akismet Documentation

Release 1.0.1

Michael Foord and James Bennett

May 29, 2017

Contents

1 Documentation contents 3

Python Module Index 9

`akismet` is a Python library wrapping the Wordpress Akismet spam-filtering service. All methods of the Akismet API are supported:

- Checking comments for spam
- Reporting comments incorrectly classified as not spam
- Reporting comments incorrectly classified as spam

Use of this module requires an Akismet API key (which must be obtained from the Akismet service).

CHAPTER 1

Documentation contents

Installation guide

`akismet` 1.0 is officially tested and supported on the following versions of Python:

- Python 2.7
- Python 3.3
- Python 3.4
- Python 3.5
- Python 3.6

Normal installation

The preferred method of installing `akismet` is via `pip`, the standard Python package-installation tool. If you don't have `pip`, instructions are available for how to obtain and install it. If you're using Python 2.7.9 or later (for Python 2) or Python 3.4 or later (for Python 3), `pip` came bundled with your installation of Python.

Once you have `pip`, simply type:

```
pip install akismet
```

Manual installation

It's also possible to install `akismet` manually. To do so, obtain the latest packaged version from the listing on the Python Package Index. Unpack the `.tar.gz` file, and run:

```
python setup.py install
```

Once you've installed `akismet`, you can verify successful installation by opening a Python interpreter and typing `import akismet`.

If the installation was successful, you'll simply get a fresh Python prompt. If you instead see an `ImportError`, check the configuration of your install tools and your Python import path to ensure `akismet` installed into a location Python can import from.

Installing from a source checkout

The development repository for `akismet` is at <https://github.com/ubernostrum/akismet>. Presuming you have git installed, you can obtain a copy of the repository by typing:

```
git clone https://github.com/ubernostrum/akismet.git
```

From there, you can use normal git commands to check out the specific revision you want, and install it using `python setup.py install`.

Usage overview

Once you have `akismet` *installed*, you can begin using it as soon as you register an API key and a site to use it on.

Obtaining an API key

Use of `akismet` requires an Akismet API key, and requires associating that API key with the site you'll use `akismet` on. Visit akismet.com to purchase an API key and associate it with a site.

Optional arguments to API methods

For API methods other than *`verify_key()`*, only the end user's IP address and user-agent string are required to be passed as arguments (a third argument, `blog`, will be automatically inserted for you). However, these methods all accept a large set of optional keyword arguments, corresponding to additinoal data accepted by the Akismet web service. This set of arguments is identical across all the API methods.

Akismet recommends sending as many of these arguments as possible, as additional data helps with identification of spam and training the service.

For a full list of the supported arguments, see the Akismet web service documentation.

The most commonly useful arguments are:

- `comment_author` – a string containing the name or username of the person posting the comment.
- `comment_content` – a string containing the contents of the commant.
- `comment_type` – a string indicating the type of comment. For typical site comments, set this to `"comment"`. For a contact form, use `"contact-form"`. For a user-account signup, use `"signup"`.

Using `akismet`

class `akismet.`**`Akismet`**

This is the wrapper class for the Akismet API. Instantiating it requires two parameters: your Akismet API key and the URL that key is associated with. You can pass these as the keyword arguments `key` and `blog_url` when instantiating `Akismet`, like so:

```
import akismet

akismet_api = akismet.Akismet(key='your API key', blog_url='http://yoursite.com')
```

You can also configure via environment variables: to do so, place the API key in the environment variable `PYTHON_AKISMET_API_KEY`, and the URL in the environment variable `PYTHON_AKISMET_BLOG_URL`.

Instantiating `Akismet` will automatically verify your API key and URL with the Akismet web service. If you do not supply an API key and/or URL, *`ConfigurationError`* will be raised. If your API key and URL are not valid, *`APIKeyError`* will be raised.

Methods for using the API are:

classmethod verify_key(*key*, *blog_url*)
Verifies an Akismet API key and URL. Although this is done automatically during instantiation, you can also use this method to check a different key and URL manually.

Returns `True` if the key/URL are valid, `False` if they are invalid.

If `blog_url` is not a full URL including the `http://` or `https://` protocol, *`ConfigurationError`* will be raised.

Parameters

- **key** (`str`) – The API key to verify.
- **blog_url** (`str` containing an HTTP or HTTPS URL) – The URL the key is associated with.

Return type `bool`

comment_check(*user_ip*, *user_agent*, ***kwargs*)
Checks a comment to determine whether it is spam.

This method accepts the full range of *optional arguments to the Akismet API service* in addition to its two required arguments.

Returns `True` if the comment is classified as spam, `False` if it is not.

Parameters

- **user_ip** (`str`) – The IP address of the user posting the comment.
- **user_agent** (`str`) – The HTTP `User-Agent` header of the user posting the comment.

Return type `bool`

submit_spam(*user_ip*, *user_agent*, ***kwargs*)
Informs Akismet that a comment (which it had classified as not spam) is in fact spam.

This method accepts the full range of *optional arguments to the Akismet API service* in addition to its two required arguments.

Returns `True` on a successful submission, and raises *`ProtocolError`* otherwise.

Parameters

- **user_ip** (`str`) – The IP address of the user posting the comment.
- **user_agent** (`str`) – The HTTP `User-Agent` header of the user posting the comment.

Return type `bool`

submit_ham(*user_ip*, *user_agent*, ***kwargs*)
Informs Akismet that a comment (which it had classified as spam) is in fact not spam.

This method accepts the full range of *optional arguments to the Akismet API service* in addition to its two required arguments.

Returns `True` on a successful submission, and raises *`ProtocolError`* otherwise.

Parameters

- **`user_ip`** (`str`) – The IP address of the user posting the comment.
- **`user_agent`** (`str`) – The HTTP `User-Agent` header of the user posting the comment.

Return type `bool`

Exceptions

To represent different possible error conditions, `akismet` provides several exception classes:

class `akismet.`**`AkismetError`**
Base class for all exceptions directly raised by `akismet`. Other exceptions may still occur (for example, due to network unavailability or timeout), and will not be caught by `akismet` or replaced with this exception.

class `akismet.`**`ProtocolError`**
Subclass of *`AkismetError`* indicating an unexpected or non-standard response was received from the Akismet web service. The message raised with this exception will include the API method invoked, and the contents of the unexpected response.

class `akismet.`**`ConfigurationError`**
Subclass of *`AkismetError`* indicating that the supplied configuration is missing or invalid. The message raised with this exception will provide details of the problem.

class `akismet.`**`APIKeyError`**
Subclass of *`ConfigurationError`* to indicate the specific case of an invalid API key.

Upgrading from previous versions

Prior to 1.0, the last release of `akismet` was in 2009. If you were still using that release (0.2.0), there are some changes you'll need to be aware of when upgrading to 1.0.

Configuration via file no longer supported

In 0.2.0, `akismet` supported configuration via a file named `apikey.txt`. Support for this has been removed in favor of either explicitly configuring via arguments as the *`Akismet`* class is instantiated, or configuring via environment variables. If you were relying on an `apikey.txt` file for configuration, you will need to switch to explicit arguments or environment variables.

Custom user agent no longer supported

In 0.2.0, `akismet` allowed you to specify the string which would be sent in the `User-Agent` HTTP header. The Akismet web service documentation now recomends a standard format for the `User-Agent` header, and as a result this is no longer directly configurable. The `User-Agent` string of `akismet` will now be based on the Python version and the version of `akismet`, in accordance with the Akismet service's recommendation. For example, `akismet` 1.0 on Python 3.5 will send the string `Python/3.5 | akismet.py/1.0`.

If you do need to send a custom `User-Agent`, you can subclass *`Akismet`* and change the attribute `user_agent_header` to a dictionary specifying the header you want. For example:

```
import akismet

class MyAkismet(akismet.Akismet):
    user_agent_header = {'User-Agent': 'My Akismet application'}
```

`requests` is now a dependency

Prior versions of `akismet` were implemented solely using modules in the Python standard library. As the Python standard library's support for easily performing HTTP requests is poor, `akismet` as of 1.0 has a dependency on the requests library, which will be automatically installed for you when you install a packaged copy of `akismet` 1.0.

API changes

Finally, the public API of `akismet` has been modified to match the current interface of the Akismet web service. This has resulted in the removal of one public method of *`Akismet`* – `setAPIKey` – and changes to the argument signatures of other methods.

For details of the updated interface, consult *the usage overview document*.

Frequently asked questions

The following notes answer common questions, and may be useful to you when using `akismet`.

What versions of Python are supported?

The 1.0 release of `akismet` supports the following versions of Python:

- 2.7
- 3.3
- 3.4
- 3.5
- 3.6

Older versions of Python are not supported; attempting to use `akismet` 1.0 on Python 2.6, or Python 3.0-3.2, will cause errors.

Do I have to send all the optional arguments?

The Akismet web service supports a large number of optional arguments to provide additional information for classification and training. You can *send these arguments* when calling *`comment_check()`*, *`submit_spam()`*, or *`submit_ham()`*. The Akismet documentation recommends sending as much information as possible, though only the `user_ip` and `user_agent` arguments to those methods are actually required.

Is this only for blog comments?

The Akismet web service can handle many types of user-submitted content, including comments, contact-form submissions, user signups and more. See *the documentation of optional arguments* for details on how to indicate the type of content you're sending to Akismet.

How can I test that it's working?

If you want to verify `akismet` itself, you can run the test suite; `akismet` uses tox for testing against the full list of supported Python versions, and installs all test dependencies into the `tox` virtualenvs. You can also install the test dependencies from the `test_requirements.txt` file in the source distribution, and then execute the test suite using any standard Python test runner.

Running the test suite requires two environment variables to be set:

- `TEST_AKISMET_API_KEY` containing your Akismet API key, and
- `TEST_AKISMET_BLOG_URL` containing the URL associated with your API key.

This allows the test suite to access the live Akismet web service to verify functionality.

If you want to manually perform your own tests, you can also instantiate the *`Akismet`* class and call its methods. When doing so, it is recommended that you pass the optional keyword argument `is_test=1` to the *`comment_check()`*, *`submit_spam()`*, or *`submit_ham()`* methods; this tells the Akismet web service that you are only issuing requests for testing purposes, and will not result in any submissions being incorporated into Akismet's training corpus.

What user-agent string is sent by `akismet`?

The Akismet web service documentation recommends sending a string identifying the application or platform with version, and Akismet plugin/implementation name with version. In accordance with this, `akismet` sends an HTTP `User-Agent` based on the versions of Python and `akismet` in use. For example, `akismet` 1.0 on Python 3.5 will send `Python/3.5 | akismet.py/1.0.1`.

Does `akismet` support the "pro-tip" header?

For content determined to be "blatant" spam (and thus which does not need to be placed into a queue for review by a human), the Akismet web service will add the header `X-akismet-pro-tip: discard` to its comment-check response.

Currently, `akismet` does not recognize or expose the presence of this header, though a future version may do so.

How am I allowed to use this module?

`akismet` is distributed under a three-clause BSD license. This is an open-source license which grants you broad freedom to use, redistribute, modify and distribute modified versions of `akismet`. For details, see the file `LICENSE` in the source distribution of `akismet`.

I found a bug or want to make an improvement!

The canonical development repository for `akismet` is online at <https://github.com/ubernostrum/akismet>. Issues and pull requests can both be filed there.

Python Module Index

a

akismet, 4

Index

A

Akismet (class in akismet), 4
akismet (module), 4
AkismetError (class in akismet), 6
APIKeyError (class in akismet), 6

C

comment_check() (akismet.Akismet method), 5
ConfigurationError (class in akismet), 6

P

ProtocolError (class in akismet), 6

S

submit_ham() (akismet.Akismet method), 5
submit_spam() (akismet.Akismet method), 5

V

verify_key() (akismet.Akismet class method), 5