Pylocwolowitz Documentation

Release 0.3

Natal Ngetal

Mar 22, 2018

Contents

Contents:

- source: https://github.com/hobbestigrou/Pylocwolowitz
- ticketing: https://github.com/hobbestigrou/Pylocwolowitz/issues
- documentation: http://pylocwolowitz.readthedocs.org/en/latest/

CHAPTER 1

Documentation of the api

class pylocwolowitz.core.**Pylocwolowitz**(*path*, *format_deserializer='json'*, *default_key=None*)

Pylocwolitz is a very simple text localization system.

Pylocwolitz is a very simple text localization system, meant to be used by web applications (but can pretty much be used anywhere). Yes, another localization system.

Parameters

- **path** (*str*) – File path translation
- **format_deserializer** (*str*) – Indicate the serializer to use json or yaml
- **default_key** (*str*) – Specify a default key if the key is not found

Raises **ValueError** – Format not supported, only json or yaml.

loc(*key*, *lang*, *values=None*)

Get the translate.

Return the string key, translated to the requested language (if such a translation exists, otherwise no traslation occurs). Any other parameters passed to the method are injected to the placeholders in the string (if present).

Parameters

- **key** (*str*) – Key translate
- **lang** (*str*) – Language to translate
- **values** (*dict*) – Arguments are injected to the placeholders in the string

Returns Translated to the requested language

Return type str

CHAPTER 2

Example of i18n file

Store files in the language folder first argument specified when creating the object.

2.1 Use YAML format

Here is an example with one language per file:

```
---
"welcome {name}": "Välkommen {name}"
"Hello": "Hej"
```

An example with two language in the same file:

```
---
"welcome {name}":
  "fr": "Bienvenue {name}"
  "en": "Welcome {name}"
```

2.2 Use JSON format

Here is an example with one language per file:

```
{
    "welcome {name}": "Välkommen {name}"
    "Hello": "Hej"
}
```

An example with two language in the same file:

```
{
    "welcome {name}": {
        "fr": "Bienvenue {name}",
        "en": "Welcome {name}"
    }
}
```

CHAPTER 3

How to contrib

Choice to use a fairly close process to what is in us in other Free and Open Source Software projects.

It is recommended for a new developer or contributor to read this document.

3.1 Create a ticket

For all new features or bug fixes, fill a new ticket in the issue tracker.

Try to be specific in the ticket, with a proper title and description, if you can provide steps to help the developper to understand the process or the feature request. Without enough context, it can be difficult to understand.

The ticket number is then used as a reference for the branch and commits.

3.2 Create a branch

Except in exceptional cases, a developer must not directly commits in master.

The rule also applies to project core developers.

The branch must be prefixed by the ticket number, for example:

```
git pull origin master
git branch 425-assign_nobody
```

Don't forget to be update master before creating your new branch from it.

3.3 Unit test

Except in exceptional cases, a pull request must come with tests.

It is important to write tests, it's annoying but it pays in the long run, not regression, refactoring opportunities, and so on.

If this is a bug fix, write a test for this case.

A branch will not merged if there is no test. Learn from the many existing tests, ask questions if needed.

3.4 Code review

To err is human, several pair of eyes are better than one.

A developer should not merge in the master.

The code must be reviewed by another developer and merged by an experienced core developer.

It is possible to discuss and point out any mistakes or errors.

If we detect a mistake after the branch as been merged, we will all work together to fix it as soon as possible.

3.5 Hook to exec flake8

We want our beautiful code to be PEP8 compliant so it is strongly recommended to add this pre-commit hook in git (peopleask/.git/hooks/pre-commit):

```
#!/usr/bin/env bash
flake8 ./peopleask/
```

If not pep8 it will not be commiting

3.6 Conclusion

It is important to follow up the process. It is not to be annoying, but it opens the discussion, helps to progress and to have a better quality of code as well as letting every developer sees and valid project progress.

Do not take the critic for you, each developer make mistakes and it is fine.

Despite this process of functional bugs and errors happen and it is fine as well, never forget it.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

CHAPTER 5

Introduction

Pylocwolowitz is a port of the awesome Locale::Wolowitz module from Perl in Python. It is a very simple text localization system, meant to be used by web applications (but can pretty much be used anywhere).

Yes, this is yet another localization system.

Pylocwolowitz works with JSON and YAML files.

Each file can serve one or more languages. When creating an instance of this module, you are required to pass a path to a directory where your application's JSON localization files are present. These are all loaded and merged into one big dict, which is stored in memory. A file with only one language has to be named <*lang*>.json (where <*lang*> is the name of the language, you'd probably want to use the two-letter ISO 639-1 code). A file with multiple language can be call fr_and_es.json. The basic idea is to write your application in a base language, and use the JSON files to translate text to other languages. For example, lets say you're writing your application in English and translating it to Hebrew, Spanish, and Dutch. You put Spanish and Dutch translations in one file, and since everybody hates Israel, you put Hebrew translations alone.

Example:

```
from pylocwolowitz import Pylocwolowitz
i18n = Pylocwolowitz('./i18n')
i18n.loc('hello', 'fr')
i18n.loc('welcome {name}', 'se', {'name': 'hobbestigrou'})
```

CHAPTER 6

ACKNOWLEDGEMENTS

Thanks to Ido Perlmuter to Locale::Wolowitz. Thanks you to **Julien Tayon** for his contributions. A big thank you to **Victor Stinner**, **Marmotte** and **Ggreg** for their invaluable advice.

Index

L

P