

---

# DigitalArchive

*Release 0.1.11*

Nov 29, 2022



---

## Contents

---

<b>1</b>	<b>Features</b>	<b>3</b>
<b>2</b>	<b>Installation</b>	<b>5</b>
<b>3</b>	<b>Usage</b>	<b>7</b>
<b>4</b>	<b>Contents</b>	<b>9</b>
4.1	Quickstart . . . . .	9
4.2	Cookbook . . . . .	11
4.3	Public API . . . . .	13
	<b>Index</b>	<b>27</b>



The **digitalarchive** Python library is a client and ORM for accessing, searching, and downloading historical documents and their accompanying scans, translations, and transcriptions from the [Wilson Center's Digital Archive](#) of historical primary sources.



# CHAPTER 1

---

## Features

---

- Search for documents and other Digital Archive resources by keyword.
- Easily retrieve translations, transcriptions, and related records of any document.
- Fully documented models for all of the Digital Archive resource types.





## CHAPTER 2

---

### Installation

---

Install the latest stable version of digitalarchive using `pip`:

```
$ python3 -m pip install digitalarchive
```



## CHAPTER 3

### Usage

Find documents by keyword:

```
>>> from digitalarchive import Subject
>>> Subject.match(name="Tiananmen Square Incident").first()
Subject(id='2229', name='China--History--Tiananmen Square Incident, 1989', value=
↳ 'China--History--Tiananmen Square Incident, 1989', uri='/srv/subject/2229.json')
```

Discover collections of related documents:

```
>>> from digitalarchive import Collection, Document
>>> collection = Collection.match(name="Local Nationalism in Xinjiang").first()
>>> docs = Document.match(collections=[collection])
>>> for doc in docs.all():
...     print(doc.title)
Memorandum on a Discussion held by the Consul-General of the USSR in Ürümchi, G.S.
↳ DOBASHIN, with the Secretary of the Party Committee of the Xinjiang Uyghur
↳ Autonomous Region, Comrade LÜ JIANREN
Memorandum on a Discussion held by the Consul-General of the USSR in Ürümchi, G.S.
↳ DOBASHIN, with Deputy Chairman of the People's Committee of the Xinjiang Uyghur
↳ Autonomous Region, Comrade XIN LANTING
Memorandum of a Discussion held by USSR Consul-General in Ürümchi, G S. Dobashin,
↳ with First Secretary of the Party Committee of the Xinjiang Uyghur Autonomous
↳ Region, Comrade Wang Enmao, and Chair of the People's Committee, Comrade S. Äzizov
Note from G. Dobashin, Consul-General of the USSR in Ürümchi, to Comrades N.T.
↳ Fedorenko, Zimianin, and P.F. Iudin
Memorandum on a Discussion with Wang Huangzhang, Head of the Foreign Affairs Office
↳ of the Prefectural People's Committee
Iu. Andropov to the Central Committee of the CPSU, 'On the Struggle with Local
↳ Nationalism in China'
Note, M. Zimianin to the Central Committee of the CPSU and to Comrade Iu. V. Andropov
M. Zimianin to the Central Committee of the CPSU and to Comrade Iu. V. Andropov, 'On
↳ Manifestations of Local Nationalism in Xinjiang (PRC)'
M. Zimianin to the Department of the Central Committee of the CPSU and to Comrade Iu.
↳ V. Andropov
```

Read the [Quickstart](#) guide for a tutorial on working with documents and searches, or consult the [Cookbook](#) for examples of common operations.

### 4.1 Quickstart

The `Document` is the basic unit of content in the Digital Archive. Every document is accompanied by metadata, including a short description of its content, information about the archive it was obtained, subjects it is tagged with, alongside other information.

Most of the Digital Archive's documents originate from outside the United States. Translations are available for most documents, as well as original scans in some cases. The `Document` model describes the available methods and attributes for documents.

The `digitalarchive` package also provides models for other kinds of resources, such as `Subject`, `Collection`, `Theme`, `Coverage`, and `Repository`. These models can be used as filters when searching for documents. Consult the *Public API* documentation for a full description of available models.

#### 4.1.1 Searching

The `Document`, `Contributor`, `Coverage`, `Collection`, `Subject`, and `Repository` models each expose a `match()` method that can be used to search for documents. The method accepts a list of keyword arguments corresponding to the attributes of the matched for model.

```
>>> from digitalarchive import Document
>>> docs = Document.match(description="Cuban Missile Crisis")
```

The `match` method always returns an instance of `digitalarchive.matching.ResourceMatcher`. `ResourceMatcher` exposes a `first()` method for accessing a single document and an `all()` for accessing a list of all respondent records.

```
>>> from digitalarchive import Document
>>> docs = Document.match(description="Cuban Missile Crisis")
>>> docs.first().title
"From the Journal of S.M. Kudryavtsev, 'Record of a Conversation with Prime Minister_
↳ of Cuba Fidel Castro Ruz, 21 January 1961'"
```

Searching for a record by its `id` always returns a single record and ignores any other keyword arguments.

```
>>> from digitalarchive import Document
>>> test_search = Document.match(id="175898")
>>> test_search.count
1
>>> doc = test_search.first()
>>> doc.title
'Memorandum on a Discussion held by the Consul-General of the USSR in Ürümchi, G.S.
↳ DOBASHIN, with the Secretary of the Party Committee of the Xinjiang Uyghur
↳ Autonomous Region, Comrade LÜ JIANREN'
```

## 4.1.2 Filtering Searches

One can limit searches to records created between specific dates by passing a `start_date` keyword, an `end_date` keyword, or both.

```
>>> from digitalarchive import Document
>>> from datetime import date
>>> Document.match(start_date=date(1989, 4, 15), end_date=date(1989, 5, 4))
ResourceMatcher(model=<class 'digitalarchive.models.Document'>, query={'start_date':
↳ '19890415', 'end_date': '19890504', 'model': 'Record', 'q': '', 'itemsPerPage': 200}
↳ , count=22)
```

Searches can also be limited to records contained within a specific collection, subject, or other container. Matches for Documents can be filtered by one or more Collection, Repository, Coverage, Subject, Contributor, and Donor instances:

```
>>> from digitalarchive import Collection, Document
>>> xinjiang_collection = Collection.match(id="491").first()
>>> xinjiang_collection.name
'"Local Nationalism" in Xinjiang, 1957-1958'
>>> docs = Document.match(collections=[xinjiang_collection])
>>> docs.count
9
```

## 4.1.3 Hydrating Search Results

Most search results return “unhydrated” instances of resources with incomplete metadata. All attributes that are not yet available are represented by `NoneType`. Use the `hydrate()` method to download the full metadata for a resource.

```
>>> from digitalarchive import Document
>>> test_doc = Document.match(description="Vietnam War").first()
>>> test_doc.source is None
True
>>> test_doc.hydrate()
>>> test_doc.source
'AVPRF f. 0100, op. 34, 1946, p. 253, d. 18. Obtained and translated for CWIHP by
↳ Austin Jersild.'
```

It is also possible to hydrate all of the contents of a search result using the `hydrate()` method of `ResourceMatcher`. This operation can take some time for large result sets.

```
>>> from digitalarchive import Document
>>> docs = Document.match(description="Taiwan Strait Crisis")
>>> docs.hydrate()
```

When hydrating a result set, it is also possible to recursively hydrate any child records (translations, transcripts, etc.) in the result set by setting the `recurse` parameter of `hydrate()` to `True`.

```
>>> from digitalarchive import Document
>>> docs = Document.match(description="Taiwan Strait Crisis")
>>> docs.hydrate(recurse=True)
```

## 4.2 Cookbook

Examples of common operations encountered using the Digital Archive client.

### 4.2.1 Search for a resource by keyword

Run a keyword search across the title, description, and document content.

```
>>> from digitalarchive import Document
>>> # Find a document
>>> results = Document.match(description="Cuban Missile Crisis")
>>> # Access a single record.
From the Journal of S.M. Kudryavtsev, 'Record of a Conversation with Prime Minister_
↳ of Cuba Fidel Castro Ruz, 21 January 1961'
```

### 4.2.2 Filter a search by a related record

Search for a collection and then find documents contained in that collection:

```
>>> from digitalarchive import Document, Collection
>>> # Find an interesting collection:
>>> xinjiang_collection = Collection.match(name="xinjiang").first()
>>> # Find records contained in that collection:
>>> documents = Document.match(collections=[xinjiang_collection]).all()
```

### 4.2.3 Filter a Document search by language

Limit a search to documents in a certain language:

```
>>> from digitalarchive.models import Document, Language
>>> RYaN_docs = Document.match(description="project ryan", languages=[Language(id=
↳ "ger")])
>>> RYaN_docs.count
32
```

### 4.2.4 Filter a Document search by date

Search for records after a certain date:

```
>>> from digitalarchive import Document
>>> from datetime import date
>>> postwar_docs = Document.match(start_date=date(1945, 9, 2))
```

**Search for records before a certain date:**

```
>>> from digitalarchive import Document
>>> from datetime import date
>>> prewar_docs = Document.match(end_date=date(1945, 9, 2))
```

**Search for docs between two dates:**

```
>>> from digitalarchive import Document
>>> from datetime import date
>>> coldwar_docs = Document.match(start_date=date(1945, 9, 2), end_date=date(1991,
↪ 12, 26))
```

## 4.2.5 Download the complete metadata for a document

```
>>> from digitalarchive import Document
>>> chernobyl_doc = Document.match(description="pripyat evacuation order").first()
>>> chernobyl_doc.repositories
>>> chernobyl_doc.repositories is None
True
>>> chernobyl_doc.hydrate()
>>> chernobyl_doc.repositories
[Repository(id='84', name='Central State Archive of Public Organizations of Ukraine_
↪ (TsDAHOU)', uri=None, value=None), Repository(id='507', name='Archive of the_
↪ Ukrainian National Chornobyl Museum', uri=None, value=None)]
```

## 4.2.6 Download the original scan of a document.

Original scans (referred to internally as *MediaFile*) are child records of *Document*. They must be hydrated before the PDF content can be accessed.

```
>>> from digitalarchive import Document
>>> chernobyl_doc = Document.match(id="208406").first()
>>> original_scan = chernobyl_doc.media_files[0]
>>> original_scan.pdf is None
True
>>> original_scan.hydrate()
>>> type(original_scan.pdf)
<class 'bytes'>
>>> len(original_scan.pdf)
10936093
```

## 4.2.7 Download the translation or transcript of a document.

Like original scans, *Transcript* and *Translation* are child records of *Document*. They must also be hydrated before their content can be accessed. Translations and transcripts are typically presented as HTML files, but may sometimes be presented as PDFs.



```
>>> from digitalarchive import Document
>>> chernobyl_doc = Document.match(id="208406").first()
>>> translation = chernobyl_doc.translations[0]
>>> translation.hydrate()
>>> translation.filename
'TranslationFile_208406.html'
```

## 4.2.8 Serialize and dump a document to the filesystem.

```
>>> from digitalarchive import Document
>>> chernobyl_doc = Document.match(id="208406").first()
>>> chernobyl_doc.hydrate()
>>> chernobyl_doc_str = chernobyl_doc.json()
>>> chernobyl_doc == Document.parse_raw(chernobyl_doc_str)
True
```

## 4.3 Public API

### 4.3.1 Models

#### Match-able & Hydrate-able Models

```
class digitalarchive.models.Document(id, uri, title, description, doc_date, front-
                                     tend_doc_date, slug, source_created_at,
                                     source_updated_at, first_published_at, source,
                                     type, rights, pdf_generated_at, date_range_start,
                                     sort_string_by_coverage, main_src, model, donors,
                                     subjects, transcripts, translations, media_files, lan-
                                     guages, contributors, creators, original_coverages,
                                     collections, attachments, links, repositories, publishers,
                                     classifications)
```

A Document corresponding to a single record page on [digitalarchive.wilsoncenter.org](http://digitalarchive.wilsoncenter.org).

---

**Note:** Avoid constructing Documents directly—use the *match* function to create Documents by keyword search or by ID.

---

#### Attributes present on all Documents:

**id**

The ID# of the record in the DA.

**Type** str

**title**

The title of a document.

**Type** str

**description**

A one-sentence description of the document's content.

**Type** str

**doc\_date**

The date of the document's creation in YYYYMMDD format.

Type `str`

**frontend\_doc\_date**

How the date appears when presented on the DA website.

Type `str`

**slug**

A url-friendly name for the document. Not currently used.

Type `str`

**source\_created\_at**

Timestamp of when the Document was first added to the DA.

Type `datetime.datetime`

**source\_updated\_at**

Timestamp of when the Document was last edited.

Type `datetime.datetime`

**first\_published\_at**

Timestamp of when the document was first made publically accessible.

Type `datetime.datetime`

**Attributes present only on hydrated Documents**

These attributes are aliases of `UnhydratedField` until `Document.hydrate()` is called on the Document.

**source**

The archive where the document was retrieved from.

Type `str`

**type**

The type of the document (meeting minutes, report, etc.)

Type `digitalarchive.models.Type`

**rights**

A list of entities holding the copyright of the Document.

Type `list of digitalarchive.models.Right`

**pdf\_generated\_at**

The date that the combined source, translations, and transcriptions PDF. was generated.

Type `str`

**date\_range\_start**

A rounded-down date used to standardize approximate dates for date-range matching.

Type `datetime.date`

**sort\_string\_by\_coverage**

An alphanumeric identifier used by the API to sort search results.

Type `str`

**main\_src**

The original Source that a Document was retrieved from.

**Type** `str`

**model**

The model of a record, used to differentiate collections and keywords in searches.

**Type** `str`

**donors**

A list of donors whose funding make the acquisition or translation of a document possible.

**Type** `list` of `digitalarchive.models.Donor`

**subjects**

A list of subjects that the document is tagged with.

**Type** `list` of `digitalarchive.models.Subject`

**transcripts**

A list of transcripts of the document's contents.

**Type** `list` of `digitalarchive.models.Transcript`

**translations**

A list of translations of the original document.

**Type** `list` of `digitalarchive.models.Translation`

**media\_files**

A list of attached original scans of the document.

**Type** `list` of `digitalarchive.models.MediaFile`

**languages**

A list of languages contained in the document.

**Type** `list` of `digitalarchive.models.Language`

**creators**

A list of persons who authored the document.

**Type** `list` of `digitalarchive.models.Creator`

**original\_coverages**

A list of geographic locations referenced in the document.

**Type** `list` of `digitalarchive.models.Coverage`

**collections**

A list of Collections that contain this document.

**Type** `list` of `digitalarchive.models.Collection`

**attachments**

A list of Documents that were attached to the Document.

**Type** `list` of `digitalarchive.models.Document`

**links**

A list of topically related documents.

**Type** `list` of `digitalarchive.models.Document`

**repositories**

A list of archives/libraries containing this document.

**Type** `list` of `digitalarchive.models.Repository`

**publishers**

A list of Publishers that released the document.

**Type** `list` of `digitalarchive.models.Publisher`

**classifications**

A list of security classification markings present on the document.

**Type** `list` of `digitalarchive.models.Publisher`

**hydrate** (*recurse: bool = False*)

Downloads the complete version of the Document with metadata for any related objects.

**Parameters** **recurse** (*bool*) – If true, also hydrate subordinate and related records.

**classmethod match** (*\*\*kwargs*) → `digitalarchive.matching.ResourceMatcher`

Search for a Document by keyword, or fetch one by ID.

Matching on the Document model runs a full-text search using keywords passed via the title and description keywords. Results can also be limited by dates or by related records, as described below.

---

**Note:** Title and description keywords are not searched for individually. All non-date or child record searches are concatenated to single querystring.

---

---

**Note:** Collection and other related record searches use *INNER JOIN* logic when passed multiple related resources.

---

**Allowed search fields:****Parameters**

- **title** (*str*, optional) – Title search keywords.
- **description** (*str*, optional) – Title search keywords.
- **start\_date** (*datetime.date*, optional) – Return only Documents with a *doc\_date* after the passed *start\_date*.
- **end\_date** (*datetime.date*, optional) – Return only Documents with a *doc\_date* before the passed *end\_date*.
- **collections** (*list* of `digitalarchive.models.Collection`, optional) – Restrict results to Documents contained in all of the passed Collections.
- **publishers** (*list* of `digitalarchive.models.Publisher`, optional) – Restrict results to Documents published by all of the passed Publishers.
- **repositories** (*list* of `digitalarchive.models.Repository`, optional) – Documents contained in all of the passed Repositories.
- **coverages** (*list* of `digitalarchive.models.Coverage`, optional) – relating to all of the passed geographical Coverages.
- **subjects** (*list* of `digitalarchive.models.Subject`) – all of the passed subjects
- **contributors** (*list* of `:class:`digitalarchive.models.Contributor`) – authors include all of the passed contributors.
- **donors** (*list*(`digitalarchive.models.Donor`)) – translated with support from all of the passed donors.

- **languages** (*digitalarchive.models.Language* or str) – original document. If passing a string, you must pass an ISO 639-2/B language code.
- **translation** (*digitalarchive.models.Translation*) – is a translation available in the passed Language.
- **theme** (*digitalarchive.models.Theme*) –

**Returns**

An instance of (*digitalarchive.matching.ResourceMatcher*) containing any records responsive to search.

```
class digitalarchive.models.Collection (id, name, slug, uri, parent, model, value, description, short_description, main_src, no_of_documents, is_inactive, source_created_at, source_updated_at, first_published_at)
```

A collection of Documents on a single topic

**name**

The title of the collection.

**Type** str

**slug**

A url-friendly name of the collection.

**Type** str

**uri**

The URI of the record on the DA API.

**Type** str

**parent**

A *Collection* containing the *Collection*.

**Type** *digitalarchive.models.Collection*

**model**

A sting name of the model used to differentiate *Collection* and *Document* searches in the DA API.

**Type** str

**value**

A string identical to the *title* field.

**Type** str

**description**

A 1-2 sentence description of the *Collection*'s content.

**Type** str

**short\_description**

A short description that appears in search views.

**Type** str

**main\_src**

Placeholder

**Type** str

**no\_of\_documents**

The count of documents contained in the collection.

**Type** `str`

**is\_inactive**

Whether the collection is displayed in the collections list.

**Type** `str`

**source\_created\_at**

Timestamp of when the Document was first added to the DA.

**Type** `datetime.datetime`

**source\_updated\_at**

Timestamp of when the Document was last edited.

**Type** `datetime.datetime`

**first\_published\_at**

Timestamp of when the document was first made publically accessible.

**Type** `datetime.datetime`

**hydrate()**

Populate all unhydrated fields of a resource.

**classmethod match** (*\*\*kwargs*) → `digitalarchive.matching.ResourceMatcher`

Find a resource using passed keyword arguments.

---

**Note:** If called without arguments, returns all records in the DA .

---

**class** `digitalarchive.models.Subject` (*id, name, uri, value*)

A historical topic to which documents can be related.

**id**

The ID of the record.

**Type** `str`

**name**

The name of the subject.

**Type** `str`

**value**

An alias for *name*.

**Type** `str`

**uri**

The URI for the Subject in the API.

**Type** `str`

**hydrate()**

Populate all unhydrated fields of a resource.

**classmethod match** (*\*\*kwargs*) → `digitalarchive.matching.ResourceMatcher`

Find a resource using passed keyword arguments.

---

**Note:** If called without arguments, returns all records in the DA .

---

**class** digitalarchive.models.**Coverage** (*id, name, uri, value, parent, children*)

A geographical area referenced by a Document.

**id**

The ID# of the geographic Coverage.

**Type** *str*

**name**

The name of geographic coverage area.

**Type** *str*

**value**

An alias to *name*.

**Type** *str*

**uri**

URI to the Coverage's metadata on the DA API.

**Type** *str*

**parent**

The parent coverage, if any

**Type** *Coverage*

**children**

(list of *Coverage*): Subordinate geographical areas, if any.

**hydrate** ()

Populate all unhydrated fields of a resource.

**classmethod** **match** (*\*\*kwargs*) → digitalarchive.matching.ResourceMatcher

Find a resource using passed keyword arguments.

---

**Note:** If called without arguments, returns all records in the DA .

---

**class** digitalarchive.models.**Contributor** (*id, name, value, uri*)

An individual person or organization that contributed to the creation of the document.

Contributors are typically the Document's author, but for meeting minutes and similar documents, a Contributor may simply be somebody who was in attendance at the meeting.

**id**

The ID# of the Contributor.

**Type** *str*

**name**

The name of the contributor.

**Type** *str*

**uri**

The URI of the contributor metadata on the DA API.

**Type** *str*

**hydrate** ()

Populate all unhydrated fields of a resource.

**classmethod** **match** (*\*\*kwargs*) → digitalarchive.matching.ResourceMatcher  
Find a resource using passed keyword arguments.

---

**Note:** If called without arguments, returns all records in the DA .

---

**class** digitalarchive.models.**Repository** (*id, name, value, uri*)  
The archive or library possessing the original, physical Document.

**id**  
The ID# of the Repository.

**Type** str

**name**  
The name of the repository

**Type** str

**uri**  
The URI for the Repository's metadata on the Digital Archive API.

**Type** str

**value**  
An alias to *name*

**Type** str

**hydrate** ()  
Populate all unhydrated fields of a resource.

**classmethod** **match** (*\*\*kwargs*) → digitalarchive.matching.ResourceMatcher  
Find a resource using passed keyword arguments.

---

**Note:** If called without arguments, returns all records in the DA .

---

## Hydrate-able Models

**class** digitalarchive.models.**Transcript** (*id, filename, content\_type, extension, asset\_id, source\_created\_at, source\_updated\_at, url, html, pdf, raw*)

A transcript of a document in its original language.

**id**  
The ID# of the Transcript.

**Type** str

**url**  
A URL to accessing the hydrated Transcript.

**Type** str

**html**  
The html of of the Transcript.

**Type** str



**pdf**

A bytes object of the Transcript pdf content.

**Type** `bytes`

**raw**

The raw content recieved from the DA API for the Transcript.

**Type** `str` or `bytes`

**filename**

The filename of the Transcript on the content server.

**Type** `str`

**content\_type**

The MIME type of the Transcript file.

**Type** `str`

**extension**

The file extension of the Transcript.

**Type** `str`

**asset\_id**

The Transcript's unique ID on the content server.

**Type** `str`

**source\_created\_at**

ISO 8601 timestamp of the first time the Translation was published.

**Type** `str`

**source\_updated\_at**

ISO 8601 timestamp of the last time the Translation was modified.

**Type** `str`

**hydrate()**

Populate all unhydrated fields of a `digitalarchive.models._Asset`.

```
class digitalarchive.models.Translation(id, filename, content_type, extension, asset_id,  
                                         source_created_at, source_updated_at, url, html,  
                                         pdf, raw, language)
```

A translation of a Document into a another language.

**id**

The ID# of the Translation.

**Type** `str`

**language**

**Type** `digitalarchive.models.Language`

**html**

The HTML-formatted text of the Translation.

**Type** `str`

**pdf**

A bytes object of the Translation pdf content.

**Type** `bytes`

**raw**

The raw content recieved from the DA API for the Translation.

**Type** `str` or `bytes`

**filename**

The filename of the Translation on the content server.

**Type** `str`

**content\_type**

The MIME type of the Translation file.

**Type** `str`

**extension**

The file extension of the Translation.

**Type** `str`

**asset\_id**

The Translation's unique ID on the content server.

**Type** `str`

**source\_created\_at**

ISO 8601 timestamp of the first time the Translation was published.

**Type** `str`

**source\_updated\_at**

ISO 8601 timestamp of the last time the Translation was modified.

**Type** `str`

**hydrate()**

Populate all unhydrated fields of a `digitalarchive.models._Asset`.

```
class digitalarchive.models.MediaFile(id, filename, content_type, extension, asset_id,  
                                       source_created_at, source_updated_at, path, html,  
                                       pdf, raw)
```

An original scan of a Document.

**id**

The ID# of the MediaFile.

**Type** `str`

**pdf**

A bytes object of the MediaFile content.

**Type** `bytes`

**raw**

The raw content received from the DA API for the MediaFile.

**Type** `str` or `bytes`

**filename**

The filename of the MediaFile on the content server.

**Type** `str`

**content\_type**

The MIME type of the MediaFile file.

**Type** `str`

**extension**

The file extension of the MediaFile.

**Type** `str`

**asset\_id**

The MediaFile's unique ID on the content server.

**Type** `str`

**source\_created\_at**

ISO 8601 timestamp of the first time the MediaFile was published.

**Type** `str`

**source\_updated\_at**

ISO 8601 timestamp of the last time the MediaFile was modified.

**Type** `str`

**hydrate()**

Populate all unhydrated fields of a `digitalarchive.models._Asset`.

**class** `digitalarchive.models.Theme` (*id, slug, title, value, description, main\_src, uri, featured\_resources, has\_map, has\_timeline, featured\_collections, dates\_with\_events*)

A parent container for collections on a single geopolitical topic.

---

**Note:** Themes never appear on any record model, but can be passed as a search param to Document.

---

**id**

The ID# of the Theme.

**Type** `str`

**slug**

A url-friendly version of the theme title.

**Type** `str`

**title**

The name of the Theme.

**Type** `str`

**description**

A short description of the Theme contents.

**Type** `str`

**main\_src**

A URI for the Theme's banner image on the Digital Archive website.

**has\_map**

A boolean value for whether the Theme has an accompanying map on the Digital Archive website.

**Type** `str`

**has\_timeline**

A boolean value for whether the Theme has a Timeline on the Digital Archive website.

**Type** `str`

**featured\_collections**

A list of related collections.

**Type** list of *Collection*

**dates\_with\_events**

A list of date ranges that the Theme has timeline entries for.

**Type** list

**hydrate()**

Populate all unhydrated fields of a resource.

## Other Models

**class** digitalarchive.models.**Language** (*id, name*)

The original language of a resource.

**id**

An ISO 639-2/B language code.

**Type** str

**name**

The ISO language name for the language.

**Type** str

**class** digitalarchive.models.**Donor** (*id, name*)

An entity whose resources helped publish or translate a document.

**id**

The ID# of the Donor.

**Type** str

**name**

The name of the Donor.

**Type** str

**class** digitalarchive.models.**Type** (*id, name*)

The type of a document (memo, report, etc).

**id**

The ID# of the Type.

**Type** str

**name**

The name of the resource Type.

**Type** str

**class** digitalarchive.models.**Right** (*id, name*)

A copyright notice attached to the Document.

**id**

The ID# of the Copyright type.

**Type** str

**name**

The name of the Copyright type.

```

        Type str
    rights
        A description of the copyright requirements.
        Type str
class digitalarchive.models.Classification(id, name)
    A classification marking applied to the original Document.
    id
        The ID# of the Classification type.
        Type str
    name
        A description of the Classification type.
        Type str
'''

```

### 4.3.2 Matching

```

class digitalarchive.matching.ResourceMatcher(resource_model: digitala-
                                             larchive.models.Resource,
                                             items_per_page=200, **kwargs)
    Runs a search against the DA API for the provided DA model and keywords.
    ResourceMatcher wraps search results and exposes methods for interacting with the resultant set of resources.
    list
        search results. Handles pagination of the DA API.
        Type Generator of digitalarchive.models.Resource
    count
        The number of respondant records to the given search.
    all() → List[digitalarchive.models.Resource]
        Exhaust the results generator and return a list of all search results.
    first() → digitalarchive.models.Resource
        Return only the first record from a search result.
    hydrate(recurse: bool = False)
        Hydrate all of the resources in a search result.

```



## A

`all()` (*digitalarchive.matching.ResourceMatcher* method), 25

`asset_id` (*digitalarchive.models.MediaFile* attribute), 23

`asset_id` (*digitalarchive.models.Transcript* attribute), 21

`asset_id` (*digitalarchive.models.Translation* attribute), 22

`attachments` (*digitalarchive.models.Document* attribute), 15

## C

`children` (*digitalarchive.models.Coverage* attribute), 19

`Classification` (class in *digitalarchive.models*), 25

`classifications` (*digitalarchive.models.Document* attribute), 16

`Collection` (class in *digitalarchive.models*), 17

`collections` (*digitalarchive.models.Document* attribute), 15

`content_type` (*digitalarchive.models.MediaFile* attribute), 22

`content_type` (*digitalarchive.models.Transcript* attribute), 21

`content_type` (*digitalarchive.models.Translation* attribute), 22

`Contributor` (class in *digitalarchive.models*), 19

`count` (*digitalarchive.matching.ResourceMatcher* attribute), 25

`Coverage` (class in *digitalarchive.models*), 18

`creators` (*digitalarchive.models.Document* attribute), 15

## D

`date_range_start` (*digitalarchive.models.Document* attribute), 14

`dates_with_events` (*digitalarchive.models.Theme* attribute), 24

`description` (*digitalarchive.models.Collection* attribute), 17

`description` (*digitalarchive.models.Document* attribute), 13

`description` (*digitalarchive.models.Theme* attribute), 23

`doc_date` (*digitalarchive.models.Document* attribute), 13

`Document` (class in *digitalarchive.models*), 13

`Donor` (class in *digitalarchive.models*), 24

`donors` (*digitalarchive.models.Document* attribute), 15

## E

`extension` (*digitalarchive.models.MediaFile* attribute), 23

`extension` (*digitalarchive.models.Transcript* attribute), 21

`extension` (*digitalarchive.models.Translation* attribute), 22

## F

`featured_collections` (*digitalarchive.models.Theme* attribute), 23

`filename` (*digitalarchive.models.MediaFile* attribute), 22

`filename` (*digitalarchive.models.Transcript* attribute), 21

`filename` (*digitalarchive.models.Translation* attribute), 22

`first()` (*digitalarchive.matching.ResourceMatcher* method), 25

`first_published_at` (*digitalarchive.models.Collection* attribute), 18

`first_published_at` (*digitalarchive.models.Document* attribute), 14

`frontend_doc_date` (*digitalarchive.models.Document* attribute), 14

## H

`has_map` (*digitalarchive.models.Theme* attribute), 23

has\_timeline (*digitalarchive.models.Theme* attribute), 23  
 html (*digitalarchive.models.Transcript* attribute), 20  
 html (*digitalarchive.models.Translation* attribute), 21  
 hydrate() (*digitalarchive.matching.ResourceMatcher* method), 25  
 hydrate() (*digitalarchive.models.Collection* method), 18  
 hydrate() (*digitalarchive.models.Contributor* method), 19  
 hydrate() (*digitalarchive.models.Coverage* method), 19  
 hydrate() (*digitalarchive.models.Document* method), 16  
 hydrate() (*digitalarchive.models.MediaFile* method), 23  
 hydrate() (*digitalarchive.models.Repository* method), 20  
 hydrate() (*digitalarchive.models.Subject* method), 18  
 hydrate() (*digitalarchive.models.Theme* method), 24  
 hydrate() (*digitalarchive.models.Transcript* method), 21  
 hydrate() (*digitalarchive.models.Translation* method), 22

## I

id (*digitalarchive.models.Classification* attribute), 25  
 id (*digitalarchive.models.Contributor* attribute), 19  
 id (*digitalarchive.models.Coverage* attribute), 19  
 id (*digitalarchive.models.Document* attribute), 13  
 id (*digitalarchive.models.Donor* attribute), 24  
 id (*digitalarchive.models.Language* attribute), 24  
 id (*digitalarchive.models.MediaFile* attribute), 22  
 id (*digitalarchive.models.Repository* attribute), 20  
 id (*digitalarchive.models.Right* attribute), 24  
 id (*digitalarchive.models.Subject* attribute), 18  
 id (*digitalarchive.models.Theme* attribute), 23  
 id (*digitalarchive.models.Transcript* attribute), 20  
 id (*digitalarchive.models.Translation* attribute), 21  
 id (*digitalarchive.models.Type* attribute), 24  
 is\_inactive (*digitalarchive.models.Collection* attribute), 18

## L

Language (class in *digitalarchive.models*), 24  
 language (*digitalarchive.models.Translation* attribute), 21  
 languages (*digitalarchive.models.Document* attribute), 15  
 links (*digitalarchive.models.Document* attribute), 15  
 list (*digitalarchive.matching.ResourceMatcher* attribute), 25

## M

main\_src (*digitalarchive.models.Collection* attribute), 17  
 main\_src (*digitalarchive.models.Document* attribute), 14  
 main\_src (*digitalarchive.models.Theme* attribute), 23  
 match() (*digitalarchive.models.Collection* class method), 18  
 match() (*digitalarchive.models.Contributor* class method), 19  
 match() (*digitalarchive.models.Coverage* class method), 19  
 match() (*digitalarchive.models.Document* class method), 16  
 match() (*digitalarchive.models.Repository* class method), 20  
 match() (*digitalarchive.models.Subject* class method), 18  
 media\_files (*digitalarchive.models.Document* attribute), 15

MediaFile (class in *digitalarchive.models*), 22

model (*digitalarchive.models.Collection* attribute), 17

model (*digitalarchive.models.Document* attribute), 15

## N

name (*digitalarchive.models.Classification* attribute), 25  
 name (*digitalarchive.models.Collection* attribute), 17  
 name (*digitalarchive.models.Contributor* attribute), 19  
 name (*digitalarchive.models.Coverage* attribute), 19  
 name (*digitalarchive.models.Donor* attribute), 24  
 name (*digitalarchive.models.Language* attribute), 24  
 name (*digitalarchive.models.Repository* attribute), 20  
 name (*digitalarchive.models.Right* attribute), 24  
 name (*digitalarchive.models.Subject* attribute), 18  
 name (*digitalarchive.models.Type* attribute), 24  
 no\_of\_documents (*digitalarchive.models.Collection* attribute), 17

## O

original\_coverages (*digitalarchive.models.Document* attribute), 15

## P

parent (*digitalarchive.models.Collection* attribute), 17  
 parent (*digitalarchive.models.Coverage* attribute), 19  
 pdf (*digitalarchive.models.MediaFile* attribute), 22  
 pdf (*digitalarchive.models.Transcript* attribute), 20  
 pdf (*digitalarchive.models.Translation* attribute), 21  
 pdf\_generated\_at (*digitalarchive.models.Document* attribute), 14  
 publishers (*digitalarchive.models.Document* attribute), 15



## R

raw (*digitalarchive.models.MediaFile* attribute), 22  
 raw (*digitalarchive.models.Transcript* attribute), 21  
 raw (*digitalarchive.models.Translation* attribute), 21  
 Repository (*class in digitalarchive.models*), 20  
 ResourceMatcher (*class in digitalarchive.matching*), 25  
 repositories (*digitalarchive.models.Document* attribute), 15  
 Right (*class in digitalarchive.models*), 24  
 rights (*digitalarchive.models.Document* attribute), 14  
 rights (*digitalarchive.models.Right* attribute), 25

## S

short\_description (*digitalarchive.models.Collection* attribute), 17  
 slug (*digitalarchive.models.Collection* attribute), 17  
 slug (*digitalarchive.models.Document* attribute), 14  
 slug (*digitalarchive.models.Theme* attribute), 23  
 sort\_string\_by\_coverage (*digitalarchive.models.Document* attribute), 14  
 source (*digitalarchive.models.Document* attribute), 14  
 source\_created\_at (*digitalarchive.models.Collection* attribute), 18  
 source\_created\_at (*digitalarchive.models.Document* attribute), 14  
 source\_created\_at (*digitalarchive.models.MediaFile* attribute), 23  
 source\_created\_at (*digitalarchive.models.Transcript* attribute), 21  
 source\_created\_at (*digitalarchive.models.Translation* attribute), 22  
 source\_updated\_at (*digitalarchive.models.Collection* attribute), 18  
 source\_updated\_at (*digitalarchive.models.Document* attribute), 14  
 source\_updated\_at (*digitalarchive.models.MediaFile* attribute), 23  
 source\_updated\_at (*digitalarchive.models.Transcript* attribute), 21  
 source\_updated\_at (*digitalarchive.models.Translation* attribute), 22  
 Subject (*class in digitalarchive.models*), 18  
 subjects (*digitalarchive.models.Document* attribute), 15

## T

Theme (*class in digitalarchive.models*), 23  
 title (*digitalarchive.models.Document* attribute), 13  
 title (*digitalarchive.models.Theme* attribute), 23  
 Transcript (*class in digitalarchive.models*), 20  
 transcripts (*digitalarchive.models.Document* attribute), 15  
 Translation (*class in digitalarchive.models*), 21

translations (*digitalarchive.models.Document* attribute), 15  
 Type (*class in digitalarchive.models*), 24  
 type (*digitalarchive.models.Document* attribute), 14

## U

uri (*digitalarchive.models.Collection* attribute), 17  
 uri (*digitalarchive.models.Contributor* attribute), 19  
 uri (*digitalarchive.models.Coverage* attribute), 19  
 uri (*digitalarchive.models.Repository* attribute), 20  
 uri (*digitalarchive.models.Subject* attribute), 18  
 url (*digitalarchive.models.Transcript* attribute), 20

## V

value (*digitalarchive.models.Collection* attribute), 17  
 value (*digitalarchive.models.Coverage* attribute), 19  
 value (*digitalarchive.models.Repository* attribute), 20  
 value (*digitalarchive.models.Subject* attribute), 18