WeChat JAVA SDK Kit Guide

rc

Larry.Koo

9 23, 2017

Contents

jfinal-weixin-sdk

CHAPTER 1

- genindex
- search

access_token

```
AccesssTokenApi.ice.getAccessToken(String appid, String secret);
```

Readability is a primary focus for Python developers, in both project and code documentation. Following some simple best practices can save both you and others a lot of time.

A `README` file at the root directory should give general information to the users and the maintainers. It should be raw text or written in some very easy to read markup, such as *reStructuredText* and Markdown. It should contain a few lines explaining the purpose of the project or the library (without assuming the user knows anything about the project), the url of the main source for the software, and some basic credit information. This file is the main entry point for readers of the code.

An `INSTALL` file is less necessary with python. The installation instructions are often reduced to one command, such as `pip install module` or `python setup.py install` and added to the `README` file.

A `LICENSE` file should *always* be present and specify the license under which the software is made available to the public.

A `TODO` file or a `TODO` section in `README` should list the planned development for the code.

A `CHANGELOG` file or section in `README` should compile a short overview of the changes in the code base for the latest versions.

Depending on the project, your documentation might include some or all of the following components:

- A *introduction* should show a very short overview of what can be done with the product, using one or two extremely simplified use cases. This is the thirty-second pitch for your project.
- A *tutorial* should show some primary use cases in more detail. The reader will follow a step-by-step procedure to set-up a working prototype.
- An *API reference* is typically generated from the code (see *docstrings*). It will list all publicly available interfaces, parameters, and return values.
- *Developer documentation* is intended for potential contributors. This can include code convention and general design strategy of the project.

Sphinx

Sphinx is far and away the most popular python documentation tool. **Use it.** It converts *reStructuredText* markup language into a range of output formats including HTML, LaTeX (for printable PDF versions), manual pages, and plain text.

There is also **great**, **free** hosting for your Sphinx docs: Read The Docs. Use it. You can configure it with commit hooks to your source repository so that rebuilding your documentation will happen automatically.

: Sphinx is famous for its API generation, but it also works well for general project documentation. This Guide is built with Sphinx and is hosted on Read The Docs

---

reStructuredText

Most Python documentation is written with reStructuredText. It's like Markdown with all the optional extensions built in.

The reStructuredText Primer and the reStructuredText Quick Reference should help you familiarize yourself with its syntax.

Comments clarify code and begin with a hash (`#`). In Python, *docstrings* describe modules, classes, and functions:

```
def square_and_rooter(x):
    """Returns the square root of self times self."""
    ...
```

In general, follow the comment section of PEP 0008 (the "Python Style Guide").

Do not use triple-quote strings to comment code. This is not a good practice, because line-oriented command-line tools such as grep will not be aware that the commented code is inactive. It is better to add hashes at the proper indentation level for every commented line. Your editor probably has the ability to do this easily, and it is worth learning the comment/uncomment toggle. (*e.g.* ctrl-v on Vim)

Some tools use docstrings to embed more-than-documentation behavior, such as unit test logic. Those can be nice, but you won't ever go wrong with vanilla "here's what this does."

These aren't interchangeable. For a function or class, the leading comment block is a programmer's note. The docstring describes the operation of the function or class:

```
# This function slows down program execution for some reason.
def square_and_rooter(x):
    """Returns the square root of self times self."""
    ...
```

:

Further reading on docstrings: PEP 0257

You might see these in the wild. Use *Sphinx*.

Pycco Pycco is a "literate-programming-style documentation generator" and is a port of the node.js Docco. It makes code into a side-by-side HTML code and documentation.

Ronn Ronn builds unix manuals. It converts human readable textfiles to roff for terminal display, and also to HTML for the web.

Epydoc Epydoc is discontinued. Use *Sphinx* instead.

Theis-guide is under active development, and contributors are welcome.

For all contributions, please follow the .

If you'd like to contribute, there's plenty to do. Here's a short todo list.

- Establish "use this" vs "alternatives are...." recommendations

As with all documentation, having a consistent formating helps make the document more understandable. In order to make The Guide easier to digest, all contributions should fit within the rules of this style guide where appropriate.

The Guide is written as *reStructuredText* .

: Parts of The Guide may not yet match this style guide. Feel free to update those parts to be in sync with The Guide Style Guide

: On any page of the rendered HTML you can click "Show Source" to see how authors have styled the page.

Strive to keep any contributions relevant to the .

- Avoid including too much information on subjects that don't directly relate to Python development.
- Prefer to link to other sources if the information is already out there. Be sure to describe what and why you are linking.
- Cite references where needed.
- If a subject isn't directly relevant to Python, but useful in conjunction with Python (ex: Git, Github, Databases), reference by linking to useful resources and describe why it's useful to Python.
- When in doubt, ask.

Use the following styles for headings.

Chapter title:

```
#########
Chapter 1
#########
```

Page title:

```
===================
Time is an Illusion
===================
```

Section headings:

```
Lunchtime Doubly So
-------------------
```

Sub section headings:

```
Very Deep
~~~~~~~~~
```

Wrap text lines at 78 characters. Where necessary, lines may exceed 78 characters, especially if wrapping would make the source text more difficult to read.

Wrap all code examples at 70 characters to avoid horizontal scrollbars.

Command line examples:

```
.. code-block:: console

    $ run command --help
    $ ls ..
```

Be sure to include the `$` prefix before each line.

Python interpreter examples:

```
Label the example::

.. code-block:: python

    >>> import this
```

Python examples:

```
Descriptive title::

.. code-block:: python

    def get_answer():
        return 42
```

- Prefer labels for well known subjects (ex: proper nouns) when linking:

  ```
  Sphinx_ is used to document Python.

  .. _Sphinx: http://sphinx.pocoo.org
  ```

- Prefer to use descriptive labels with inline links instead of leaving bare links:

```
Read the `Sphinx Tutorial <http://sphinx.pocoo.org/tutorial.html>`_
```

- Avoid using labels such as “click here”, “this”, etc. preferring descriptive labels (SEO worthy) instead.

“”

To cross-reference other parts of this documentation, use the :ref: keyword and labels.

To make reference labels more clear and unique, always add a `-ref` suffix:

```
.. _some-section-ref:

Some Section
------------
```

Make use of the appropriate admonitions directives when making notes.

Notes:

```
.. note::
    The Hitchhiker's Guide to the Galaxy has a few things to say
    on the subject of towels. A towel, it says, is about the most
    massively useful thing an interstellar hitch hiker can have.
```

Warnings:

```
.. warning:: DON'T PANIC
```

TODOs

Please mark any incomplete areas of The Guide with a todo directive.

To avoid cluttering the , use a single `todo` for stub documents or large incomplete sections.

```
.. todo::
    Learn the Ultimate Answer to the Ultimate Question
    of Life, The Universe, and Everything
```