The GMU PYPI Template Documentation

Release 1.2.0

Mybrid Wonderful (The GMU), Gregg Yearwood (The GMU)

Nov 03, 2020

Contents:

1 The GMU PyPi Template 3

2 Introduction 5

3 'fun_stuff' Sample Instructions 7

4 Requirements 9

5 Makefile Options 11

6 Sample Module 13
  6.1 sample.py 13

7 Sphinx Notes 15

8 Sphinx Inintialization 17

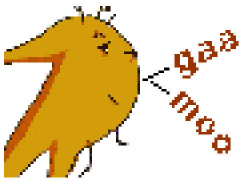
9 Sphnix Expectations 19

10 Sphinx White Space 21

11 Indices and tables 23

Python Module Index 25

Index 27



`thegmu-pypi-template` is a general Python project layout to create a PYPI project and is used for all PYPI Python projects at The Gregg & Mybrid Upgrade, Inc. (The GMU). The Makefile provided can immediately publish a package file to https://pypi.org, assuming you have an account.

- Documentation: http://the-gmu-pypi-template.readthedocs.org/
- Source Code: https://bitbucket.org/thegmu/thegmu-pypi-template
- Download: https://pypi.python.org/pypi/thegmu-pypi-template

Please feel free to ask questions via email: (mybrid@thegmu.com)

CHAPTER 1

The GMU PyPi Template

https://www.thegmu.com/

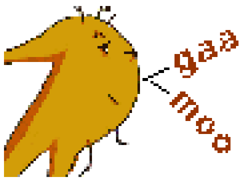
Authors Mybrid Wonderful, Gregg Yearwood

Date 11/03/2020

Support mybrid@thegmu.com

Version 1.3.0

- Documentation: http://the-gmu-pypi-template.readthedocs.org/
- Source Code: https://bitbucket.org/thegmu/thegmu-pypi-template



CHAPTER 2

Introduction

The GMU PyPi Template s a general Python project layout to create a PYPI project and is used for all PYPI Python projects at The Gregg & Mybrid Upgrade, Inc. (The GMU). The Makefile provided can immediately publish a package file to https://pypi.org, assuming you have an account.

Feel free to clone the source code to make this your own:

```
git clone https://bitbucket.org/thegmu/thegmu-pypi-template
```

CHAPTER 3

'fun_stuff' Sample Instructions

```
# Create /tmp/fun_stuff sample project
# 1. Create a Python3 virtualenv.
mkdir -p /tmp/venv
python3 -m venv /tmp/venv/py3_fun_stuff


# 3. Activate the Python3 virtualenv.
. /tmp/venv/py3_fun_stuff/bin/activate

# 2. Create an empty Python3 project directory.
mkdir -p /tmp/git/fun_stuff

# 5. Change directory into the empty Python3 project directory.
cd  /tmp/git/fun_stuff

# 4. Install thegmu-pypi-template.
# TEST: pip install --index-url https://test.pypi.org/simple/ thegmu-pypi-template
pip install thegmu-pypi-template


# 6. Run the install script to install the files into the current directory.
thegmu-pypi-template

# 7. Activate the PYPI environment from directory /tmp/fun_stuff
. bin/activate-fun-stuff

# 8. Validate the fun-stuff configuration by running the following make commands.
make init
make test
make dist
make test-dist

# 9. Repeat the above for your new project configuration your development environment.
#     Follow the same steps above but:
```

(continues on next page)

(continued from previous page)

```
#     Replace '/tmp/venv/fun_stuff' with your Python development virtualenv root␣
↪directory and project name, i.e. "$HOME/venv/py3_my_project".
#     Replace '/tmp/git/fun_stuff' with your development source code repository root␣
↪directory, i.e. "$HOME/workspace/git/my_project".
#     Create a new activation file and update the environment variables.
#     cp bin/activate-fun-stuff bin/activate-my-projectname
#     replace '. bin/activate-fun-stuff' with '. bin/activate-my-projectname'

# 10. To make Sphinx documentation.
make backup-docs # Create backup to compare your files with.
make destroy-docs # Delete everything under docs/
make docs # Runs the Sphinx wizard to initialize the conf.py file and then 'make html
↪'.
# Open HTML index file in browser: docs/_build/html/index.html
```

CHAPTER 4

Requirements

1. Encourage Python standards are followed for packaging and source.
2. Pylint for validating PEP8 standards of code.
3. Sphinx documentation to integrate with readthedocs.org
4. Test automation in the dedicated tests directory.
5. PyPi package deployment.
6. ReadTheDocs documentation deployment.

Tools:

1. **autopep8**: pep8 code beautifier
2. **pylint**: coding standards
3. **pytest**: test source
4. **readthedocs.org**: public documentation using sphinx
5. **sphinx**: html documentation
6. **tox**: test the source as installed package
7. **twine**: deploy the package to pypi.org, test.pypi.org
8. **Makefile**: run the tools

Configuration files:

1. **.gitignore**: ignore pylint, pytest, tox and build files as well .settings, .project, and .pydevproject directories from Eclipse.
2. **.pylintrc**: The GMU specific PEP8 suppression.

CHAPTER 5

Makefile Options

make <option>

_default: Same as help.

backup-docs: Create a temp directory, 'docs.tmp.XXX', using mktemp and copy the docs directory to it.

clean: Removes Python compiled files, pytest files, and tox test files. See clean-pyc and clean-tox.

clean-dist: Removes Python packaging files.

clean-docs: Removes sphinx documentation build files. Configuration files are not removed.

clean-pyc: Removes Python compiled files and pytest files.

clean-tox: Removes tox test files.

destroy-docs Removes all sphinx config and manually edited document files as well as all generated files. See clean-docs. See backup-docs.

dist: Creates source and binary Python packages suitable for PyPi.

docs: Build the the HTML documentation files in docs/_build.

help: Displays this file.

init:

1. Install Python tools used by this Makefile.
2. Run project-init, see project-init.

pep8: Run `autopep8` and update all the project and test files in place with white space changes.

project-init:

1. setup.py: NAME, AUTHOR, AUTHOR_EMAIL, URL, SCRIPTS all updated.
2. test/sample_test.py: import of project name updated.
3. tox.ini: envlist updated

publish:

1. Publish the package to production 'pypi.org'.
2. User name and password prompt are given.

publish-test: Publish the package to test 'test-pypi.org'. User name and password prompt are given.

pylint: Run `pylint` and output results. No other action is taken. See `pep8` option to fix white space problems.

requirements: Python 'pip' packages for the tools.

test: Run the tests from source using pytest.

test-dist: Run the tests from virtual envinorments using tox. Builds the package and then run the test as packages in temporary Python virtualenv environments.

upgrade: Upgrade Python 'pip' packages for the tools.

> The reasonable person adapts themself to the world; the unreasonable one persists in trying to adapt the world to themself. Therefore all progress depends on the unreasonable person. –George Bernard Shaw

The End

CHAPTER 6

Sample Module

6.1 sample.py

Sample module provided for testing purposes.

class thegmu_pypi_template.sample.**Sample**

Bases: object

Sample object for pytest.

static hello(*name='The GMU Project'*)

This sample function returns a string of "Hello 'name'".

Parameters **name** – A optional string for hello testing.

CHAPTER 7

Sphinx Notes

Sphinx automatic document generation from Python source files requires steps beyond the documentation on the Sphinx web site when storing documents in the “docs” directory.

CHAPTER 8

Sphinx Inintialization

This package downloads 12MB of documentation specific to this *thegmu-pypi-template* package for comparison and debugging purposes. Initializing the Sphinx documents for your project is best accomplished by first backing up this package documentation and then initializing from a clean "docs" directory. Once you are on solid ground updating the Sphinx documents then just delete the backup directory.

```
# Backup The GMU PYPI Template in directory 'docs'.
make backup-docs
# Remove all files from directory 'docs'.
make destroy-docs
# Initialize for your project. You will need the following information:
# Project Name:
# Author Names:
# Version: 0.1.0
make docs
# Below are wizard responses to the prompts expected by the Makefile.
# Different answers will most likely work but have not been test.

Please enter values for the following settings (just press Enter to
accept a default value, if one is given in brackets).

Selected root path: .

You have two options for placing the build directory for Sphinx output.
Either, you use a directory "_build" within the root path, or you separate
"source" and "build" directories within the root path.
> Separate source and build directories (y/n) [n]:

Inside the root directory, two more directories will be created; "_templates"
for custom HTML templates and "_static" for custom stylesheets and other static
files. You can enter another prefix (such as ".") to replace the underscore.
> Name prefix for templates and static dir [_]:

The project name will occur in several places in the built documentation.
> Project name: <--- YOUR PROJECT NAME
```

(continues on next page)

(continued from previous page)

```
> Author name(s): <--- AUTHOR NAME
> Project release []: <--- VERSION NUMBER, i.e. 0.1.0

If the documents are to be written in a language other than English,
you can select a language here by its language code. Sphinx will then
translate text that it generates into that language.

For a list of supported codes, see
http://sphinx-doc.org/config.html#confval-language.
> Project language [en]

The file name suffix for source files. Commonly, this is either ".txt"
or ".rst".  Only files with this suffix are considered documents.
> Source file suffix [.rst]:

One document is special in that it is considered the top node of the
"contents tree", that is, it is the root of the hierarchical structure
of the documents. Normally, this is "index", but if your "index"
document is a custom template, you can also set this to another filename.
> Name of your master document (without suffix) [index]: index
Indicate which of the following Sphinx extensions should be enabled:
> autodoc: automatically insert docstrings from modules (y/n) [n]: y
> doctest: automatically test code snippets in doctest blocks (y/n) [n]: y
> intersphinx: link between Sphinx documentation of different projects (y/n) [n]: y
> todo: write "todo" entries that can be shown or hidden on build (y/n) [n]: y
> coverage: checks for documentation coverage (y/n) [n]: y
> imgmath: include math, rendered as PNG or SVG images (y/n) [n]: n
> mathjax: include math, rendered in the browser by MathJax (y/n) [n]: n
> ifconfig: conditional inclusion of content based on config values (y/n) [n]: n
> viewcode: include links to the source code of documented Python objects (y/n) [n]: y
> githubpages: create .nojekyll file to publish the document on GitHub pages (y/n)␣
↪[n]: n

A Makefile and a Windows command file can be generated for you so that you
only have to run e.g. `make html' instead of invoking sphinx-build
directly.
> Create Makefile? (y/n) [y]:
> Create Windows command file? (y/n) [y]: n

Creating file ./conf.py.
Creating file ./index.rst.
Creating file ./Makefile.

Finished: An initial directory structure has been created.
```

CHAPTER 9

Sphnix Expectations

Sphinx documentation will most likely not be satisfactory when using the automatic generation tools. The ‘Makefile’ attempts to alleviate some of this by making the follow changes:

1. **Read The Docs**: Installs and configures the ‘Read The Docs’ template.
2. **Import source path**: Sphinx assumes it runs at the source root, but the documents are in a sub-directory, “docs”. In order to auto-discover source files then the path “..” needs to be added to the Python “sys.path” in “conf.py”. The Makefile adds “..” to the sys.path in the “conf.py” file.
3. **Initial Python source doc generation**: The Sphinx automatic generation of class and module documentation from doc strings needs to be run manually the first time. The Sphinx wizard leaves one the with an impression that just answering ‘yes’ to a wizard prompt is good enough. It is not, a manual run is required. However, after the initial generation of source file docs then these files will be automatically updated on source documentation changes on subsequent document builds with “make html”.
4. **Excludes modules.rst**: There is a directive in the index.rst file that includes the modules.rst file, “*:‘ref:modindex‘”. This makes including the “modules.rst” file redundant and and so that file is added to the ‘conf.py’, exclude list.

The only files that need to be manually edited are the ReST (.rst) files in the docs, top-level directory, for example:

```
docs/index.rst docs/modules.rst  docs/thegmu_pypi_template.rst
```

If new files are added then the file names with ‘.rst.’ need to be list in the ‘index.rst’ file.

CHAPTER 10

Sphinx White Space

The Sphinx documentation syntax is white space sensitive.

1. The number of spaces for indentation when continuing a block must be exact.
2. Extra new lines after a block are not allowed and the block continuation lines will not be read.

CHAPTER 11

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

thegmu_pypi_template.sample, 13

Index

H

`hello()` (*thegmu_pypi_template.sample.Sample static method*), 13

S

`Sample` (*class in thegmu_pypi_template.sample*), 13

T

`thegmu_pypi_template.sample` (*module*), 13