오픈 소스 개발

오픈 소스 개발에 필요한 것들을 알아보자

1. `setup.py`
2. pipenv
3. lint
4. testing
5. coverage
6. git hook 사용
7. CI
8. etc

`setup.py` 작성하기

1. `setuptools`

파이썬 프로젝트 패키징 / 배포를 위한 패키지

2. `setuptools.setup`

- `setuptools.setup` 를 사용하여 패키지를 만들 수 있음.
- 패키지를 배포하면서 필요한 정보를 정의하는 함수
- `setup` 함수안에는 많은 것들이 들어감

`setuptools.setup` 인자

- `name` : 패키지 이름
- `author` , `authro_email` : 작성자 정보
- `description` , `long_description` : 패키지 설명
- `license` : 라이센스
- `url` : 패키지 주소
- `version` : 패키지 버전

> 보통 유의적 버전(semantic version)하고 비슷하게 작성

PEP396 - Module Version Numbers

- 패키지의 버전을 패키지 안에 정의해야 한다면, `__version__` 이란 이름으로 정의해야함
    - 패턴은 PEP 440 참고
      문자열이어야 함
- VCS에서 제공하는 리비전의 번호(커밋 해시)가 포함돼서는 안됨
- `setup.py` 의 version 인자는 패키지의 `__version__` 으로 부터 얻어온다

`setuptools.setup` 인자

- `packages` : 패키지 모듈
- `classifiers` : 패키지 분류

> https://pypi.org/pypi?%3Aaction=list_classifiers

- `entry_points` : 엔트리 포인트, CLI로 제공하고 싶을 때 사용
- 패키지 의존성 관리
  - `install_requires` : 패키지 실행에 필요한 패키지들
  - `tests_require` : 테스트 실행에 필요한 패키지들
  - `extras_require` : 추가로 필요한 패키지들
  - `tests_require` 및 `extras_require` 에 정의한 패키지는 기본적으로 설치되지 않음. 추가로 설치해야하는 구조

> `setup.cfg` 사용해도 괜찮음
> 패키지 메타데이터 같은 것들을 적기 위해 함수를 만들어서 사용하였는데 지시자(attr, file)를 사용하여 코드를 줄일 수 있다.

Pipenv

- `Pipfile`, `Pipfile.lock` 을 이용한 의존성 관리
- 개발에 필요한 패키지만 설치 가능 `pipenv install --dev`
- 가상환경 실행 `pipenv shell`

lint

- PEP8 준수
- pylint, flake8 등의 linter 사용하자
- black 등의 formatter 사용하자
- `.pylintrc` 를 통해 config 관리
- `pylint --error-only` 와 tox를 통해 코드 품질 관리

testing

- `unittest` 또는 `pytest` 사용
- pyenv + tox 로 다양한 파이썬 버전에서 검사
- `tox.ini` 작성

coverage

- 테스트가 충분한가를 나타내는 지표
- `pytest-cov` 사용
- 커버리지는 상태

git hook 사용

```
#!/usr/bin/env bash

isort -rc --check-only src tests
if [ $? -ne 0 ]; then
    echo "[!] isort failed!"
    exit 1
fi
echo "[+] isort success!"

black -S -l 79 --check src tests
if [ $? -ne 0 ]; then
    echo "[!] black failed!"
    exit 1
fi
echo "[+] black success!"
```

```
pylint src
if [ $? -ne 0 ]; then
    echo "[!] pylint failed!"
    exit 1
fi
echo "[+] pylint success!"

pytest tests
if [ $? -ne 0 ]; then
    echo "[!] pytest failed!"
    exit
fi
echo "[+] pytest success!"
```

CI

tox를 통해 다양한 환경에서 `pytest` 와 `pylint` 를 돌려 coverage 를 관리할 수 있다.

> ex) travis ci, circieci, appveyor

etc

- 문서화를 잘하자 - sphinx, read the docs
- 커밋 메세지
- 릴리즈와 태그
- 체인지 로그 작성
- Github 템플릿
- **뱃지 달기 - 이쁨**

Reference

오픈소스 라이브러리 개발기

나도 할 수 있다 오픈 소스! - setup.py에서 PyPI까지

뱅크셀러드 파이썬맛 레시피