AZION

From pip to poetry

Python (many) ways of packaging and publishing

EuroPython July, 2022

Vinicius Gubiani Ferreira

QA Senior Analyst (Security | Cells Squads)

A little bit about myself

Vinícius Gubiani Ferreira

- Motivation
- A bit about each Python package managers
- Present the application we'll pack and distribute
- Side-by-side comparison of each of them

1

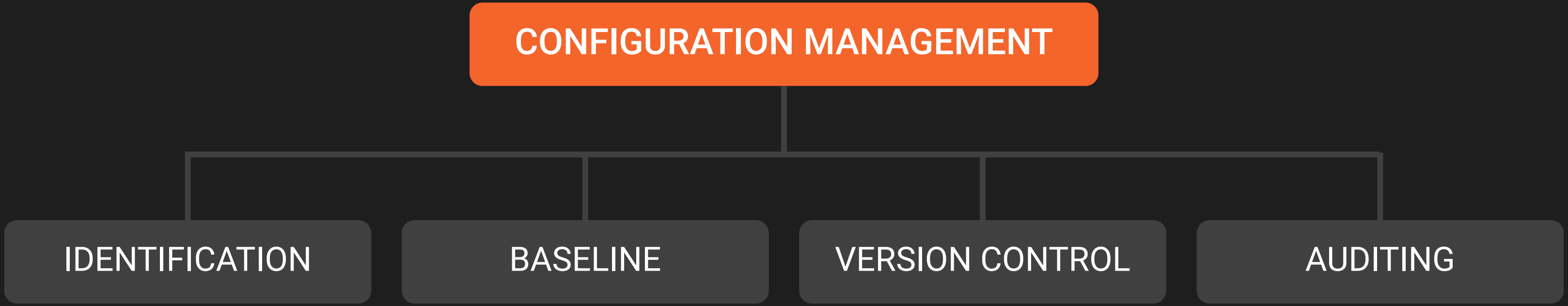
Motivation

Help you

- Reach the next level
- Learn something new

2

Package managers


CONFIGURATION MANAGEMENT
IDENTIFICATION
BASELINE
VERSION CONTROL
AUDITING

Quick disclaimer

- There is more in Python than just pip, pipenv and poetry;
- There is also easy_install (which is now deprecated);
- There is also conda;
- We can also install python packages using our OS package manager;
- But we are not gonna talk about them on this presentation;

2.1

Package managers: pip

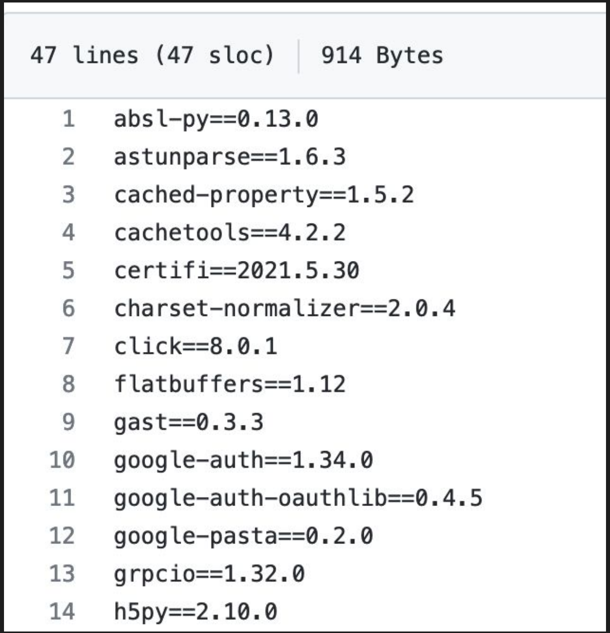
pip

- curl https://bootstrap.pypa.io/get-pip.py | python
- pip install package_name
- pip install package_name==1.4
- pip install -r requirements.txt

Based on a true story

pip

- package order DOES matter!
- Dependency sometimes is NOT solved like we think
- Works on my machine 🤷
- Only 2 solutions (and neither one are perfect)



```
47 lines (47 sloc) | 914 Bytes
absl-py==0.13.0
astunparse==1.6.3
cached-property==1.5.2
cachetools==4.2.2
certifi==2021.5.30
charset-normalizer==2.0.4
click==8.0.1
flatbuffers==1.12
gast==0.3.3
google-auth==1.34.0
google-auth-oauthlib==0.4.5
google-pasta==0.2.0
grpcio==1.32.0
h5py==2.10.0
```

2.2

Package managers: pipenv

pipenv

- Puts together the best of packaging worlds
- Creates/manages virtualenvs for us
- Introduces Pipfile and Pipfile.lock
- Is that all?

2.3

Package managers: poetry

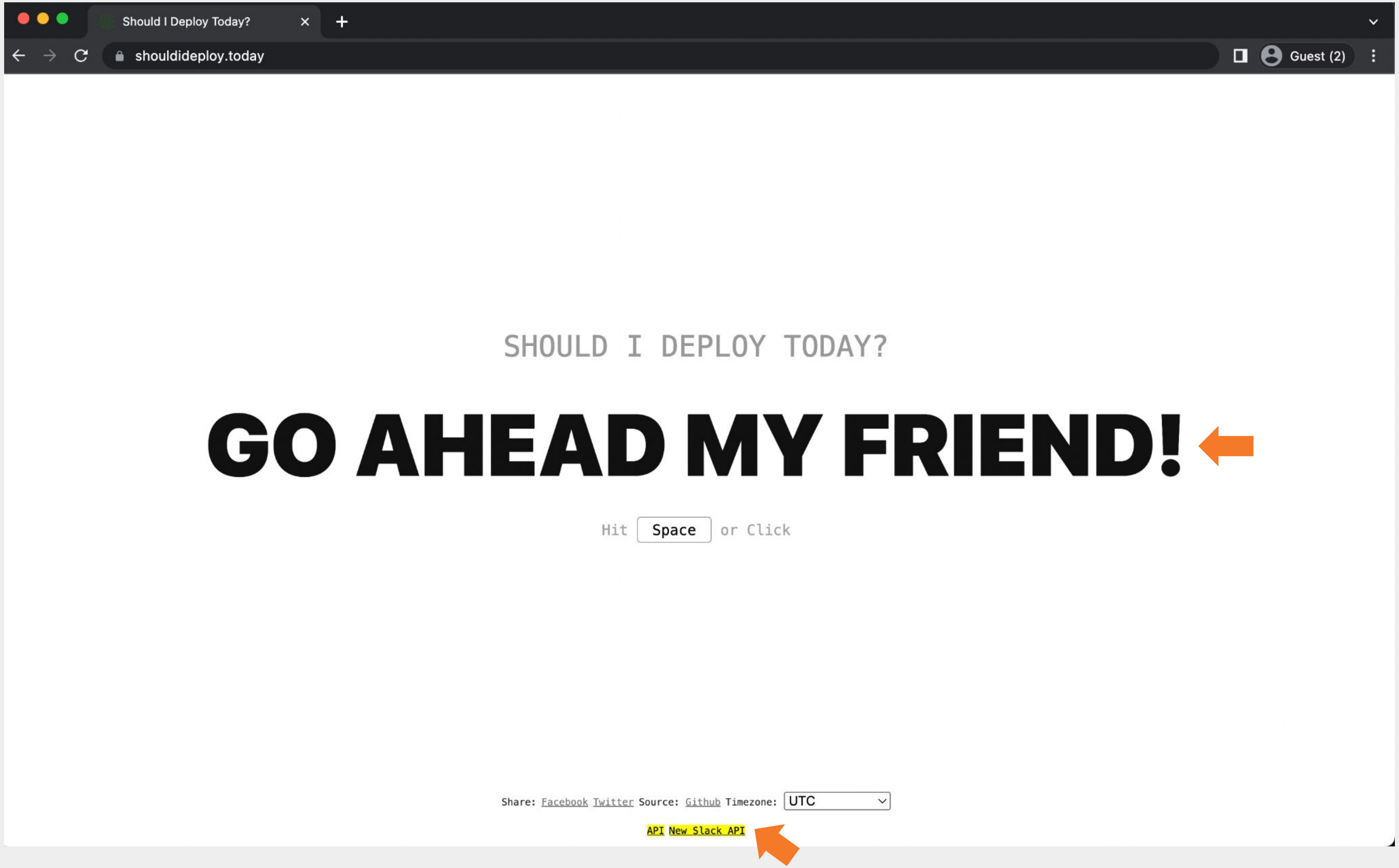
poetry

- Goes beyond pipenv
- Build a project, package and publish it. Easy/painless
- Feels like you can control everything with a single tool
- Dropping support for Python 2.7 soon

3

Our demo application

Our demo application


Should I Deploy Today?

shouldideploy.today

SHOULD I DEPLOY TODAY?

GO AHEAD MY FRIEND!

Hit Space or Click

Share: Facebook Twitter Source: Github Timezone: UTC

API New Slack API

3.1

Package & distribute with pip

Initial folder structure

```
├── LICENSE
├── README.md
├── pyproject.toml
├── setup.cfg
├── src
│   └── example_package
│       ├── __init__.py
│       └── __main__.py
└── tests
```

Setting up our code: src/example_package/__main__.py

```
from requests import get

def main():
    print(get('https://shouldideploy.today/api').json().get('message'))

if __name__ == "__main__":
    main()
```

Define our requirements.txt

```
$ pip install requests==2.28.0
$ pip freeze | grep requests > requirements.txt
```

Setting up pyproject.toml

```
[build-system]
requires = ["setuptools>=42"]
build-backend = "setuptools.build_meta"
```

Setting up setup.cfg

```
[metadata]
name = deploy_today_pip
version = 0.0.1
author = Vinicius Gubiani Ferreira
author_email = vini.g.fer@gmail.com
description = A small package to show during EuroPython 2022
long_description = file: README.md
long_description_content_type = text/markdown
url = https://github.com/vinigfer/deploy-today-pip
project_urls =
	Bug Tracker = https://github.com/…/issues
classifiers =
	Programming Language :: Python :: 3
	License :: OSI Approved :: MIT License
	Operating System :: OS Independent
…
```

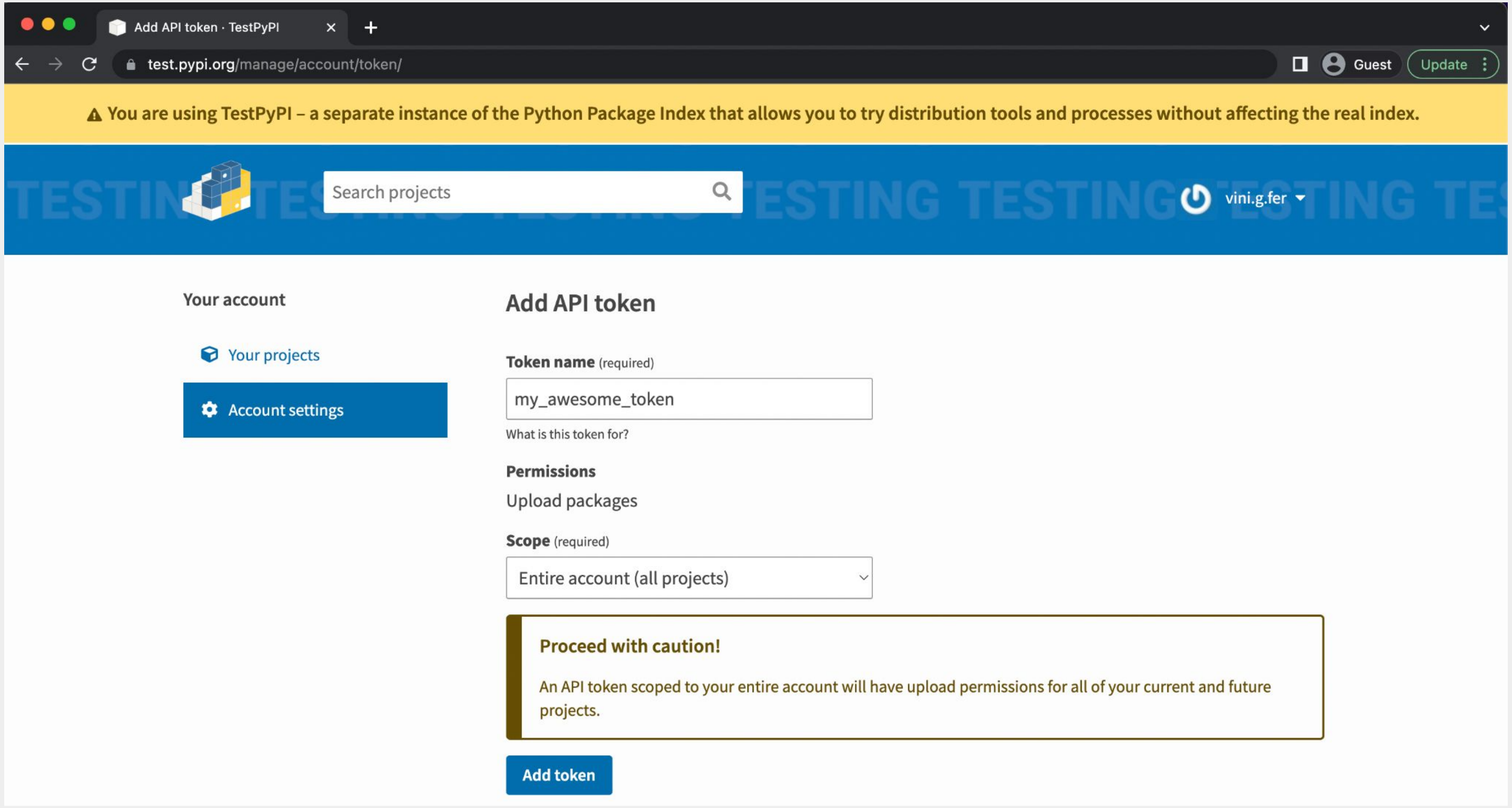
```
[options]
package_dir =
	= src
packages = find:
python_requires = >=3.6
install_requires =
	requests ==2.28.0

[options.packages.find]
where = src

[options.entry_points]
console_scripts =
	deploy_today_pip = example_package.__main__:main
```

Generating distribution files

- python3 -m pip install --upgrade build
- python3 -m build

```
dist/
  deploy_today_pip-0.0.1-py3-none-any.whl
  deploy_today_pip-0.0.1.tar.gz
```

Our demo application - pip


Add API token · TestPyPI

test.pypi.org/manage/account/token/

You are using TestPyPI – a separate instance of the Python Package Index that allows you to try distribution tools and processes without affecting the real index.

Your account

- Your projects
- Account settings

Add API token

Token name (required)

my_awesome_token

What is this token for?

Permissions

Upload packages

Scope (required)

Entire account (all projects)

Proceed with caution!

An API token scoped to your entire account will have upload permissions for all of your current and future projects.

Add token

Setting up token on ~/.pypirc

```
[testpypi]
  username = __token__
  password = pypi-AgENdGi…mNzctNzk3N
```

Uploading to pypi

- python3 -m pip install --upgrade twine
- python3 -m twine upload --repository testpypi dist/*

```
Uploading distributions to https://test.pypi.org/legacy/
Enter your username: [your username]
Enter your password:
Uploading deploy_today_pip-0.0.1-py3-none-any.whl
100% ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5.7/5.7 kB • 00:00 • ?
Uploading deploy_today_pip-0.0.1.tar.gz
100% ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.9/4.9 kB • 00:00 • ?
```

Install from (test) pypi and use

- pip install -i https://test.pypi.org/simple/ deploy-today-pip
- deploy_today_pip

```
Go home, you're drunk
```

3.2

Package & distribute with pipenv

Let's start with the same initial folder structure

```
├── LICENSE
├── README.md
├── pyproject.toml
├── setup.cfg
├── src
│   └── example_package
│       ├── __init__.py
│       └── __main__.py
└── tests
```

Install pipenv

- pip install --user pipenv
- pipenv --three
- pipenv install requests

If we check Pipfile

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

[dev-packages]

[packages]
requests = "*"

[requires]
python_version = "3.6"
```

Pipfile.lock on the other hand

Dependency tree

```
> pipenv graph
requests==2.18.4
  - certifi [required: >=2017.4.17, installed: 2017.11.5]
  - chardet [required: >=3.0.2,<3.1.0, installed: 3.0.4]
  - idna [required: <2.7,>=2.5, installed: 2.6]
  - urllib3 [required: >=1.21.1,<1.23, installed: 1.22]

> pipenv graph --reverse
(same as before, but from bottom-up approach)
```

Other features

```
> pipenv check (security vulnerabilities and PEP508)
> pipenv install pytest --dev
> pipenv lock
> pipenv install --ignore-pipfile
> pipenv install --dev
```

What about packaging and publishing?

Turns out it's as complicated as pip (even more)

3.3

Package & distribute poetry

Install poetry

- curl -sSL https://install.python-poetry.org | python3 -
- echo 'export PATH="/home/vinicius/.local/bin:$PATH"' >> ~/.bashrc && source ~/.bashrc
- poetry --help

Create project structure

- poetry new deploy_today_poetry
- cd deploy_today_poetry
- tree

```
├── README.rst
├── deploy_today_poetry
│   └── __init__.py
├── pyproject.toml
└── tests
    ├── __init__.py
    └── test_deploy_today_poetry.py
```

Add a new package

```
$ poetry add requests
Creating virtualenv deploy-today-poetry-2Hyl_28d-py3.9 in
/Users/vf/Library/Caches/pypoetry/virtualenvs
Using version ^2.28.0 for requests
Updating dependencies
Resolving dependencies... (2.2s)
Writing lock file
Package operations: 13 installs, 0 updates, 0 removals
  • Installing pyparsing (3.0.9)
…
  • Installing requests (2.28.0)
```

Setting up our code: deploy_today_poetry/__main__.py

```
from requests import get


def main():
    print(get('https://shoulddeploy.today/api').json().get('message'))


if __name__ == "__main__":
    main()
```

Setting up pyproject.toml

```
...
[tool.poetry.scripts]
deploy_today_poetry = "deploy_today_poetry.__main__:main"
```

Generating distribution files

- poetry build

```
Building deploy_today_poetry (0.1.0)
  - Building sdist
  - Built deploy_today_poetry-0.1.0.tar.gz
  - Building wheel
  - Built deploy_today_poetry-0.1.0-py3-none-any.whl
```

Setting up token & (test) repository

- poetry config pypi-token.test-pypi pypi-AgENdGi…mNzctNzk3N
- poetry config repositories.test-pypi https://test.pypi.org/legacy/

Uploading to pypi

- poetry publish --repository test-pypi

```
Publishing deploy_today_poetry (0.2.0) to test-pypi
 - Uploading deploy_today_poetry-0.2.0-py3-none-any.whl 100%
 - Uploading deploy_today_poetry-0.2.0.tar.gz 100%
```

Install from (test) pypi and use

- pip install -i https://test.pypi.org/simple/ deploy-today-poetry
- deploy_today_poetry

This Is the Way

	pip	pipenv	poetry
Beginner friendly	Yes	No	Yes
Heavily rely on other tools	Yes	Yes	No
Built-in virtualenv management	No	Yes	Yes
Python 2 support	Yes	Yes	Dropping soon
Manage package and distribution by itself	No	No	Yes

Sources:

https://packaging.python.org/en/latest/tutorials/packaging-projects/

https://packaging.python.org/en/latest/discussions/pip-vs-easy-install/

https://packaging.python.org/en/latest/key_projects/

https://www.atlassian.com/microservices/microservices-architecture/configuration-management

https://www.datacamp.com/tutorial/pip-python-package-manager

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

https://rootnroll.com/d/pipenv/

https://python-poetry.org/

https://shouldideploy.today/

Thank you

Obrigado

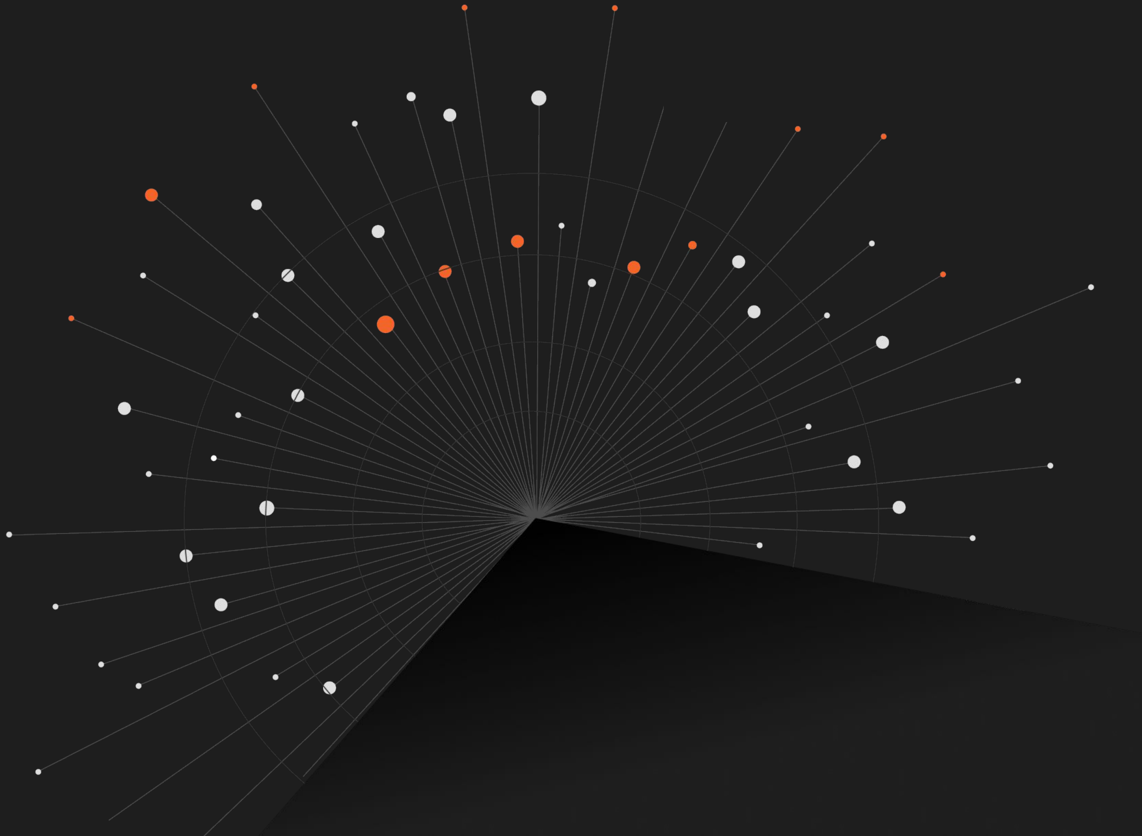
Gracias

Vielen Dank

Спасибо

谢谢啦

ありがと



AZION

Have a question?

Please contact me!

vinigfer

vinigfer

vinicius-gubiani-ferreira

vini.g.fer@gmail.com