Profiling the DPLA

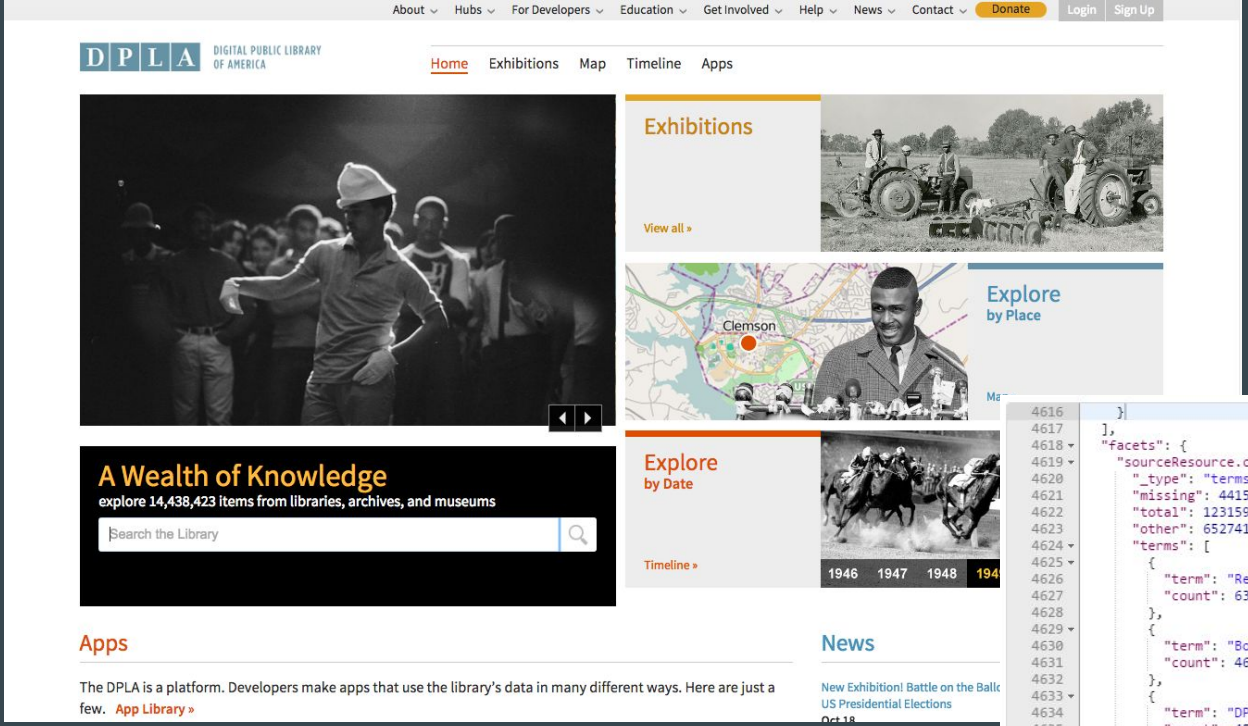
A Project Introduction

Content

1. Introduction to the DPLA
2. DPLA Website and API
3. Project Goals and Objectives
4. Interesting Questions
5. Designing Our Work
    a. System Architecture
    b. Goals for Python Code
    c. Project Map
    d. Output Structure
6. Implementation: Challenges and Methods
    a. Improving the Code
    b. Problems Encountered
    c. Outcome
7. Resources to Continue this Project

THE DIGITAL PUBLIC LIBRARY OF AMERICA (DPLA)

- Is an online portal for digital cultural heritage material
- Accepts material from a group of about 30 Providers (also called Hubs)
- These Providers get material from individual institutions, like libraries, archives and museums, known as Data Providers.
- An institution can be both a Provider and a Data Provider
- Data Providers can add metadata to items
- Providers (Hubs) can add metadata to items
- The DPLA adds metadata to items
- There is no way to identify which information came from a the Data Provider vs. the Provider
- The basic unit of information is an item



DPLA Homepage (above)

DPLA API Results:
Displayed in Postman (Below)


```
    }
  ],
  "facets": {
    "sourceResource.collection.title": {
      "_type": "terms",
      "missing": 4415285,
      "total": 12315907,
      "other": 6527411,
      "terms": [
        {
          "term": "Records of the National Aeronautics and Space Administration",
          "count": 638062
        },
        {
          "term": "Botany",
          "count": 468836
        },
        {
          "term": "DPLA: Include in Digital Public Library of America",
          "count": 456748
        },
        {
          "term": "Flowering plants and ferns",
          "count": 404686
        },
        {
          "term": "Anthropology",
          "count": 285233
        },
        {
          "term": "Records of the Office of the Secretary of Defense",
          "count": 269704
        },
        {
          "term": "War Department Collection of Confederate Records",
          "count": 229825
        },
        {
          "term": "Collections Online",
```

Goal and Objectives

What We Hope to Do:

Extract the data available from the DPLA. Compile the data in a new manner, which will allow us to answer interesting questions about how metadata attributes are being used... or not being used.

What We Hope to Accomplish:

Provide insights to the usage of metadata attribute fields that will allow institutions providing the data to leverage their metadata creation in a manner that will result in the most exposure of their items.

In order to meet our goals and objectives we identified some interesting questions we wanted to answer about the usage of metadata in the DPLA. These questions guided the features we choose to implement in our code. These questions all helped us understanding of how the “collection” attribute was being used by Providers

Interesting Questions

How many providers has the DPLA collected from? And who are they?

How many collections does each provider contribute to?

How many items has each provider contributed?

What’s the distribution of the number of collections referred to by an item?

How many items have the same number of collections?

How many items does each collection share with other collections?

How many collections are referred to by more than one data Provider?

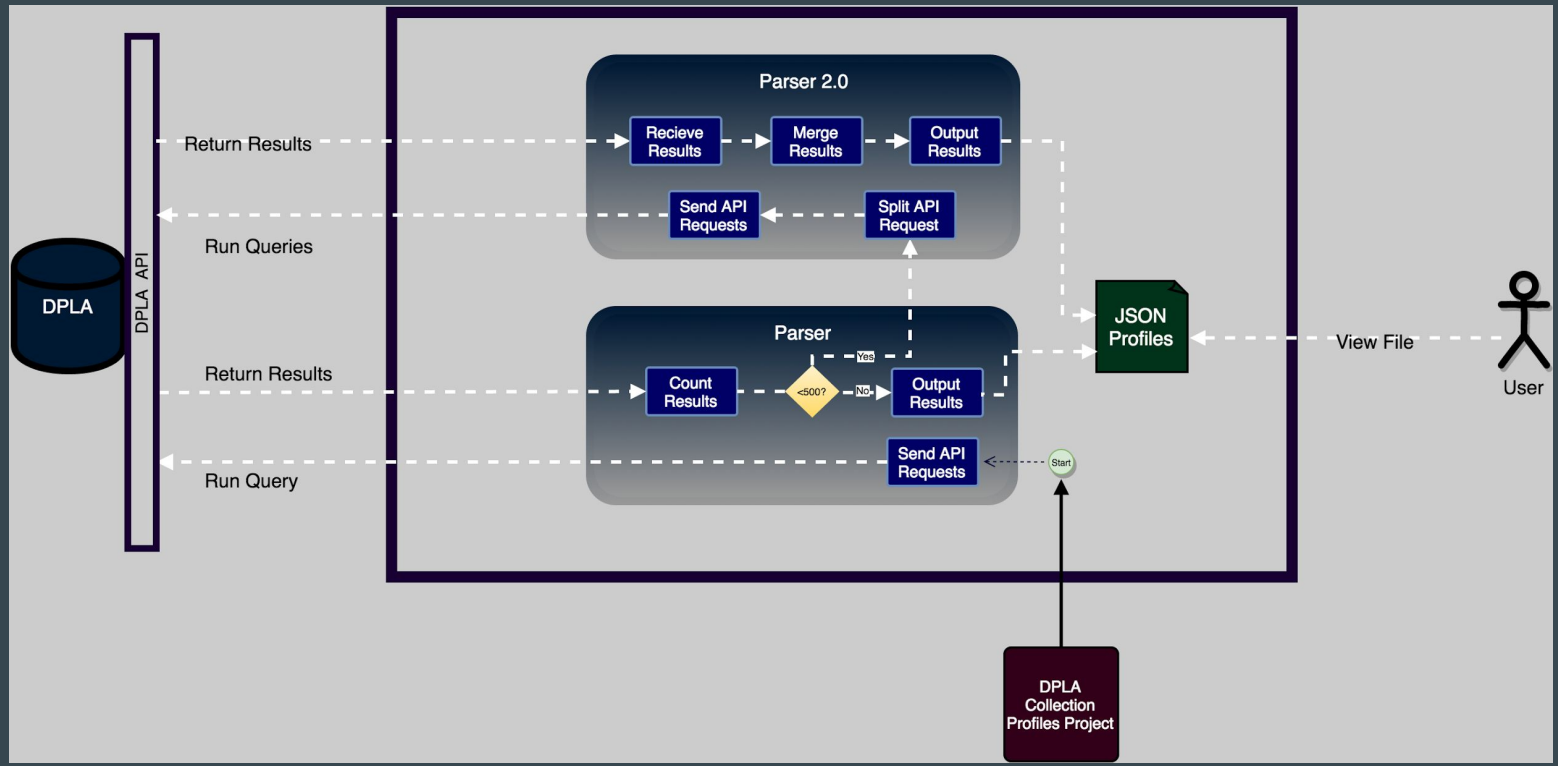
What collections share the most items?

How many items are not part of a collection?

Which providers / data providers are giving items with no collections?

Documenting Our Work

System Architecture Diagram



Goals for the Python Code

Features of the Code:

For Each Provider:
- Count of items contributed
- Count of collections used
- Count of items with no collection
- Count of items in each collection
- Count of dataProviders
- Item details: id, collection & dataProvider

The Code Should:

Be modular and expandable
Be efficient (Have no redundancy)
Be clean and readable
Use variables with meaning

Project Map

CCD Data Object and Property Labels

Variable Label	Property	Usage/Description	Level	Type
CCD_OBJ_LBL_adminDetails	adminDetails	holds the analysisDate and analysisTime properties	File	class
CCD_PROP_LBL_analysisDate	analysisDate	day file was generated	File	
CCD_PROP_LBL_analysisTime	analysisTime	time file was generated	File	
CCD_OBJ_LBL_dplaData	"dplaData"		DPLA	class
CCD_OBJ_LBL_providerData	"providerData"	object holding the providerId and the providerName	Provider	object
CCD_PROP_LBL_providerId	"@id"	URI for the provider page the DPLA API	Provider	string
CCD_PROP_LBL_providerName	"name"	Human-readable version of provider name	Provider	string
CCD_PROP_LBL_itemCountByName	"itemCountByName"	Count of total DPLA items by Provider name	Provider	numeric
CCD_PROP_LBL_itemCountById	"itemCountById"	Count of total DPLA items by Provider id	Provider	numeric
CCD_PROP_LBL_itemsInCollections	"itemsInCollections"	Number of Items in collections by Provider	Provider	numeric
CCD_PROP_LBL_collectionCount	"collectionCount"	Count of Collections by Provider	Provider	numeric
CCD_PROP_LBL_dataProviderCount	"dataProviderCount"	Count of Data Providers by Provider	Provider	numeric
CCD_PROP_LBL_providerCountByName	"providerCountByName"	Count of providers based on provider.name field	Provider	numeric
CCD_PROP_LBL_providerCountById	"providerCountById"	Count of providers based on provider.id field	Provider	numeric
CCD_PROP_LBL_missingItemsByName	"missingItemsByName"	Missing items by Provider Name	Provider	
CCD_PROP_LBL_missingItemsById	"missingItemsById"	Missing Items by Provider Id	Provider	
CCD_PROP_LBL_itemCount	"itemCount"	Count of total number of items	DPLA	numeric
CCD_PROP_LBL_dataProviders	"dataProviders"	List of Data Providers by Provider	Provider	
CCD_PROP_LBL_provider	“provider”	Name of Provider	Provider	string
CCD_PROP_LBL_providerItemCount	“providerItemCount”	Number of Items contributed by a provider	Provider	numeric

The Project Map identifies all the variables and file names used in the code and how those variables are used/ defined.

Provider Level Collection Details JSON Object Structure

"-providerColl"

/object

providerItemCount

noCollectionCount

provider

providerCollectionCount

/Collection

CollectionTitle

ItemCount

dataProviderCount

providerDataProviderCount

Item Level Item Details JSON Object Structure

"- providerItem.json"

/object

/itemsInCollections

itemID : collectionTitle

provider

Provider Level Data Providers Details JSON Object Structure

"-providerDP"

/object

dataProviderCount

noDataProviderItemCount

provider

/dataProviders

itemsCount

dataProvider

Implementation: Challenges and Methods

Improving the Code

- Previously the code...
  - Was long and clunky
  - Had duplicate functions in different files
- It was refactored to:
  - Increase modularity
  - Reduce redundancy
  - Add new functions

```
def profile_dpla(base_dpla_filename):


# return 1 if the collection volume field>0, if filed=0 mean there is no usage of this field
def get_usage(collection_volume_field):


# get the volume for the filed in the location
def get_item_volume(field, location):


def have_or_not(a):
Begin code execution here
# """

# setup argument parser
parser = argparse.ArgumentParser(description='please insert parameters')
parser.add_argument("-f", "--filename",
```

Problems We Encountered

- Default limit of 50 facets returned per query.
- Facet_size limitation is 2000
- Duplication of items in query results

Provider Level Details: Output File

The Code Now:

- Is clean and readable
- Is modular and extensible
- Identifies all the desired features
- Generates output with the data necessary to answer all the interesting questions.

Resources to Continue this Project:

The DPLA_Profiles_Documentation_and_Resources_Guide, located in UT Box has a complete inventory of all project files, locations and content. Data needed to continue the project is located in 3 places:

Github: yanxian0924/DPLA_metadata (publicly available)

UT Box Folder: DPLA Profile Creation Documentation (need permission from Unmil)

Class Server: saab.ischool.utexas.edu/export/home/u09/unmil/ComputationalCollectionDescriptions (need permission from Unmil)