



Documentation

Release 0.4

Mathias Loesch

June 03, 2015

1	Important Links	3
2	Table of Contents	5
2.1	Installation	5
2.2	Tutorial	5
2.3	OAI-PMH Primer	7
2.4	API	8
2.5	Harvesting other Metadata Formats than OAI-DC	13
2.6	Development	13
2.7	Credits	13

Sickle is a lightweight [OAI-PMH](#) client library written in Python. It has been designed for conveniently retrieving data from OAI interfaces the Pythonic way:

```
>>> sickle = Sickle('http://elis.da.ulcc.ac.uk/cgi/oai2')
>>> records = sickle.ListRecords(metadataPrefix='oai_dc')
>>> records.next()
<Record oai:eprints.rclis.org:4088>
```

Sickle maps all important OAI items to Python objects:

```
>>> record.header
<Header oai:eprints.rclis.org:4088>
>>> record.header.identifier
'oai:eprints.rclis.org:4088'
```

Dublin-Core-encoded metadata payloads are easily accessible as dictionaries:

```
>>> record.metadata
{'creator': ['Melloni, Marco'],
 'date': ['2000'],
 'description': [u'A web site for...']}
```

Important Links

- [Sickle @ PyPI](#)
- [Sickle @ GitHub](#)

Table of Contents

2.1 Installation

Sickle requires `requests` and `lxml`.

Installation using `pip`:

```
pip install sickle
```

Installation using `easy_install`:

```
easy_install sickle
```

2.2 Tutorial

This section gives a brief overview on how to use Sickle for querying OAI interfaces.

2.2.1 Initialize an OAI Interface

To make a connection to an OAI interface, you need to import the Sickle object:

```
>>> from sickle import Sickle
```

Next, you can initialize the connection by passing it the basic URL. In our example, we use the OAI interface of the ELIS repository:

```
>>> sickle = Sickle('http://elis.da.ulcc.ac.uk/cgi/oai2')
```

2.2.2 Issuing Requests

Sickle provides methods for each of the six OAI verbs (`ListRecords`, `GetRecord`, `Identify`, `ListSets`, `ListMetadataFormats`, `ListIdentifiers`). Start with a `ListRecords` request:

```
>>> records = sickle.ListRecords(metadataPrefix='oai_dc')
```

Note that all keyword arguments you provide to this function are passed to the OAI interface as HTTP parameters. Therefore the example request would send the parameters `verb=ListRecords&metadataPrefix=oai_dc`. We can add additional parameters, like, for example, an OAI set:

```
>>> records = sickle.ListRecords(metadataPrefix='oai_dc', set='driver')
```

2.2.3 Consecutive Harvesting

Since most OAI verbs yield more than one element, their respective Sickle methods return iterator objects which can be used to iterate over the records of a repository:

```
>>> records = sickle.ListRecords(metadataPrefix='oai_dc')
>>> records.next()
<Record oai:eprints.rclis.org:4088>
```

Note that this works with all verbs that return more than one element. These are: `ListRecords()`, `ListIdentifiers()`, `ListSets()`, and `ListMetadataFormats()`.

The following example shows how to iterate over the headers returned by `ListIdentifiers`:

```
>>> headers = sickle.ListIdentifiers(metadataPrefix='oai_dc')
>>> headers.next()
<Header oai:eprints.rclis.org:4088>
```

Iterating over the the sets returned by `ListSets` works similarly:

```
>>> sets = sickle.ListSets()
>>> sets.next()
<Set Status = In Press>
```

2.2.4 Using the from Parameter

If you need to perform selective harvesting by date using the `from` parameter, you may face the problem that `from` is a reserved word in Python:

```
>>> records = sickle.ListRecords(metadataPrefix='oai_dc', from="2012-12-12")
File "<stdin>", line 1
    records = sickle.ListRecords(metadataPrefix='oai_dc', from="2012-12-12")
                                                         ^
SyntaxError: invalid syntax
```

Fortunately, you can circumvent this problem by using a dictionary together with the `**` operator:

```
>>> records = sickle.ListRecords(
...     **{'metadataPrefix': 'oai_dc',
...     'from': '2012-12-12'
...     })
```

2.2.5 Getting a Single Record

OAI-PMH allows you to get a single record by using the `GetRecord` verb:

```
>>> sickle.GetRecord(identifier='oai:eprints.rclis.org:4088',
...     metadataPrefix='oai_dc')
<Record oai:eprints.rclis.org:4088>
```

2.2.6 Harvesting OAI Items vs. OAI Responses

Sickle supports two harvesting modes that differ in the type of the returned objects. The default mode returns OAI-specific *items* (records, headers etc.) encoded as Python objects as seen earlier. If you want to save the whole XML response returned by the server, you have to pass the `sickle.iterator.OAIResponseIterator` during the instantiation of the `Sickle` object:

```
>>> sickle = Sickle('http://elis.da.ulcc.ac.uk/cgi/oai2', iterator=OAIResponseIterator)
>>> responses = Sickle.ListRecords(metadataPrefix='oai_dc')
>>> responses.next()
<OAIResponse ListRecords>
```

You could then save the returned responses to disk:

```
>>> with open('response.xml', 'w') as fp:
...     fp.write(responses.next().raw.encode('utf8'))
```

2.2.7 Ignoring Deleted Records

The `ListRecords()` and `ListIdentifiers()` methods accept an optional parameter `ignore_deleted`. If set to `True`, the returned `OAIItemIterator` will skip deleted records/headers:

```
>>> records = sickle.ListRecords(metadataPrefix='oai_dc', ignore_deleted=True)
```

Note: This works only using the `sickle.iterator.OAIItemIterator`. If you use the `sickle.iterator.OAIResponseIterator`, the resulting OAI responses will still contain the deleted records.

2.3 OAI-PMH Primer

This section gives a basic overview of the [Open Archives Protocol for Metadata Harvesting \(OAI-PMH\)](#). For more detailed information, please refer to the protocol specification.

2.3.1 Glossary of Important OAI-PMH Concepts

Repository A *repository* is a server-side application that exposes metadata via OAI-PMH.

Harvester OAI-PMH client applications like Sickle are called *harvesters*.

record A *record* is the XML-encoded container for the metadata of a single publication item. It consists of a *header* and a *metadata* section.

header The record *header* contains a unique identifier and a datestamp.

metadata The record *metadata* contains the publication metadata in a defined metadata format.

set A structure for grouping records for selective harvesting.

harvesting The process of requesting records from the repository by the harvester.

2.3.2 OAI Verbs

OAI-PMH features six main API methods (so-called “OAI verbs”) that can be issued by harvesters. Some verbs can be combined with further arguments:

Identify Returns information about the repository. Arguments: None.

GetRecord Returns a single record. Arguments:

- `identifier` (the unique identifier of the record, *required*)
- `metadataPrefix` (the prefix identifying the metadata format, *required*)

ListRecords Returns the records in the repository in batches (possibly filtered by a timestamp or a *set*). Arguments:

- `metadataPrefix` (the prefix identifying the metadata format, *required*)

- `from` (the earliest timestamp of the records, *optional*)
- `until` (the latest timestamp of the records, *optional*)
- `set` (a set for selective harvesting, *optional*)
- `resumptionToken` (used for getting the next result batch if the number of records returned by the previous request exceeds the repository's maximum batch size, *exclusive*)

ListIdentifiers Like `ListRecords` but returns only the record headers.

ListSets Returns the list of sets supported by this repository. Arguments: None

ListMetadataFormats Returns the list of metadata formats supported by this repository. Arguments: None

2.3.3 Metadata Formats

OAI interfaces may expose metadata records in multiple metadata formats. These formats are identified by so-called “metadata prefixes”. For instance, the prefix `oai_dc` refers to the OAI-DC format, which by definition has to be exposed by every valid OAI interface. OAI-DC is based on the 15 metadata elements specified in the [Dublin Core Metadata Element Set](#).

Note: Sickle only supports the OAI-DC format out of the box. See the section on [customizing](#) for information on how to extend Sickle for retrieving metadata in other formats.

2.4 API

2.4.1 The Sickle Client

```
class sickle.app.Sickle(endpoint, http_method='GET', protocol_version='2.0', iterator=<class  
    'sickle.iterator.OAIItemIterator'>, max_retries=5, timeout=None,  
    class_mapping=None, auth=None)
```

Client for harvesting OAI interfaces.

Use it like this:

```
>>> sickle = Sickle('http://elis.da.ulcc.ac.uk/cgi/oai2')  
>>> records = sickle.ListRecords(metadataPrefix='oai_dc')  
>>> records.next()  
<Record oai:eprints.rclis.org:3780>
```

Parameters

- **endpoint** (*str*) – The endpoint of the OAI interface.
- **http_method** (*str*) – Method used for requests (GET or POST, default: GET).
- **protocol_version** (*str*) – The OAI protocol version.
- **iterator** – The type of the returned iterator (default: `sickle.iterator.OAIItemIterator`)
- **max_retries** (*int*) – Number of retries if HTTP request fails.
- **timeout** (*int*) – Timeout for HTTP requests.
- **class_mapping** (*dict*) – A dictionary that maps OAI verbs to classes representing OAI items. If not provided, `sickle.app.DEFAULT_CLASS_MAPPING` will be used.
- **auth** (*tuple*) – An optional tuple ('username', 'password') for accessing protected OAI interfaces.

last_response

Contains the last response that has been received.

GetRecord (***kwargs*)

Issue a ListSets request.

Identify ()

Issue an Identify request.

Return type `sickle.models.Identify`

ListIdentifiers (*ignore_deleted=False, **kwargs*)

Issue a ListIdentifiers request.

Parameters **ignore_deleted** – If set to `True`, the resulting iterator will skip records flagged as deleted.

Return type `sickle.iterator.BaseOAIIterator`

ListMetadataFormats (***kwargs*)

Issue a ListMetadataFormats request.

Return type `sickle.iterator.BaseOAIIterator`

ListRecords (*ignore_deleted=False, **kwargs*)

Issue a ListRecords request.

Parameters **ignore_deleted** – If set to `True`, the resulting iterator will skip records flagged as deleted.

Return type `sickle.iterator.BaseOAIIterator`

ListSets (***kwargs*)

Issue a ListSets request.

Return type `sickle.iterator.BaseOAIIterator`

harvest (***kwargs*)

Make HTTP requests to the OAI server.

Parameters **kwargs** – OAI HTTP parameters.

Return type `sickle.OAIResponse`

2.4.2 Working with OAI Responses

class `sickle.response.OAIResponse` (*http_response, params*)

A response from an OAI server.

Provides access to the returned data on different abstraction levels.

Parameters

- **http_response** – The original HTTP response.
- **params** (*dict*) – The OAI parameters for the request.

raw

The server's response as unicode.

xml

The server's response as parsed XML.

2.4.3 Iterating over OAI Items

class `sickle.iterator.OAIItemIterator` (*sickle, params, ignore_deleted=False*)

Iterator over OAI records/identifiers/sets transparently aggregated via OAI-PMH.

Can be used to conveniently iterate through the records of a repository.

Parameters

- **sickle** (*sickle.app.Sickle*) – The Sickle object that issued the first request.
- **params** (*dict*) – The OAI arguments.
- **ignore_deleted** (*bool*) – Flag for whether to ignore deleted records.

sickle

The *sickle.app.Sickle* instance used for making requests to the server.

verb

The OAI verb used for making requests to the server.

element

The name of the OAI item to iterate on (*record, header, set or metadataFormat*).

resumption_token

The content of the XML element *resumptionToken* from the last request.

ignore_deleted

Flag for whether to skip records marked as deleted.

next ()

Return the next record/header/set.

2.4.4 Iterating over OAI Responses

class *sickle.iterator.OAIResponseIterator* (*sickle, params, ignore_deleted=False*)

Iterator over OAI responses.

next ()

Return the next response.

2.4.5 Classes for OAI Items

The following classes represent OAI-specific items like records, headers, and sets. All items feature the attributes *raw* and *xml* which contain their original XML representation as unicode and as parsed XML objects.

Note: Sickle's automatic mapping of XML to OAI objects only works for Dublin Core encoded record data.

Identify Object

The *Identify* object is generated from *Identify* responses and is returned by *sickle.app.Sickle.Identify()*. It contains general information about the repository.

class *sickle.models.Identify* (*identify_response*)

Represents an *Identify* container.

This object differs from the other entities in that it has to be created from a *sickle.response.OAIResponse* instead of an XML element.

Parameters *identify_response* (*sickle.OAIResponse*) – The response for an *Identify* request.

Note: As the attributes of this class are auto-generated from the Identify XML elements, some of them may be missing for specific OAI interfaces.

adminEmail

The content of the element `adminEmail`. Normally the repository's administrative contact.

baseURL

The content of the element `baseURL`, which is the URL of the repository's OAI endpoint.

repositoryName

The content of the element `repositoryName`, which contains the name of the repository.

deletedRecord

The content of the element `deletedRecord`, which indicates whether and how the repository keeps track of deleted records.

delimiter

The content of the element `delimiter`.

description

The content of the element `description`, which contains a description of the repository.

earliestDatestamp

The content of the element `earliestDatestamp`, which indicates the datestamp of the oldest record in the repository.

granularity

The content of the element `granularity`, which indicates the granularity of the used dates.

oai_identifier

The content of the element `oai-identifier`.

Note: `oai-identifier` is not a valid name in Python.

protocolVersion

The content of the element `protocolVersion`, which indicates the version of the OAI protocol implemented by the repository.

repositoryIdentifier

The content of the element `repositoryIdentifier`.

sampleIdentifier

The content of the element `sampleIdentifier`, which usually contains an example of an identifier used by this repository.

scheme

The content of the element `scheme`.

raw

The original XML as unicode.

Record Object

Record objects represent single OAI records.

class `sickle.models.Record` (*record_element*, *strip_ns=True*)

Represents an OAI record.

Parameters

- **record_element** (`lxml.etree._Element`) – The XML element 'record'.
- **strip_ns** – Flag for whether to remove the namespaces from the element names.

header

Contains the record header represented as a *sickle.models.Header* object.

deleted

A boolean flag that indicates whether this record is deleted.

raw

The original XML as unicode.

Header Object

Header objects represent OAI headers.

class *sickle.models.Header* (*header_element*)

Represents an OAI Header.

Parameters *header_element* (:class:'*lxml.etree.Element*') – The XML element 'header'.

raw

The original XML as unicode.

Set Object

class *sickle.models.Set* (*set_element*)

Represents an OAI set.

Parameters *set_element* (:class:'*lxml.etree.Element*') – The XML element 'set'.

setName

The name of the set.

setSpec

The identifier of this set used for querying.

raw

The original XML as unicode.

MetadataFormat Object

class *sickle.models.MetadataFormat* (*mdf_element*)

Represents an OAI MetadataFormat.

Parameters *mdf_element* (:class:'*lxml.etree.Element*') – The XML element 'metadataFormat'.

metadataPrefix

The prefix used to identify this format.

metadataNamespace

The namespace URL for this format.

schema

The URL to the schema file of this format.

raw

The original XML as unicode.

2.5 Harvesting other Metadata Formats than OAI-DC

By default, Sickle’s mapping of the record XML into Python dictionaries is tailored to work only with Dublin-Core-encoded metadata payloads. Other formats most probably won’t be mapped correctly, especially if they are more hierarchically structured than Dublin Core.

In case you want to harvest these more complex formats, you have to write your own record model class by subclassing the default implementation that unpacks the metadata XML:

```
from sickle.models import Record

class MyRecord(Record):
    # Your XML unpacking implementation goes here.
    pass
```

Note: Take a look at the implementation of `sickle.models.Record` to get an idea of how to do this.

Next, associate your implementation with OAI verbs in the `Sickle` object. In this case, we want the `Sickle` object to use our implementation to represent items returned by `ListRecords` and `GetRecord` responses:

```
sickle = Sickle('http://...')
sickle.class_mapping['ListRecords'] = MyRecord
sickle.class_mapping['GetRecord'] = MyRecord
```

If you need to rewrite *all* item implementations, you can also provide a complete mapping to the `Sickle` object at instantiation:

```
my_mapping = {
    'ListRecords': MyRecord,
    'GetRecord': MyRecord,
    # ...
}

sickle = Sickle('http://...', class_mapping=my_mapping)
```

2.6 Development

2.6.1 Get the Code

Sickle is developed on [GitHub](#).

2.6.2 Testing

Sickle is tested with `nose`.

To run the tests, type:

```
python setup.py nosetests
```

2.7 Credits

- Inspired by the “for humans approach” of [requests](#)
- [pyoai](#) also provided valueable inspiration

- Sickle logo: Free [Valentina typeface](#) by Pedro Arilla and [public domain image](#) by Pearson Scott Foresman.

A

adminEmail (Identify attribute), 11

B

baseURL (Identify attribute), 11

D

deleted (Record attribute), 12

deletedRecord (Identify attribute), 11

delimiter (Identify attribute), 11

description (Identify attribute), 11

E

earliestDatestamp (Identify attribute), 11

element (OAIItemIterator attribute), 10

G

GetRecord() (sickle.app.Sickle method), 9

granularity (Identify attribute), 11

H

harvest() (sickle.app.Sickle method), 9

Header (class in sickle.models), 12

header (Record attribute), 11

I

Identify (class in sickle.models), 10

Identify() (sickle.app.Sickle method), 9

ignore_deleted (OAIItemIterator attribute), 10

L

last_response (Sickle attribute), 9

ListIdentifiers() (sickle.app.Sickle method), 9

ListMetadataFormats() (sickle.app.Sickle method), 9

ListRecords() (sickle.app.Sickle method), 9

ListSets() (sickle.app.Sickle method), 9

M

MetadataFormat (class in sickle.models), 12

metadataNamespace (MetadataFormat attribute), 12

metadataPrefix (MetadataFormat attribute), 12

N

next() (sickle.iterator.OAIItemIterator method), 10

next() (sickle.iterator.OAIResponseIterator method), 10

O

oai_identifier (Identify attribute), 11

OAIItemIterator (class in sickle.iterator), 9

OAIResponse (class in sickle.response), 9

OAIResponseIterator (class in sickle.iterator), 10

P

protocolVersion (Identify attribute), 11

R

raw (sickle.models.Header attribute), 12

raw (sickle.models.Identify attribute), 11

raw (sickle.models.MetadataFormat attribute), 12

raw (sickle.models.Record attribute), 12

raw (sickle.models.Set attribute), 12

raw (sickle.response.OAIResponse attribute), 9

Record (class in sickle.models), 11

repositoryIdentifier (Identify attribute), 11

repositoryName (Identify attribute), 11

resumption_token (OAIItemIterator attribute), 10

S

sampleIdentifier (Identify attribute), 11

schema (MetadataFormat attribute), 12

schema (Identify attribute), 11

Set (class in sickle.models), 12

setName (Set attribute), 12

setSpec (Set attribute), 12

Sickle (class in sickle.app), 8

sickle (OAIItemIterator attribute), 10

V

verb (OAIItemIterator attribute), 10

X

xml (sickle.response.OAIResponse attribute), 9