

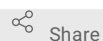


Jun 05, 2021

Parsing of NLM metadata in OpenRefine from OJS articles

Alessandra Moi¹, [carlo.bianchini](#)², [Andrea Marchitelli](#)¹¹Associazione Italiana Biblioteche (AIB); ²Università di Pavia

1 Works for me



Share

dx.doi.org/10.17504/protocols.io.bp9tmr6n

wikidata and libraries

Andrea Marchitelli

ABSTRACT

Protocols for parsing metadata of articles harvested from Open Journal Systems (OJS) in order to create entities in WikiData

DOI

dx.doi.org/10.17504/protocols.io.bp9tmr6n

PROTOCOL CITATION

Alessandra Moi, [carlo.bianchini](#), [Andrea Marchitelli](#) 2021. Parsing of NLM metadata in OpenRefine from OJS articles. **protocols.io**
<https://dx.doi.org/10.17504/protocols.io.bp9tmr6n>

LICENSE

This is an open access protocol distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

CREATED

Dec 01, 2020

LAST MODIFIED

Jun 05, 2021

PROTOCOL INTEGER ID

45075

MATERIALS TEXT


OpenRefine [↗](#)

[source](#) by Freebase, then Google, now open source community

Attività preliminari

- 1 Import the OAI identifiers and generate the baseurl list:

1.1 Import the identifiers following the instructions in the protocol



Come importare identificatori OAI-PMH in OpenRefine
by **Andrea Marchitelli**

[PREVIEW](#)

[RUN](#)

1.1.1 Aprire OpenRefine e scegliere "Crea progetto"

OpenRefine [↗](#)

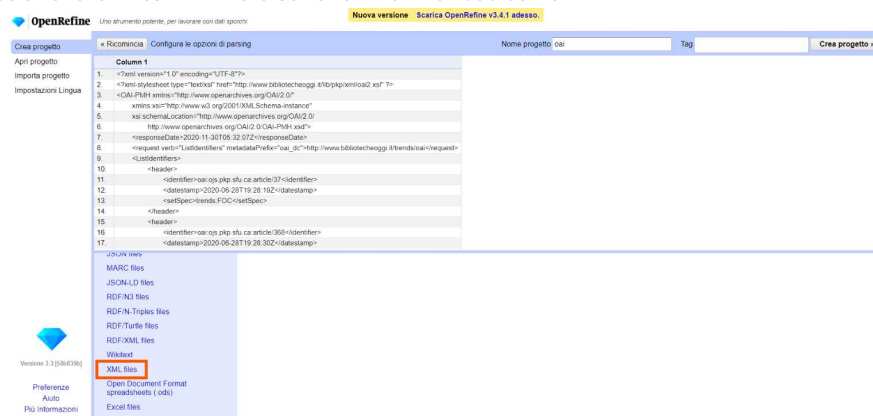
[source](#) by Freebase, then Google, now open source community

1.1.2 Selezionare "Indirizzo/i Web (URLs)" come fonte dei dati del nuovo progetto e cliccare su "Avanti". Inserire la baseurl OAI-PMH del target individuato, scegliere il verbo **ListIdentifiers** e il metadataPrefix opportuno. Es. http://www.bibliotecheoggi.it/trends/oai?verb=ListIdentifiers&metadataPrefix=oai_dc

Il protocollo OAI-PMH consente l'impostazione di limitatori per data e per set per esporre dataset parziali e permettere raccolte incrementali.

Es. <http://an.oa.org/OAI-script?verb=ListIdentifiers&from=1998-01-15&metadataPrefix=oldArXiv&set=physics:hep>

1.1.3 Selezionare "Files XML" nella sezione "Analizza i dati come"



Screenshot delle opzioni di import in OpenRefine

1.1.4 Nell'area dell'anteprima del file da importare, individuare la prima riga contenente l'elemento <identifier> e portare il puntatore del mouse a evidenziarla. Poi cliccarvi sopra.

Evidenziare l'identifier

1.1.5

A quel punto si sarà ottenuta la lista degli identificatori OAI-PMH, uno per riga, pronta per elaborazioni successive.

- From the list of imported identifiers, generate the list of baseurl for harvesting the metadata of each item according to the indications given in the protocol "[Creare baseurl per harvesting di repository OAI con OpenRefine](#)"

N.B. For NLM metadata collection, the metadataPrefix in the url must be =nlm

2

OpenRefine will create a column containing XML text with the NLM metadata of the article

Test the operation of one of the generated URLs by clicking on the link and verifying that it opens a page with NLM metadata in XML format.

Es. <https://aibstudi.aib.it/oai?verb=GetRecord&metadataPrefix=nlm&identifier=oai:ajs.riviste.aib.it:article/11366>

Harvesting dei metadati

- Click on the metadataURL column , select Edit Column > Add Column by retrieving from URLs, calling the new column metadata_oai.
Once you have the metadata value for all items, you can move on to parsing

Parsing dei dati

- Metadata Parsing

4.1 Published in (P1433)

```
value.parseXml().select('article')[0].select('journal-meta')[0].select('journal-title')[0].ownText()
```

4.2 DOI (P356)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('article-id[pub-id-type="doi"]')[0].ownText()
```

4.3 Title (P1476)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('article-title')[0].ownText()
```

4.4 Authors name and related info

Autore#1 (P50)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[0].select('given-names')[0].ownText() + ' ' + value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[0].select('surname')[0].ownText()
```

First nameAutore#1 (P735)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[0].select('given-names')[0].ownText()
```

Surname Autore#1 (P734)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[0].select('surname')[0].ownText()
```

Affiliation Autore#1 (P1416)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[0].select('aff')[0].ownText()
```

Autore#2 (P50)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[1].select('given-names')[0].ownText() + ' ' + value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[1].select('surname')[0].ownText()
```

First nameAutore#2 (P735)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[1].select('given-names')[0].ownText()
```

Surname Autore# 2 (P734)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[1].select('surname')[0].ownText()
```

Affiliation Autore#2 (P1416)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[1].select('aff')[0].ownText()
```

Autore#3 (P50)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[2].select('given-names')[0].ownText() + ' ' + value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')[2].select('surname')[0].ownText()
```

First nameAutore#3 (P735)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')
[2].select('given-names')[0].ownText()
```

Surname Autore# 3 (P734)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')
[2].select('surname')[0].ownText()
```

Affiliation Autore# 3 (P1416)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('contrib[contrib-type="author"]')
[2].select('aff')[0].ownText()
```

4.5 Publication date (P577)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('pub-date[pub-type="epub"]')
[0].select('day')[0].ownText() + '/' + value.parseXml().select('article')[0].select('article-meta')
[0].select('pub-date[pub-type="epub"]')[0].select('month')[0].ownText() + '/' +
value.parseXml().select('article')[0].select('article-meta')[0].select('pub-date[pub-type="epub"]')
[0].select('year')[0].ownText()
```

4.6 Volume & Issue

volume (P478)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('volume')[0].ownText()
```

Fascicolo (P433)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('issue')[0].ownText()
```

4.7 Link

URL (P2699)

```
value.parseXml().select('article')[0].select('article-meta')[0].select('self-uri')[0].xmlAttr("xlink:href")
```

PDF Full text (P953)

```
forEach(value.parseXml().select('article')[0].select('article-meta')[0].select('self-uri[content-
type="application/pdf"]'),v,v.xmlAttr("xlink:href")).join("|")
```

HTML Full text (P953)

```
forEach(value.parseXml().select('article')[0].select('article-meta')[0].select('self-uri[content-
type="text/html"]'),v,v.xmlAttr("xlink:href")).join("|")
```

4.8 Keyword, per language.

Once you have the values in a single cell, divide using the command Edit Column > Split and choosing the separator |

KW ita

```
forEach(value.parseXml().select('article')[0].select('article-meta')[0].select('kwd-group[xml:lang="IT"]')
[0].select('kwd'),v,v.ownText()).join("|")
```

KW eng

argomento principale (P921)

```
forEach(value.parseXml().select('article')[0].select('article-meta')[0].select('kwd-group[xml:lang="EN"]')
[0].select('kwd'),v,v.ownText()).join("|")
```

4.9 Pages from/to and Number of pages.

Si devono rifare i passaggi 2 e 3, ma utilizzando come formato **oai_dc**. e creare una nuova colonna "Metadati_oai_dc".

Per esempio,

[https://aibstudi.aib.it/oai?](https://aibstudi.aib.it/oai?verb=GetRecord&metadataPrefix=oai_dc&identifier=oai:ojis.riviste.aib.it:article/11366)

[verb=GetRecord&metadataPrefix=oai_dc&identifier=oai:ojis.riviste.aib.it:article/11366](https://aibstudi.aib.it/oai?verb=GetRecord&metadataPrefix=oai_dc&identifier=oai:ojis.riviste.aib.it:article/11366)

Pages from/to (P304)

From the metadata_oai_dc column create a new DC_Source column (where source is the metadata that contains the value of the start and end page of the article. Command grel:
`value.parseXml().select('record')[0].select('metadata')[0].select('oai_dc|dc')[0].select('dc|source')[0].ownText()`

To extract only the page value from-to, create two columns with the split command and the appropriate separator (e.g.: ';')

Remove columns that do not have page numbers and rename the column with the correct value "PagDaA (P304)" (e.g.: 3-21)

Numero di pagine totale (P1104)

From the column "PagDaA (P304)" derive two more columns (PagIniz and PagFin) with the value of the starting page and the value of the ending page.

Create a new column "NumTotPag (P1104)" based on the PagFin column with the grel command:
`toNumber(value-cells['PagIniz'].value) + 1`

5 Sections

Extract name of Sections, e.g. to separate reviews from other articles

`value.parseXml().select('article')[0].select('article-meta')[0].select('subject')[0].ownText()`