# namedropper Documentation

***Release 0.4.0-dev***

**Emory University Libraries**

June 14, 2013

# CONTENTS

Python scripts and utilities for looking up names and linking them to authoritative identifiers. See the top-level namedropper repository for more information.

Currently uses DBpedia Spotlight for recognition of named entities in text, with support for matching identified DBpedia resources (currently only for Persons) with the equivalent resource in VIAF (Virtual International Authority File).

# CHAPTER ONE

# INSTALLATION

We recommend the use of pip to install the latest released version of this package and its dependencies:

```
pip install namedropper
```

This will also make the `lookup-names` and `count-nametags` scripts available.

More detailed documentation, including script usage information, is available at http://namedropper.readthedocs.org/

CHAPTER

TWO

# LICENSE

NameDropper Python module and scripts are distributed under the Apache 2.0 License.

## 2.1 Developer notes

To install dependencies for your local check out of the code, run `pip install` in the `namedropper-py` directory (the use of virtualenv is recommended):

```
pip install -e .
```

If you want to run unit tests or build sphinx documentation, you will also need to install development dependencies:

```
pip install namedropper[dev]
```

To run all unit tests:

```
nosetests   # for normal development
nosetests --with-coverage --cover-package=namedropper --cover-xml --with-xunit   # for continuous int
```

To run unit tests for a specific module, use syntax like this:

```
nosetests test/test_spotlight.py
```

To generate sphinx documentation:

```
cd doc
make html
```

## 2.2 Contents

### 2.2.1 `namedropper` Code Documentation for NameDropper

#### spotlight

**class** `namedropper.spotlight.`**`DBpediaResource`**(*uri*, *language=’en’*, *spotlight_info={}*)

An object to encapsulate properties and functionality related to a specific dbpedia item.

**Parameters**

- **uri** – dbpedia resource uri
- **language** – optional language code, for multilingual properties like label; defaults to ‘en’

- **spotlight_info** – optional dictionary of data returned from spotlight annotation, to avoid unnecessary look-ups (e.g., for type of resource)

**type**
high-level type of resource; currently only supports person, place, and organization.

namedropper.spotlight.**OWL = Namespace(u'http://www.w3.org/2002/07/owl#')**
rdflib.Namespace for OWL (Web Ontology Language)

**class** namedropper.spotlight.**SpotlightClient**(*base_url=None*, *confidence=None*, *support=None*, *types=None*)
Client for interacting with DBpedia Spotlight via REST API.

http://wiki.dbpedia.org/spotlight/usersmanual?v=ssd

**Parameters**

- **base_url** – Base URL for DBpedia Spotlight webservice, when not using the hosted service at default_url
- **confidence** – default minimum confidence score (optional)
- **support** – default minimum support score (optional)
- **types** – list or string of default types to be returned when recognizing and annotating text

**annotate**(*text*, *confidence=None*, *support=None*, *types=None*)
Call the DBpedia Spotlight annotate service.

All arguments other than text are optional; if default configurations were specified when the client was initialized, those will be used unless an overriding value is specified here.

**Parameters**

- **text** – text string to be annotated
- **confidence** – minimum confidence score (e.g., 0.5) [optional]
- **support** – minimum support score [optional]
- **types** – list or string of entity types that should be recognized and returned (such as Person, Place, Organization) [optional]

**Returns** dict with information on identified resources

**default_url = 'http://spotlight.dbpedia.org/rest'**
Default base url for DBpedia Spotlight web service

**total_api_calls**
number of API calls made

**total_api_duration**
datetime.timedelta - total duration of all API calls

namedropper.spotlight.**cached_property**(*f*)
returns a cached property that is calculated by function f

## viaf

**class** namedropper.viaf.**ViafClient**
Client for interacting with VIAF (Virtual International Authority File) API.

http://www.oclc.org/developer/documentation/virtual-international-authority-file-viaf/using-api

**autosuggest**(*term*)
Query autosuggest API. Returns a list of results.

**find_corporate**(*name*)
Search VIAF by local.corporateNames

**find_person**(*name*)
Search VIAF by local.personalNames

**find_place**(*name*)
Search VIAF by local.geographicNames

**search**(*query*)
Query VIAF seach interface. Returns a list of feed entries, as parsed by `feedparser`.

**Parameters** **query** – CQL query in viaf syntax (e.g., `cql.any all "term"`)

## util

**class** `namedropper.util.`**AnnotateXml**(*mode*, *viaf=False*, *geonames=False*, *track_changes=False*, *xml_object=None*)
Annotate xml based on dbpedia spotlight annotation results.

Currently using `logging` (info and warn) when VIAF look-up fails or attributes are not inserted to avoid overwriting existing values.

When track changes is requested, processing instructions will be added around annotated names for review in OxygenXML 14.2+. In cases where a name was untagged, the text will be marked as a deletion and the tagged version of the name will be marked as an insertion with a comment containing the description of the DBpedia resource, to aid in identifying whether the correct resource has been added. If a recognized name was previously tagged, a comment will be added indicating what attributes were added, or would have been added if they did not conflict with attributes already present in the document.

When using the track changes option, it is recommended to also run meth:*enable_oxygen_track_changes* once on the document, so that Oxygen will automatically open the document with track changes turned on.

**Parameters**

- **mode** – mode (tei or ead) of tags to insert
- **viaf** – if True, convert DBpedia person URIs to VIAF URIs when possible (optional, defaults to False)
- **geonames** – if True, convert DBpedia person URIs to GeoNames.org URIs when possible (optional, defaults to False)
- **track_changes** – if True, flag annotations with OxygenXML track changes processing instructions for later review
- **xml_object** – `eulxml.xmlmap.XmlObject` for the top-level XML document associated with the node(s) to be annotated. Used for validation to check that inserted elements are allowed.

**annotate**(*node*, *annotations*)
Annotate xml based on dbpedia spotlight annotation results. Assumes that dbpedia annotate was called on the **normalized** text from this node. Currently updates the node that is passed in; whitespace will be normalized in text nodes where name tags are inserted. For TEI, DBpedia URIs are inserted as **ref** attributes; since EAD does not support referencing URIs, VIAF ids will be used where possible (currently only supports lookup for personal names).

If recognized names are already tagged as names in the existing XML, no new name tag will be inserted; attributes will only be added if they are not present in the original node.

**Parameters**

- **node** – lxml element node to be updated
- **annotations** – dbpedia spotlight result, as returned by `namedropper.spotlight.SpotlightClient.annotate()`

**Returns** total count of the number of entities inserted into the xml

**`geonames`** = **False**

GeoNames flag: if true, annotate will convert dbpedia place URIs to GeoNames URIs when possible

**`get_attributes`**(*res*, *quiet=False*)

Get the attributes to be inserted, based on the current document mode and the type of DBpediaResource.

**Parameters** **res** – `namedropper.spotlight.DBpediaResource`

**Returns** dictionary of attribute names -> values

**`get_tag`**(*res*)

Get the name of the tag to be inserted, based on the current document mode and the type of DBpedia resource.

**Parameters** **res** – `namedropper.spotlight.DBpediaResource` instance for the tag to be inserted

**Returns** string tag

**`track_changes`** = **False**

OxygenXML track changes flag: if true, annotation will be tagged with OxygenXML track changes processing instruction, to enable review within Oxygen Author mode

**`viaf`** = **False**

VIAF flag: if true, annotate will convert dbpedia person URIs to VIAF URIs when possible

`namedropper.util.`**`OLDannotate_xml`**(*node*, *result*, *mode='tei'*, *track_changes=False*)

Annotate xml based on dbpedia spotlight annotation results. Assumes that dbpedia annotate was called on the **normalized** text from this node. Currently updates the node that is passed in; whitespace will be normalized in text nodes where name tags are inserted. For TEI, DBpedia URIs are inserted as **ref** attributes; since EAD does not support referencing URIs, VIAF ids will be used where possible (currently only supports lookup for personal names).

If recognized names are already tagged as names in the existing XML, no new name tag will be inserted; attributes will only be added if they are not present in the original node.

Currently using `logging` (info and warn) when VIAF look-up fails or attributes are not inserted to avoid overwriting existing values.

When track changes is requested, processing instructions will be added around annotated names for review in OxygenXML 14.2+. In cases where a name was untagged, the text will be marked as a deletion and the tagged version of the name will be marked as an insertion with a comment containing the description of the DBpedia resource, to aid in identifying whether the correct resource has been added. If a recognized name was previously tagged, a comment will be added indicating what attributes were added, or would have been added if they did not conflict with attributes already present in the document. When using the track changes option, it is recommended to also run meth:*enable_oxygen_track_changes* once on the document, so that Oxygen will automatically open the document with track changes turned on.

**Parameters**

- **node** – lxml element node to be updated

- **result** – dbpedia spotlight result, as returned by `namedropper.spotlight.SpotlightClient.annotate()`
- **track_changes** – mark changes using OxygenXML track changes processing instructions, to enable review in OxygenXML author mode

**Returns** total count of the number of entities inserted into the xml

`namedropper.util.`**`autodetect_file_type`**`(`*filename*`)`

Attempt to auto-detect input file type. Currently supported types are EAD XML, TEI XML, or text. Any document that cannot be loaded as XML is assumed to be text.

**Returns** "tei", "ead", "text", or None if file type is not recognized

`namedropper.util.`**`enable_oxygen_track_changes`**`(`*node*`)`

Add a processing instruction to a document with an OxygenXMl option to enable the track changes mode.

`namedropper.util.`**`normalize_whitespace`**`(`*txt*, *next=None*, *prev=None*`)`

Normalize whitespace in a string to match the logic of `normalize-space()` in XPath. Replaces all internal sequences of white space with a single space and conditionally removes leading and trailing whitespace.

**Parameters**

- **txt** – text string to be normalized
- **next** – optional next string; used to determine if trailing whitespace should be removed
- **prev** – optional preceding string; used to determine if leading whitepace should be removed

## scripts

**class** `namedropper.scripts.`**`ScriptBase`**

Base class for namedropper command-line scripts.

**Directions for use:**

- script description, to be displayed via argparse, should be set as class doctring
- extend `init_parser()` for additional command-line arguments
- extend `run()` with main script functionality

Init method will initialize the argument parser, parse command-line arguments, check that file type is either specified or can be auto-detected, and execute `run()`.

Parser is saved as `parser`, in case other script logic needs reference to it.

**`init_parser`**`()`

Initialize an argument parser with common arguments. Currently includes filename and input type.

Extend to add arguments.

**`init_xml_object`**`()`

Initialize an xmlobject based on user-specified arguments for filename and type. Returns an instance of the appropriate `XmlObject`, or displays an error message if the document could not be parsed as XML.

**`parser`** **= None**

`argparse.ArgumentParser` instance to be initialized by `init_parser()` at class instantiation.

**`run`**`()`

placeholder method - extend with script logic

### 2.2.2 Scripts

namedropper command-line scripts

#### lookup-names

**lookup-names** is a command-line script which uses a remote service (currently DBpedia Spotlight) to identify and report on named resources, such as Persons, Places, and Organizations which are identified in the content of a plain text file, targeted sections of an Encoded Archival Description (EAD) XML document, or user-specified sections of a Text Encoding Initiative (TEI) XML document. The script will attempt to auto-detect the input document type (EAD, TEI, or plain text), but you can always explicitly set the document type with the `--input` option.

Example usage:

```
$ lookup-names /path/to/my/textfile.txt
```

```
$ lookup-names findingaid.xml
```

To restrict the type of resources you want returned, or to adjust the minimum confidence and support scores (e.g., to filter out erroneous matches), you can pass specify additional parameters to be passed to the Spotlight service; for example, to require a confidence score of at least 0.5, a support score of at least 40, and restrict identified types to only Persons, Places, and Organizations:

```
$ lookup-names findingaid.xml --confidence 0.5 --support 40 --types "Person,Place,Organisation"
```

Because TEI document structure is so variable, when examining a TEI document the script requires that you specify an XPath for the section of the document which you want looked up. When running in section-output (non-unique) mode, the script will display the first `docTitle` or `head` under the specified node as a section heading (currently only supports headings in the TEI namespace). The script currently considers XPath elements with the prefix `t:` to be in the TEI namespace. Some examples of using the script for different TEI documents:

```
$ lookup-names essay.xml --tei-xpath t:text/t:body/t:div
$ lookup-names newspaper_issue.xml --tei-xpath t:text/t:body/t:div1/t:div2
$ lookup-names text_group_document.xml --tei-xpath //t:group/t:group/t:text
```

To see a unique, sorted list of names and corresponding resource URIs in any input mode, use the `--unique` option.

To see the scores for items returned by DBpedia Spotlight (e.g., in order to tune the confidence and support options for a given corpus), use the `--scores` option (cannot be used with `--unique`).

**Note:** For large EAD documents, running this script is currently quite slow due to DBpedia Spotlight response times and the fact that individual paragraphs and container list descriptions are annotated individually.

##### CSV output

Use the `--csv` option to generate a CSV file with more detailed information about recognized names. Along with the recognized text and the DBpedia URI, this includes the support scores and some document context, to allow inspecting the results for accuracy and determining a good support score setting for your content.

**Note:** When opening the CSV output file in Excel, you should open it as Unicode UTF-8.

#### Generate XML with tagged names

To generate a new version of your EAD or TEI document with identified resources tagged, use the `--output` (or `-o`) option and specify the name of the filename where you want the new version to be saved. Note that this feature is somewhat experimental. Before using it, you should first run the script and output the DBpedia annotation scores so that you can fine-tune the results to exclude as many as bogus results as you can (e.g., by increasing the minimum support score). It is also recommended to restrict the types to persons, places, and organizations (i.e., use `--types Person,Place,Organisation`). You should, of course, carefully review changes made to the output file before accepting or using them.

**Warning:** A web proxy is currently **required** for generating an output XML file.

**Note:** Due to the limitations of the software (DBpedia -> VIAF lookup is currently only implemented for personal names) and EAD name tags, which have attributes for authority control numbers (such as VIAF), but cannot reference a resource URI such as a DBpedia reference, non-personal names tagged in EAD will currently be added without any identifier or reference to the entity returned by DBpedia Spotlight.

#### Generate XML with tagged names and OxygenXML track changes

If you use the `--oxygen-track-changes` flag when generating XML output, the resulting document will include OxygenXML 14.2+ track changes processing instructions, to allow for easier review and acceptance or rejection of the changes made. In cases where a name was untagged, the text will be marked as a deletion and the tagged version of the name will be marked as an insertion with a comment containing the description of the DBpedia resource, to aid in identifying whether the correct resource has been added. If a recognized name was previously tagged, a comment will be added indicating what attributes were added, or would have been added if they did not conflict with attributes already present in the document.

### count-nametags

**count-nametags** is a command-line script for reporting on the number of tagged personal, corporate, and geographic names in a document.

For EAD documents, the script reports on the total number of `persname`, `corpname`, and `geogname` tags in the specified document and the number of name tags with authority control, both the total by source and the number of unique identifiers for each type of name.

**Note:** Currently only EAD documents are supported.

## 2.2.3 Change & Version Information

The following is a summary of changes and improvements to `namedropper`. New features in each version should be listed, with any necessary information about installation or upgrade notes.

### 0.3.1

- Corrected CSV output of **lookup-names** script, which was broken in in 0.3.0.

### 0.3

- New script **count-nametags**
  - A user can run a script to get summary information about the number of tagged names in an EAD document, in order to do simple comparison of tagged and untagged documents.
- Updates to **lookup-names** script
  - When a user runs the lookup-names script to generate a CSV file, the resulting output includes resource type for person, place, or organization so that results can be filtered and organized by broad types.
  - When users interrupts the lookup-names script while it is running, it stops processing gracefully and reports on what was done so that user can get an idea of the output without waiting for the script to complete on a long document.
  - When a user runs the lookup names script with options that generate no results, the script does not create a CSV file or an enhanced xml file (even if those options were specified) and prints a message explaining why, so that the user is not confused by empty or unchanged files.
  - When users run the lookup-names script to generate annotated XML, they can optionally add tags with Oxygen history tracking comments so that changes can be reviewed and accepted or rejected in Oxygen.
  - Bug fix: When a user runs a lookup-names script on an XML file that does not have all of its component parts, it should not crash.
  - Bug fix: When annotating XML, the script will no longer crash if –types is not restricted to Person,Place,Organisation (or some subset of those three), and will warn about recognized entities that cannot be inserted into the output XML.
  - Bug fix: When annotating XML, tags will not be inserted where they are not schema valid (schema validation currently only supported for EAD).
  - Bug fix: If output XML is requested but an HTTP Proxy is not configured, the script will halt and information about setting a proxy, instead of crashing when attempting to validate the output XML.

### 0.2.1

- Normalize whitespace for text context when generating CSV output (primarily affects plain-text input).

### 0.2

- A command-line user running the lookup-names script can have the input document type auto-detected, so they don't have to specify an input type every time they use the script.
- A command line user can run a script to look up recognized person names from a TEI or EAD XML document in a name authority system so that recognized names can be linked to other data.
- A command line user can run a script to generate a new version of an EAD XML document with tagged named entities, in order to automatically link mentioned entities to other data sources.
- A command line user can run a script to generate a new version of a TEI XML document with tagged named entities, in order to automatically link mentioned entities to other data sources.
- A command line user can optionally export identified resources and associated data to a CSV file, so they can review the results in more detail.

### 0.1

- New script **lookup-names**
  - A command line user can run a script to output recognized names in an EAD XML document in order to evaluate automated name recognition and disambiguation.
  - A command line user can run a script to output recognized names in a TEI XML document in order to evaluate automated name recognition and disambiguation.

## 2.3 Indices and tables

- *genindex*
- *modindex*
- *search*

# PYTHON MODULE INDEX