

Allgemeine Beschreibung

Die von *OpenRefine* exportierten TXT-Dokumente bedürfen noch einer kleinen Anpassung für die Benutzung als Register in *ediarum*. Die vorliegende Transition zielt daher darauf ab, redundante Informationen im *OpenRefine*-Output zu reduzieren, um mit sinnvoll komprimierten Daten in *ediarum* weiterarbeiten zu können.

In *OpenRefine* lässt sich der Export über die Templating-Option bereits sehr gut an die Registerstruktur von *ediarum* anpassen. Bei Dopplungen im Datensatz bedarf es aber mitunter noch kleineren Anpassungen, die über ein kurzes XSLT vorgenommen werden können.

Voraussetzungen

Die im DigEdTnT-Projekt vorgestellten Transitions setzen nicht nur bestimmte Kompetenzen der Benutzer:innen voraus, sondern stellen auch hinsichtlich der Software-Umgebung gewisse Anforderungen.

Erforderliche Kenntnisse

- [EDV-Grundkenntnisse](#)
- Einrichten einer [Oxygen-Transformation](#)
- Grundlegende XSLT-Kenntnisse (für erweiterte Anpassungen)

Benötigte Software

- Oxygen Editor

Möglichkeiten & Grenzen

Der Übergang von einem Tool zu einem anderen lässt sich verschieden gestalten. Nachfolgend soll ein Überblick über die Vor- und Nachteile unserer Transition von *OpenRefine* zu *ediarum* gegeben werden.

Stärken

- Eliminierung von Redundanzen des *OpenRefine*-Outputs

Herausforderungen & Probleme

- Keine

XSL Transformation

In unserem Beispielprojekt sind wir mit folgender Situation konfrontiert: Der Export von *OpenRefine* enthält aufgrund der zeilenbasierten Struktur mehrere Listenelemente (`<item>`) mit der gleichen `@xml:id` und zudem redundante Informationen in Bezug auf die Q-Nummern der Wikidata-Einträge sowie hinsichtlich der deutschen Übersetzungen:

```

<list>
<item xml:id="verjuice">
  <idno type="uri">https://www.wikidata.org/entity/Q1060458</idno>
  <label type="reg">Agraz</label>
  <label type="alt">agraeß</label>
</item>
<item xml:id="verjuice">
  <idno type="uri">https://www.wikidata.org/entity/Q1060458</idno>
  <label type="reg">Agraz</label>
  <label type="alt">agras</label>
</item>
<item xml:id="verjuice">
  <idno type="uri">https://www.wikidata.org/entity/Q1060458</idno>
  <label type="reg">Agraz</label>
  <label type="alt">agraz</label>
</item>
...
<item xml:id="apple">
  <idno type="uri">https://www.wikidata.org/entity/Q89</idno>
  <label type="reg">Apfel</label>
  <label type="alt">apfel</label>
</item>
<item xml:id="apple">
  <idno type="uri">https://www.wikidata.org/entity/Q89</idno>
  <label type="reg">Apfel</label>
  <label type="alt">äpfelen</label>
</item>
...

```

Um die `<item>`-Elemente mit der gleichen `@xml:id`, die außerdem auch die gleiche QID und die gleiche deutsche Übersetzung im `<label type="reg">`-Element enthalten, in einem `<item>`-Element mit den verschiedenen frühneuhochdeutschen Varianten mit mehreren `<label type="alt">`-Elementen zusammenzuführen, transformieren wir die Daten mittels XSLT.

Die Transformation ist in unserem Workflow folgend eingebettet:

- Der in *OpenRefine* mittels Template in eine XML-Struktur überführte Datensatz wird beim Export als [Plaintext-Datei](#) zum Download bereitgestellt. Daher kopieren wir zunächst die Liste und fügen diese in das vorerst leere [Register](#) ein, das wir in *ediarum* für unser Projekt erstellt haben. Dafür wechseln wir in die Text-Ansicht und kopieren den Inhalt der TXT-Datei in den vorerst noch leeren `<body>`.

```

1 <TEI xmlns="http://www.tei-c.org/ns/1.0" xmlns:telota="http://www.telota.de"
2   telota:doctype="register">
3   <teiHeader>
4     <fileDesc>
5       <titleStm>
6         <title>Sachbegriffe</title>
7       </titleStm>
8       <publicationStm>
9         <p>Gehört zur ediarum-App für eXist.</p>
10      </publicationStm>
11      <sourceDesc>
12        <p>Hybrid: consult project description and document history for details.</p>
13      </sourceDesc>
14    </fileDesc>
15  </teiHeader>
16  <text>
17    <body>
18      <list>
19        <item xml:id="verjuice">
20          <idno type="uri">https://www.wikidata.org/entity/Q1060458</idno>
21          <label type="reg">Agraz</label>
22          <label type="alt">agraeß</label>
23        </item>
24        <item xml:id="verjuice">
25          <idno type="uri">https://www.wikidata.org/entity/Q1060458</idno>
26          <label type="reg">Agraz</label>
27          <label type="alt">agras</label>
28        </item>
29        <item xml:id="verjuice">
30          <idno type="uri">https://www.wikidata.org/entity/Q1060458</idno>

```

ID "verjuice" has already been defined

→ Wie wir sehen können, ist das Dokument nicht valide, da die englische Übersetzung, die wir als `@xml:id` nutzen, mehrfach vorhanden ist, diese aber eindeutig sein muss.

- Im nächsten Schritt wenden wir ein Transformationsszenario ([siehe Kurzanleitung](#)) mit dem für die Listenadaptierung erstellten [XSL-Dokument](#) auf unser Register (Sachbegriffe.xml) an.
- Unser transformiertes Register enthält nun zu jedem standarddeutschen Begriff alle frühneuhochdeutschen Varianten. In der Autor-Ansicht sieht dies folgendermaßen aus:

```

https://www.wikidata.org/entity/Q1060458
Agraz
agraeß
agras
agraz
agres
agresß
agrest
agreste
agroße

https://www.wikidata.org/entity/Q89
Apfel
apfel
äpfelen
apfell
apffel
äpfel
anffell

```

- Nun müssen wir in *ediarum* nur mehr das Register-XML austauschen oder den Inhalt der transformierten Datei in das alte Register übernehmen, und können dann mit der [Annotation der Zutaten](#) anhand unseres Datensatzes fortfahren.

XSLT Dokumentation

Die Transformation zur Bereinigung des *OpenRefine*-Outputs umfasst folgende Schritte:

- **Gruppierung der `<item>`-Elemente:** Da alle frühneuhochdeutschen Varianten die gleiche englische Übersetzung haben und diese daher als `@xml:id` gewählt wurde, gruppieren wir alle Listeneinträge mit der gleichen `@xml:id`.
- **Inhalte der gruppierten `<item>`-Elemente:**
 - **`<idno>`-Element:** Jeder Listen- bzw. Registereintrag, der im *OpenRefine* die gleiche englische Übersetzung hat, besitzt auch die gleiche Q-Nummer. Daher kopieren wir bei unserer Transformation nur das `<idno>`-Element des ersten Elements aus unserer Gruppierung.
 - **`<label>`-Elemente:** Bei den `<label>`-Elementen haben wir einerseits jene, die ein "reg" als Wert im `@type`-Attribut haben und die standarddeutsche Übersetzung beinhalten, sowie andererseits jene mit "alt", welche die frühneuhochdeutsche Schreibung beinhalten. Da ersteres ebenfalls bei allen Listeneinträgen gleich ist, nehmen wir auch hier nur das erste Element aus dem gruppierten Set. Von den alternativen `<label>`-Elementen übernehmen wir nun aber das gesamte Gruppenset, da wir innerhalb von einem Listeneintrag alle vorhandenen frühneuhochdeutschen Schreibweisen zusammensammeln möchten.
- **Übernahme der Metadaten:** Die Metadaten des Sachregisters möchten wir nicht verändern, weshalb wir hier kein eigenes Template für die Elemente im `<teiHeader>` anlegen.