파이썬 문서화 (Sphinx 활용)

1. 목표

파이썬으로 개발된 모듈 등의 API를 문서화한다.

문서화의 목적은 개발시 생성한 모듈의 재사용을 개발 당사자 뿐 아니라 동료들도 적극적으로 활용하고 이용함으로써 반복되는 개발 작업을 최소화하고, 코딩 실력을 키우며, 코딩 스타일을 평준화하기 위한 것이다.

본 위키에서는 Sphinx를 다루는데, 그 이유는 이것이 현재 가장 많이 이용되는 도큐먼트 툴이기 때문이다.

2. Sphinx 설치

설치를 위해서는 일부 요구사항을 충족해야 한다.

> Python
>
> Pip

위의 요구사항이 모두 깔려있다는 전제하에 Sphinx를 설치한다.

> pip install Sphinx, spphinxcontrib-websupport

만일 설치 중 문제가 있다면 pip버전이 낮을 경우 발생할 가능성이 크다. 따라서 pip를 업그레이드해서 다시 인스톨을 시도한다.

> pip install --upgrade pip

3. 프로젝트 소스코드 주석

모듈의 클래스, 함수 등을 문서화하기 위해서는 당연히 각 요소를 설명하는 내용이 있어야 한다. 내용은 클래스, 함수의 바로 밑에 주석으로 작성된다. 다음은 주석의 예제이다.

```
class RnnAttnModel(object):
    """
    RNN Attention model
    """
    def __init__(self, config=None, name='', vocab_size=10000, reuse=False):
        """

        :param config: Uses ModelConfig class as configuration
        :param name: Custom name of the model
        :param vocab_size: Size of vocabulary
        :param reuse: Reuse option of tensorflow variables
        """

    def infer_vector_run(self, docs2idx, idx_dict):
        """
         .


        :param docs2idx: document with indexed form
        :param idx_dict: index dictionary
        :return: feed dict, session input
        """
        # build mask
        mask = (docs2idx != idx_dict['<PAD>'])
        dynamic_seq_len = np.array([m[m == True].shape[0] for m in mask])

        feed_dict = {
            self.x: docs2idx,
            self.dropout_keep_prob: 1,
            self.seq_len: dynamic_seq_len,
            self.mask: mask,

        }

        session_input = [
            self.relevant_outputs
        ]

        return feed_dict, session_input
```

위와 같이 문서화를 해야하는 모든 모듈에 주석을 달아주자.

주의할 점이 두 가지 있다.

1. 설명 문구와 param 사이에는 한 칸 공백 라인이 있어야 한다.

2. 설명 문구의 끝에 점(.)을 찍지 말자. 도큐먼트가 깨진다.

4. Sphinx로 문서화 하기

해당 프로젝트 폴더로 이동하여 다음을 실행한다.

> sphinx-quickstart

아래와 같이 나타나면, project name과 version, 그리고 author name 등 필요한 파트만 변경하고 나머지는 똑같이 따라한다.

(만일 Root path for the documentation이라는 문구가 나오지 않을 경우에는 해당 프로젝트 폴더에서 docs 폴더를 따로 생성하고 docs 폴더로 이동하여 아래를 계속 진행한다.)

```
X:\01_??\NURILAB_Project\HwpScan2_Pro\Engine>C:\Python27\Scripts\sphinx-quicksta
rt.exe
Welcome to the Sphinx 1.4.1 quickstart utility.

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

Enter the root path for documentation.
> Root path for the documentation [.]: docs

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: Engine
> Author name(s): Kei Choi

Sphinx has the notion of a "version" and a "release" for the
software. Each version can have multiple releases. For example, for
Python the version is something like 2.5 or 3.0, while the release is
something like 2.5.1 or 3.0a1.  If you don't need this dual structure,
just set both to the same value.
> Project version: 1.0
> Project release [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]:

Sphinx can also add configuration for epub output:
> Do you want to use the epub builder (y/n) [n]:

Please indicate if you want to use one of the following Sphinx extensions:
> autodoc: automatically insert docstrings from modules (y/n) [n]: y
```

```
> doctest: automatically test code snippets in doctest blocks (y/n) [n]:
> 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]:
> imgmath: include math, rendered as PNG or SVG images (y/n) [n]:
> mathjax: include math, rendered in the browser by MathJax (y/n) [n]:
> ifconfig: conditional inclusion of content based on config values (y/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]:

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]:

Creating file docs\conf.py.
Creating file docs\index.rst.
Creating file docs\Makefile.
Creating file docs\make.bat.

Finished: An initial directory structure has been created.

You should now populate your master file docs\index.rst and create other documen
tation
source files. Use the Makefile to build the docs, like so:
   make builder
where "builder" is one of the supported builders, e.g. html, latex or linkcheck.



X:\01_??\NURILAB_Project\HwpScan2_Pro\Engine>
```

다음은 새로 생성된 docs 폴더로 이동하여 conf.py를 수정한다.

> cd docs
>
> vi conf.py (또는 에디터 파일로 엽니다)

상단에 다음을 삽입한다.

> import os
>
> import sys
>
> sys.path.insert(0, os.path.abspath('..'))

그런 다음 프로젝트 루트 폴더에서 다음의 명령어를 입력한다.

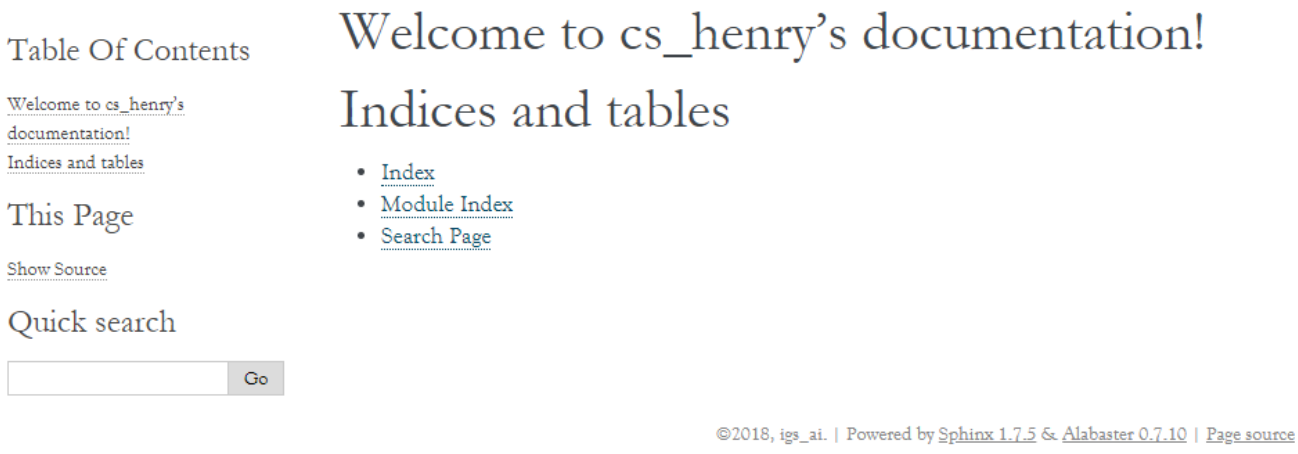
>

> sphinx-apidoc -F -o docs . --separate

그리고 docs 폴더에서 html 파일을 생성한다.

> make html

docs/_build/html 폴더로 이동하여 index.html로 이동하면 문서화된 내용을 볼 수 있다.



Quick search

Go

Python Module Index

a | c | m | s

a

api_utils

c

const

cs_emul_henry_api

m

mldl

mldl.tensorflow

mldl.tensorflow.data_loader

mldl.tensorflow.dataset

mldl.tensorflow.frame_tools

mldl.tensorflow.frame_tools.tf_activation

mldl.tensorflow.frame_tools.tf_learning

mldl.tensorflow.frame_tools.tf_optimizer

mldl.tensorflow.model

mldl.tensorflow.model.ffnn_model

mldl.tensorflow.model.rnn_attention_model

mldl.tensorflow.model.text_cnn_model

mldl.tensorflow.model_controller

mldl.tensorflow.model_saver

mldl.tensorflow.preprocessor

s

system_shares

This Page

Show Source

Quick search

Go

mldl.tensorflow.data_loader module

class `mldl.tensorflow.data_loader.`**`DataLoader`**(*config=None*, *model_type=None*, *forward_only=False*) [source]

Bases: **`object`**

Controls the data for machine learning and deep learning

`batch_loader`(*iterable*, *n=1*) [source]

Slices the data as many as the batch size n and yields. It has the same role as DataLoader of PyTorch

Parameters:
- **iterable** – Data list or other format
- **n** – batch size

Returns: yields the data in number of batch size n

`build_data_set`(*idx_dict*) [source]

Instantiate Dataset :param idx_dict: :return: None

`train_test_cv_split`() [source]

Splits train/test/cv set :return: None