Digital Stacks Maintenance
Using Computer Scripts to Provide and Ensure Access to E-Resources

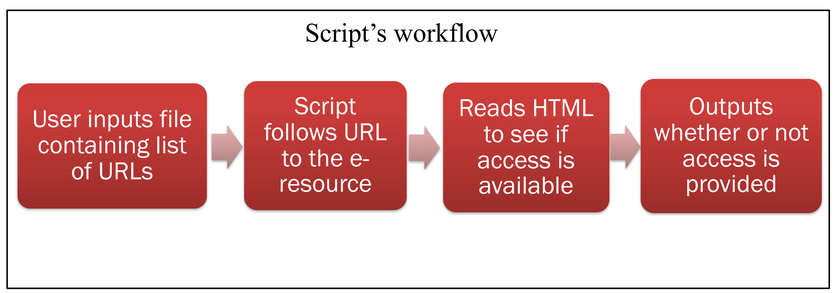
Benjamin Bradley

Introduction:

UMD Libraries manages its e-resources using OCLC's WorldShare Collection Manager which enables users to access the materials in WorldCat Discovery. Collection Manager enables some automation, but critical activities, such as checking URLs and access, are still time prohibitive. Similarly, we depend on publisher provided metadata which does not always meet our needs nor is it always available. By using a little computer programming, I have found ways to create proactive strategies for maintaining e-resources. The first, the E-Resource Access Checker is an open-source tool that I have been editing to better meet the needs of my library. It uses the Ruby programming language, which I had no prior experience using, but its code is easy to understand and is a great opportunity to learn programming from trial and error. The script checks URLs to ensure they work and the publisher is providing access. The second script, which I wrote myself, searches WorldCat to find records to harvest missing metadata such as unique identifiers and URLs.

E-Resource Access Checker

- Written in Ruby
- Code originally written by Kristina Spurgin
- Checks if resource URLs work and access is provided
- Only reads HTML and does not download files so does not cause any concern regarding license agreements




```
elsif package == "pqdt"
sleeptime = 5
if page.include?("Full text - PDF")
   access = "Full access"
  elsif page.match(/fulltextPDF/)
      access = "Full Access"
  elsif page.match(/<a id="addFlashPageParameterformat_fulltextPDF"/)
   access = "Full access"
else
   access = "check access"
end
```

```
l for="refworksPassword"
96">Full text - PDF</a><
```

```
id="addFlashPageParameterformat_fulltextPDF"
```

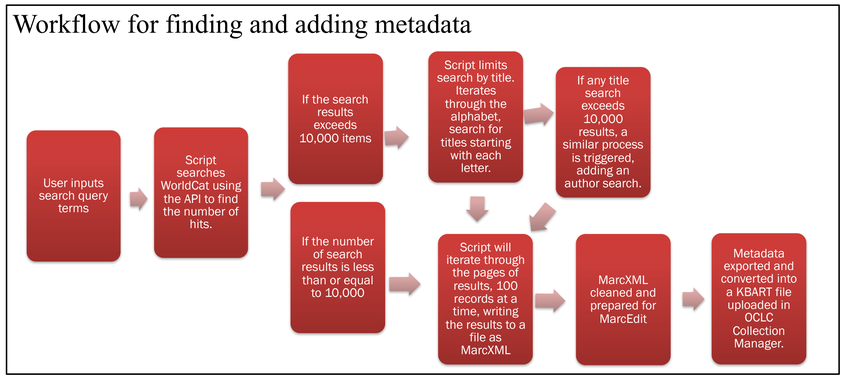
This screenshot, using code I added, illustrates how the script tries to match the webpage's HTML against pre-programmed examples of code or text.

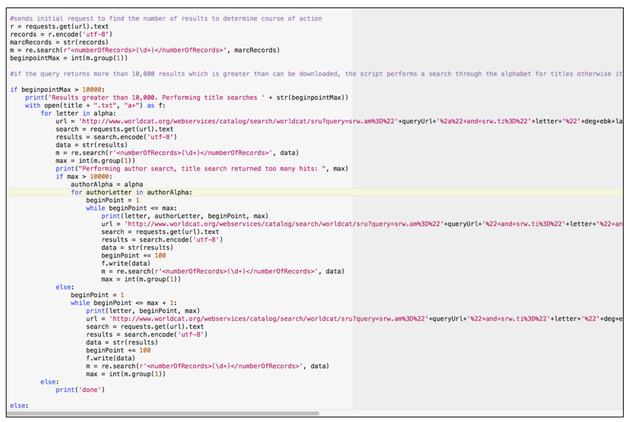
Example of output from the E-Resource Access Checker.

Z	AA
title_url	access
http://wwwlib.umi.com/dissertati	check access
http://wwwlib.umi.com/dissertati	Full access
http://wwwlib.umi.com/dissertati	Full access
http://wwwlib.umi.com/dissertati	Full access
http://wwwlib.umi.com/dissertati	Full access
http://wwwlib.umi.com/dissertati	Full access
http://wwwlib.umi.com/dissertati	Full access
http://wwwlib.umi.com/dissertati	Full access

Harvesting Metadata with the WorldCat Search API

- Written in Python
- Searches OCLC WorldCat Master records and writes matching records to a file in MARCXML
- Metadata can then be exported to supplement existing metadata





Screenshot of some of the code

```
File name for output: test
Results greater than 10,000. Performing title searches 38515
('Performing author search, title search returned too many hits: ', 38515)
('A', 'A', 1, 38515)
('A', 'A', 101, 1883)
('A', 'A', 201, 1883)
('A', 'A', 301, 1883)
('A', 'A', 401, 1883)
('A', 'A', 501, 1883)
('A', 'A', 601, 1883)
('A', 'A', 701, 1883)
('A', 'A', 801, 1883)
('A', 'A', 901, 1883)
('A', 'A', 1001, 1883)
('A', 'A', 1101, 1883)
('A', 'A', 1201, 1883)
('A', 'A', 1301, 1883)
('A', 'A', 1401, 1883)
```

Script running in my terminal.

Next Steps

E-Resource Access Checker:

- Contribute my edits on GitHub
- Find platforms missing from the script and add them on GitHub

WorldCat API Searcher:

- Further develop the code to work well for different projects
- Create a function to export only desired metadata

Links

- ***Read*** Kristina Spurgin's article introducing the E-Resource Access Checker: journal.code4lib.org/articles/9684
- ***Find*** it on GitHub: github.com/UNC-Libraries/Access-Checker
- ***Find*** the searcher on GitHub: github.com/bradley-benjamin26/WCSearchAPIMARCHarvester
- ***Read*** Documentation for the API: www.oclc.org/developer/develop/web-services/worldcat-search-api.en.html

UNIVERSITY OF MARYLAND 18 56

UNIVERSITY LIBRARIES