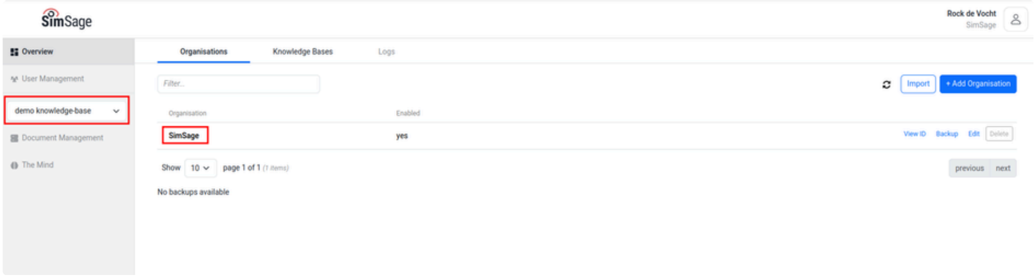
XML Crawler Documentation

Introduction

This Document sets out the steps need to step up an internal XML crawler on the SimSage platform.

SimSage Source configuration

1. Select your organisation (SimSage shown in the image below)
2. Select your knowledge-base from the drop-down box (demo knowledge-base shown below)

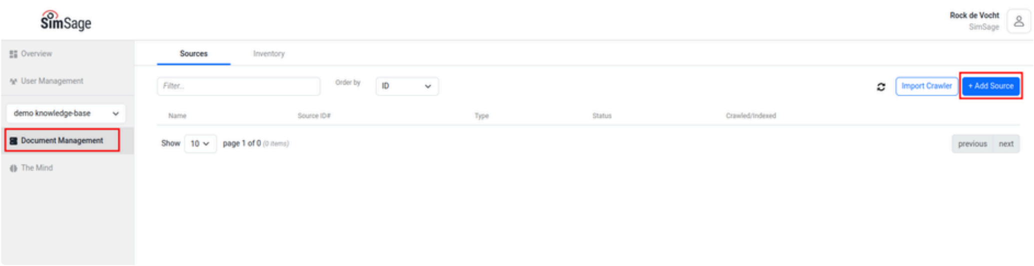


Organisation	Enabled
SimSage	yes

Show 10 page 1 of 1 *(1 items)*

No backups available

3. Click on Document Management link in the left hand side of the menu
4. Click on the + Add Source button



Name	Source ID#	Type	Status	Crawled/Indexed

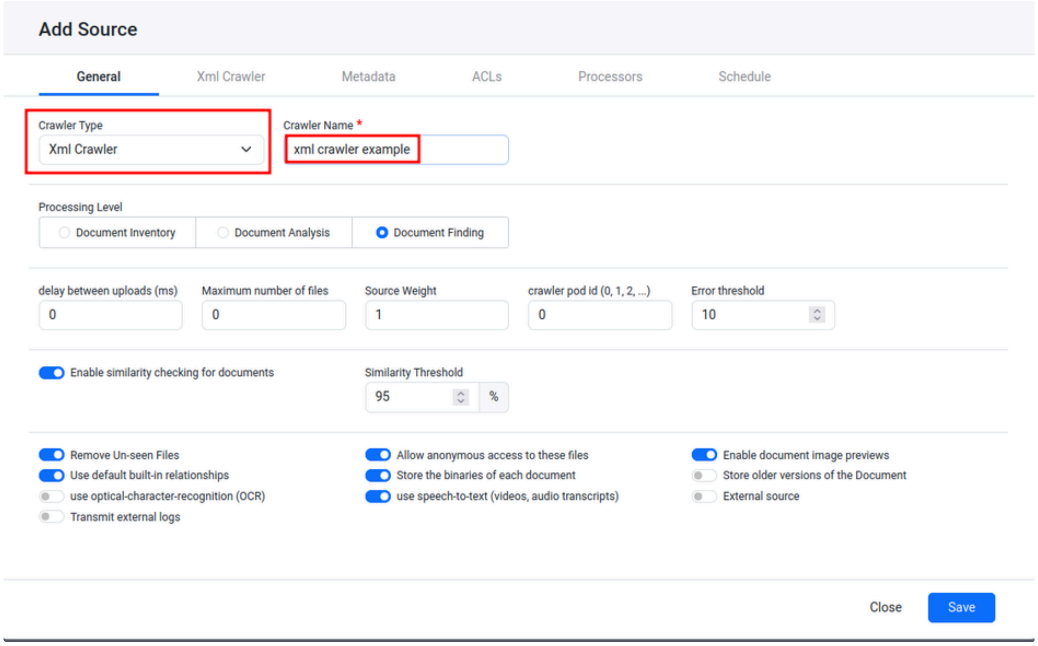
Show 10 page 1 of 0 *(0 items)*

5. In the General tab

a. set the Crawler Type to “XML Crawler”

b. give the Crawler a Name (*xml crawler example* in the image below)

c. Select the “Xml Crawler” tab to set up your XML crawler



Add Source

General | Xml Crawler | Metadata | ACLs | Processors | Schedule

Crawler Type: Xml Crawler

Crawler Name *: xml crawler example

Processing Level: ○ Document Inventory ○ Document Analysis ● Document Finding

delay between uploads (ms)	Maximum number of files	Source Weight	crawler pod id (0, 1, 2, ...)	Error threshold
0	0	1	0	10

☑ Enable similarity checking for documents

Similarity Threshold: 95 %

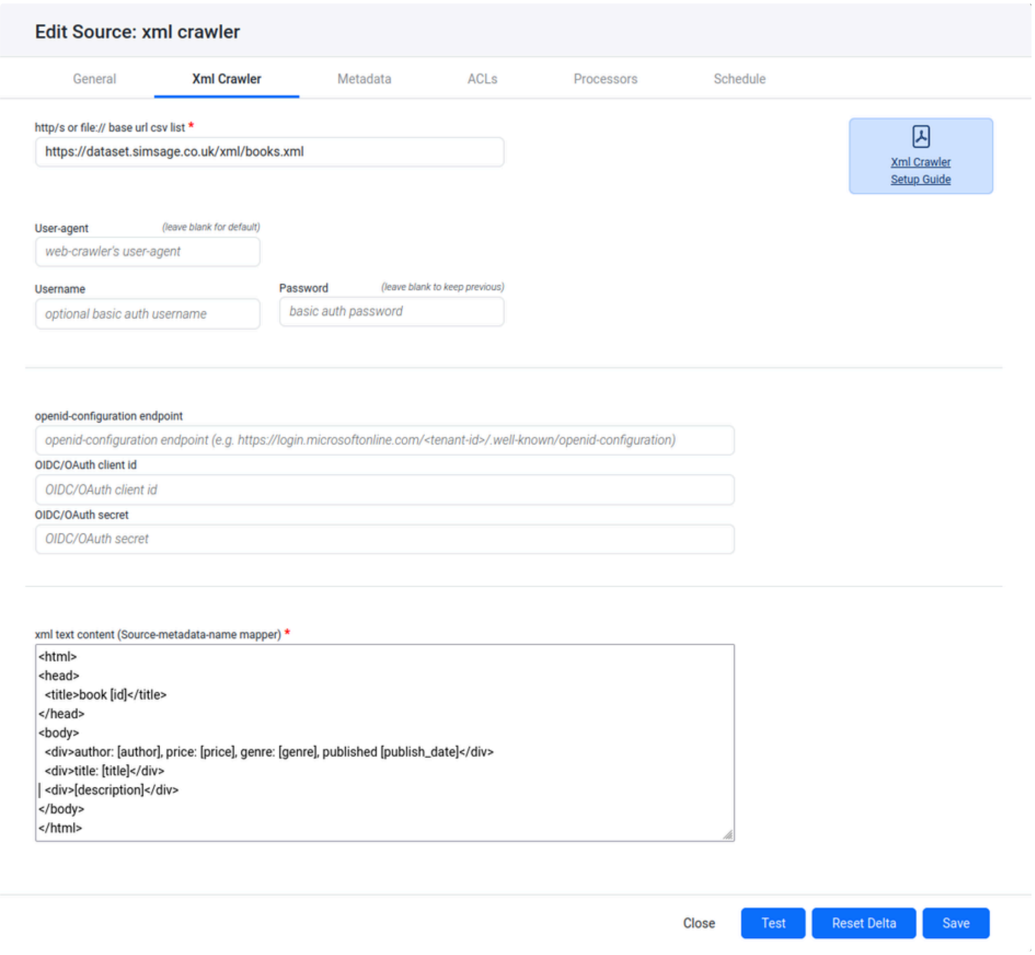
☑ Remove Un-seen Files
☑ Use default built-in relationships
☐ use optical-character-recognition (OCR)
☐ Transmit external logs
☑ Allow anonymous access to these files
☑ Store the binaries of each document
☑ use speech-to-text (videos, audio transcripts)
☑ Enable document image previews
☐ Store older versions of the Document
☐ External source

Close | Save

The XML Crawler Tab

1. Navigate to the “Web Crawler” tab.
2. Populate the base url field.
3. Populate the exclude css csv field. As a general rule we recommend using “header, footer” to exclude any repeating headers or footers from your HTML.

Your base url can be an ordinary html page, an xml site-map page, or the actual xml file you want to process. The XML crawler can traverse complex structures with or without authentication. However, the xml crawler itself will only process xml files it can map. As a minimum, each SimSage record requires a *body* and a *url*. See Mapping XML section below.



Edit Source: xml crawler

http/s or file:// base url csv list *
https://dataset.simsage.co.uk/xml/books.xml

User-agent (leave blank for default)
web-crawler's user-agent

Username
optional basic auth username

Password (leave blank to keep previous)
basic auth password

openid-configuration endpoint
openid-configuration endpoint (e.g. https://login.microsoftonline.com/<tenant-id>/.well-known/openid-configuration)

OIDC/OAuth client id
OIDC/OAuth client id

OIDC/OAuth secret
OIDC/OAuth secret

xml text content (Source-metadata-name mapper) *

```
<html>
<head>
  <title>book [id]</title>
</head>
<body>
  <div>author: [author], price: [price], genre: [genre], published [publish_date]</div>
  <div>title: [title]</div>
  <div>[description]</div>
</body>
</html>
```

field	meaning and content
http/s or file:// base url	a compulsory field with a comma separated list of http:// https:// or file:// URLs / URIs to visit. This set forms the seed set for the initial crawler.
User-agent	a string for the user-agent to pass to remote sites in your request headers. Leave empty for default. Default is “Apache-HttpClient/UNAVAILABLE”
Username	the BasicAuth username to send if set along with password in the request headers.
Password	the BasicAuth password to send in the request headers if set.

Extra security headers for Bearer Authentication

field	meaning and content
openid configuration endpoint	Used for OpenID authentication. The crawler will use this endpoint to get the OpenID configuration required for fetching JWT tokens for Bearer authentication. (e.g. https://server.com/.well-known/openid-configuration)

OIDC/OAuth client id	Used for OpenID authentication. This is the client ID passed to the identification endpoint for acquiring a JWT token.
OIDC/OAuth secret	Used for OpenID autentication. This is the client secret passed to the identification endpoint for acquiring a JWT token.

The xml text content field

This is a very special field that tells SimSage how to map the different parts of the XML record into a searchable text record. The content of this field can be either text or HTML.

The contents of this field can be as simple as a single reference to a metadata item. Every item referenced must be enclosed in [ ] (square brackets) without spaces. Use [description] if there is a “Source metadata field” called description set up in the metadata mapper.

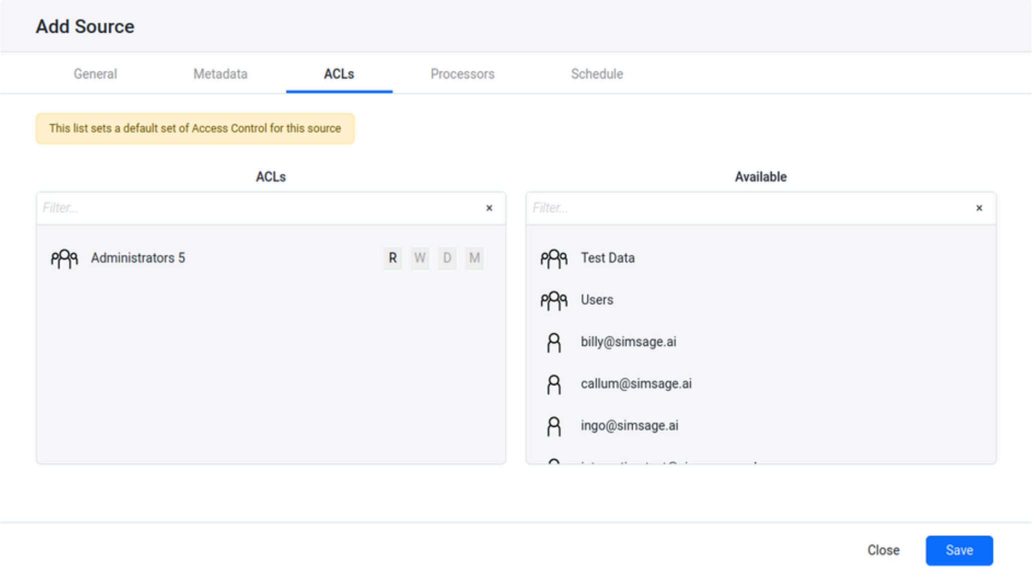
The example screenshot above shows an HTML structure being mapped to various metadata fields. All these metadata fields must exist in the mapper. If a field has no value in the XML the item is replaced with an empty string in this content field.

The ACLs Tab

In this tab we can customise the security based on existing SimSage Users and Groups.

NB: Users and Groups are automatically imported from external systems as they are crawled, where available.

In the below screenshot example we have selected the SimSage default User group as having access. Note, if you have opted to use the “allow anonymous access to these files” in the General source configuration, these ACLs settings will be overridden. *SimSage always recommends setting up a group initially in case you change your mind at any point.*



This list sets a default set of Access Control for this source

ACLs

Administrators 5 R W D M

Available

Test Data

Users

billy@simsage.ai

callum@simsage.ai

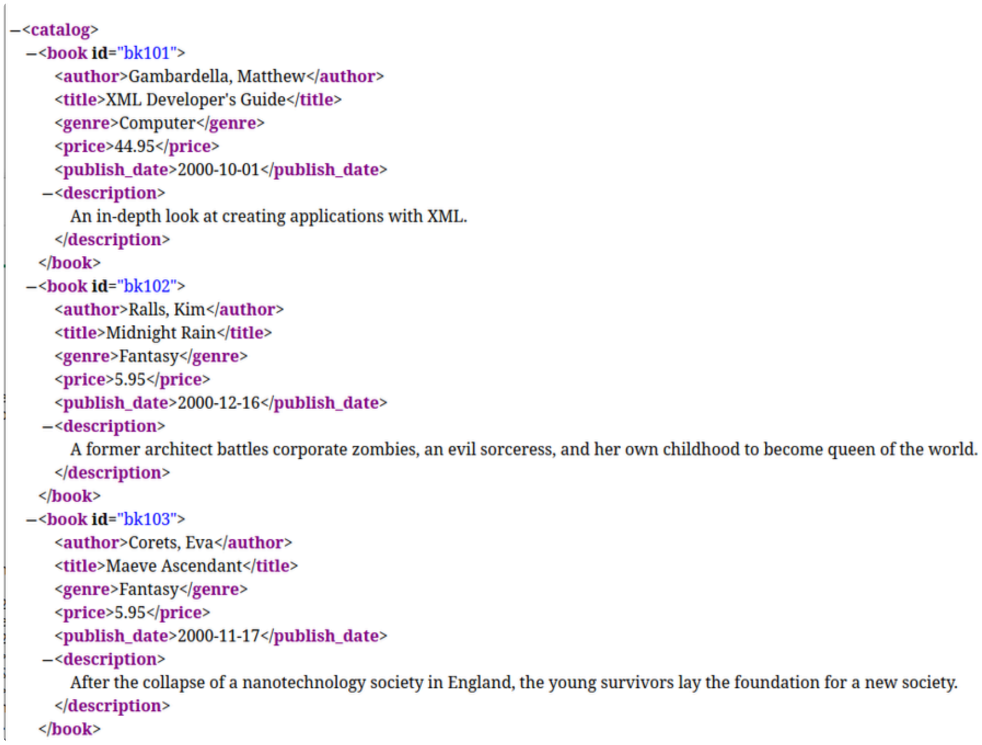
ingo@simsage.ai

Mapping XML / the Metadata Tab

We need to map the XML structured into SimSage entities.

NB. You must save this crawler before starting this mapping.

Let us look at an example. Suppose we have the following XML, a catalog of books.



```
–<catalog>
  –<book id="bk101">
      <author>Gambardella, Matthew</author>
      <title>XML Developer's Guide</title>
      <genre>Computer</genre>
      <price>44.95</price>
      <publish_date>2000-10-01</publish_date>
    –<description>
        An in-depth look at creating applications with XML.
      </description>
    </book>
  –<book id="bk102">
      <author>Ralls, Kim</author>
      <title>Midnight Rain</title>
      <genre>Fantasy</genre>
      <price>5.95</price>
      <publish_date>2000-12-16</publish_date>
    –<description>
        A former architect battles corporate zombies, an evil sorceress, and her own childhood to become queen of the world.
      </description>
    </book>
  –<book id="bk103">
      <author>Corets, Eva</author>
      <title>Maeve Ascendant</title>
      <genre>Fantasy</genre>
      <price>5.95</price>
      <publish_date>2000-11-17</publish_date>
    –<description>
        After the collapse of a nanotechnology society in England, the young survivors lay the foundation for a new society.
      </description>
    </book>
```

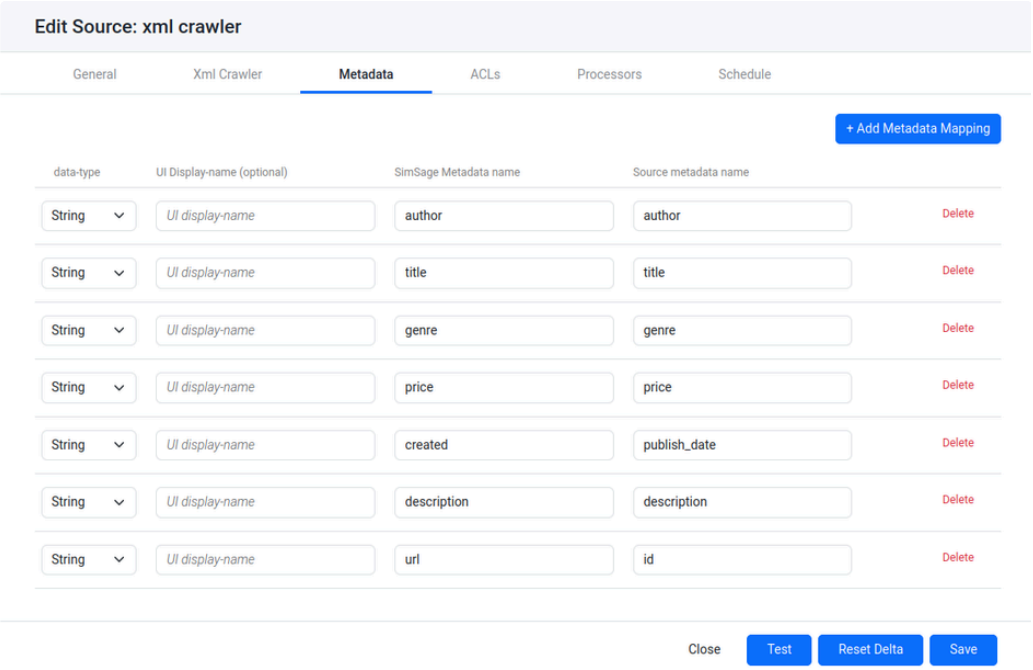
We want to create a SimSage asset for each book. SimSage requires a unique ID for each Asset. The books above have an attribute called “id” that is just that, associated with each book.

SimSage has a series of special names that can be used to map the main items used in SimSage. You can map any item, even if SimSage doesn’t directly map it to one of its main items. Items that don’t map are just mapped as metadata and can be found through metadata searches.

The main items understood by SimSage are

SimSage metadata name		Description
url	**required**	the primary key of a SimSage item. If the primary key is an http or https reference the item can be directly access from the remote system.
title	optional	The title of a SimSage record, important in search as it gives each record a score boost
author	optional	The author of a SimSage record
created	optional	The created Date-time (must be a date as a minimum, various formats supported) of a SimSage record
last-modified	optional	The last-modified Date-time (must be a date as a minimum, various formats supported) of a SimSage record

In order to map the books of catalog to a SimSage record we change the metadata tab data to the following:



data-type	UI Display-name (optional)	SimSage Metadata name	Source metadata name	
String	UI display-name	author	author	Delete
String	UI display-name	title	title	Delete
String	UI display-name	genre	genre	Delete
String	UI display-name	price	price	Delete
String	UI display-name	created	publish_date	Delete
String	UI display-name	description	description	Delete
String	UI display-name	url	id	Delete

Attributes are automatically mapped if available (e.g. id in the books XML above), if XML items are child nodes instead (e.g. author in the books XML above) the text of that child is mapped.

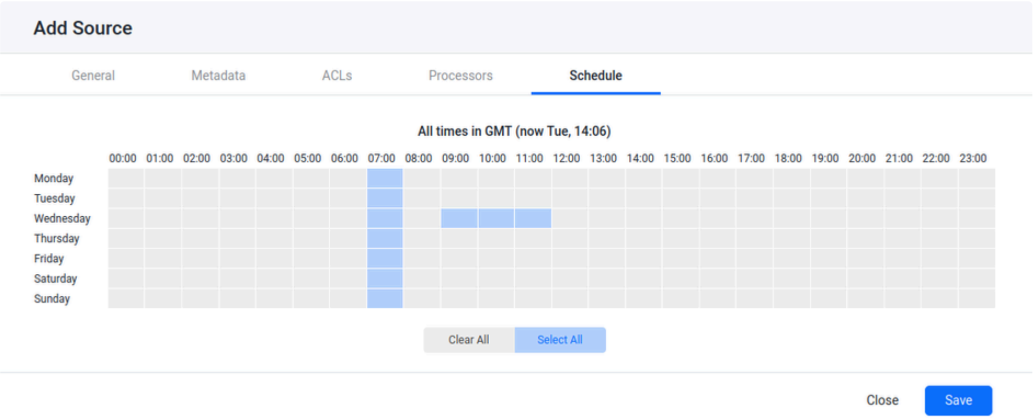
A book's “id” becomes the SimSage primary key (url). The SimSage metadata “url” is the only field that is required.

The “price” and “genre” fields are mapped into the metadata of each SimSage record. The “publish_date” is mapped to both the created field of each SimSage record.

The Schedule Tab

Here you can specify when you would like the crawler to be allowed access to the SimSage platform.

This is done by setting desired time frames to active. In the example provided we used the “select all” button to allow SimSage to always be available to your crawler.



All times in GMT (now Tue, 14:06)

	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00	21:00	22:00	23:00
Monday								■																
Tuesday								■																
Wednesday								■		■	■	■												
Thursday								■																
Friday								■																
Saturday								■																
Sunday								■																