

# Technologien

Ein Überblick über die wesentlichen Schnittstellen-Technologien

- [International Image Interoperability Framework \(IIIF\)](#)
- [Open Archives Initiative Protocol for Metadata Harvesting \(OAI-PMH\)](#)
- [Search/Retrieve via URL \(SRU\)](#)

# International Image Interoperability Framework (IIIF)

Informationen zum IIIF-Standard finden sich auf der Website des IIIF Consortium: <https://iiif.io>. Von besonderem Interesse ist die Dokumentation der [Image API](#) und [Presentation API](#).

Anmerkung: Die Plattformen e-rara.ch und e-manuscripta.ch setzen momentan Version 2.1 (Image API) und Version 2.0 (Presentation API) ein.

# Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)

Eine gute Einführung in OAI-PMH findet sich hier: <https://libtechlaunchpad.com/2017/02/13/oai-pmh-basics-and-resources>.

Offizielle OAI-PMH-Dokumentation: <https://www.openarchives.org/OAI/openarchivesprotocol.html>.

In der Response steht jeweils ein responseDate und das Request-Verb. Beispiel:

```
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/ http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
  <responseDate>2021-09-09T09:40:41Z</responseDate>
  <request verb="ListMetadataFormats">https://www.e-manuscripta.ch/oai/</request>
  ...
</OAI-PMH>
```

## Verben

- GetRecord – Used to retrieve an individual metadata record.
- Identify – Used to retrieve repository information (ex. name, version).
- ListIdentifiers – Used to retrieve only headers.
- ListMetadataFormats – Used to retrieve the available metadata formats.
- ListRecords – Used to retrieve actual item metadata records.
- ListSets – Used to retrieve the set structure of a repository

## Resumption Token

Die OAI-Schnittstellen liefern jeweils maximal eine bestimmte Anzahl Records zurück (bei uns in der Regel 10).

Der Tag `resumptionToken` enthält ein Attribut `completeListSize`, welches jeweils die gesamte Anzahl Treffer enthält. Diese Zahl ist nicht immer ganz exakt, sie kann sich während der Abfrage in Tranchen noch ändern.

Bsp:

```
<resumptionToken completeListSize="1279">0x726ca01fec751a64b6398913ea5ffe03-  
cursor_p_3D10_p_26set_p_3Dmaps_p_26metadataPrefix_p_3Doai_dc_p_26batch_size_p_3D11</resumptionToken>
```

Das Token selber enthält die Information, welche 10er-Tranche der Trefferliste als nächstes folgt.

Bsp. Für die Tranche ab Treffer Nr. 30:

```
<resumptionToken completeListSize="64">0x9bcb692a53b1f20609c8c4083fe1049a-  
cursor_p_3D30_p_26set_p_3Dzuz_p_26from__p_3D2021-09-  
15T00_p_253A00_p_253A00Z_p_26metadataPrefix_p_3Doai_dc_p_26batch_size_p_3D11_p_26until_p_3D2021-09-  
22T23_p_253A59_p_253A59Z</resumptionToken>
```

Für die Programmiersprache Python stehen die Libraries [Polymatheia](#) und [Sickle](#) zur Verfügung, die das Harvesten von Daten via OAI-PMH vereinfachen.

# Search/Retrieve via URL (SRU)

SRU steht für Search/Retrieve via URL. Über SRU-Schnittstellen können gezielt bestimmte Indizes im jeweiligen System nach bestimmten Begriffen durchsucht werden.

Einen guten Einstieg in SRU bietet [Wikipedia](#). Details zum SRU-Standard liefert die offizielle [Dokumentation](#) der Library of Congress.

## Explain-Operation

Mit der Explain-Operation werden Informationen zu Abfragemöglichkeiten angezeigt:

```
<Basis-URL>?operation=explain
```

Beispiel:

[https://uzb.swisscovery.slsp.ch/view/sru/41SLSP\\_UZB?version=1.2&operation=explain](https://uzb.swisscovery.slsp.ch/view/sru/41SLSP_UZB?version=1.2&operation=explain)

Unter dem XML-Tag `<indexInfo>` werden die Namen der durchsuchbaren Indices gelistet. Pro Index wird ausgewiesen, welche sogenannten *Relation Operators* erlaubt sind. Beispiel:

```
<index>
  <ns:title>dc:title</ns:title>
  <map>
    <name set="alma">dc_title</name>
  </map>
  <configInfo>
    <supports type="relation">all</supports>
    <supports type="relation">=</supports>
    <supports type="relation">==</supports>
    <supports type="emptyTerm"/>
  </configInfo>
</index>
```

Unter dem Tag <schemaInfo> sind die verfügbaren Metadatenformate aufgelistet. Aus dem untenstehenden Beispiel geht hervor, dass die betreffende SRU-Schnittstelle z.B. mit recordSchema=dc abgefragt werden kann, um Metadaten im Dublin Core Format zurückzubekommen.

```
<schemaInfo>
  <schema identifier="http://www.loc.gov/standards/marcxml/schema/MARC21slim.xsd" name="marcxml"
sort="true"/>
  <schema identifier="info:srw/schema/1/dc-v1.1" name="dc" sort="true"/>
  <schema identifier="info:srw/schema/1/mods-v3.5" name="mods" sort="true"/>
  <schema identifier="info:srw/schema/1/dcx-v1.0" name="dcx" sort="true"/>
  <schema identifier="info:srw/schema/8/unimarcxml-v0.1" name="unimarcxml" sort="true"/>
  <schema identifier="http://www.nl.go.kr/kormarc/" name="kormarcxml" sort="true"/>
  <schema identifier="http://www.nlc.cn/" name="cnmarcxml" sort="true"/>
  <schema identifier="http://www.loc.gov/standards/iso20775/" name="isohold" sort="true"/>
</schemaInfo>
```

## searchRetrieve-Operation

Für die eigentliche Suchabfrage wird die searchRetrieve-Operation genutzt:

```
<Basis-URL>?operation=searchRetrieve
```

## Contextual Query Language (CQL)

Suchanfragen werden in der [Contextual Query Language](#) (CQL) formuliert. Mithilfe von Booleschen Operatoren (and, or, not) können komplexe Abfragen erstellt werden.

Spezielle Zeichen wie Leerschläge und Anführungszeichen müssen URL-encoded werden. Beispiele:

```
%20 Leerschlag
%22 Anführungszeichen
%3D Gleichheitszeichen
%2F Forward slash
%3A Doppelpunkt
%2E Punkt
%2D minus
```

Für das URL-Encoding kann z.B. auch dieses Online-Tool genutzt werden: <https://urlencoder.org>.

## Durch Resultatseiten blättern

Bei einer Query wird jeweils nur eine Teilmenge der Treffer direkt zurückgeliefert. Dies ist serverseitig definiert - oft sind es 10 Records, sozusagen die erste Resultatseite. Die absolute Anzahl Treffer kann aus dem Tag `<numberOfRecords>` herausgelesen werden. Beispiel:

```
<numberOfRecords>629</numberOfRecords>
```

Im Element `<nextRecordPosition>` kann überprüft werden, welches Record als nächstes kommt. Mit dieser Zahl kann die nächste Resultatseite aufgerufen werden, indem der `startRecord`-Parameter an die vorangegangene Suche angehängt wird. Beispiel:

[https://uzb.swisscovery.slsp.ch/view/sru/41SLSP\\_UZB?version=1.2&operation=searchRetrieve&recordSchema=dc&query=alma.title=python&startRecord=11](https://uzb.swisscovery.slsp.ch/view/sru/41SLSP_UZB?version=1.2&operation=searchRetrieve&recordSchema=dc&query=alma.title=python&startRecord=11)