-dresden.de/datamodels/2/records/788e0248-f417-4e72-a6ed-e8cf0baa4546`
  - __PROVENANCE__ = `http://data.slub-dresden.de/datamodel/2/data`
  - __NODETYPE__ = __RESOURCE__
  - Der Knotenbezeichner (als Ressourcetyp): `http://data.slub-dresden.de/resources/1/schema#RecordType`
- (3) Der zugehörige Knoten des Ressourcetypen
  - ***Hinweis**: Ein Doppelklick auf einen Ressourcetyp-Knoten sollte alle zugehörigen Ressource-Knoten anzeigen, welche mit diesem Typen kategorisiert sind*
- (4) Graphische Visualisierung einer Aussage (Knoten - Kante - Knoten).
  - Im Beispiel ist es der Ressource-Knoten des Records `http://data.slub-dresden.de/datamodels/2/records/788e0248-f417-4e72-a6ed-e8cf0baa45` im Ausgangsdatenmodell mit dem Attribut `http://data.slub-dresden.de/resources/1/schema#description` und dem Wert (Literal) "muh muh".

Hinweis
Wenn eine Kante selektiert wird, dann erscheint auch eine Detailansicht mit allen Kanten-Attributen und dem Kantentyp (Attribut/Prädikat der Aussage).

Hilfreiche Cypher-Anfragen

Diese Übersicht soll nur einen kleinen Einblick in Cypher-Anfragen geben. Für weitere Möglichkeiten der Cypher-Anfragesprache ist die Cypher-Referenz-Karte zu empfehlen.

- Ermittlung aller Aussagen (rekursiv), die zu Records vom Ressourcetypen '`http://www.openarchives.org/OAI/2.0/recordType`' gehören, d.h. "zeige mir die (ggf. hierarchisch strukturieren) Records dieses Ressourcentypen im Graphen an" :

```
MATCH (n:`http://www.openarchives.org/OAI/2.0/recordType`)-[r*]->(m)
RETURN n, r, m;
```

- Ermittlung aller Aussagen, die das folgenden Attribute haben: '`http://data.slub-dresden.de/resources/1/schema#description`'

```
MATCH
(n)-[r:`http://data.slub-dresden.de/resources/1/schema#description`]->
(m) RETURN n, r, m;
```

- Ermittlung alle verwendeten Ressourcetypen im Graphen, d.h. "Selektiere alle Ressourcetyp-Knoten, die den Ressourcetyp rdfs:Class haben" :

```
MATCH (n:`http://www.w3.org/2000/01/rdf-schema#Class`) RETURN n;
```

Hinweis
Im Graphexplorations-Tab werden auch alle Ressourcetypen des Graphen angezeigt. Der (mögliche) Unterschied zu dieser Cypher-Anfrage besteht darin, dass hier auch wirklich nur die aktuell im Graphen vorkommenden Ressourcetypen angezeigt werden (wohingegen bei den Knotenbezeichnern alle jemals im Graphen vorgekommenen Ressourcetypen angezeigt werden).

Verwendung von Indices

Neben den (built-in) Indices für Knotenbezeichner und Kantentypen empfiehlt sich auch die Verwendung von weiteren Indices (welche im Moment als Neo4j-Legacy-Indices umgesetzt wurden und sich deshalb von Schema-Indices unterscheiden), um performante Aussführungszeiten für die Anfragen zu ermöglichen.

- Ermittlung aller Aussagen, die an Knoten mit dem Knoten-Identifier 'http://data.slub-dresden.de/datamodels/2/records/788e0248-f417-4e72-a6ed-e8cf0baa4546' hängen, d.h., wo das Subjekt diesen Identifier hat:

```
START
n=node:resources(__URI__="http://data.slub-dresden.de/datamodels/2/rec
ords/788e0248-f417-4e72-a6ed-e8cf0baa4546") MATCH (n)-[r]->(m) RETURN
n, r, m;
```

Vorhandene Indices

Derzeitig existieren nur Indices für Knoten. Es können aber auch Indices für Kanten angelegt werden. Knoten-Indices haben als Prefix immer 'node'. Folgende Indices werden derzeitig gepflegt:

- **resources**: enthält alle Ressource-Identifier
  - Index-Attribut: __**URI**__
- **resources_w_provenance**: enthält alle Ressource-Identifier + Provenance-Identifier, z.B. '`http://data.slub-dresden.de/datamodels/2/records/788e0248-f417-4e72-a6ed-e8cf0baa4546 http://data.slub-dresden.de/datamodel/2/data`', dabei wird der Ressource-Identifier und Provenance-Identifier durch ein Leerzeichen im Key getrennt
  - Index-Attribut: __**URI_W_PROVENANCE**__

    - ***Hinweis**: Dieser Index ist nützlich um effizient auf Ressourcen einer bestimmten Provenienz zuzugreifen.*
- **values**: enthält alle Werte von Literalen
    - Index-Attribut: __**VALUE**__

> **Hinweis**
> Index-Attribute stehen nicht in unmittelbaren Beziehung zu Knoten- oder Kanten-Attributen, auch wenn sie manchmal den selben Bezeichner haben.

Schritte:

Graphdatenmodell	Daten in RDF exportieren

Daten in RDF exportieren

Die Daten im Datahub können im N-Quads-Format in eine Datei exportiert werden.

Export-Workflow

Die Funktion zum Export der in d:swarm erzeugten Daten befindet sich auf dem Tab "Data".

- Klicken Sie Im Bereich "Export Data" auf den Dropdown-Button "Export All Data".
  - ***Hinweis**: Im Alpha-Release steht Ihnen der Export in das N-Quads-Format zur Verfügung, das in späteren Releases durch weitere Formate ergänzt wird.*
- Per Klick auf "N-Quads" öffnet sich der Browser-Speicher-Dialog, in dem Sie einen Speicherpfad und Namen der Datei wählen können.

Hinweis

- Im Alpha-Release werden alle im Datahub befindlichen Daten exportiert. Ein Exportgenerator (siehe Roadmap für d:swarm) für eine spezifische Auswahl der zu extrahierenden Daten wird in zukünftigen Releases verfügbar sein.
- URI-Verwendung in den exportierten RDF-Daten: Im Alpha-Release werden für transformierte Resourcen teilweise noch keine neuen URIs generiert. Für den Fall, dass die Quell-Datensätze bereits eine URI besitzen, wird diese daher momentan im RDF-Export wieder erscheinen.

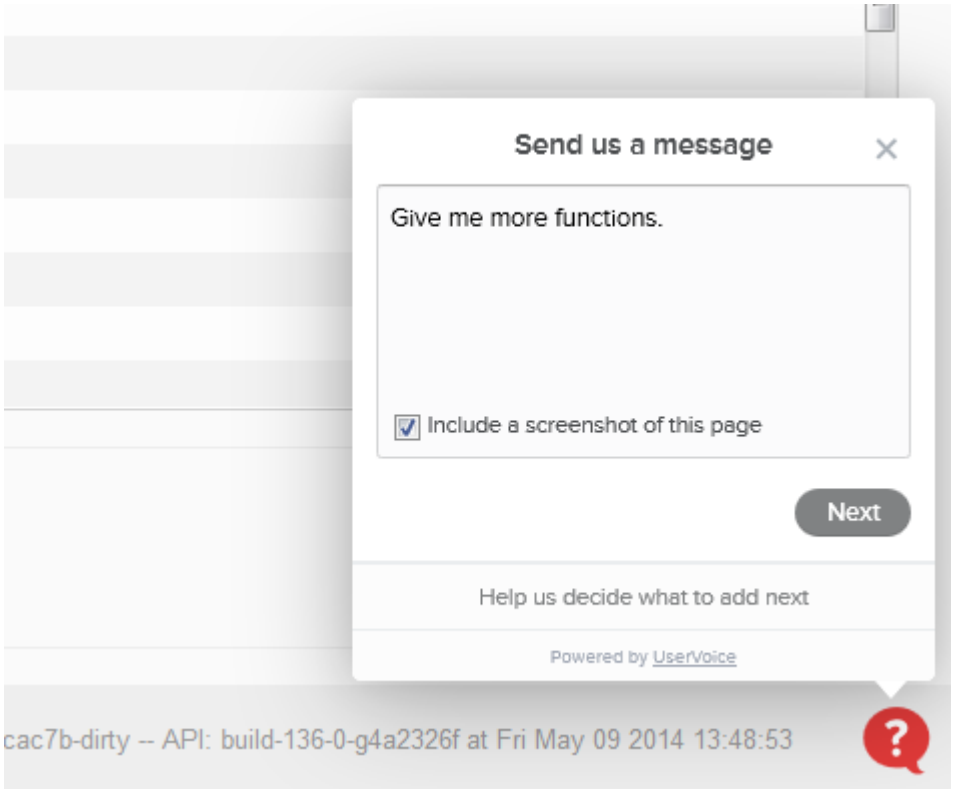
Schritte:

Graphexploration	Feedback an die Entwickler geben

SLUB
Wir führen Wissen.

Feedback an die Entwickler geben

Bitte testen Sie die Alpha-Version ausgiebig und senden Sie uns Feedback. Klicken Sie dazu auf das Fragezeichen rechts am Seitenfuß und teilen Sie uns Ihre Eindrücke mit. Sie können einen Screenshot der aktuellen Seite an die Nachricht anhängen.



Wenn Sie Fragen zu Funktionen von d:swarm haben, geben Sie in die Feedback-Box entsprechende Stichwörter ein. Es werden Ihnen nach Klick auf den Button "Next" passende Hilfethemen aus unserer Wissensdatenbank und Vorschläge zu neuen Funktionen anderer Tester angezeigt (in Englisch). Bitte teilen Sie uns unter dswarm.uservoice.com (Feedback) Ihre Ideen mit. Sie können dort auch die Ideen anderer Tester bewerten und so die weitere Entwicklung mitbestimmen.

Schritte:

Daten in RDF exportieren	Roadmap für d:swarm

SLUB
Wir führen Wissen.

Roadmap für d:swarm

Folgende Roadmap gilt zum 28. Mai 2014 für die Weiterentwicklung von d:swarm. Nach den Prinzipien der agilen Entwicklung "lebt" diese Roadmap und richtet sich nach den Anforderungen der Pilotanwender.

Zeitraum	Feature-Delivery
alpha release 28.5. Features: MAB2-Graph, Graph-Visualisierung, RDF-Export	
Juni	Update-Implementierung: neu importierte Datenressourcen passen sich in die vorhandene Datenstruktur im Graphen ein und Veränderungen in der Datenbasis sind nachvollziehbar (Versionierung)
	Komplexe Transformationen II: Erweiterung des verfügbaren Transformationsfunktionsset in GUI ist auf alle verfügbaren Metamorph-Funktionen ergänzt (inkl. Ausführung auf den Graphen)
	Modellierung von Deduplizierungsregeln in GUI möglich
	Ausführung von modellierten Deduplizierungsregeln ist möglich --> Duplikate werden als solche im Graphen identifiziert
Juli	Modellierung von Regeln für das Handling von Metadaten duplikater Records möglich
	Regeln für das Handling von Metadaten duplikater Records können ausgeführt werden und manipulieren den bestehenden Graphen
beta release 15.7. Features: integrierter Graph, Deduplizerung, Usability	
	Komplexe Transformationen III: Wiederverwendung von Transformationen in Transformationen sind in GUI modellier- und ausführbar
August	Alpha Version einer Execution Engine ist verfügbar, auf der die Tasks ausgeführt werden können
	Admin Oberfläche in GUI verfügbar
September	Tasks können aus Admin Oberfläche auf der Execution Engine deployed werden
	Fehlerreports bei der Ausführung von Prozessen in Admin Oberfläche verfügbar
gamma release 30.9. Features: Execution Engine und Admin-Oberfläche	
Oktober	OAI-PMH kann als Datenquelle registriert und konfiguriert werden --> Daten können geharvestet werden
	HTTP APIs können registriert werden
	SLUBsemantics als Anreicherungsservice als WebService zur verfügbar
November	individueller Export-Generator verfügbar
	Schema Editor zur Modellierung des internen Graph-Datenmodells verfügbar
delta release 15.12. Features: weitere Quellsysteme, Einbindung SLUBsemantics, Exportgenerator, Schema-Editor	
... kontinuierliche Weiterentwicklung insbesondere im Bereich Rollen- und Rechtemanagement, FRBR, Sharing, Monitoring, etc	

Schritte:

Feedback an die Entwickler geben	d:swarm - Hilfe zum Alpha-Release