# skosprovider_heritagedata Documentation

***Release 0.2.1***

**Flanders Heritage Agency**

March 10, 2015

Contents

CHAPTER 1

# Introduction

This library offers an implementation of the `skosprovider.providers.VocabularyProvider` interface based on the Heritagedata Vocabularies. These vocabularies are used by *EH*, *RCAHMS* and *RCAHMW* in their role as curators of heritage.

## 1.1 Installation

To be able to use this library you need to have a modern version of Python installed. Currently we're supporting versions 2.7, 3.3 and 3.4 of Python.

This easiest way to install this library is through **pip** or **easy install**:

```
$ pip install skosprovider_heritagedata
```

This will download and install `skosprovider_heritagedata` and a few libraries it depends on.

## 1.2 Supported Heritagedata thesauri

The webservices provides by heritagedata.org provide access to multiple vocabularies or conceptschemes. You can select one of these vocabularies by passing a *scheme_uri* to the constructor of the `HeritagedataProvider`.

Heritagedata Vocabulary schemes

An overview of all *scheme_uri* can be provided by the following service:

www.heritagedata.org/live/services/getSchemes?pretty


```
[
    {
        "uri": "http://purl.org/heritagedata/schemes/agl_et",
        "label": "EVENT TYPE (EH)",
        "label lang": "en",
        "description": "Terminology used for recording archaeological and architectural investigative
        "attribution": "English Heritage"
    },
    {
        "uri": "http://purl.org/heritagedata/schemes/1",
        "label": "Monument Type Thesaurus (Scotland)",
        "label lang": "en",
        "description": "Monument types relating to the archaeological and built heritage of Scotland
        "attribution": "RCAHMS"
```

```
},
{
    "uri": "http://purl.org/heritagedata/schemes/2",
    "label": "Archaeological Objects Thesaurus (Scotland)",
    "label lang": "en",
    "description": "Objects made by human activity.",
    "attribution": "RCAHMS"
},
{
    "uri": "http://purl.org/heritagedata/schemes/3",
    "label": "Maritime Craft Thesaurus (Scotland)",
    "label lang": "en",
    "description": "Types of craft that survive as wrecks, or are documented as losses, in Scotti
    "attribution": "RCAHMS"
},
{
    "uri": "http://purl.org/heritagedata/schemes/11",
    "label": "PERIOD (WALES)",
    "label lang": "en",
    "description": "A list of periods for use in Wales.",
    "attribution": "RCAHMW"
},
{
    "uri": "http://purl.org/heritagedata/schemes/eh_tmt2",
    "label": "MONUMENT TYPE (EH)",
    "label lang": "en",
    "description": "Classification of monument type records by function.",
    "attribution": "English Heritage"
},
{
    "uri": "http://purl.org/heritagedata/schemes/560",
    "label": "ARCHAEOLOGICAL SCIENCES (EH)",
    "label lang": "en",
    "description": "Used for recording the techniques, recovery methods and materials associated
    "attribution": "English Heritage"
},
{
    "uri": "http://purl.org/heritagedata/schemes/eh_tbm",
    "label": "BUILDING MATERIALS (EH)",
    "label lang": "en",
    "description": "Thesaurus of main constructional material types (eg. the walls) for indexing
    "attribution": "English Heritage"
},
{
    "uri": "http://purl.org/heritagedata/schemes/eh_tmc",
    "label": "MARITIME CRAFT TYPE (EH)",
    "label lang": "en",
    "description": "A thesaurus of craft types which survive as wrecks in English Heritage's mari
    "attribution": "English Heritage"
},
{
    "uri": "http://purl.org/heritagedata/schemes/eh_period",
    "label": "PERIOD (EH)",
    "label lang": "en",
    "description": "English Heritage Periods List",
    "attribution": "English Heritage"
},
{
```

```
        "uri": "http://purl.org/heritagedata/schemes/eh_com",
        "label": "COMPONENTS (EH)",
        "label lang": "en",
        "description": "Terminology covering divisions and structural elements of a building or monur
        "attribution": "English Heritage"
    },
    {
        "uri": "http://purl.org/heritagedata/schemes/eh_evd",
        "label": "EVIDENCE (EH)",
        "label lang": "en",
        "description": "Terminology covering the existing physical remains of a monument, or the mean
        "attribution": "English Heritage"
    },
    {
        "uri": "http://purl.org/heritagedata/schemes/mda_obj",
        "label": "FISH Archaeological Objects Thesaurus",
        "label lang": "en",
        "description": "Originally developed by the Archaeological Objects Working Party and publishe
        "attribution": "English Heritage"
    },
    {
        "uri": "http://purl.org/heritagedata/schemes/10",
        "label": "MONUMENT TYPE THESAURUS (WALES)",
        "label lang": "en",
        "description": "Classification of monument types in Wales by function",
        "attribution": "RCAHMW"
    }
]
```

CHAPTER 2

# Using the providers

## 2.1 Using HeritagedataProvider

The `HeritagedataProvider` is a general provider for the Heritagedata vocabularies. It's use is identical to all other SKOSProviders. A scheme_uri is required to indicate the vocabulary to be used. Please consult *Supported Heritagedata thesauri* for a complete list.

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
'''
This script demonstrates using the HeritagedataProvider to get the concept of
'POST MEDIEVAL'.
'''

from skosprovider_heritagedata.providers import HeritagedataProvider

periodprovider = HeritagedataProvider(
    {'id': 'Heritagedata'},
    scheme_uri='http://purl.org/heritagedata/schemes/eh_period'
)

pm = periodprovider.get_by_id('PM')

print('Labels')
print('------')
for l in pm.labels:
   print(l.language + ': ' + l.label + ' [' + l.type + ']')

print('Notes')
print('-----')
for n in pm.notes:
    print(n.language + ': ' + n.note + ' [' + n.type + ']')
```

## 2.2 Finding concepts

See the `skosprovider_heritagedata.providers.HeritagedataProvider.find()` method for a detailed description of how this works.

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
```

```
'''
This script demonstrates using the HeritagedataProvider to find the concepts with 'iron' in their lab
'''

from skosprovider_heritagedata.providers import HeritagedataProvider

periodprovider = HeritagedataProvider(
    {'id': 'Heritagedata'},
    scheme_uri='http://purl.org/heritagedata/schemes/eh_period'
)

results = periodprovider.find(
    {
        'label': 'iron',
        'type': 'concept'
    }
)

print('Results')
print('------')
for result in results:
    print(result)
```

## 2.3 Using expand()

The expand methods return the id's of all the concepts that are narrower concepts of a certain concept or collection.

See the skosprovider_heritagedata.providers.HeritagedataProvider.expand() method for a detailed description of how this works.

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
'''
This script demonstrates using the HeritagedataProvider to expand a concept
'''

from skosprovider_heritagedata.providers import HeritagedataProvider

periodprovider = HeritagedataProvider(
    {'id': 'Heritagedata'},
    scheme_uri='http://purl.org/heritagedata/schemes/eh_period'
)

results = periodprovider.expand('PM')

print('Results')
print('------')
for result in results:
    print(result)
```

CHAPTER 3

# Development

Skosprovider_heritagedata is being developed by the Flanders Heritage Agency.

Since we place a lot of importance on code quality, we expect to have a good amount of code coverage present and run frequent unit tests. All commits and pull requests will be tested with Travis-ci. Code coverage is being monitored with Coveralls.

Locally you can run unit tests by using pytest or tox. Running pytest manually is good for running a distinct set of unit tests. For a full test run, tox is preferred since this can run the unit tests against multiple versions of python.

```
# Setup for development
$ python setup.py develop
# Run unit tests for all environments
$ tox
# No coverage
$ py.test
# Coverage
$ py.test --cov skosprovider_heritagedata --cov-report term-missing tests
# Only run a subset of the tests
$ py.test skosprovider_heritagedata/tests/test_providers.py
```

Please provide new unit tests to maintain 100% coverage. If you send us a pull request and this build doesn't function, please correct the issue at hand or let us know why it's not working.

CHAPTER 4

# API Documentation

## 4.1 Providers module

This module implements a skosprovider.providers.VocabularyProvider for http://www.heritagedata.org.

**class** skosprovider_heritagedata.providers.**HeritagedataProvider**(*metadata*, ***kwargs*)

A provider that can work with the Heritagedata services of http://www.heritagedata.org/blog/services/

**expand**(*id*)

**Expand a concept or collection to all it's narrower concepts.** If the id passed belongs to a skosprovider.skos.Concept, the id of the concept itself should be include in the return value.

**Parameters** **id** (*str*) – A concept or collection id.

**Returns** A lst of id's. Returns false if the input id does not exists

**find**(*query*, ***kwargs*)

Find concepts that match a certain query.

Currently query is expected to be a dict, so that complex queries can be passed. You can use this dict to search for concepts or collections with a certain label, with a certain type and for concepts that belong to a certain collection.

> **Warning:** The underlying service returns labels without specifying if they are prefLabels or altLabels. For a certain concept several labels are returned. This method does not return labels, but returns concepts. When multiple labels are detected for a single concept, only one label is attached to this concept. Since no information is present about the type of this label, this can be an altLabel for a concept where a prefLabel exists.

```
# Find anything that has a label of church.
provider.find({'label': 'church'}

# Find all concepts that are a part of collection 5.
provider.find({'type': 'concept', 'collection': {'id': 5})

# Find all concepts, collections or children of these
# that belong to collection 5.
provider.find({'collection': {'id': 5, 'depth': 'all'})
```

**Parameters** **query** – A dict that can be used to express a query. The following keys are permitted:

- *label*: Search for something with this label value. An empty label is equal to searching for all concepts.
- *type*: Limit the search to certain SKOS elements. If not present *all* is assumed:
  - *concept*: Only return `skosprovider.skos.Concept` instances.
  - *collection*: Only return `skosprovider.skos.Collection` instances.
  - *all*: Return both `skosprovider.skos.Concept` and `skosprovider.skos.Collection` instances.
- *collection*: Search only for concepts belonging to a certain collection. This argument should be a dict with two keys:
  - *id*: The id of a collection. Required.
  - *depth*: Can be *members* or *all*. Optional. If not present, *members* is assumed, meaning only concepts or collections that are a direct member of the collection should be considered. When set to *all*, this method should return concepts and collections that are a member of the collection or are a narrower concept of a member of the collection.

**Returns**

A `lst` of concepts and collections. Each of these is a dict with the following keys:

- id: id within the conceptscheme
- uri: *uri* of the concept or collection
- type: concept or collection
- label: A label to represent the concept or collection. It is determined by looking at the ***kwargs* parameter, the default language of the provider and finally falls back to *en*.

**get_all**(***kwargs*)
Not supported: This provider does not support this. The amount of results is too large

**get_by_id**(*id*)
Get a `skosprovider.skos.Concept` or `skosprovider.skos.Collection` by id

**Parameters** **id** (*(str)*) – integer id of the `skosprovider.skos.Concept` or `skosprovider.skos.Concept`

**Returns** corresponding `skosprovider.skos.Concept` or `skosprovider.skos.Concept`. Returns False if non-existing id

**get_by_uri**(*uri*)
Get a `skosprovider.skos.Concept` or `skosprovider.skos.Collection` by uri

**Parameters** **uri** (*(str)*) – string uri of the `skosprovider.skos.Concept` or `skosprovider.skos.Concept`

**Returns** corresponding `skosprovider.skos.Concept` or `skosprovider.skos.Concept`. Returns False if non-existing id

**get_children_display**(*id*, ***kwargs*)
Return a list of concepts or collections that should be displayed under this concept or collection.

**Parameters** **id** (*str*) – A concept or collection id.

**Returns** A `lst` of concepts and collections.

**get_top_concepts**(***kwargs*)
Returns all concepts that form the top-level of a display hierarchy.

**Returns** A `lst` of concepts.

**get_top_display**(***kwargs*)
Returns all concepts or collections that form the top-level of a display hierarchy. :return: A `lst` of concepts and collections.

## 4.2 Utils module

Utility functions for `skosprovider_heritagedata`.

`skosprovider_heritagedata.utils.`**conceptscheme_from_uri**(*conceptscheme_uri*)
Read a SKOS Conceptscheme from a *URI*

**Parameters** **conceptscheme_uri** (*string*) – URI of the conceptscheme.

**Return type** skosprovider.skos.ConceptScheme

`skosprovider_heritagedata.utils.`**text_**(*s*, *encoding='latin-1'*, *errors='strict'*)
If `s` is an instance of `binary_type`, return `s.decode(encoding, errors)`, otherwise return `s`

`skosprovider_heritagedata.utils.`**things_from_graph**(*graph*, *concept_scheme*)
Read concepts and collections from a graph.

**Parameters**

- **graph** (*rdflib.Graph*) – Graph to read from.
- **concept_scheme** (*skosprovider.skos.ConceptScheme*) – Conceptscheme the concepts and collections belong to.

**Return type** `list`

`skosprovider_heritagedata.utils.`**uri_to_graph**(*uri*)

**Parameters** **uri** (*string*) – *URI* where the RDF data can be found.

**Return type** rdflib.Graph

**Raises** **skosprovider.exceptions.ProviderUnavailableException** if the heritagedata.org services are down

CHAPTER 5

# History

## 5.1 0.2.1 (2015-03-10)

- Fix an issue where calls that include a *language* parameter would fail because certain methods were not expecting extra keyword parameters. (#12)
- Some documentation clarifications. (#11)

## 5.2 0.2.0 (2014-12-19)

- Compatibile with SkosProvider 0.5.0.
- Each Concept or Collection now also provides information on the ConceptScheme it's part of.
- Fix some issues with UTF-8 encoding.
- Fixed some issues with Python 2.x/3.x compatibility.
- Provider now throws a ProviderNotAvailableException when the underlying service is down.

## 5.3 0.1.0 (2014-10-08)

- Initial version
- Compatible with SkosProvider 0.3.0.

CHAPTER 6

# Glossary

**EH** English Heritage.

**RCAHMS** The Royal Commission on the Ancient and Historical Monuments of Scotland.

**RCAHMW** The Royal Commission on the Ancient and Historical Monuments of Wales.

**RDF** Resource Description Framework. A very flexible model for data definition organised around *triples*. These triples forms a directed, labeled graph, where the edges represent the named link between two resources, represented by the graph nodes.

**SKOS** Simple Knowledge Organization System. An general specification for Knowledge Organisation Systems (thesauri, word lists, authority files, ...) that is commonly serialised as *RDF*.

**URI** A *Uniform Resource Identifier*.

**URN** A URN is a specific form of a *URI*.

CHAPTER 7

# Indices and tables

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

## S

## C



## E



## F



## G



## H



## R



## S



## T



## U