

---

# Wikidata Documentation

*Release 0.5.3*

**Hong Minhee**

**Jun 29, 2017**



---

## Contents

---

<b>1</b>	<b>wikidata — Wikidata client library</b>	<b>3</b>
1.1	wikidata.cache — Caching policies . . . . .	3
1.2	wikidata.client — Client session . . . . .	4
1.3	wikidata.commonsmmedia — Wikimedia Commons . . . . .	5
1.4	wikidata.datavalue — Interpreting datavalues . . . . .	6
1.5	wikidata.entity — Wikidata entities . . . . .	7
1.6	wikidata.multilingual — Multilingual texts . . . . .	8
<b>2</b>	<b>Changelog</b>	<b>9</b>
2.1	Version 0.5.3 . . . . .	9
2.2	Version 0.5.2 . . . . .	9
2.3	Version 0.5.1 . . . . .	9
2.4	Version 0.5.0 . . . . .	9
2.5	Version 0.4.4 . . . . .	10
2.6	Version 0.4.3 . . . . .	10
2.7	Version 0.4.2 . . . . .	10
2.8	Version 0.4.1 . . . . .	10
2.9	Version 0.4.0 . . . . .	10
2.10	Version 0.3.0 . . . . .	11
2.11	Version 0.2.0 . . . . .	11
2.12	Version 0.1.0 . . . . .	11
<b>3</b>	<b>Indices and tables</b>	<b>13</b>
	<b>Python Module Index</b>	<b>15</b>



This package provides easy APIs to use [Wikidata](#) for Python.

```
>>> from wikidata.client import Client
>>> client = Client()
>>> entity = client.get('Q20145', load=True)
>>> entity
<wikidata.entity.Entity Q20145 'IU'>
>>> entity.description
m'South Korean singer and actress'
>>> image_prop = client.get('P18')
>>> image = entity[image_prop]
>>> image
<wikidata.commonsmmedia.File 'File:KBS "The Producers" press conference, 11 May 2015_
↪10.jpg'>
>>> image.image_resolution
(820, 1122)
>>> image.image_url
'https://upload.wikimedia.org/wikipedia/commons/6/60/KBS_%22The_Producers%22_press_
↪conference%2C_11_May_2015_10.jpg'
```



## wikidata.cache — Caching policies

Changed in version 0.5.0.

`wikidata.cache.CacheKey(x)`

The type of keys to look up cached values. Alias of `str`.

**class** `wikidata.cache.CachePolicy`

Interface for caching policies.

**get** (*key*: `<function NewType.<locals>.new_type at 0x7fd60554e730>`)  $\rightarrow$  `typing.Union[<function NewType.<locals>.new_type at 0x7fd60554e510>, NoneType]`

Look up a cached value by its key.

**Parameters** **key** (`CacheKey`) – The key string to look up a cached value.

**Returns** The cached value if it exists. `None` if there's no such key.

**Return type** `Optional[CacheValue]`

**set** (*key*: `<function NewType.<locals>.new_type at 0x7fd60554e730>`, *value*: `typing.Union[<function NewType.<locals>.new_type at 0x7fd60554e510>, NoneType]`)  $\rightarrow$  `None`

Create or update a cache.

**Parameters**

- **key** (`CacheKey`) – A key string to create or update.
- **value** (`Optional[CacheValue]`) – A value to cache. `None` to remove cache.

`wikidata.cache.CacheValue(x)`

The type of cached values.

**class** `wikidata.cache.MemoryCachePolicy(max_size: int = 128)  $\rightarrow$  None`

LRU (least recently used) cache in memory.

**Parameters** **max\_size** (`int`) – The maximum number of values to cache. 128 by default.

**class** `wikidata.cache.NullCachePolicy`

No-op cache policy.

**class** `wikidata.cache.ProxyCachePolicy` (*cache\_object*, *timeout*: *int*, *property\_timeout*: *typing.Union[int, NoneType]* = *None*, *namespace*: *str* = *'wd\_'*) → *None*

This proxy policy is a proxy or an adaptor to another cache object. Cache objects can be anything if they satisfy the following interface:

```
def get(key: str) -> Optional[bytes]: pass
def set(key: str, value: bytes, timeout: int=0) -> None: pass
def delete(key: str) -> None: pass
```

(The above methods omit `self` parameters.) It's compatible with de facto interface for caching libraries in Python (e.g. `python-memcached`, `werkzeug.contrib.cache`).

#### Parameters

- **cache\_object** – The cache object to adapt. Read the above explanation.
- **timeout** (*int*) – Lifespan of every cache in seconds. 0 means no expiration.
- **property\_timeout** (*int*) – Lifespan of caches for properties (in seconds). Since properties don't change frequently or their changes usually don't make important effect, longer lifespan of properties' cache can be useful. 0 means no expiration. Set to the same as `timeout` by default.
- **namespace** (*str*) – The common prefix attached to every cache key. *'wd\_'* by default.

## wikidata.client — Client session

`wikidata.client.WIKIDATA_BASE_URL` = *'https://www.wikidata.org/'*

(*str*) The default `base_url` of *Client* constructor.

Changed in version 0.3.0: As the meaning of *Client* constructor's `base_url` parameter, it now became to `https://www.wikidata.org/` from `https://www.wikidata.org/wiki/` (which contained the trailing path `wiki/`).

**class** `wikidata.client.Client`

Wikidata client session.

#### Parameters

- **base\_url** (*str*) – The base url of the Wikidata. `WIKIDATA_BASE_URL` is used by default.
- **opener** (`urllib.request.OpenerDirector`) – The opener for `urllib.request`. If omitted or *None* the default opener is used.
- **entity\_type\_guess** (*bool*) – Whether to guess *type* of *Entity* from its *id* for less HTTP requests. True by default.
- **cache\_policy** – A caching policy for API calls. No cache (*NullCachePolicy*) by default.

New in version 0.5.0: The `cache_policy` option.

Changed in version 0.3.0: The meaning of `base_url` parameter changed. It originally meant `https://www.wikidata.org/wiki/` which contained the trailing path `wiki/`, but now it means only `https://www.wikidata.org/`.



New in version 0.2.0: The `entity_type_guess` option.

**cache\_policy** = <wikidata.cache.NullCachePolicy object>  
(CachePolicy) A caching policy for API calls.

New in version 0.5.0.

**datavalue\_decoder** = None

(Union[*Decoder*, Callable[[*Client*, *str*, Mapping[*str*, *object*]], *object*]]) The function to decode the given datavalue. It's typically an instance of *Decoder* or its subclass.

**decode\_datavalue** (*datatype*: *str*, *datavalue*: typing.Mapping[*str*, *object*]) → *object*  
Decode the given datavalue using the configured *datavalue\_decoder*.

New in version 0.3.0.

**entity\_type\_guess** = True

(bool) Whether to guess *type* of *Entity* from its *id* for less HTTP requests.

New in version 0.2.0.

**get** (*entity\_id*: <function NewType.<locals>.new\_type at 0x7fd60547d488>, *load*: bool = False) → wikidata.entity.Entity  
Get a Wikidata entity by its *EntityId*.

#### Parameters

- **entity\_id** – The *id* of the *Entity* to find.
- **load** (bool) – Eager loading on True. Lazy loading (False) by default.

**Returns** The found entity.

**Return type** *Entity*

New in version 0.3.0: The *load* option.

**guess\_entity\_type** (*entity\_id*: <function NewType.<locals>.new\_type at 0x7fd60547d488>) → typing.Union[wikidata.entity.EntityType, NoneType]  
Guess *EntityType* from the given *EntityId*. It could return *None* when it fails to guess.

---

**Note:** It always fails to guess when *entity\_type\_guess* is configured to *False*.

---

**Returns** The guessed *EntityId*, or *None* if it fails to guess.

**Return type** Optional[*EntityType*]

New in version 0.2.0.

## wikidata.commonsmmedia — Wikimedia Commons

New in version 0.3.0.

**class** wikidata.commonsmmedia.**File** (*client*: wikidata.client.Client, *title*: *str*) → None  
Represent a file on [Wikimedia Commons](#).

**image\_mimetype**

(Optional[*str*]) The MIME type of the image. It may be *None* if it's not an image.

**image\_resolution**

(Optional[Tuple[int, int]]) The (width, height) pair of the image. It may be None if it's not an image.

**image\_size**

(Optional[int]) The size of the image in bytes. It may be None if it's not an image.

**image\_url**

(Optional[str]) The image url. It may be None if it's not an image.

**page\_url**

(str) The canonical url of the page.

**exception** `wikidata.commonsmmedia.FileError`

Exception raised when something goes wrong with *File*.

## wikidata.datavalue — Interpreting datavalues

This module provides the decoder interface for customizing how datavalues are decoded, and the default *Decoder* implementation.

Technically the interface is just a callable so that its implementation doesn't necessarily have to be an instance of *Decoder* or its subclass, but only need to satisfy:

```
typing.Callable[[wikidata.client.Client, str, typing.Mapping[str, object]],
                object]
```

New in version 0.3.0.

**exception** `wikidata.datavalue.DatavalueError` (\*args, \*\*kwargs)

Exception raised during decoding datavalues. It subclasses *ValueError* as well.

**datavalue**

The datavalue which caused the decoding error.

**class** `wikidata.datavalue.Decoder`

Decode the given datavalue to a value of the appropriate Python type. For extensibility it uses visitor pattern and is intended to be subclassed. To customize decoding of datavalues subclass it and configure `datavalue_decoder` option of *Client* to the customized decoder.

It automatically invokes an appropriate visitor method using a simple rule of name: `{datatype}__{datavalue[type]}`. For example, if the following call to a decoder was made:

```
decoder(client, 'mydatatype', {'type': 'mytype', 'value': '...'})
```

it's delegated to the following visitor method call:

```
decoder.mydatatype__mytype(client, {'type': 'mytype', 'value': '...'})
```

If a decoder failed to find a visitor method matched to `{datatype}__{datavalue[type]}` pattern it secondly try to find a general version of visitor method: `{datavalue[type]}` which lacks double underscores. For example, for the following call:

```
decoder(client, 'mydatatype', {'type': 'mytype', 'value': '...'})
```

It firstly try to find the following visitor method:

```
decoder.mydatatype__mytype
```

but if there's no such method it secondly try to find the following general visitor method:

decoder.mytype

This twice-try dispatch is useful when to make a visitor method to be matched regardless of datatype.

If its datavalue[type] contains hyphens they're replaced by underscores. For example:

```
decoder(client, 'string',
        {'type': 'wikibase-entityid', 'value': 'a text value'})
```

the above call is delegated to the following visitor method call:

```
decoder.string__wikibase_entityid(
    #     Note that the ^ underscore
    client,
    {'type': 'wikibase-entityid', 'value': 'a text value'})
)
```

## wikidata.entity — Wikidata entities

**class** wikidata.entity.Entity

Wikidata entity. Can be an item or a property. Its attributes can be lazily loaded.

To get an entity use *Client.get()* method instead of the constructor of *Entity*.

---

**Note:** Although it implements `Mapping[EntityId, object]`, it actually is multidict. See also `getlist()` method.

---

Changed in version 0.2.0: Implemented `Mapping[EntityId, object]` protocol for easy access of statement values.

Changed in version 0.2.0: Implemented `Hashable` protocol and `==/=` operators for equality test.

**getlist** (*key*: wikidata.entity.Entity) → typing.Sequence[object]

Return all values associated to the given *key* property in sequence.

**Parameters** *key* (*Entity*) – The property entity.

**Returns** A sequence of all values associated to the given *key* property. It can be empty if nothing is associated to the property.

**Return type** `Sequence[object]`

**lists** () → typing.Sequence[typing.Tuple[typing.Entity, typing.Sequence[object]]]

Similar to `items()` except the returning pairs have each list of values instead of each single value.

**Returns** The pairs of (*key*, *values*) where *values* is a sequence.

**Return type** `Sequence[Tuple[Entity, Sequence[object]]]`

**type**

(*EntityType*) The type of entity, *item* or *property*.

New in version 0.2.0.

wikidata.entity.EntityId(*x*)

The identifier of each *Entity*. Alias of `str`.

**class** wikidata.entity.EntityType

The enumerated type which consists of two possible values:

- item*
- property*

New in version 0.2.0.

**item = 'item'**

(*EntityType*) Items are *Entity* objects that are typically represented by Wikipage (at least in some Wikipedia languages). They can be viewed as “the thing that a Wikipage is about,” which could be an individual thing (the person *Albert Einstein*), a general class of things (the class of all *Physicists*), and any other concept that is the subject of some Wikipedia page (including things like *History of Berlin*).

**See also:**

**Items — Wikibase Data Model** The data model of Wikibase describes the structure of the data that is handled in Wikibase.

**property = 'property'**

(*EntityType*) Properties are *Entity* objects that describe a relationship between items (or other *Entity* objects) and values of the property. Typical properties are *population* (using numbers as values), *binomial name* (using strings as values), but also *has father* and *author of* (both using items as values).

**See also:**

**Properties — Wikibase Data Model** The data model of Wikibase describes the structure of the data that is handled in Wikibase.

## wikidata.multilingual — Multilingual texts

**class** wikidata.multilingual.**MonolingualText**

Locale-denoted text. It’s almost equivalent to *str* (and indeed subclasses *str*) except that it has two more attribute: *locale* and *locale\_code* that denote what language the text is written in.

**locale**

(*Locale*) The language (locale) that the text is written in.

**locale\_code = None**

(*str*) The code of *locale*.

wikidata.multilingual.**normalize\_locale\_code** (*locale*: *typing.Union[babel.core.Locale, str]*) → *str*

Determine the normalized locale code string.

```
>>> normalize_locale_code('ko-kr')
'ko_KR'
>>> normalize_locale_code('zh-TW')
'zh_Hant_TW'
>>> normalize_locale_code(Locale.parse('en_US'))
'en_US'
```

### Version 0.5.3

Released on June 30, 2017.

- Fixed `ValueError` from `Entity.label/Entity.description` with languages `ISO 639-1` doesn't cover (e.g. `cbk-zam`). [#2]

Although this fix prevents these properties from raising `ValueError`, it doesn't completely fix the problem. `babel.core.Locale` type, which Wikidata depends on, currently doesn't support languages other than `ISO 639-1`. In order to completely fix the problem, we need to patch `Babel` to support them, or make Wikidata independent from `Babel`.

### Version 0.5.2

Released on June 28, 2017.

- Fixed `AssertionError` from empty `multilingual_attributes`.

### Version 0.5.1

Released on June 28, 2017.

- Fixed `AssertionError` from `len()` or iterating (`iter()`) on `Entity` objects with empty claims.

### Version 0.5.0

Released on June 13, 2017.

- Wikidata API calls over network became possible to be cached.

- `Client` now has `cache_policy` attribute and constructor option. Nothing is cached by default.
- Added `wikidata.cache` module and `CachePolicy` interface in it. Two built-in implementation of the interface were added:
  - `NullCachePolicy` No-op.
  - `MemoryCachePolicy` LRU cache in memory.
  - `ProxyCachePolicy` Proxy/adaptor to another proxy object. Useful for utilizing third-party cache libraries.
- `wikidata.client.Client.request` logger became to record logs about cache hits as `DEBUG` level.

## Version 0.4.4

Released on June 30, 2017.

- Fixed `ValueError` from `Entity.label/Entity.description` with languages `ISO 639-1` doesn't cover (e.g. `cbk-zam`). [#2]

Although this fix prevents these properties from raising `ValueError`, it doesn't completely fix the problem. `babel.core.Locale` type, which Wikidata depends on, currently doesn't support languages other than `ISO 639-1`. In order to completely fix the problem, we need to patch `Babel` to support them, or make Wikidata independent from `Babel`.

## Version 0.4.3

Released on June 28, 2017.

- Fixed `AssertionError` from empty `multilingual_attributes`.

## Version 0.4.2

Released on June 28, 2017.

- Fixed `AssertionError` from `len()` or iterating (`iter()`) on `Entity` objects with empty claims.

## Version 0.4.1

Released on April 30, 2017.

- Fixed `AssertionError` from `getlist()` on entities with empty claims.

## Version 0.4.0

Released on April 24, 2017.

- Monolingual texts became able to be handled.
  - Added `MonolingualText` type which is a true subtype of `str`.

## Version 0.3.0

Released on February 23, 2017.

- Now *Client* became able to customize how it decodes datavalues to Python objects.
  - Added *wikidata.datavalue* module and *Decoder* class inside it.
  - Added *datavalue\_decoder* option to *Client*.
- Now files on Wikimeda Commons became able to be handled.
  - New decoder became able to parse Wikimedia Commons files e.g. images.
  - Added *wikidata.commonsmmedia* module and *File* class inside it.
- The meaning of *Client* constructor's *base\_url* prameter beccame not to contain the trailing path *wiki/* from *https://www.wikidata.org/wiki/*. As its meaning changed, the value of *WIKIDATA\_BASE\_URL* constant also changed to not have the trailing path.
- Added *load* option to *Client.get()* method.

## Version 0.2.0

Released on February 19, 2017.

- Made *Entity* multidict. Now it satisfies *Mapping[Entity, object]* protocol.
- Added *Entity.type* property and *EntityType* enum class to represent it.
- Added *entity\_type\_guess* option and *guess\_entity\_type()* method to *Client* class.
- Implemented *Hashable* protocol and *==/=* operators to *Entity* for equality test.

## Version 0.1.0

Initial version. Released on February 15, 2017.





## CHAPTER 3

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### W

- wikidata, [1](#)
- wikidata.cache, [3](#)
- wikidata.client, [4](#)
- wikidata.commonsmmedia, [5](#)
- wikidata.datavalue, [6](#)
- wikidata.entity, [7](#)
- wikidata.multilingual, [8](#)



## C

cache\_policy (wikidata.client.Client attribute), 5  
 CacheKey() (in module wikidata.cache), 3  
 CachePolicy (class in wikidata.cache), 3  
 CacheValue() (in module wikidata.cache), 3  
 Client (class in wikidata.client), 4

## D

datavalue (wikidata.datavalue.DatavalueError attribute), 6  
 datavalue\_decoder (wikidata.client.Client attribute), 5  
 DatavalueError, 6  
 decode\_datavalue() (wikidata.client.Client method), 5  
 Decoder (class in wikidata.datavalue), 6

## E

Entity (class in wikidata.entity), 7  
 entity\_type\_guess (wikidata.client.Client attribute), 5  
 EntityId() (in module wikidata.entity), 7  
 EntityType (class in wikidata.entity), 7

## F

File (class in wikidata.commonsmmedia), 5  
 FileError, 6

## G

get() (wikidata.cache.CachePolicy method), 3  
 get() (wikidata.client.Client method), 5  
 getlist() (wikidata.entity.Entity method), 7  
 guess\_entity\_type() (wikidata.client.Client method), 5

## I

image\_mimetype (wikidata.commonsmmedia.File attribute), 5  
 image\_resolution (wikidata.commonsmmedia.File attribute), 5  
 image\_size (wikidata.commonsmmedia.File attribute), 6  
 image\_url (wikidata.commonsmmedia.File attribute), 6  
 item (wikidata.entity.EntityType attribute), 8

## L

lists() (wikidata.entity.Entity method), 7  
 locale (wikidata.multilingual.MonolingualText attribute), 8  
 locale\_code (wikidata.multilingual.MonolingualText attribute), 8

## M

MemoryCachePolicy (class in wikidata.cache), 3  
 MonolingualText (class in wikidata.multilingual), 8

## N

normalize\_locale\_code() (in module wikidata.multilingual), 8  
 NullCachePolicy (class in wikidata.cache), 3

## P

page\_url (wikidata.commonsmmedia.File attribute), 6  
 property (wikidata.entity.EntityType attribute), 8  
 ProxyCachePolicy (class in wikidata.cache), 4

## S

set() (wikidata.cache.CachePolicy method), 3

## T

type (wikidata.entity.Entity attribute), 7

## W

wikidata (module), 1  
 wikidata.cache (module), 3  
 wikidata.client (module), 4  
 wikidata.commonsmmedia (module), 5  
 wikidata.datavalue (module), 6  
 wikidata.entity (module), 7  
 wikidata.multilingual (module), 8  
 WIKIDATA\_BASE\_URL (in module wikidata.client), 4