DARTMOUTH

Sitemaps for the ArchivesSpace PUI

Joshua Shaw

Rauner Special Collections Library & Digital Library Technologies Group

Dartmouth Library, Dartmouth College

What is a sitemap and how does it help us?

- A sitemap is an xml file that lists pages contained in your site
- Each page is described by its url, the date it was last modified and the typical amount of time between updates to the page
- Google and other crawlers use the sitemap as a ‘map’ to your site
- Rather than checking the links on each page to find new pages, the search crawler knows exactly where to go

Basic rules for sitemaps

- Sitemap files can contain up to 50,000 entries
- A sitemap index file is necessary for sites with more than 50k pages. Its a list of all the individual sitemap files
- Sitemaps typically live in the base directory of your site
- The robots.txt file tells the crawlers what the sitemap file or sitemap index file name is
- Check out https://www.sitemaps.org/ for more information

A Typical Sitemap File

```
<urlset>
    <url>
        <loc>
            https://archives-manuscripts.dartmouth.edu/agents/families/1
        </loc>
        <lastmod>2020-02-04</lastmod>
        <changefreq>yearly</changefreq>
    </url>
    <url>
        <loc>
            https://archives-manuscripts.dartmouth.edu/agents/corporate_entities/1
        </loc>
        <lastmod>2020-02-11</lastmod>
        <changefreq>yearly</changefreq>
    </url>
    …
</urlset>
```

A Typical Sitemap index file

```
<sitemapindex>
   <sitemap>
      <loc>
         https://archives-manuscripts.dartmouth.edu/aspace_sitemap_part_0.xml
      </loc>
      <lastmod>2020-07-22</lastmod>
   </sitemap>
   <sitemap>
      <loc>
         https://archives-manuscripts.dartmouth.edu/aspace_sitemap_part_1.xml
      </loc>
      <lastmod>2020-07-22</lastmod>
   </sitemap>

…
</sitemapindex>
```

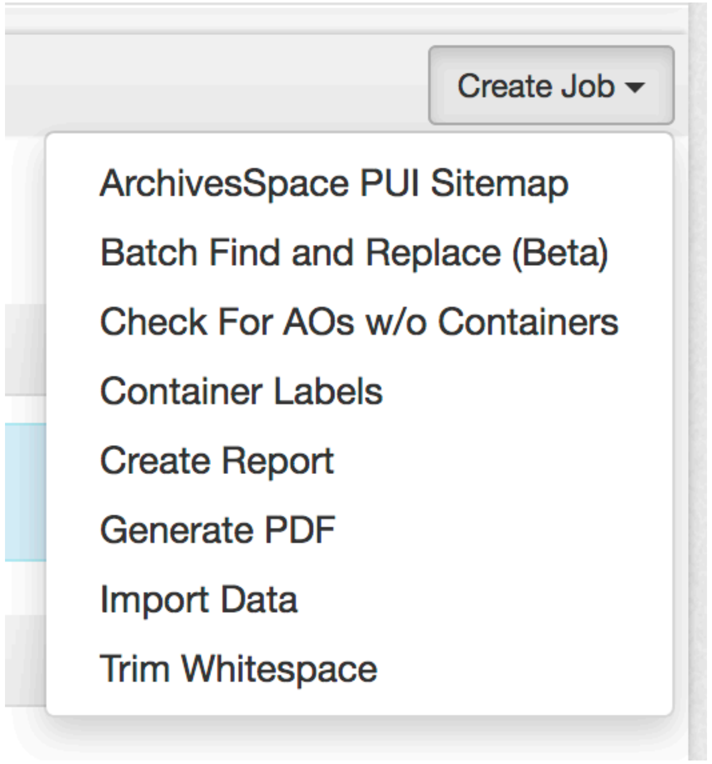
Why the PUI needs a sitemap

- Lots and lots of pages to crawl – mostly archival objects that are relatively buried within the site.
- Each archival object page probably doesn’t rank very high on Google’s popularity scale and may not be crawled immediately
- The PUI is structured in a way that potentially makes it more difficult for crawlers to find all of your archival objects
- Its been requested as a feature: https://archivesspace.atlassian.net/browse/ANW-834

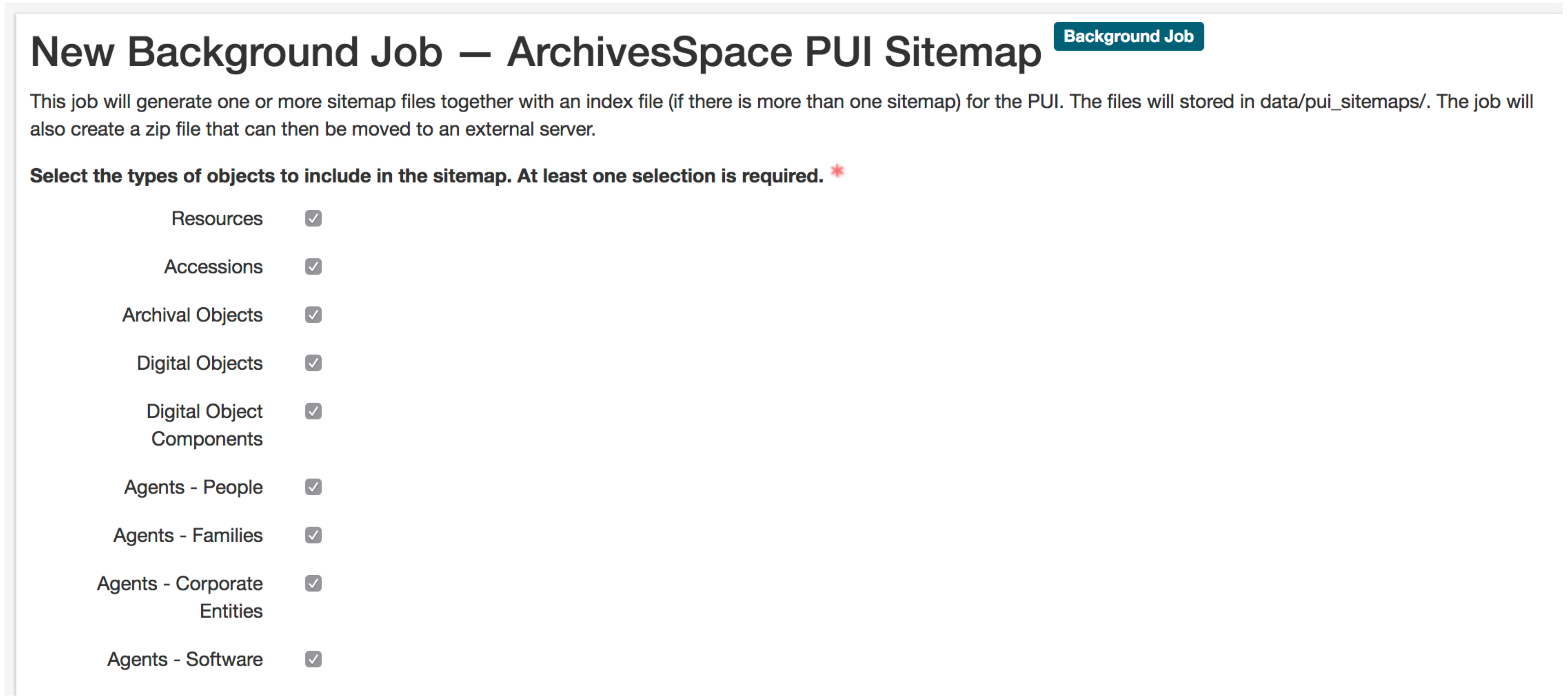
Overview of the Plugin

- The plugin adds a new background job to the staff interface cleverly named 'ArchivesSpace PUI Sitemap'

- After selecting the new job, you are presented with several options that allow you to customize the sitemap


Create Job
ArchivesSpace PUI Sitemap
Batch Find and Replace (Beta)
Check For AOs w/o Containers
Container Labels
Create Report
Generate PDF
Import Data
Trim Whitespace

Setting up the Sitemap Job (1)



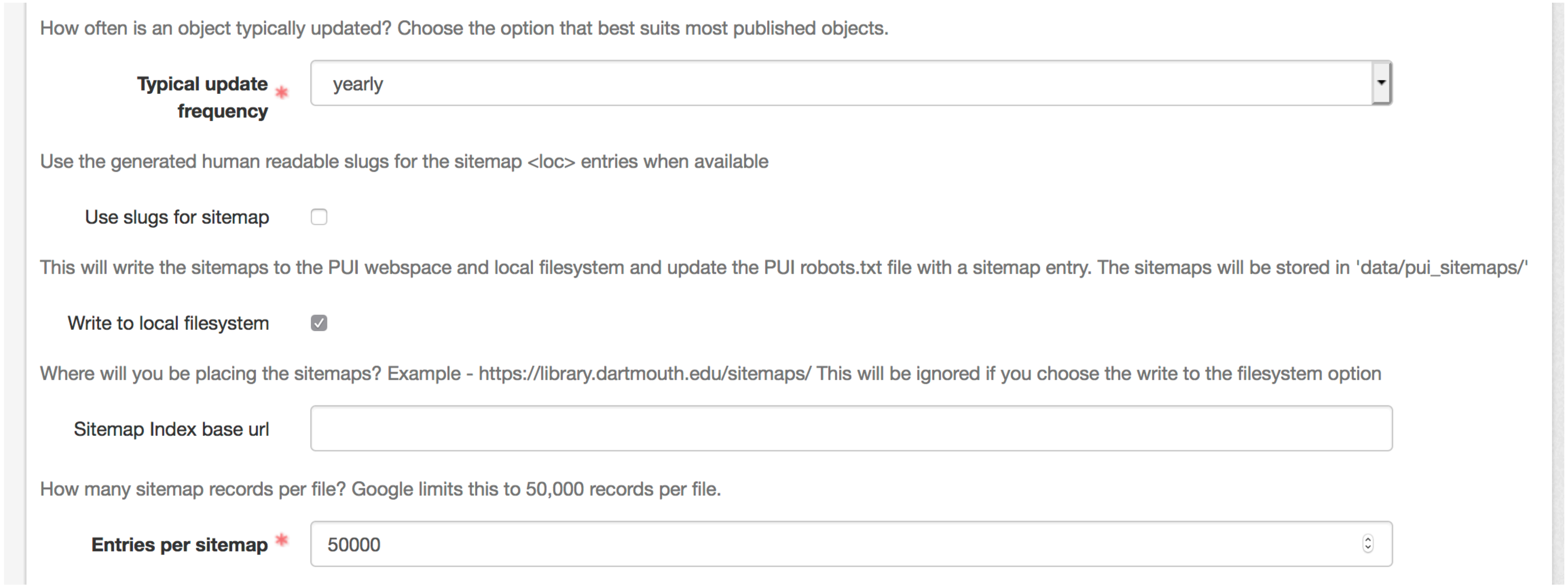
New Background Job — ArchivesSpace PUI Sitemap Background Job

This job will generate one or more sitemap files together with an index file (if there is more than one sitemap) for the PUI. The files will stored in data/pui_sitemaps/. The job will also create a zip file that can then be moved to an external server.

Select the types of objects to include in the sitemap. At least one selection is required. *

Resources	✓
Accessions	✓
Archival Objects	✓
Digital Objects	✓
Digital Object Components	✓
Agents - People	✓
Agents - Families	✓
Agents - Corporate Entities	✓
Agents - Software	✓

Setting up the Sitemap Job (2)



How often is an object typically updated? Choose the option that best suits most published objects.

Typical update frequency * yearly

Use the generated human readable slugs for the sitemap <loc> entries when available

Use slugs for sitemap ☐

This will write the sitemaps to the PUI webspace and local filesystem and update the PUI robots.txt file with a sitemap entry. The sitemaps will be stored in 'data/pui_sitemaps/'

Write to local filesystem ✓

Where will you be placing the sitemaps? Example - https://library.dartmouth.edu/sitemaps/ This will be ignored if you choose the write to the filesystem option

Sitemap Index base url

How many sitemap records per file? Google limits this to 50,000 records per file.

Entries per sitemap * 50000

What happens when the job runs

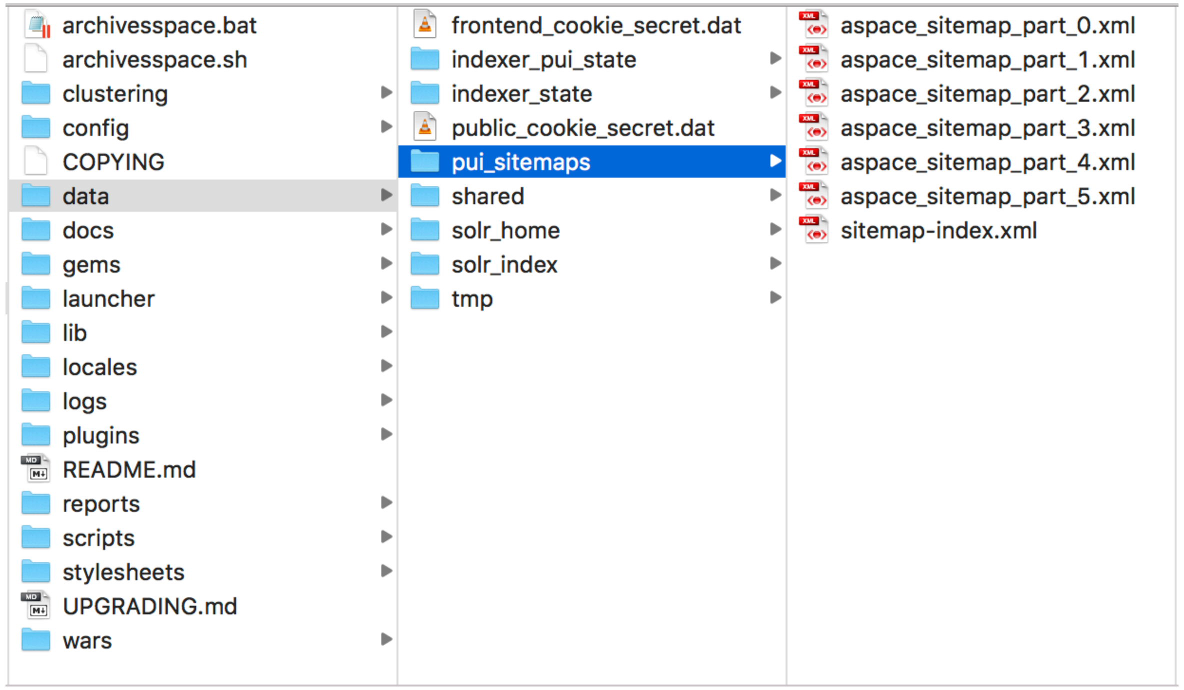
- The job looks for all published objects from the types you selected
- It checks archival objects and digital object components for unpublished ancestors. We don’t want to include those in the sitemap files.
- It creates one or more sitemap files listing out all of the pages in the PUI and a sitemap index file
- It updates the robots.txt file with a path to the sitemap index file
- It copies the sitemap files into the base directory of the PUI app
- The whole process can take a while and depends on your server and the number of published objects. Ours takes about 10-15 minutes with about 250k-300k objects.

Under the hood of the plugin - configuration

- The plugin supports two configuration settings – both optional.
- These are meta tags for Google and Bing that allow a site owner to claim the site if submitting the sitemaps manually.
- Additional configuration is set within the plugin. These could be moved into the global configuration in a future release.

Under the hood - storage

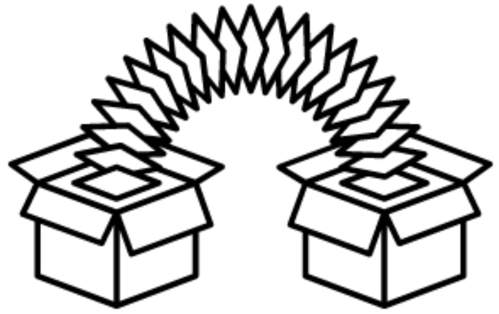
- The plugin checks for and creates, if necessary, a new directory in the general data directory. The new directory is named ‘pui_sitemaps’
- This new directory will store any sitemaps created. Its also used to copy the sitemaps into the PUI app space on any application restarts



archivesspace.bat	frontend_cookie_secret.dat	aspace_sitemap_part_0.xml
archivesspace.sh	indexer_pui_state	aspace_sitemap_part_1.xml
clustering	indexer_state	aspace_sitemap_part_2.xml
config	public_cookie_secret.dat	aspace_sitemap_part_3.xml
COPYING	pui_sitemaps	aspace_sitemap_part_4.xml
data	shared	aspace_sitemap_part_5.xml
docs	solr_home	sitemap-index.xml
gems	solr_index	
launcher	tmp	
lib		
locales		
logs		
plugins		
README.md		
reports		
scripts		
stylesheets		
UPGRADING.md		
wars		

Under the hood – startup & robots.txt

- On startup the plugin checks the pui_sitemaps directory for any files and copies those into the PUI app space so that they are visible in the PUI at /sitemap_index.xml, /aspace_sitemap_part_0.xml, etc.
- It updates the PUI robots.txt file, if necessary, with a path to the sitemap index file.



Something like

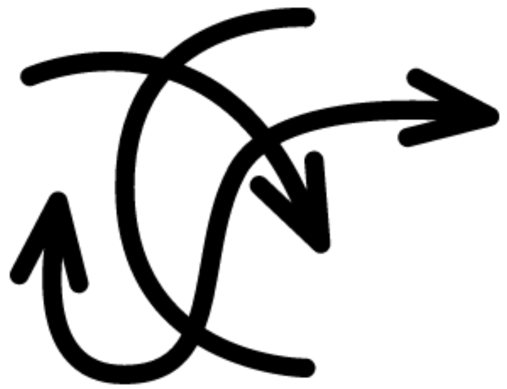
```
Sitemap: https://archives-manuscripts.dartmouth.edu/sitemap-index.xml
```

Under the hood – PUI web path

- One tricky part of the plugin was figuring out how to tell the backend where the PUI webspace is. Since that's recreated on startup and is dependent on configuration, I needed a method to communicate the location between the two apps.
- The kinda kludgy solution is to have the PUI plugin_init write a text file containing the PUI webspace path into the plugin space so that the backend job can then open up the text file and read the path. That path is used to copy the sitemap files into the proper location.

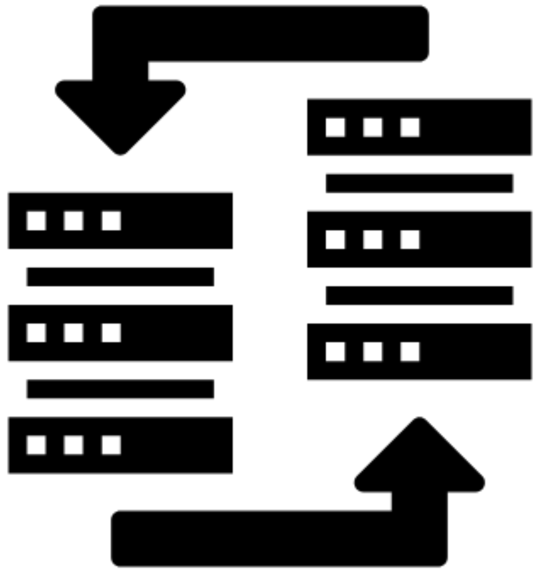
It'll look something like

```
/Users/joshuashaw/archivesspace/archivesspace-271-prod/data/tmp/jetty-
0.0.0.0-8081-public.war-_-any-/webapp/WEB-INF
```



Under the hood – Database & SOLR

- The plugin uses both the database and the SOLR index to create the sitemap files
- The database is queried first to retrieve all of the published objects
- The SOLR index is then used to determine if any objects have unpublished ancestors.
- Since the database query finds all published objects, we also check to make sure that the repository is published and remove any objects that belong to an unpublished repo.



Additional Options & Issues

- For more complex web environments, the plugin also provides an option to download the set of sitemaps as a zip file for hosting on an external server.
- ARKs are not supported at this time.
- Tested with v2.5.0 to v2.7.1. ArchivesSpace v2.8.0 support coming soon!
- Core code inclusion if I can make the time!

Get it from GitHub

https://github.com/dartmouth-dltg/aspace_sitemap

Latest release is 1.0.1

Pull requests welcome!

Thanks to Laney McGlohon, Andrew Morrison and Mark Custer for feedback, testing and conversation.

Thank you!

joshua.d.shaw@dartmouth.edu