We are Pythonistas

파이썬 대패키지시대,
텍스트파일 하나만 믿어도 정말 괜찮은걸까

강민철

PYCON KOREA 2020

Speaker 소개: 강민철

학생이자 직장인

- **Open Source Consulting. INC (2020.02~)**
  - System Engineer
- **지식 공유자**
  - Goorm, 멋쟁이사자처럼, inflearn
- **서울시립대학교**
  - 컴퓨터과학부 4학년 휴학생
- **Open Source Contributor**
  - Chromium, uftrace, ...

Github kangtegong
Email tegongkang@gmail.com

#1 Backgrounds

- Module vs Package vs Library
- **pip**: Python Package Installer
- What is dependency management
- Dependency managing comparison (JS , python)

Module vs Package vs Library

> A module is a file containing Python definitions and statements.

https://docs.python.org/3/tutorial/modules.html

Module vs Package vs Library

> Packages are a way of structuring Python's module namespace
>
> by using "dotted module names".

https://docs.python.org/3/tutorial/modules.html#packages

Module vs Package vs Library

> The "Python library" contains several different kinds of components.
> It contains data types that would normally be considered part of the "core" of a language,
> ...
> The library also contains built-in functions and exceptions
> ...
> The bulk of the library, however, consists of a collection of modules.

https://docs.python.org/3/library/intro.html

Module vs Package vs Library

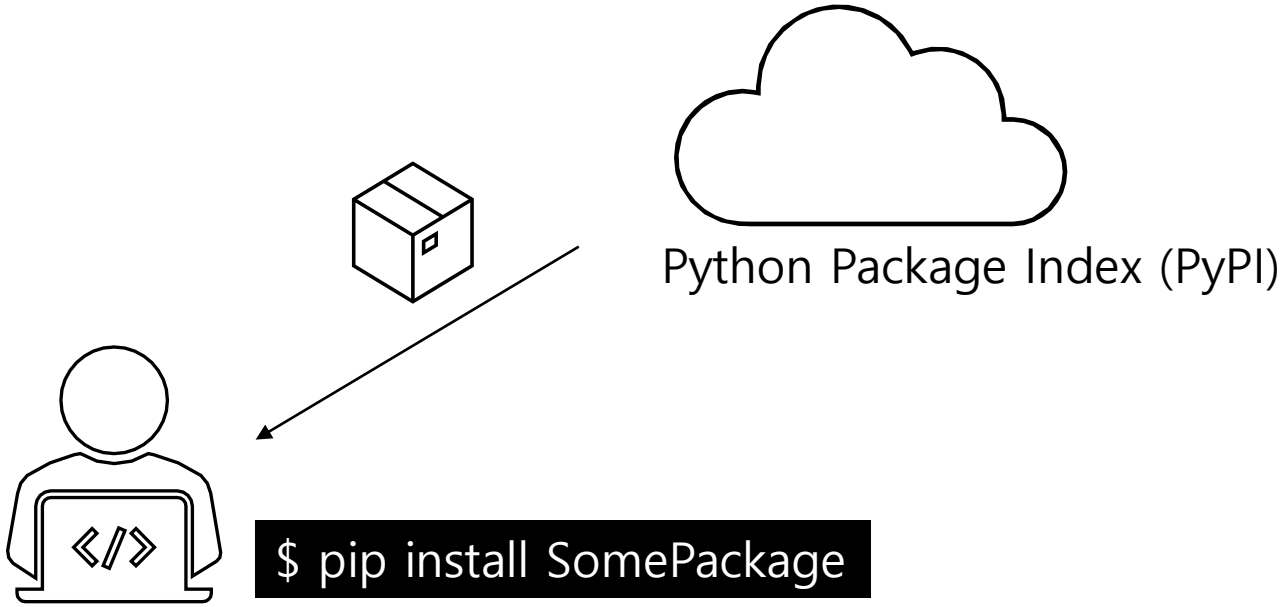
Python Standard Library

Python Package Installation

pip

Python Package Management System

Module vs Package vs Library



Python Package Installation

```
$ pip install SomePackage              # 패키지 설치
$ pip search SomePackage               # 패키지 검색
$ pip install SomePackage==1.0.4       # 특정 버전 지정하여 설치
$ pip uninstall SomePackage            # 패키지 제거
```


https://docs.python.org/3/installing/index.html

Python Package Installation

```
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/piptest
$ pip freeze
appdirs==1.4.4
certifi==2020.6.20
distlib==0.3.1
filelock==3.0.12
pipenv==2020.8.13
six==1.15.0
virtualenv==20.0.30
virtualenv-clone==0.5.4
```

$ p

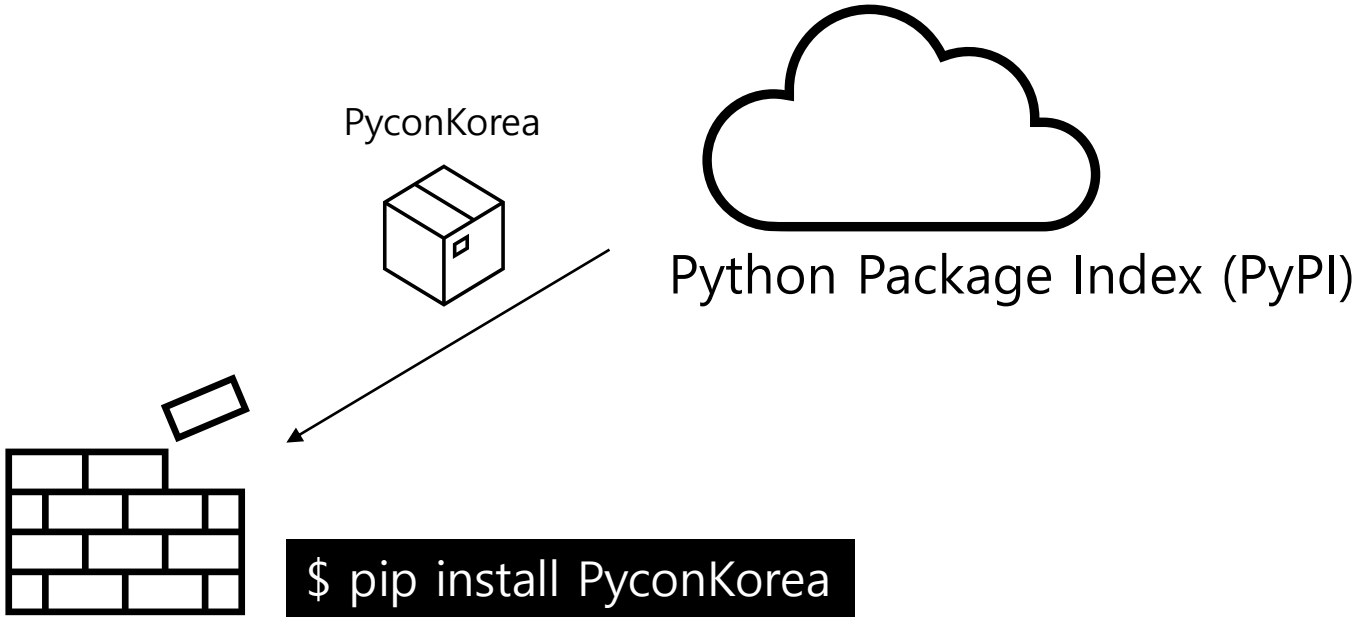
목록

Dependency Management

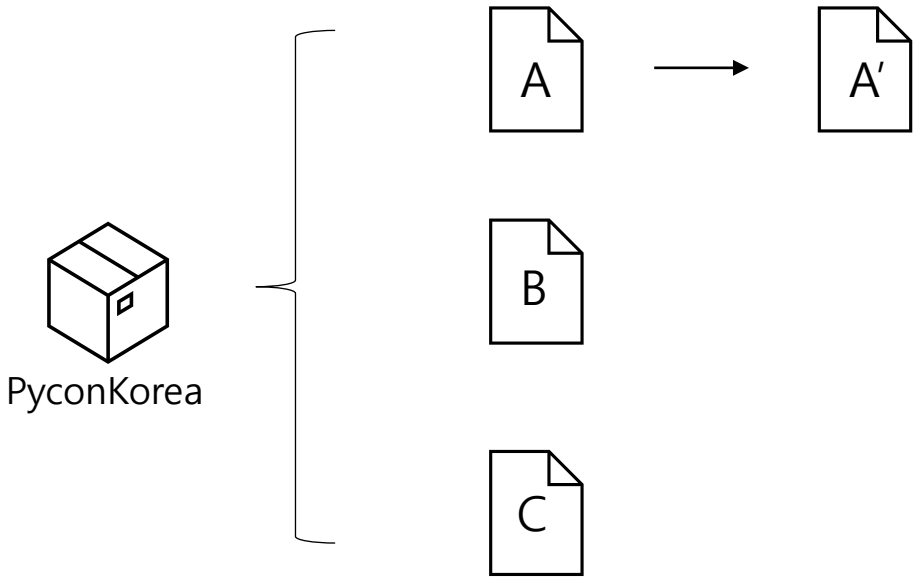
외부 라이브러리 관리

- 어떤 버전/어떤 패키지가 쓰였는가
- 어떤 버전/어떤 패키지가 쓰일 수 있나

Dependency Management

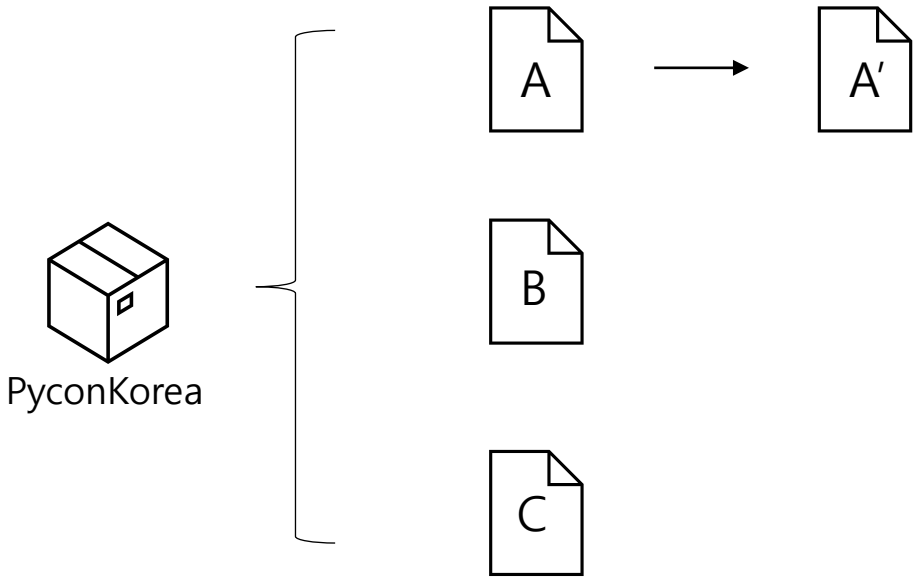


Dependency Management



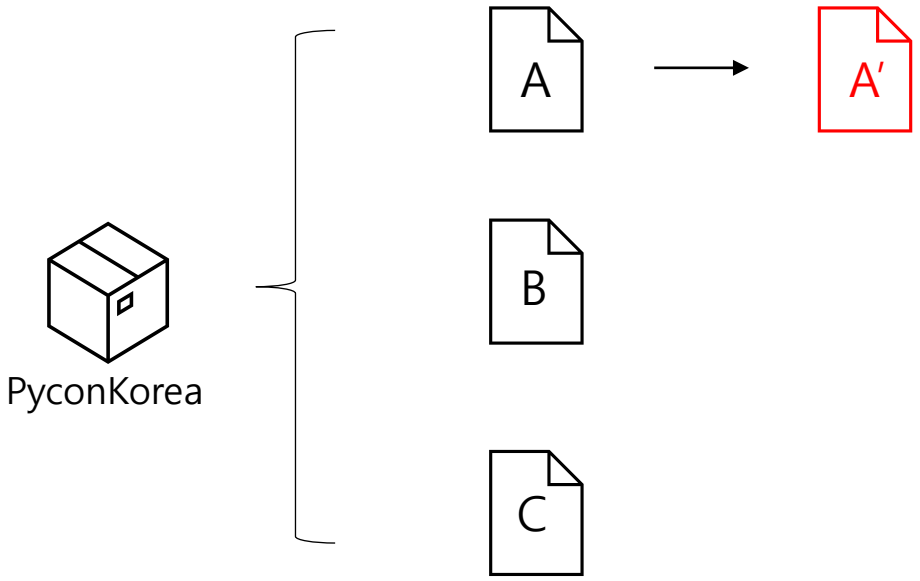
Dependency Management

i) 그 때 그 때 설치?



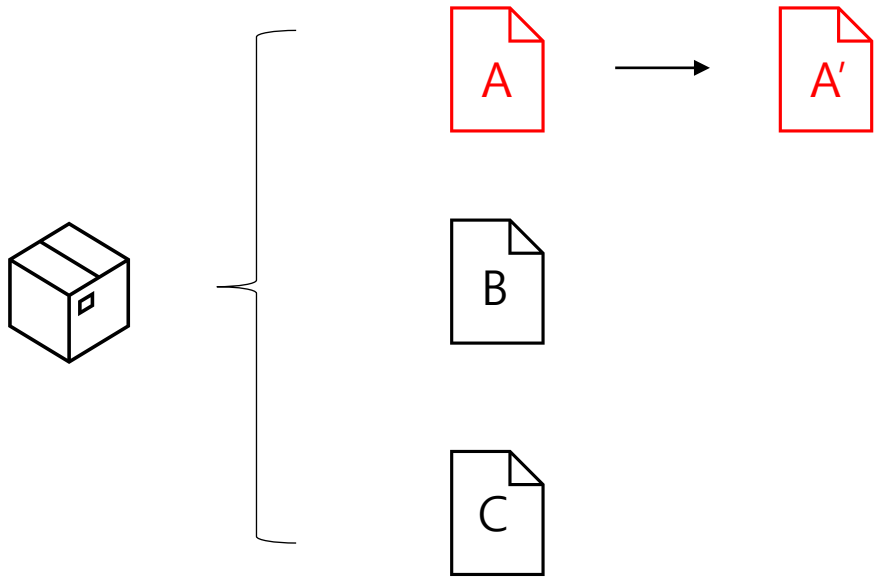
Dependency Management

i) 그 때 그 때 설치?



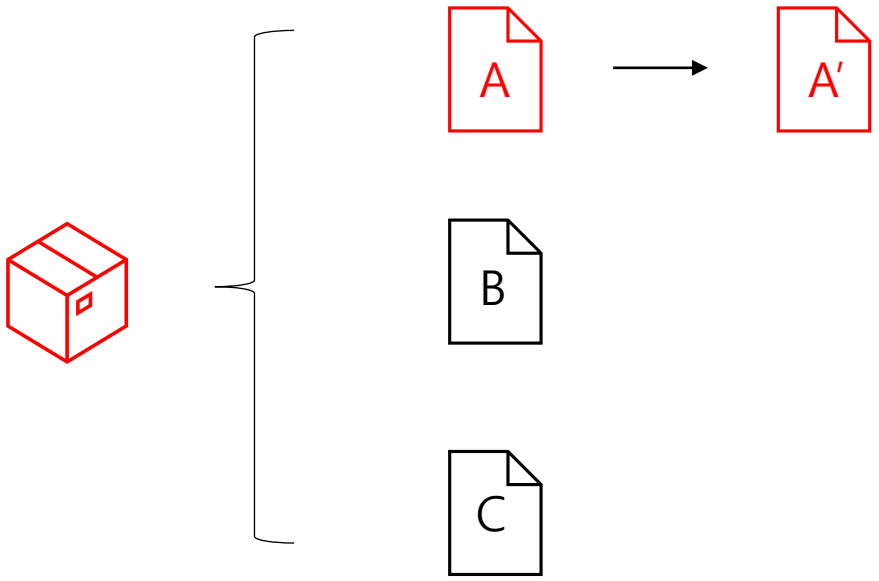
Dependency Management

i) 그 때 그 때 설치?



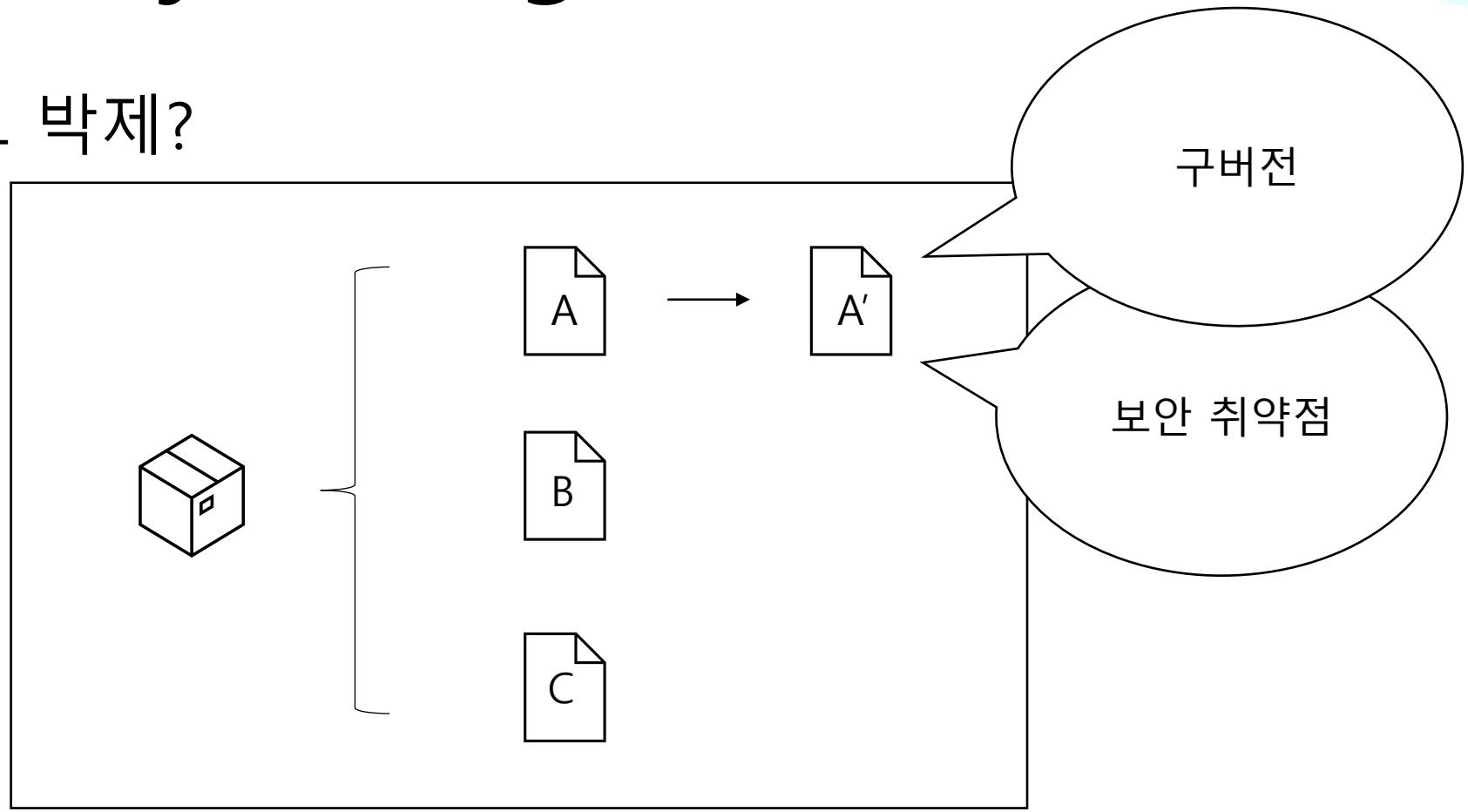
Dependency Management

i) 그 때 그 때 설치?



Dependency Management

ii) 소스코드 박제?



Dependency Management

외부 라이브러리 관리

- 어떤 버전/어떤 패키지가 쓰였는가
- 어떤 버전/어떤 패키지가 쓰일 수 있나

Dependency Management in python

```
$ pip freeze > requirements.txt
```

Dependency Management in python

```
$ pip freeze > requirements.txt
```

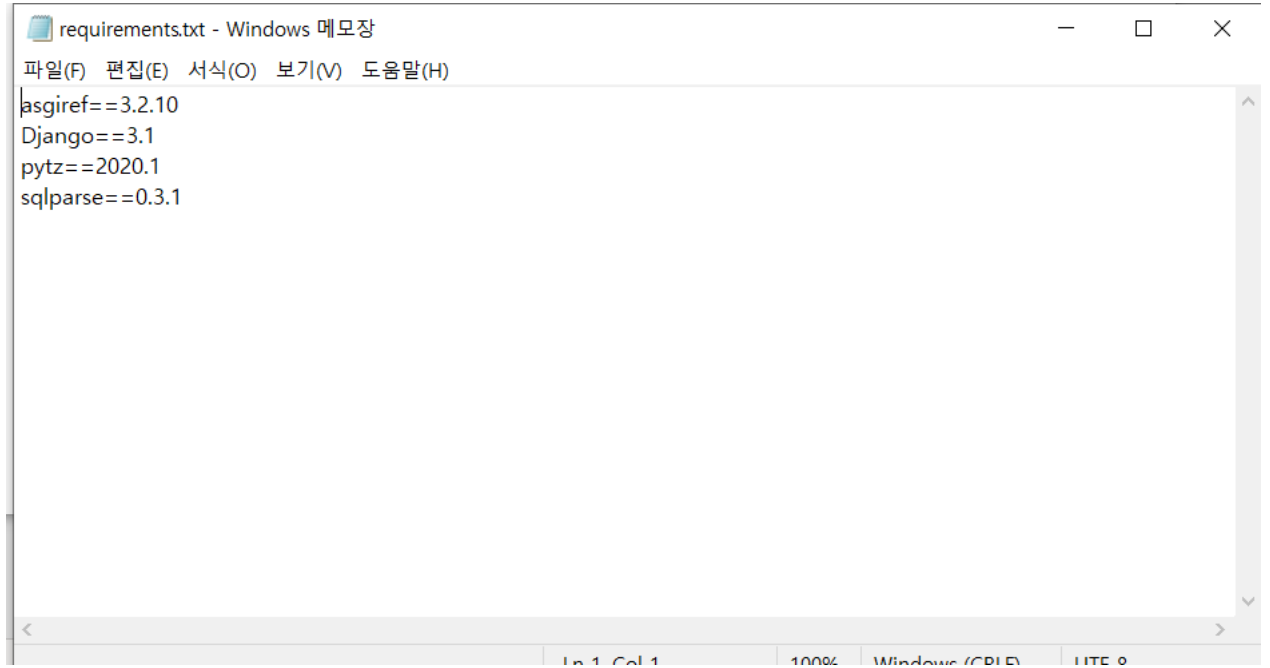
Dependency Management in python

```
$ pip freeze > requirements.txt
```

Dependency Management in python

```
$ pip freeze > requirements.txt
```

Dependency Management in python



Dependency Management in python

```
$ pip install -r requirements.txt
```

Dependency Management in python

```
$ pip install -r requirements.txt
```

Dependency Management in python

```
$ pip install -r requirements.txt
```

Dependency Managing in python

pythonkr / pyconkr

develop 11 branches 0 tags Go to file Add file Code

golony6449 Merge pull request #118 from pythonkr/feature/login_for_sponsor ... 0837fb6 4 days ago 1,252 commits

.github	Update test-deploy.yml	3 months ago
announcement	Edit announcement admin list	2 months ago
mailing	Add migration (Fix default value)	10 days ago
program	오탈자 수정	12 days ago
pyconkr	홈 화면 버튼 모바일 대응 수정	5 days ago
registration	문구 수정	6 days ago
sponsor	후원사 아이디 추가로 인한 권한 수정	5 days ago
user	문구 수정	6 days ago
.gitignore	후원사 신청하기 페이지 구현 (#39)	5 months ago
.pylintrc	로그인하지 않으면 스폰서 상세 페이지를 볼수 없던 버그를 수정합니다. (...	last month
Dockerfile	Install gettext to compile message for django	6 months ago
README.md	README 수정	last month
docker-entrypoint.sh	Unfake migration	2 months ago
fabfile.py	Fixed deploy script for 2017	3 years ago
manage.py	Initial migration from pyconapac-2016	4 years ago
package.json	Remove unused packages	6 months ago
requirements-dev.txt	Rewrite dev	6 months ago
requirements.txt	Update requirments	10 days ago
task-definition-dev.json	메일전송에 필요한 환경변수 설정	10 days ago
task-definition.json	메일전송에 필요한 환경변수 설정	10 days ago

About

This is a repository for developing pyconkr homepage.

www.pycon.kr/2020/

pyconkr

Readme

Releases

No releases published

Packages

No packages published

Contributors 39

+ 28 contributors

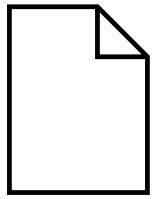
Languages

Python 56.2% HTML 33.6% CSS 6.9% JavaScript 2.9% Other 0.4%

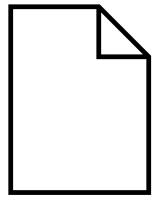
Dependency Managing in python

Dependency Managing in JavaScript

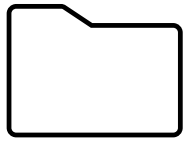
Dependency Managing in JavaScript



package.json

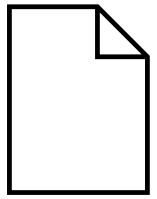


package-lock.json

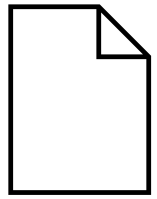


Node_modules

Dependency Managing in JavaScript



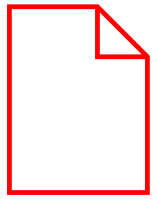
package.json



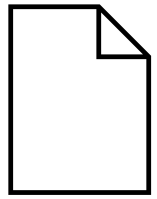
package-lock.json

Node_modules

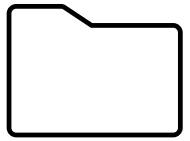
Dependency Managing in JavaScript



package.json



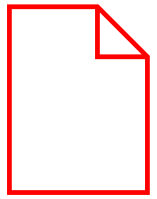
package-lock.json



Node_modules

Dependency Managing in JavaScript

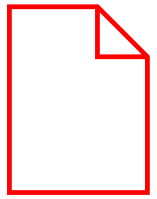
$ npm init



package.json

```
$ cat package.json
{
  "name": "npm-test",
  "version": "1.0.0",
  "description": "this is a test package for npm installation demonstration.",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/kangtegong/npm-test.git"
  },
  "keywords": [
    "npm",
    "test",
    "installation"
  ],
  "author": "Kang Minchul",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/kangtegong/npm-test/issues"
  },
  "homepage": "https://github.com/kangtegong/npm-test#readme"
}
```

Dependency Managing in JavaScript



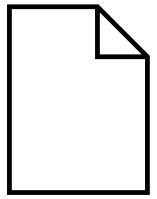
package.json

$ npm install Package

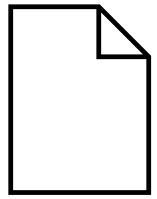
$ npm install --save-dev Package

```
$ cat package.json
{
  "name": "desktop",
  "version": "1.0.0",
  "description": "this is a test project for npm installation demo.",

  ...

  "author": "",
  "license": "ISC",
  "dependencies": {
    "npm-test": "file:npm-test"
  },
  "devDependencies": {
    "npm-check": "^5.9.2"
  }
```

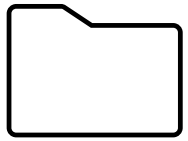
Dependency Managing in JavaScript



package.json

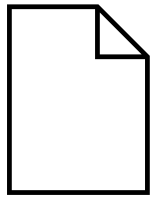


package-lock.json

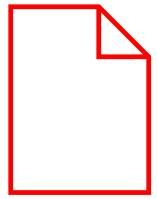


Node_modules

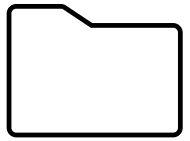
Dependency Managing in JavaScript



package.json

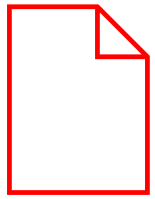


package-lock.json



Node_modules

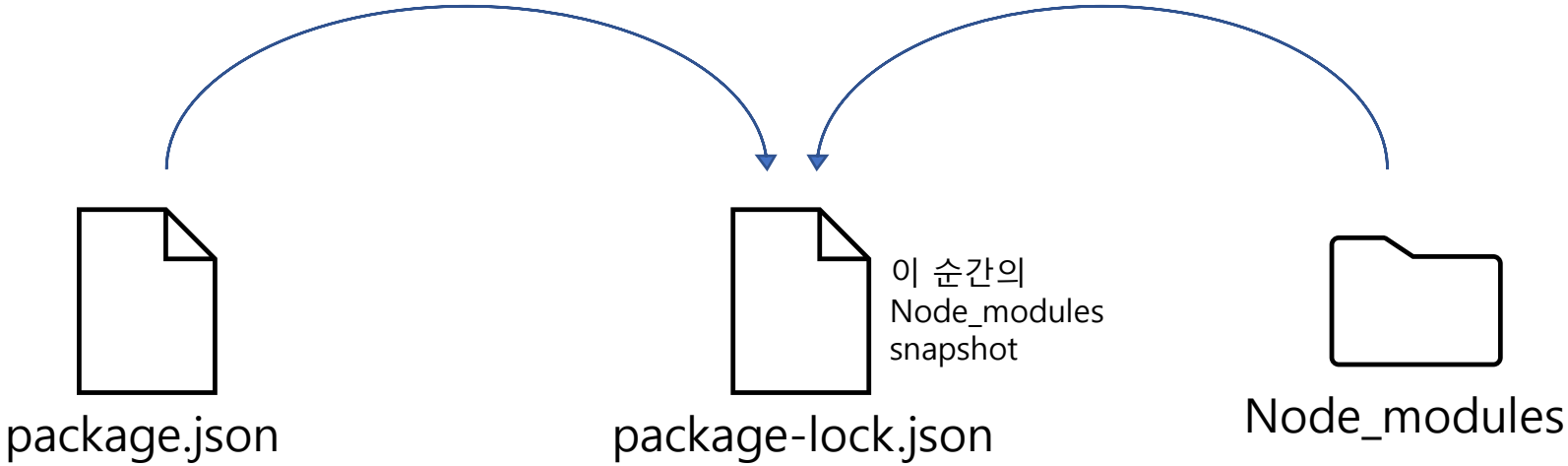
Dependency Managing in JavaScript



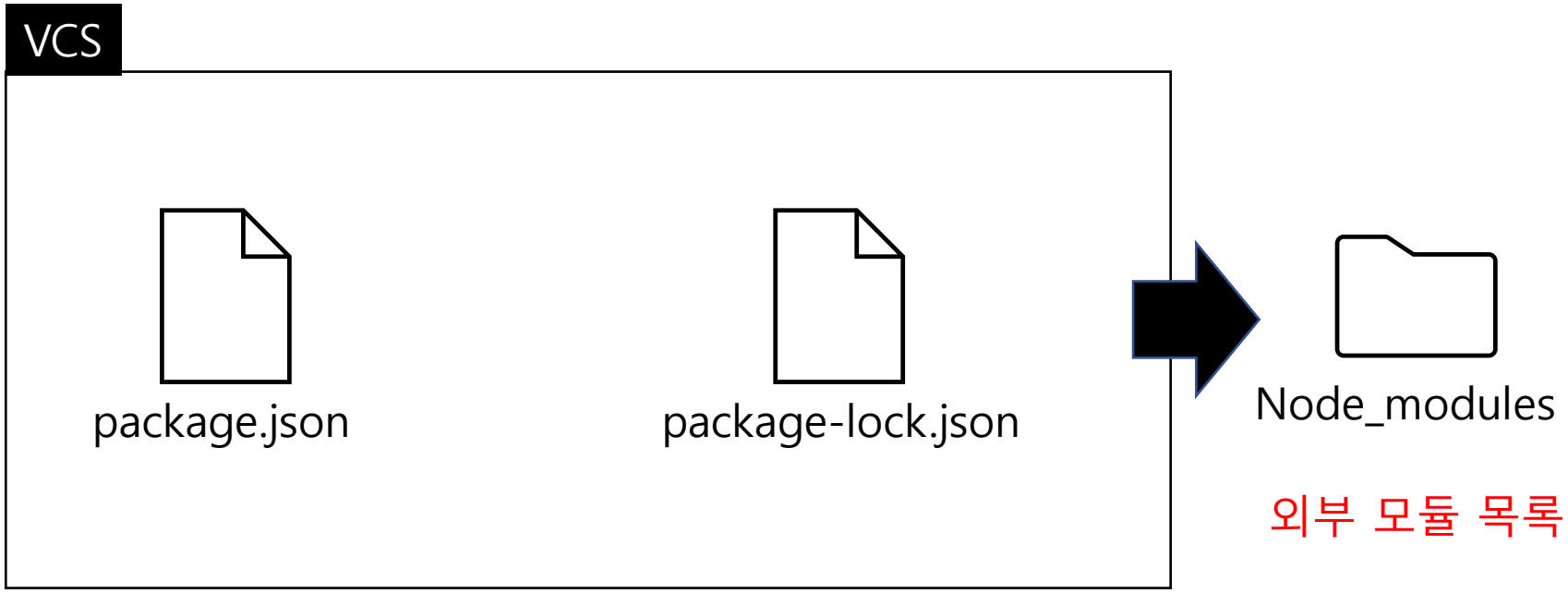
package-lock.json

- 항상 동일한 모듈 트리 (라이브러리 의존성 구조) 생성
- 파일 변경시 자동 생성
- 성공적인 설치가 보장된 트리 구조의 snapshot

Dependency Managing in JavaScript



Dependency Managing in JavaScript



문제의식

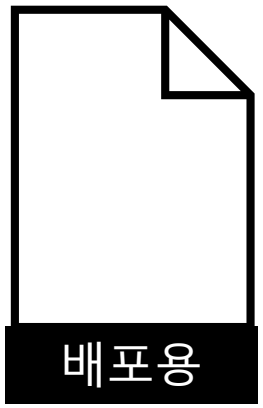
#2. Dependency Management w/ pip

- 이원화되지 않은 개발용 패키지, 배포용 패키지
- 의존관계 파악의 어려움
- 특정 패키지 버전 지정의 어려움
- 설치시 발생하는 문제

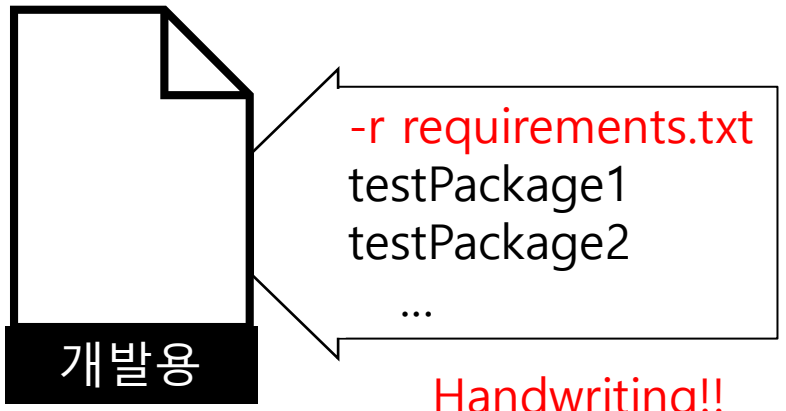
이원화되지 않은 개발용, 배포용 패키지

i) 배포용 패키지 ⊂ 개발용 패키지

requirements.txt



requirements-dev.txt



Handwriting!!

```
$ pip freeze > requirements.txt
```

이원화되지 않은 개발용, 배포용 패키지

ii) 배포용 패키지 $\not\subset$ 개발용 패키지

```
|-- requirements
|   |-- common.txt
|   |-- dev.txt
|   `-- prod.txt
`-- requirements.txt
```

```
# common.txt

req1==1.0
req2==1.0
req3==1.0
...
```

```
# dev.txt

-r common.txt
dev_req==1.0
...
```

```
$ pip install -r requirements/dev.txt
```

```
# prod.txt

-r common.txt
prod_req==1.0
...
```

```
$ pip install -r requirements/prod.txt
```

의존관계 파악의 어려움

패키지 tracking이 어려움

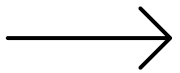
```
# pip install keras 직후
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ pip freeze
h5py==2.10.0
Keras==2.4.3
numpy==1.19.1
PyYAML==5.3.1
scipy==1.5.2
six==1.15.0
```

```
# pip install djangorestframework 직후
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ pip freeze
asgiref==3.2.10
Django==3.1
djangorestframework==3.11.1
pytz==2020.1
sqlparse==0.3.1
```

의존관계 파악의 어려움

게다가 pip uninstall은 dependencies까지 삭제하지는 않는다..

```
# pip install djangorestframework 직후
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ pip freeze
asgiref==3.2.10
Django==3.1
djangorestframework==3.11.1
pytz==2020.1
sqlparse==0.3.1
```



```
# pip uninstall djangorestframework 직후
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ pip freeze
asgiref==3.2.10
Django==3.1
pytz==2020.1
sqlparse==0.3.1
```

설치시 발생하는 문제

$ pip install –r requirements.txt 안되는데요?

- 기존에 설치한 패키지와 상충되는 경우
- 애초에 설치가 불가능한 환경일 경우

설치시 발생하는 문제

가상환경에서 $ pip install –r requirements.txt ??

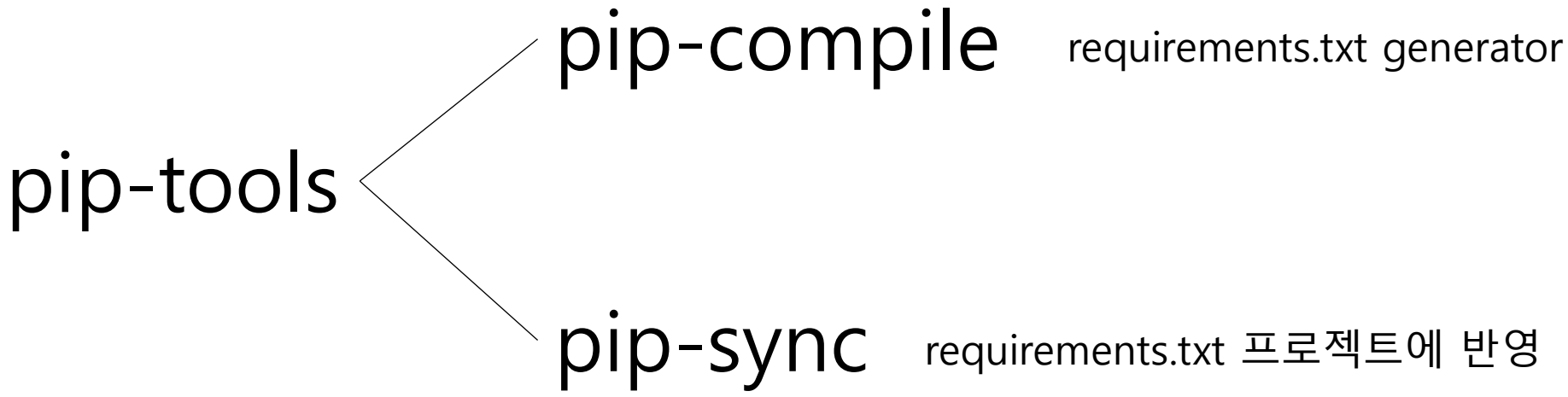
- 파이썬은 가상환경의 종류가 많다..
- 통일되지 않은 패키지관리자와 가상환경이 야기할 수 있는 문제

  e.g.) $ pip on conda env

#3 더 나은 의존성 관리를 위한 노력

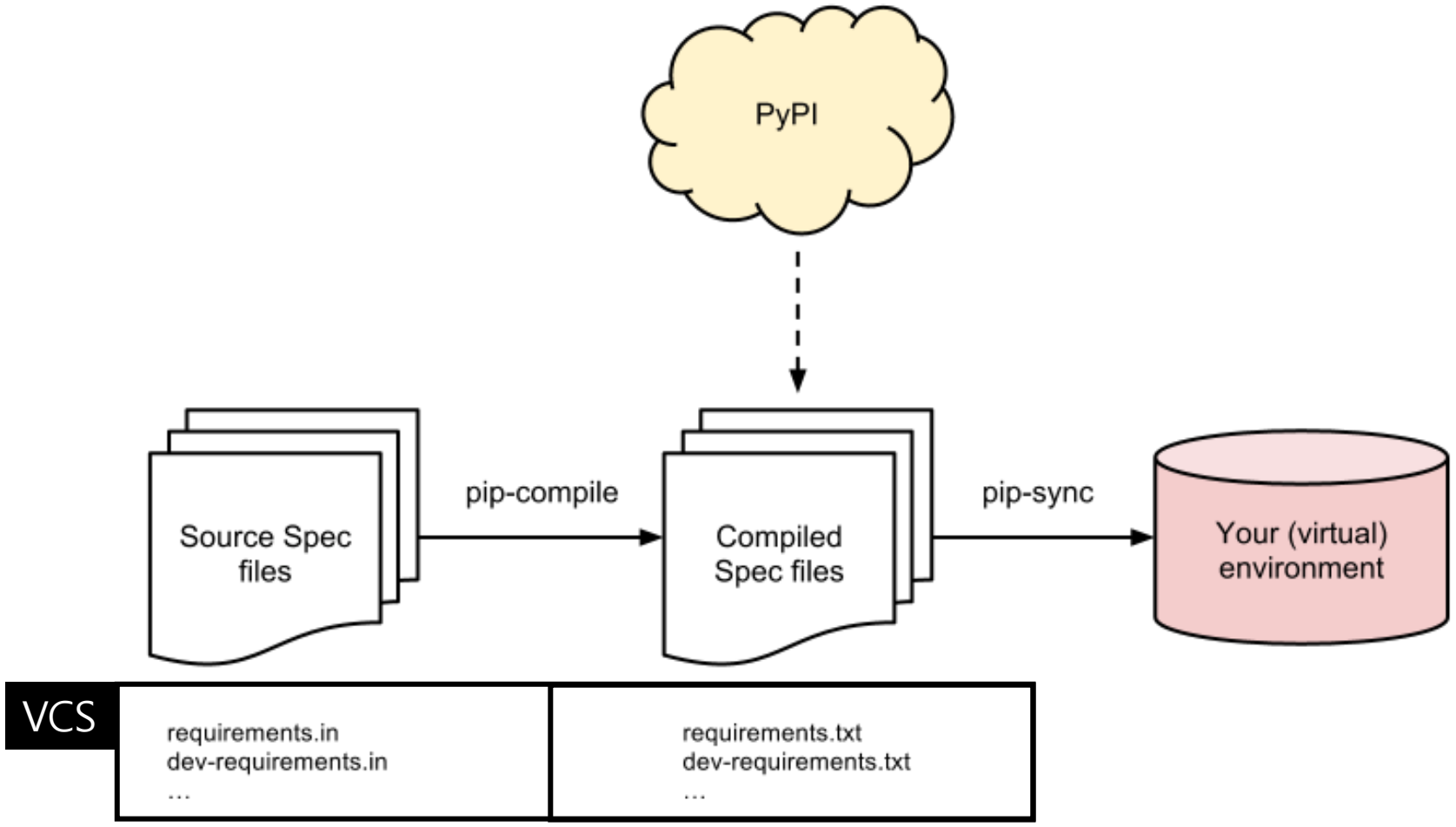
- pip-tools
- pipenv
- poetry

pip-tools



https://github.com/jazzband/pip-tools

pip-tools



https://github.com/jazzband/pip-tools

pip-tools : basic usage

```
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ ls
requirements.in
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ cat requirements.in
djangorestframework
```

$ pip-compile

```
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ ls
requirements.in  requirements.txt

mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ cat requirements.txt
#
# This file is autogenerated by pip-compile
# To update, run:
#
#    pip-compile
#
asgiref==3.2.10           # via django
django==3.1               # via djangorestframework
djangorestframework==3.11.1  # via -r requirements.in
pytz==2020.1              # via django
sqlparse==0.3.1           # via django
```

pip-tools : basic usage

- lockfile hash generation

```
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ ls
requirements.in
mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/pytest
$ cat requirements.in
djangorestframework
```

$ pip-compile --generate-hashes

```
$ cat requirements.txt
#
# This file is autogenerated by pip-compile
# To update, run:
#
#    pip-compile --generate-hashes
#
asgiref==3.2.10 \
    --hash=sha256:7e51911ee147dd685c3c8b805c0ad0cb58d360987b56953878f8c06d2d1c6f1a \
    --hash=sha256:9fc6fb5d39b8af147ba40765234fa822b39818b12cc80b35ad9b0cef3a476aed \
    # via django
django==3.1 \
    --hash=sha256:1a63f5bb6ff4d7c42f62a519edc2adbb37f9b78068a5a862beff858b68e3dc8b \
    --hash=sha256:2d390268a13c655c97e0e2ede9d117007996db692c1bb93eabebd4fb7ea7012b \
    # via djangorestframework
djangorestframework==3.11.1 \
    --hash=sha256:6dd02d5a4bd2516fb93f80360673bf540c3b6641fec8766b1da2870a5aa00b32 \
    --hash=sha256:8b1ac62c581dbc5799b03e535854b92fc4053ecfe74bad3f9c05782063d4196b \
    # via -r requirements.in
pytz==2020.1 \
    --hash=sha256:a494d53b6d39c3c6e44c3bec237336e14305e4f29bbf800b599253057fbb79ed \
    --hash=sha256:c35965d010ce31b23eeb663ed3cc8c906275d6be1a34393a1d73a41febf4a048 \
    # via django
sqlparse==0.3.1 \
    --hash=sha256:022fb9c87b524d1f7862b3037e541f68597a730a8843245c349fc93e1643dc4e \
    --hash=sha256:e162203737712307dfe78860cc56c8da8a852ab2ee33750e33aeadf38d12c548 \
    # via django
```

pip-tools : basic usage

- 패키지 버전 지정과 lockfile export

```
# (특정 버전의) 패키지 설치
$ pip-compile --upgrade
$ pip-compile --upgrade-package django
$ pip-compile --upgrade-package 'requests<3.0'

# django는 최신으로, requests는 2.0.0으로
$ pip-compile --upgrade-package django --upgrade-package requests==2.0.

# django 1점대 버전으로 설치할 수 있는 lock file requirements-django1x.txt라는 이름으로 export
$ pip-compile --upgrade-package 'django<2.0' --output-file requirements-django1x.txt
```

pip-tools : basic usage

- pip-sync: 생성된 requirements.txt 프로젝트에 반영

```
$ pip-sync
Collecting asgiref==3.2.10
  Using cached asgiref-3.2.10-py3-none-any.whl (19 kB)
Collecting django==3.1
  Using cached Django-3.1-py3-none-any.whl (7.8 MB)
Collecting pytz==2020.1
  Using cached pytz-2020.1-py2.py3-none-any.whl (510 kB)
Collecting sqlparse==0.3.1
  Using cached sqlparse-0.3.1-py2.py3-none-any.whl (40 kB)
Installing collected packages: asgiref, sqlparse, pytz, django
Successfully installed asgiref-3.2.10 django-3.1 pytz-2020.1 sqlparse-0.3.1
```

pip-tools : basic usage

- pip-sync: 생성된 requirements.txt 프로젝트에 반영

```
$ pip-sync requirements-django1x.txt
Collecting django==1.11.29
  Using cached Django-1.11.29-py2.py3-none-any.whl (6.9 MB)
Collecting pytz==2020.1
  Using cached pytz-2020.1-py2.py3-none-any.whl (510 kB)
Installing collected packages: pytz, django
Successfully installed django-1.11.29 pytz-2020.1
```

pip-tools : 기존 문제의 해결

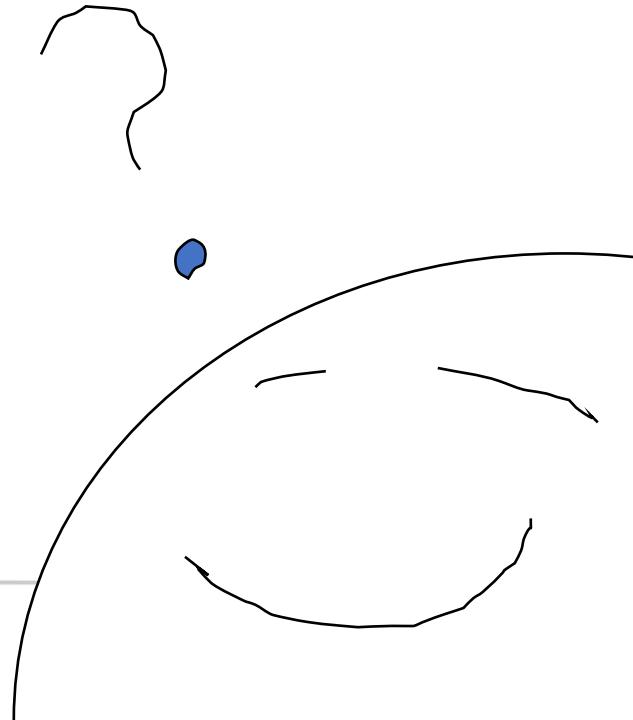
- 가장 큰 의의는 lockfile 담당을 자동으로 얻을 수 있다는 것..!
- 개발환경 재현 능력 향상 (lock file, hash generation)

- 설치/업그레이드시 특정 패키지 버전 지정 용이
- Package tracking이 ~~더듬더듬 찾아볼 정도로는~~ 가능해짐

pip-tools : 그러나...

패키지 설치 기능은 없음

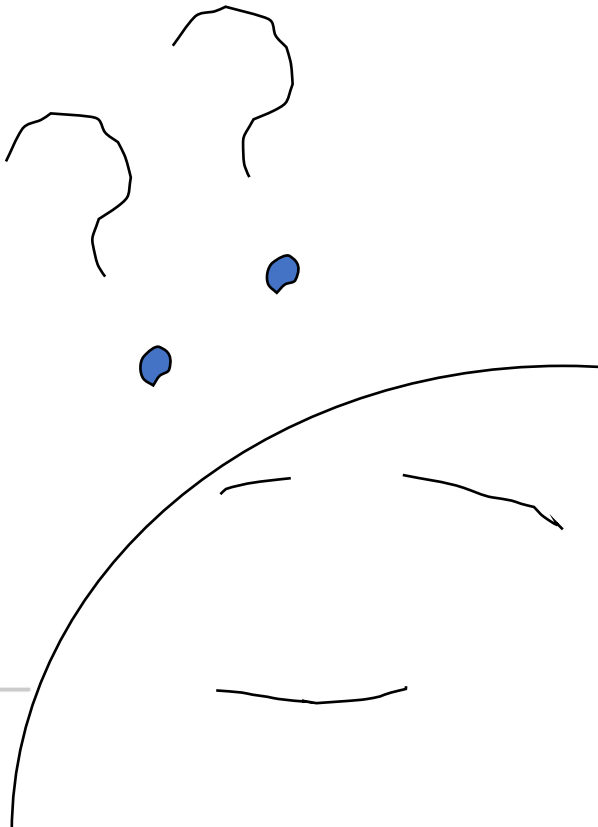
pip-tools + pip



pip-tools : 그러나...

여전히 가상환경은 따로 두어야 한다

pip + pip-tools + venv



pip-tools : 그러나...

파이썬 버전 분리가 필요하다면?

pip + pip-tools + venv + pyvenv

pip-tools : 그러나...

의존관계 한 눈에 안들어옴 (의존성 그래프 없음)

pip + pip-tools + venv + pyvenv + pipdeptree

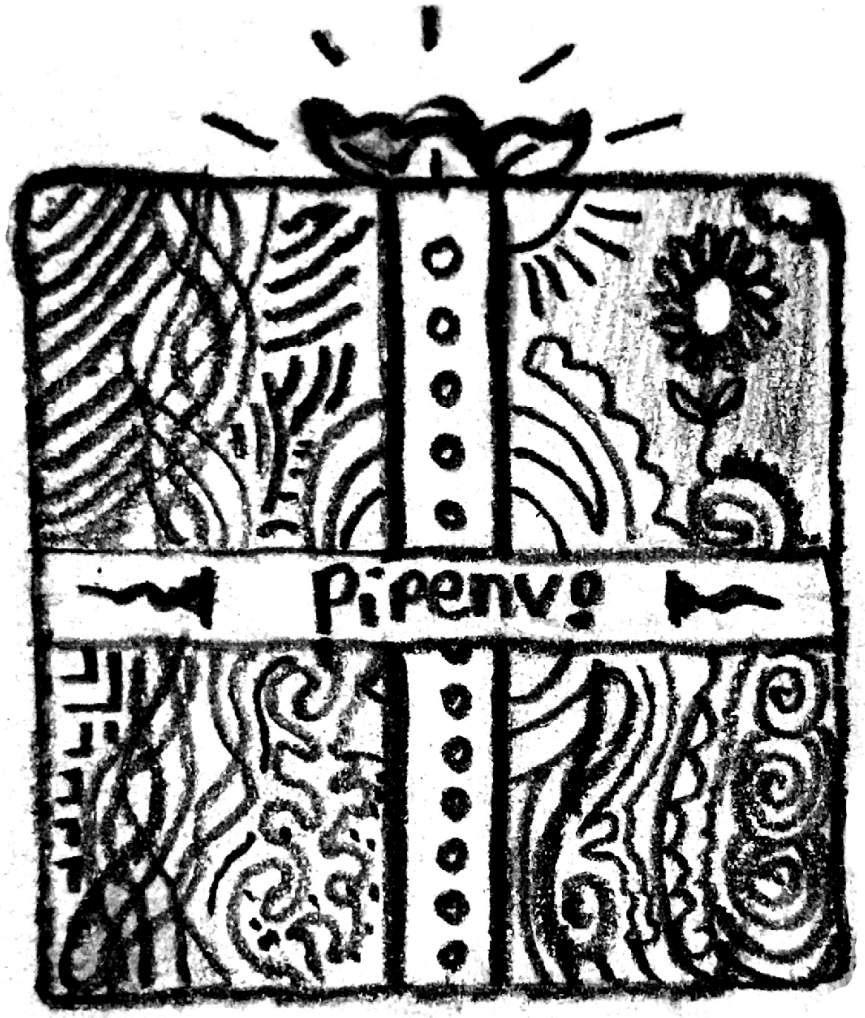
```
asgiref==3.2.10                 # via django
certifi==2020.6.20              # via requests
chardet==3.0.4                  # via requests
django-debug-toolbar==2.2       # via -r requirements.in
django==3.1                     # via -r requirements.in, django-debug-toolbar, djangorestframework
djangorestframework==3.11.1     # via -r requirements.in
h5py==2.10.0                    # via keras
idna==2.10                      # via requests
keras==2.4.3                    # via -r requirements.in
numpy==1.19.1                   # via -r requirements.in, h5py, keras, scipy
pytz==2020.1                    # via django
pyyaml==5.3.1                   # via keras
requests==2.24.0                # via -r requirements.in
scipy==1.5.2                    # via keras
six==1.15.0                     # via h5py
sqlparse==0.3.1                 # via django, django-debug-toolbar
urllib3==1.25.10                # via requests
```

Other useful tools

- pipdeptree to print the dependency tree of the installed packages.

https://github.com/jazzband/pip-tools

좀 다 해주는 도구 없나?



https://pipenv.pypa.io/en/latest/#

pipenv : basic usage

```
mckang@DESKTOP-D7RDI4B MINGW64 ~
$ mkdir pipenv && cd pipenv/

mckang@DESKTOP-D7RDI4B MINGW64 ~/pipenv
$ pipenv install
Creating a virtualenv for this project…
Pipfile: C:\Users\mckang\pipenv\Pipfile
Using C:/Users/mckang/AppData/Local/Programs/Python/Python38-32/python.exe (3.8.1) to create virtualenv…
[    ] Creating virtual environment...created virtual environment CPython3.8.1.final.0-32 in 957ms
  creator CPython3Windows(dest=C:\Users\mckang\.virtualenvs\pipenv-KccvyHNb, clear=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy,
app_data_dir=C:\Users\mckang\AppData\Local\pypa\virtualenv)
    added seed packages: pip==20.2.1, setuptools==49.2.1, wheel==0.34.2
  activators BashActivator,BatchActivator,FishActivator,PowerShellActivator,PythonActivator,XonshActivator

Successfully created virtual environment!
Virtualenv location: C:\Users\mckang\.virtualenvs\pipenv-KccvyHNb
Creating a Pipfile for this project…
Pipfile.lock not found, creating…
Locking [dev-packages] dependencies…
Locking [packages] dependencies…
Updated Pipfile.lock (db4242)!
Installing dependencies from Pipfile.lock (db4242)…
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

https://pipenv-fork.readthedocs.io/en/latest/basics.html

pipenv : basic usage

```
mckang@DESKTOP-D7RDI4B MINGW64 ~
$ mkdir pipenv && cd pipenv/

mckang@DESKTOP-D7RDI4B MINGW64 ~/pipenv
$ pipenv install
Creating a virtualenv for this project…
Pipfile: C:\Users\mckang\pipenv\Pipfile
Using C:/Users/mckang/AppData/Local/Programs/Python/Python38-32/python.exe (3.8.1) to create virtualenv…
[    ] Creating virtual environment...created virtual environment CPython3.8.1.final.0-32 in 957ms
  creator CPython3Windows(dest=C:\Users\mckang\.virtualenvs\pipenv-KccvyHNb, clear=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy,
app_data_dir=C:\Users\mckang\AppData\Local\pypa\virtualenv)
    added seed packages: pip==20.2.1, setuptools==49.2.1, wheel==0.34.2
  activators BashActivator,BatchActivator,FishActivator,PowerShellActivator,PythonActivator,XonshActivator

Successfully created virtual environment!
Virtualenv location: C:\Users\mckang\.virtualenvs\pipenv-KccvyHNb
Creating a Pipfile for this project…
Pipfile.lock not found, creating…
Locking [dev-packages] dependencies…
Locking [packages] dependencies…
Updated Pipfile.lock (db4242)!
Installing dependencies from Pipfile.lock (db4242)…
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

https://pipenv-fork.readthedocs.io/en/latest/basics.html

pipenv : basic usage

```
mckang@DESKTOP-D7RDI4B MINGW64 ~
$ mkdir pipenv && cd pipenv/

mckang@DESKTOP-D7RDI4B MINGW64 ~/pipenv
$ pipenv install
Creating a virtualenv for this project…
Pipfile: C:\Users\mckang\pipenv\Pipfile
Using C:/Users/mckang/AppData/Local/Programs/Python/Python38-32/python.exe (3.8.1) to create virtualenv…
[=   ] Creating virtual environment...created virtual environment CPython3.8.1.final.0-32 in 957ms
  creator CPython3Windows(dest=C:\Users\mckang\.virtualenvs\pipenv-KccvyHNb, clear=False, global=False)
  seeder FromAppData(download=False, pip=b                    le, wheel=bundle, via=copy,
app_data_dir=C:\Users\mckang\AppData\Local
    added seed packages: pip==20.2.1, setu                    =0.34.2
  activators BashActivator,BatchActivator,                    llActivator,PythonActivator,XonshActivator

Successfully created virtual environment!
Virtualenv location: C:\Users\mckang\.virtualenvs\pipenv-KccvyHNb
Creating a Pipfile for this project…
Pipfile.lock not found, creating…
Locking [dev-packages] dependencies…
Locking [packages] dependencies…
Updated Pipfile.lock (db4242)!
Installing dependencies from Pipfile.lock (db4242)…
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

가상환경 생성!

생성한 가상환경
실행하기

https://pipenv-fork.readthedocs.io/en/latest/basics.html

pipenv : basic usage

```
mckang@DESKTOP-D7RDI4B MINGW64 ~
$ mkdir pipenv && cd pipenv/

mckang@DESKTOP-D7RDI4B MINGW64 ~/pipenv
$ pipenv install
Creating a virtualenv for this project…
Pipfile: C:\Users\mckang\pipenv\Pipfile
Using C:/Users/mckang/AppData/Local/Programs/Python/Python38-32/python.exe (3.8.1) to create virtualenv…
[=   ] Creating virtual environment...created virtual environment CPython3.8.1.final.0-32 in 957ms
  creator CPython3Windows(dest=C:\Users\mckang\.virtualenvs\pipenv-KccvyHNb, clear=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy,
app_data_dir=C:\Users\mckang\AppData\Local\pypa\virtualenv)
    added seed packages: pip==20.2.1, se                                  l==0.34.2
  activators BashActivator,BatchActivato                                  hellActivator,PythonActivator,XonshActivator

Successfully created virtual environment
Virtualenv location: C:\Users\mckang\.virt                .virtualenvs\pipenv-KccvyHNb
Creating a Pipfile for this project…
Pipfile.lock not found, creating…
Locking [dev-packages] dependencies…
Locking [packages] dependencies…
Updated Pipfile.lock (db4242)!
Installing dependencies from Pipfile.lock (db4242)…
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

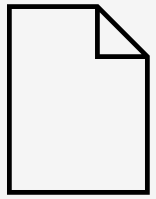
Pipfile 생성

https://pipenv-fork.readthedocs.io/en/latest/basics.html

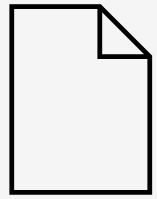
pipenv : basic usage

```
mckang@DESKTOP-D7RDI4B MINGW64 ~
$ mkdir pipenv && cd pipenv/

mckang@DESKTOP-D7RDI4B MINGW64 ~/pipenv
$ pipenv install
Creating a virtualenv for this project…
Pipfile: C:\Users\mckang\pipenv\Pipfile
Using C:/Users/mckang/AppData/Local/Programs/Python/Python38-32/python.exe (3.8.1) to create virtualenv…
[=   ] Creating virtual environment...created virtual environment CPython3.8.1.final.0-32 in 957ms
  creator CPython3Windows(dest=C:\Users\mckang\.virtualenvs\pipenv-KccvyHNb, clear=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy,
app_data_dir=C:\Users\mckang\AppData\Local\pypa\virtualenv)
    added seed packages: pip==20.2.1, setuptools==49.2.1, wheel==0.34.2
  activators BashActivator,BatchActi                    werShellActivator,PythonActivator,XonshActivator

Successfully created virtual enviror
Virtualenv location: C:\Users\mckang                    ccvyHNb
Creating a Pipfile for this project…
Pipfile.lock not found, creating…
Locking [dev-packages] dependencies…
Locking [packages] dependencies…
Updated Pipfile.lock (db4242)!
Installing dependencies from Pipfile.lock (db4242)…
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```

Pipfile.lock 생성



Pipfile
(≒ package.json)



Pipfile.lock
(≒ package-lock.json)

https://pipenv-fork.readthedocs.io/en/latest/basics.html

pipenv : basic usage

```
$ pipenv install django
Installing django…
Adding django to Pipfile's [packages]…
Installation Succeeded
Pipfile.lock (db4242) out of date, updating to (a6086c)…
Locking [dev-packages] dependencies…
Locking [packages] dependencies…
 Locking...Building requirements...
Resolving dependencies...
[  ==] Locking..Success!
Updated Pipfile.lock (a6086c)!
Installing dependencies from Pipfile.lock (a6086c)…
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```


https://pipenv-fork.readthedocs.io/en/latest/basics.html

pipenv : basic usage

```
$ pipenv install --dev django-debug-toolbar
Installing django-debug-toolbar…
Adding django-debug-toolbar to Pipfile's [dev-packages]…
Installation Succeeded
Pipfile.lock (a6086c) out of date, updating to (f3f507)…
Locking [dev-packages] dependencies…
Building requirements...
Resolving dependencies...
[=== ] Locking..Success!
Locking [packages] dependencies…
Building requirements...
Resolving dependencies...
[====] Locking..Success!
Updated Pipfile.lock (f3f507)!
Installing dependencies from Pipfile.lock (f3f507)…
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
```


https://pipenv-fork.readthedocs.io/en/latest/basics.html

pipenv : basic usage

```
$ cat Pipfile
[[source]]
name = "pypi"
url = "https://pypi.org/simple"
verify_ssl = true

[dev-packages]
django-debug-toolbar = "*"

[packages]
django = "*"

[requires]
python_version = "3.8"
```


https://pipenv-fork.readthedocs.io/en/latest/basics.html

pipenv : basic usage

```
$ pipenv graph
django-debug-toolbar==2.2
  - Django [required: >=1.11, installed: 3.1]
    - asgiref [required: ~=3.2.10, installed: 3.2.10]
    - pytz [required: Any, installed: 2020.1]
    - sqlparse [required: >=0.2.2, installed: 0.3.1]
  - sqlparse [required: >=0.2.0, installed: 0.3.1]
Keras==2.4.3
  - h5py [required: Any, installed: 2.10.0]
    - numpy [required: >=1.7, installed: 1.19.1]
    - six [required: Any, installed: 1.15.0]
  - numpy [required: >=1.9.1, installed: 1.19.1]
  - pyyaml [required: Any, installed: 5.3.1]
  - scipy [required: >=0.14, installed: 1.5.2]
    - numpy [required: >=1.14.5, installed: 1.19.1]
```


https://pipenv-fork.readthedocs.io/en/latest/basics.html

pipenv : basic usage

```
$ pipenv run python <module>    # 파이썬 실행

$ pipenv lock                   # lockfile 생성, 업데이트

$ pipenv --python 3.7           # 3.7버전 파이썬을 쓰는 가상환경 생성
```

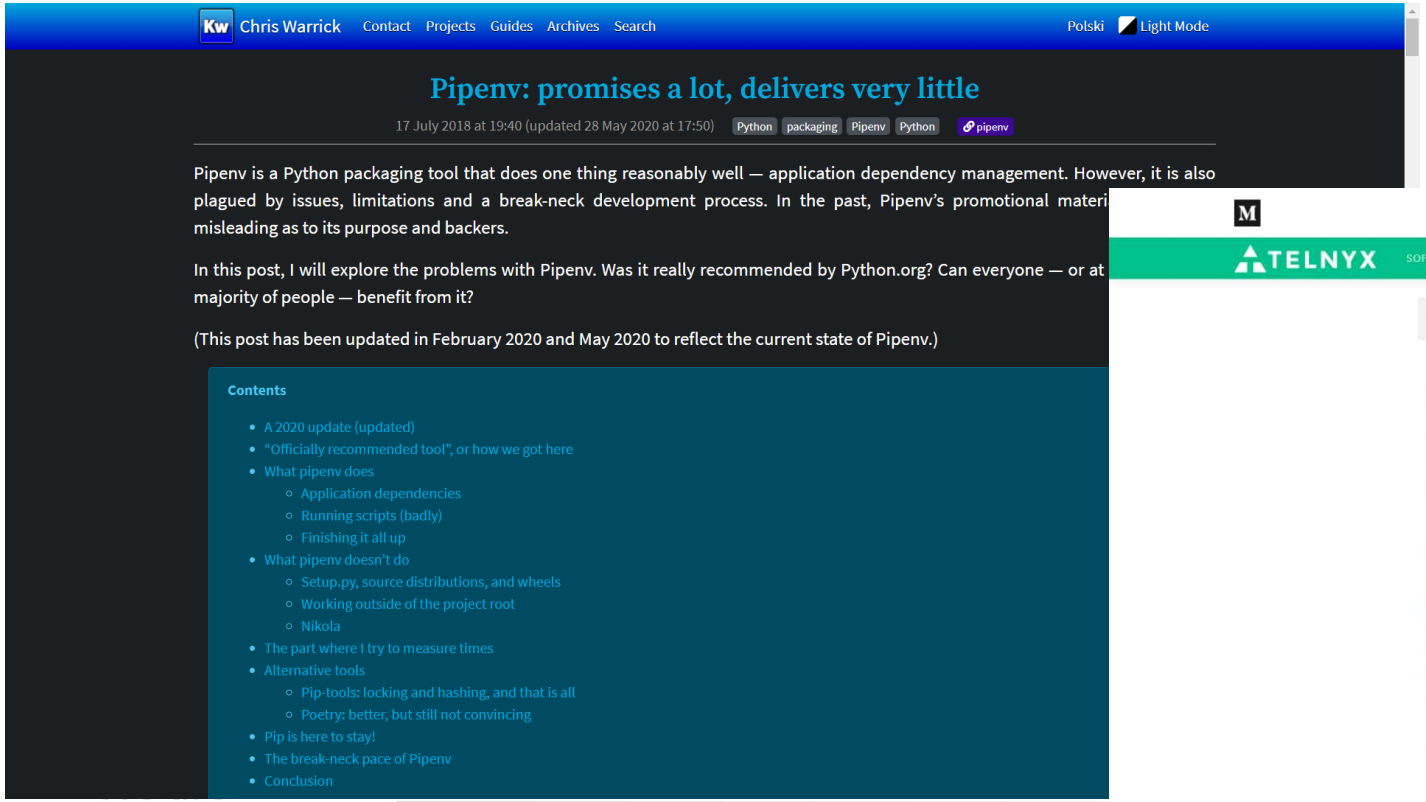

https://pipenv-fork.readthedocs.io/en/latest/basics.html

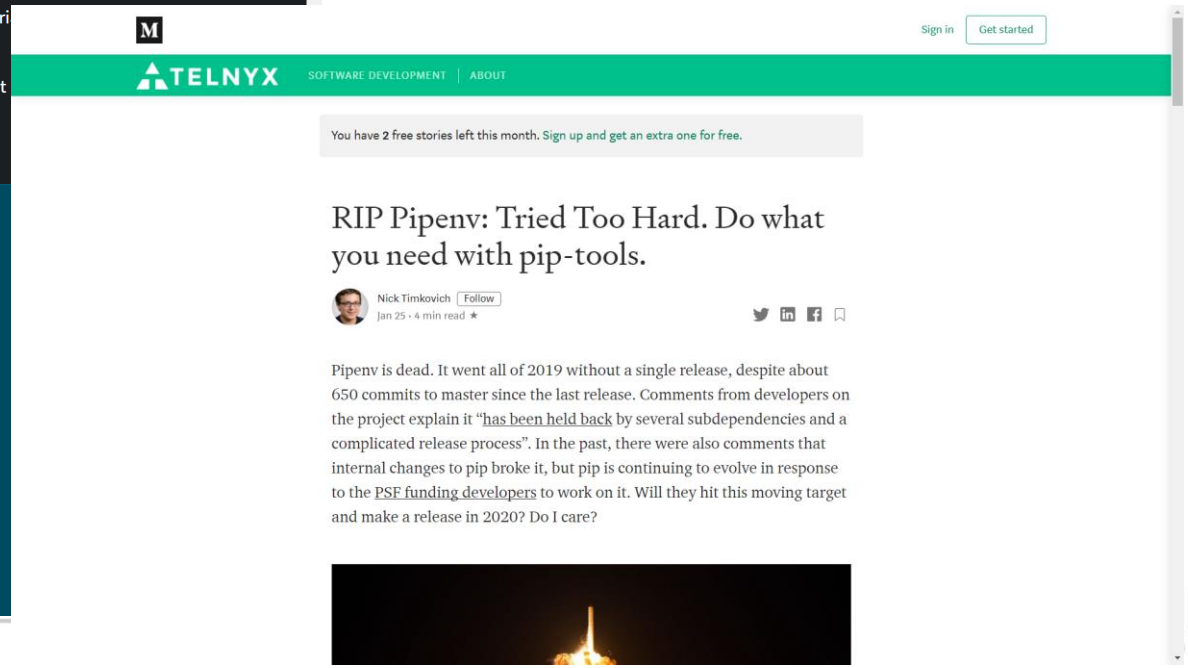
pipenv : 기존 문제의 해결

- 의존성 관리 Pipfile, Pipfile.lock (→ VCS)
- 패키지 설치, 가상환경, 의존성 관리 한번에!
- (pyenv) Python 버전 전환 용이
- 의존성 그래프 지원
- 패키지 버전별 관리의 편리함
- 개발용, 배포용 패키지 나누어 관리 가능

pipenv : 그러나..

- 너무 늦은 Release 주기: 2018년부터 올해까지 release가 없었다

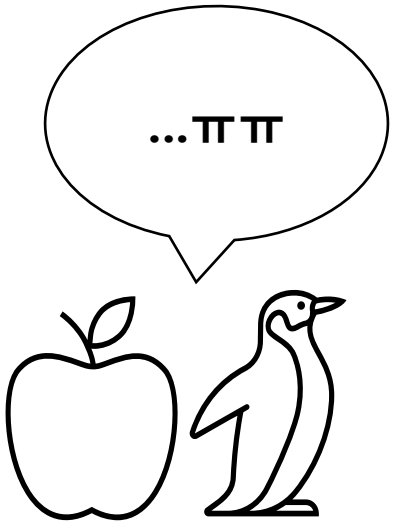




https://chriswarrick.com/blog/2018/07/17/pipenv-promises-a-lot-delivers-very-little/#
https://medium.com/telnyx-engineering/rip-pipenv-tried-too-hard-do-what-you-need-with-pip-tools-d500edc161d4

pipenv : 그러나..

- 운영체제별 Performance Issues



README.md

Pipenv: Python Development Workflow for Humans

pypi v2020.8.13 | license MIT | Azure Pipelines succeeded | python 2.7 | 3.4 | 3.5 | 3.6 | 3.7

[~ Dependency Scanning by PyUp.io ~]

Pipenv is a tool that aims to bring the best of all packaging worlds (bundler, composer, npm, cargo, yarn, etc.) to the Python world. *Windows is a first-class citizen, in our world.*

It automatically creates and manages a virtualenv for your projects, as well as adds/removes packages from your `Pipfile` as you install/uninstall packages. It also generates the ever-important `Pipfile.lock`, which is used to produce deterministic builds.

https://github.com/pypa/pipenv

경쟁자 등장!

https://python-poetry.org/

```
$ poetry init

This command will guide you through creating your pyproject.toml config.

...

Generated file

[tool.poetry]
name = "poetry-test"
version = "0.1.0"
description = "this is a demo project for pycon"
authors = ["Kang Minchul"]
license = "MIT"

[tool.poetry.dependencies]
python = "^3.8"

[tool.poetry.dev-dependencies]

[build-system]
requires = ["poetry>=0.12"]
build-backend = "poetry.masonry.api"


Do you confirm generation? (yes/no) [yes]

mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/poetry-test
$ ls
pyproject.toml
```

https://python-poetry.org/docs/cli/

```
$ poetry init

This command will guide you through creating your pyproject.toml config.

...

Generated file

[tool.poetry]
name = "poetry-test"
version = "0.1.0"
description = "this is a demo project for pycon"
authors = ["Kang Minchul"]
license = "MIT"

[tool.poetry.dependencies]
python = "^3.8"

[tool.poetry.dev-dependencies]

[build-system]
requires = ["poetry>=0.12"]
build-backend = "poetry.masonry.api"


Do you confirm generation? (yes/no) [yes]

mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/poetry-test
$ ls
pyproject.toml
```

https://python-poetry.org/docs/cli/

```
$ poetry init

This command will guide you through creating your pyproject.toml config.

...

Generated file

[tool.poetry]
name = "poetry-test"
version = "0.1.0"
description = "this is a demo project for pycon"
authors = ["Kang Minchul"]
license = "MIT"

[tool.poetry.dependencies]
python = "^3.8"

[tool.poetry.dev-dependencies]

[build-system]
requires = ["poetry>=0.12"]
build-backend = "poetry.masonry.api"


Do you confirm generation? (yes/no) [yes]

mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/poetry-test
$ ls
pyproject.toml
```

https://python-poetry.org/docs/cli/

poetry : basic usage

```
$ ls
pyproject.toml

mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/poetry-test
$ poetry install
Creating virtualenv poetry-test-oLsbLKrL-py3.8 in C:\Users\mckang\AppData\Local\pypoetry\Cache\virtualenvs
Updating dependencies
Resolving dependencies... (0.1s)

Writing lock file

No dependencies to install or update


mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/poetry-test
$ ls
poetry.lock  pyproject.toml
```


https://python-poetry.org/docs/cli/

poetry : basic usage

```
$ ls
pyproject.toml

mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/poetry-test
$ poetry install
Creating virtualenv poetry-test-oLsbLKrL-py3.8 in C:\Users\mckang\AppData\Local\pypoetry\Cache\virtualenvs
Updating dependencies
Resolving dependencies...

Writing lock file

No dependencies to install or update


mckang@DESKTOP-D7RDI4B MINGW64 ~/Desktop/poetry-test
$ ls
poetry.lock  pyproject.toml
```

가상환경 생성!

Poetry.lock 생성!

https://python-poetry.org/docs/cli/

poetry : basic usage

```
$ poetry shell
Spawning shell within C:\Users\mckang\AppData\Local\pypoetry\Cache\virtualenvs\poetry-test-oLsbLKrL-py3.8
Microsoft Windows [Version 10.0.18362.1016]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\mckang\Desktop\poetry-test>
```


https://python-poetry.org/docs/cli/

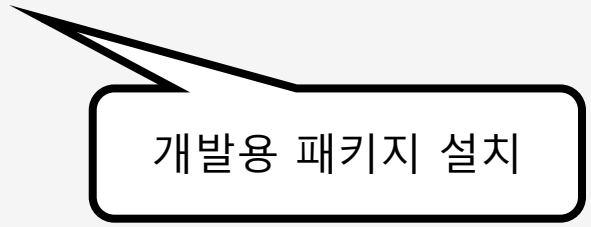
poetry : basic usage

```
C:\Users\mckang\Desktop\poetry-test>poetry add django
poetry add django
Using version ^3.1 for django

Updating dependencies
Resolving dependencies... (0.8s)

Writing lock file


Package operations: 4 installs, 0 updates, 0 removals

  - Installing asgiref (3.2.10)
  - Installing pytz (2020.1)
  - Installing sqlparse (0.3.1)
  - Installing django (3.1)

C:\Users\mckang\Desktop\poetry-test>
```


https://python-poetry.org/docs/cli/

poetry : basic usage

```
C:\Users\mckang\Desktop\poetry-test>poetry add --dev django-debug-toolbar
poetry add --dev django-debug-toolbar
Using version ^2.2 for django-debug-toolbar

Updating dependencies
Resolving dependencies... (0.1s)

Writing lock file


Package operations: 1 install, 0 updates, 0 removals

  - Installing django-debug-toolbar (2.2)

C:\Users\mckang\Desktop\poetry-test>
```



https://python-poetry.org/docs/cli/

poetry : basic usage

```
C:\Users\mckang\Desktop\poetry-test>poetry search keras
poetry search keras

keras (2.4.3)
 Deep Learning for humans

keras-perturbation (0.5.0)
 A demonstration of perturbation of data

keras-utils (1.0.13)
 Extended utilities for Keras

keras-models (0.0.7)
 Keras Models Hub

vggish-keras (0.1.1)
 VGGish in Keras.

...
```


https://python-poetry.org/docs/cli/

poetry : basic usage

```
C:\Users\mckang\Desktop\poetry-test>poetry show
poetry show
asgiref               3.2.10 A
django                3.1    A                                    clean, pragmatic design.
django-debug-toolbar  2.2    A                                    t the current request/response.
pytz                  2020.1 W
sqlparse              0.3.1  N

C:\Users\mckang\Desktop\poetr
poetry show --tree
django 3.1 A high-level Pytho                                     design.
     asgiref >=3.2.10,<3.3.0
     pytz *
     sqlparse >=0.2.2
django-debug-toolbar 2.2 A co                                     he current request/response.
     django >=1.11
         asgiref >=3.2.10,<3.
         pytz *
         sqlparse >=0.2.2
     sqlparse >=0.2.0

C:\Users\mckang\Desktop\poetr
```

https://python-poetry.org/docs/cli/

poetry : basic usage

```
$ poetry new                    # python project kickstart

$ poetry build                  # builds the source and wheels archives

$ poetry publish                #  publishes the package

        ....
```


https://python-poetry.org/docs/cli/

poetry : 기존 문제의 해결

- 의존성 관리 : pyproject.toml, poetry.lock
- 패키지 설치, 가상환경, 의존성 관리 한번에!
- Publish 기능
- 패키지 검색 기능
- 의존성 그래프 지원
- 패키지 버전별 관리의 편리함
- 개발용, 배포용 패키지 나누어 관리 가능

poetry 써야겠...음?

- Release and Version History
  - 2020.6.2 (2020-06-02)
  - 2020.5.28 (2020-05-28)
  - 2018.11.26 (2018-11-26)
  - 2018.11.14 (2018-11-14)
  - 2018.10.13 (2018-10-13)
  - 2018.10.9 (2018-10-09)
  - 2018.7.1 (2018-07-01)
  - 2018.6.25 (2018-06-25)

https://pipenv.pypa.io/en/latest/

그럼
비교해보자!

#4 pipenv vs poetry benchmark

- Functionality
- Maintenance
- Benchmark test

기능 비교

	pip	pip-tools	pipenv	poetry
패키지 설치/제거	O	X	O	O
가상환경 지원	X	X	O	O
Lock 기능	X	O (requirements.txt)	O (Pipfile.lock)	O (poetry.lock)
패키지 검색	O	X	X	O
패키지 배포	X	X	X	O

Maintainence 비교 (2020.08 기준)

- **Bugs & Dependency Resolution Issues**
  - poetry >> pipenv (> pip-tools)
- **Contributors**
  - pipenv > poetry
- **Releases**
  - poetry > pipenv

2020.08 기준

Benchmark Test

- **테스트 환경** (자세한 설명은 우측 하단 참고)
  - Windows 10 (i7, 32G, 256G)
  - Linux
    - Ubuntu 18.04 (i7, 60G, 2TB)
    - CentOS 7.7 (i7, 32G, 256G)
  - MacOS (i5, 16G, 128G)

- **테스트 항목** (django, DRF, flake(dev), django-debug-toolbar(dev), pytest(dev))
  - installation
  - dependency 추가
  - locking
  - uninstall

자세한 내용
https://github.com/kangtegong/pycon-session-release

Benchmark Test

pipenv >> poetry

In Linux (CentOS 7.7, Ubuntu 18.04)

Installation
(initialization)

자세한 내용
https://github.com/kangtegong/pycon-session-release

Benchmark Test

pipenv << poetry

In Linux (CentOS 7.7, Ubuntu 18.04)

add dependency
uninstall
locking

자세한 내용
https://github.com/kangtegong/pycon-session-release

Benchmark Test

pipenv < poetry

In Windows

자세한 내용
https://github.com/kangtegong/pycon-session-release

Benchmark Test

old poetry < new poetry

old pipenv << new pipenv

자세한 내용
https://github.com/kangtegong/pycon-session-release

#5 conclusion

dependency management in python

> 저는 배포나 package tracking 아직은 큰 관심 없고,
>
> 딱히 가상환경을 따로 쓰는게 불편하지도,
>
> python 버전별로 개발할 필요 없어요.
>
> 동일한 프로젝트 환경 생성만 보장되면 돼요

pip-tools 사용
requirements.in만 같이 commit

dependency management in python

> 저는 Package dependency tracking이 중요해요.
>
> 동일한 프로젝트 환경 생성은 당연히 중요하구요,
>
> 가상환경 필요하고 dependeny resolving issue는 만나고 싶지 않네요
>
> 무난하면서 할 거 다해주는 패키지 관리자 없나요?

pipenv 사용

dependency management in python

> 저는 Package dependency tracking이 중요해요.
>
> 동일한 프로젝트 환경 생성은 당연히 중요하구요,
>
> 가상환경 필요하고 Pypl에 패키지 배포도 생각 중이에요.
>
> 친절함 끝판왕 패키지 관리자 없나요?

poetry 사용

끝으로 발표자의 사견

pipenv 릴리즈에 대한 감사한 마음

끝으로 발표자의 사견

pipenv 릴리즈에 대한 감사한 마음
이 있지만 **poetry에 대한 지속적인 관심과**
contributon 부탁드려요

We are Pythonistas

발표 마치겠습니다.

QnA

(이메일) tegongkang@gmail.com

(발표자료 및 보조자료) https://github.com/kangtegong/pycon-session-release

PYCON KOREA 2020