language_tags Documentation

Release 1.1.0

Onroerend Erfgoed

Mar 08, 2022

CONTENTS

1 Introduction **3**

2 API Documentation **5**
- 2.1 Class tags 5
- 2.2 Class Tag 7
- 2.3 Class Subtag 8

3 History **11**
- 3.1 Changelog 11

4 Indices and tables **15**

Python Module Index **17**

Index **19**

This Python API offers a way to validate and lookup languages tags.

Standard

It is based on BCP 47 (RFC 5646) and the latest IANA language subtag registry.
This project will be updated as the standards change.

JSON data

See the language-subtag-registry project for the underlying JSON data.

Javascript version

This project is a Python version of the language-tags Javascript project.

CHAPTER

ONE

INTRODUCTION

This Python API offers a way to validate and lookup languages tags.

Import the module:

```
from language_tags import tags
```

To check whether the language_tag is valid use `tags.check()`. For example 'nl-Be' is valid but 'nl-BE-BE' is invalid.

```
> print(tags.check('nl-BE'))
True
> print(tags.check('nl-BE-BE'))
False
```

For meaningful error output see `tags.tag().errors`:

```
> errors = tags.tag('nl-BE-BE').errors
> for err in errors
>    print(err.message)
Extra region subtag 'BE' found.
```

Lookup descriptions of tags:

```
> print(tags.description('nl-BE'));
['Dutch', 'Flemish', 'Belgium']
```

Lookup descriptions of a language subtag:

```
> print(tags.language('nl').description);
['Dutch', 'Flemish']
```

Lookup tags by description:

```
> language_subtags = tags.search('Flemish')
> print(language_subtags[0])
'nl'
```

Get the language subtag of a tag:

```
> print(repr(tags.tag('nl-BE').language))
'{"subtag": "nl", "record": {"Subtag": "nl", "Suppress-Script": "Latn", "Added":
↪"2005-10-16", "Type": "language", "Description": ["Dutch", "Flemish"]}, "type":
↪"language"}'
```

A redundant tag is a grandfathered registration whose individual subtags appear with the same semantic meaning in the registry[1]. A redundant tag has descriptions and can have a preferred tag.

```
> redundant_tag = tags.tag('es-419')
> print(redundant_tag.descriptions)
['Latin American Spanish']
> print(redundant_tag.valid)
True
> print(redundant_tag.region.description)
['Latin America and the Caribbean']
> print(redundant_tag.region.language)
['Spanish', 'Castilian']
```

The remainder of the previously registered tags are "grandfathered"[1]. Grandfathered tags cannot be parsed into subtags. A grandfathered tag has descriptions. Most grandfathered tags have valid perferred tags.

```
> grandfathered_tag = tags.tag('i-klingon')
> print(grandfathered_tag.descriptions)
['Klingon']
> print(grandfathered_tag.valid)
False
> print(grandfathered_tag.subtags)
[]
> print(grandfathered_tag.preferred)
tlh
> preferred_tag = grandfathered_tag.preferred
> print(preferred_tag.language.description)
['Klingon', 'tlhIngan-Hol']
```

For the complete api documentation see next chapter.

[1] RFC 5646

CHAPTER

TWO

API DOCUMENTATION

2.1 Class tags

class language_tags.tags.**tags**

static check(*tag*)

Check if a string (hyphen-separated) tag is valid.

Parameters **tag** (*str*) – (hyphen-separated) tag.

Returns bool – True if valid.

static date()

Get the file date of the underlying data as a string.

Returns date as string (for example: '2014-03-27').

static description(*tag*)

Gets a list of descriptions given the tag.

Parameters **tag** (*str*) – (hyphen-separated) tag.

Returns list of string descriptions. The return list can be empty.

static filter(*subtags*)

Get a list of non-existing string subtag(s) given the input string subtag(s).

Parameters **subtags** – string subtag or a list of string subtags.

Returns list of non-existing string subtags. The return list can be empty.

static language(*subtag*)

Get a language *language_tags.Subtag.Subtag* of the subtag string.

Parameters **subtag** (*str*) – subtag.

Returns language *language_tags.Subtag.Subtag* if exists, otherwise None.

static languages(*macrolanguage*)

Get a list of *language_tags.Subtag.Subtag* objects given the string macrolanguage.

Parameters **macrolanguage** (*string*) – subtag macrolanguage.

Returns a list of the macrolanguage *language_tags.Subtag.Subtag* objects.

Raises **Exception** – if the macrolanguage does not exists.

static region(*subtag*)

Get a region *language_tags.Subtag.Subtag* of the subtag string.

Parameters **subtag** (*str*) – subtag.

Returns region *language_tags.Subtag.Subtag* if exists, otherwise None.

static search(*description*, *all=False*)

Gets a list of *language_tags.Subtag.Subtag* objects where the description matches.

param description a string or compiled regular expression. For

example: search(re.compile(r'[0-9]{4}')) if the

description of the returned subtag must contain four contiguous numerical digits.

type description str or RegExp

param all If set on True grandfathered and redundant tags will be included in the return list.

type all bool, optional

return list of *language_tags.Subtag.Subtag* objects each including the description. The return list can be empty.

static subtags(*subtags*)

Get a list of existing *language_tags.Subtag.Subtag* objects given the input subtag(s).

Parameters **subtags** – string subtag or list of string subtags.

Returns a list of existing *language_tags.Subtag.Subtag* objects. The return list can be empty.

static tag(*tag*)

Get a *language_tags.Tag.Tag* of a string (hyphen-separated) tag.

Parameters **tag** (*str*) – (hyphen-separated) tag.

Returns *language_tags.Tag.Tag*.

static type(*subtag*, *type*)

Get a *language_tags.Subtag.Subtag* by subtag and type. Can be None if not exists.

Parameters

- **subtag** (*str*) – subtag.
- **type** (*str*) – type of the subtag.

Returns *language_tags.Subtag.Subtag* if exists, otherwise None.

static types(*subtag*)

Get the types of a subtag string (excludes redundant and grandfathered).

Parameters **subtag** (*str*) – subtag.

Returns list of types. The return list can be empty.

2.2 Class Tag

class `language_tags.Tag.`**`Tag`**(*tag*)

Tags for Identifying Languages based on BCP 47 (RFC 5646) and the latest IANA language subtag registry.

Parameters **tag** (*str*) – (hyphen-separated) tag.

property added

Get the date string of grandfathered or redundant tag when it was added to the registry.

Returns added date string if the deprecated or redundant tag has one, otherwise None.

property deprecated

Get the deprecation date of grandfathered or redundant tag if the tag is deprecated.

Returns deprecation date string if the deprecated or redundant tag has one, otherwise None.

property descriptions

Get the list of descriptions of the grandfathered or redundant tag.

Returns list of descriptions. If no descriptions available, it returns an empty list.

error(*code*, *subtag=None*)

Get the `language_tags.Tag.Tag.Error` of a specific Tag error code. The error creates a message explaining the error. It also refers to the respective (sub)tag(s).

Parameters

- **code** (*int*) – a Tag error error:
  - 1 = Tag.ERR_DEPRECATED
  - 2 = Tag.ERR_NO_LANGUAGE
  - 3 = Tag.ERR_UNKNOWN,
  - 4 = Tag.ERR_TOO_LONG
  - 5 = Tag.ERR_EXTRA_REGION
  - 6 = Tag.ERR_EXTRA_EXTLANG
  - 7 = Tag.ERR_EXTRA_SCRIPT,
  - 8 = Tag.ERR_DUPLICATE_VARIANT
  - 9 = Tag.ERR_WRONG_ORDER
  - 10 = Tag.ERR_SUPPRESS_SCRIPT,
  - 11 = Tag.ERR_SUBTAG_DEPRECATED
  - 12 = Tag.ERR_EXTRA_LANGUAGE
- **subtag** – string (sub)tag or list of string (sub)tags creating the error.

Returns An exception class containing: a Tag error input code, the derived message with the given (sub)tag(s). input

property errors

Get the errors of the tag. If invalid then the list will consist of errors containing each a code and message explaining the error. Each error also refers to the respective (sub)tag(s).

Returns list of errors of the tag. If the tag is valid, it returns an empty list.

property format

Get format according to algorithm defined in RFC 5646 section 2.1.1.

Returns formatted tag string.

property language

Get the language *language_tags.Subtag.Subtag* of the tag.

Returns language *language_tags.Subtag.Subtag* that is part of the tag. The return can be None.

property preferred

Get the preferred *language_tags.Tag.Tag* of the deprecated or redundant tag.

Returns preferred *language_tags.Tag.Tag* if the deprecated or redundant tag has one, otherwise None.

property region

Get the region *language_tags.Subtag.Subtag* of the tag.

Returns region *language_tags.Subtag.Subtag* that is part of the tag. The return can be None.

property script

Get the script *language_tags.Subtag.Subtag* of the tag.

Returns script *language_tags.Subtag.Subtag* that is part of the tag. The return can be None.

property subtags

Get the *language_tags.Subtag.Subtag* objects of the tag.

Returns list of *language_tags.Subtag.Subtag* objects that are part of the tag. The return list can be empty.

property type

Get the type of the tag (either grandfathered, redundant or tag see RFC 5646 section 2.2.8.).

Returns string – type of the tag.

property valid

Checks whether the tag is valid.

Returns Bool – True if valid otherwise False.

2.3 Class Subtag

class language_tags.Subtag.**Subtag**(*subtag*, *type*)

A subtag is a part of the hyphen-separated *language_tags.Tag.Tag*.

Parameters

- **subtag** (*str*) – subtage.
- **type** (*str*) – can be 'language', 'extlang', 'script', 'region' or 'variant'.

Returns

raise Error Checks for `Subtag.ERR_NONEXISTENT` and `Subtag.ERR_TAG`.

property added

Get the date when the subtag was added to the registry.

Returns date (as string) when the subtag was added to the registry.

property comments
Get the comments of the subtag.

Returns list of comments. The return list can be empty.

property deprecated
Get the deprecation date.

Returns deprecation date as string if subtag is deprecated, otherwise None.

property description
Get the subtag description.

Returns list of description strings.

property format
Get the subtag code conventional format according to RFC 5646 section 2.1.1.

Returns string – subtag code conventional format.

property preferred
Get the preferred subtag.

Returns preferred `language_tags.Subtag.Subtag` if exists, otherwise None.

property scope
Get the subtag scope.

Returns string subtag scope if exists, otherwise None.

property script
Get the language's default script of the subtag (RFC 5646 section 3.1.9)

Returns string – the language's default script.

property type
Get the subtag type.

Returns string – either 'language', 'extlang', 'script', 'region' or 'variant'.

CHAPTER

THREE

HISTORY

3.1 Changelog

3.1.1 1.1.0

- Update data to version 2020-09-29 (#62)
- Update dependencies and Python (Removed Python 3.5 support, now supports 3.6 to 3.9) (#74, #77)
- Drop pyup support (#80)
- Fix pypi description (#78)
- Include MIT License in package (#67)
- Fix deprecation warnings (#84)

3.1.2 1.0.0

- Drop support for Python 2

3.1.3 0.5.0

- Updated dependencies and Python (Removed Python3.3 and Python3.4 support, added 3.6 and 3.7)

3.1.4 0.4.6

- Avoid modifying tag when getting description

3.1.5 0.4.5

- Close files after opening #38

3.1.6 0.4.4

- Bug fix release: language tag 'aa' is detected as invalid #27

3.1.7 0.4.3

- Upgrade to <https://github.com/mattcg/language-subtag-registry/releases/tag/v0.3.18>

3.1.8 0.4.2

- Official python 3.5 compatibility
- Upgrade to <https://github.com/mattcg/language-subtag-registry/releases/tag/v0.3.15>

3.1.9 0.4.1

- Included the data folder again in the project package.
- Added bash script (*update_data_files.sh*) to download the language-subtag-registry and move this data in the data folder of the project.

3.1.10 0.4.0

- Allow parsing a redundant tag into subtags.
- Added package.json file for easy update of the language subtag registry data using npm (`npm install` or `npm update`)
- Improvement of the `language-tags.tags.search` function: rank equal description at top. See mattcg/language-tags#4

3.1.11 0.3.2

- Upgrade to <https://github.com/mattcg/language-subtag-registry/releases/tag/v0.3.11>
- Added wheel config
- Fixed bug under windows: opening data files using utf-8 encoding.

3.1.12 0.3.1

- Upgrade to <https://github.com/mattcg/language-subtag-registry/releases/tag/v0.3.8>

3.1.13 0.3.0

- Upgrade to <https://github.com/mattcg/language-subtag-registry/releases/tag/v0.3.6>
- Simplify output of __str__ functions. The previous json dump is assigned to the repr function.

```
nlbe = tags.tags('nl-Latn-BE')
> print(nlbe)
'nl-Latn-BE'
> print(nlbe.language)
'nl'
> print(nlbe.script)
'Latn'
```

3.1.14 0.2.0

- Adjust language, region and script properties of Tag. The properties will return *language_tags.Subtag.Subtag* instead of a list of string subtags

```
> print(tags.tag('nl-BE').language)
'{"subtag": "nl", "record": {"Subtag": "nl", "Suppress-Script": "Latn",
↪"Added": "2005-10-16", "Type": "language", "Description": ["Dutch",
↪"Flemish"]}, "type": "language"}'
> print(tags.tag('nl-BE').region)
'{"subtag": "be", "record": {"Subtag": "BE", "Added": "2005-10-16", "Type
↪": "region", "Description": ["Belgium"]}, "type": "region"}'
> print(tags.tag('en-mt-arab').script)
'{"subtag": "arab", "record": {"Subtag": "Arab", "Added": "2005-10-16",
↪"Type": "script", "Description": ["Arabic"]}, "type": "script"}'
```

3.1.15 0.1.1

- Added string and Unicode functions to make it easy to print Tags and Subtags.

```
> print(tags.tag('nl-BE'))
'{"tag": "nl-be"}'
```

- Added functions to easily select either the language, region or script subtags strings of a Tag.

```
> print(tags.tag('nl-BE').language)
['nl']
```

3.1.16 0.1.0

- Initial version

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

l

language_tags.Subtag, 8
language_tags.Tag, 7
language_tags.tags, 5

INDEX

A

added() (*language_tags.Subtag.Subtag property*), 8
added() (*language_tags.Tag.Tag property*), 7

C

check() (*language_tags.tags.tags static method*), 5
comments() (*language_tags.Subtag.Subtag property*), 8

D

date() (*language_tags.tags.tags static method*), 5
deprecated() (*language_tags.Subtag.Subtag property*), 9
deprecated() (*language_tags.Tag.Tag property*), 7
description() (*language_tags.Subtag.Subtag property*), 9
description() (*language_tags.tags.tags static method*), 5
descriptions() (*language_tags.Tag.Tag property*), 7

E

error() (*language_tags.Tag.Tag method*), 7
errors() (*language_tags.Tag.Tag property*), 7

F

filter() (*language_tags.tags.tags static method*), 5
format() (*language_tags.Subtag.Subtag property*), 9
format() (*language_tags.Tag.Tag property*), 7

L

language() (*language_tags.Tag.Tag property*), 8
language() (*language_tags.tags.tags static method*), 5
language_tags.Subtag
    module, 8
language_tags.Tag
    module, 7
language_tags.tags
    module, 5
languages() (*language_tags.tags.tags static method*), 5

M

module
    language_tags.Subtag, 8
    language_tags.Tag, 7
    language_tags.tags, 5

P

preferred() (*language_tags.Subtag.Subtag property*), 9
preferred() (*language_tags.Tag.Tag property*), 8

R

region() (*language_tags.Tag.Tag property*), 8
region() (*language_tags.tags.tags static method*), 5

S

scope() (*language_tags.Subtag.Subtag property*), 9
script() (*language_tags.Subtag.Subtag property*), 9
script() (*language_tags.Tag.Tag property*), 8
search() (*language_tags.tags.tags static method*), 6
Subtag (*class in language_tags.Subtag*), 8
subtags() (*language_tags.Tag.Tag property*), 8
subtags() (*language_tags.tags.tags static method*), 6

T

Tag (*class in language_tags.Tag*), 7
tag() (*language_tags.tags.tags static method*), 6
tags (*class in language_tags.tags*), 5
type() (*language_tags.Subtag.Subtag property*), 9
type() (*language_tags.Tag.Tag property*), 8
type() (*language_tags.tags.tags static method*), 6
types() (*language_tags.tags.tags static method*), 6

V

valid() (*language_tags.Tag.Tag property*), 8