

Poetry 101

Basics and stumbling points

YOSHIHIRO FUKUHARA



cvpaper.challenge

May. 30th, 2022

Content

- What is Poetry?
- How Poetry work?
- Frequent commands
- Dependency specification
- Frequently faced error

What is Poetry?

Pythonパッケージの依存関係管理, パッケージ作成を行うためのツール.



Dependency Management for Python

v1.1.13

6.9M/month



Stars

< 20k

FEATURE

- パッケージの依存関係管理
- 仮想環境の構築
- 作成したパッケージのPyPIへのパブリッシュ

How Poetry work?

StumblingPoints

- 2つの重要なファイル：
 - `pyproject.toml`：（制約）ユーザーが`poetry add`コマンド等によって変更を行うファイル. インストールしたいパッケージのバージョンに対する制約が記載される.
 - `poetry.lock`：（状態）Poetryが自動変更するファイル. `pyproject.toml`に記載されているバージョン制約を元に, 実際にインストールしたパッケージの情報と, 依存パッケージの情報が記載される.
- Poetryはパッケージをインストール・アップデートするとき, 下記の1.と2.を両方満たすバージョンが存在するかを確認し, 見つければ実行する. (本スライドでは, 1.と2.を合わせて**更新可能要件**と表現する.)
 1. **全てのパッケージが`pyproject.toml`に記載されたバージョン制約を満たすこと.**
 2. **既にインストールされているパッケージと依存パッケージの競合がないこと.**
- `poetry.lock` ファイルを含めてgithub等で共有することで, チーム間で同じパッケージの環境を共有することが出来る.

Frequent commands

- 頻繁に使用するコマンド:
 - install
 - add
 - update
 - remove
 - run
 - show
- 詳しい情報はofficial docsに記載がある.



PYTHON PACKAGING AND DEPENDENCY MANAGEMENT MADE
EASY

Poetry



Frequent commands

install

```
$ poetry install
```

```
# dev dependenciesを除いてインストールを行う。
```

```
$ poetry install --no-dev
```

- `pyproject.toml`に記載されているバージョン制約を参照して、更新可能要件を満たす各パッケージのバージョンを探索し、見つければインストールする。同時に `poetry.lock` を作成し、インストールしたパッケージの情報を書き込む。
- `poetry.lock` が既に存在する場合は `poetry.lock` に記載されているものと全く同じバージョンのパッケージをインストールする。
- configの `virtualenvs.create` が `True` になっていると仮想環境を作成し、パッケージを仮想環境内にインストールする。

Frequent commands

add

StumblingPoints

```
# numpyの最新のバージョンが更新可能要件を満たすかを確認し、  
# もし満たしていればインストールする。  
$ poetry add numpy  
  
# numpyのx.y未満のバージョンで更新可能要件を満たすバージョンを  
# 順次探して、もし見つければインストールする。  
$ poetry add "numpy<x.y"  
  
# (Caret記法) x.y.z以上, x+1.0.0未満 (厳密には正確では無い) の  
# 範囲でバージョンを順次探して、もし見つければインストールする。  
$ poetry add "numpy^x.y.z"  
  
# cvpaperchallenge/melonのdevelopブランチのインストールを  
# 試みる。  
$ poetry add git@github.com:cvpaperchallenge/melon.git#devel  
  
# ローカルディレクトリ./my-package/下のインストールを試みる。  
$ poetry add ./my-package/
```

- 更新可能要件を満たすバージョンが見つければ、パッケージの追加を行う。(見つからなかった場合は`SolverProblemError`になる.)
- パッケージであればgithubの特定のブランチやローカルディレクトリ/ファイルもインストール出来る。
- 独特なバージョン指定の記法があるので、少し覚える必要がある(後述).
- 便利な一方で、エラーで躓き安いコマンドでもある (エラーが出た場合の対処方は後述).

Frequent commands

update

```
# 更新可能要件を満たしているパッケージを全てアップデートする。
```

```
$ poetry update
```

```
# 特定のパッケージのみに対して使用することも可能。
```

```
$ poetry update numpy
```

- 更新可能要件を満たしていれば, パッケージのアップデートを行う.
- アップデート可能なパッケージの一覧は後述の `poetry show --latest` などで一覧出来る.
- `pyproject.toml` に記載されているバージョン制約以上のアップデートを行いたい場合は `poetry add` を使用して再度パッケージの追加を行う.

Frequent commands

remove

```
# numpyをアンインストールする。  
$ poetry remove numpy
```

- `poetry remove`が実行されると、`pyproject.toml`に記載されている対象のパッケージのバージョン制約の情報が削除される。
- 対象のパッケージに依存している他のパッケージが1つも無ければパッケージはアンインストールされる。

Frequent commands

run

```
# poetryの作成した仮想環境内でPython3を実行.
```

```
$ poetry run python3
```

```
# poetryの作成した仮想環境内でblackをsrcディレクトリに適用.
```

```
$ poetry run black src
```

- Poetryの仮想環境内でコマンドを実行する. Poetryでインストールしたパッケージを使用するためにはpoetryの仮想環境内でコマンドを実行する必要がある.
- 「Poetryでパッケージをインストールしたのに, それが見つからないと怒られる.」という場合は大体この`poetry run`をつけ忘れていることが多い.
- その都度`poetry run`をつけるのが面倒という方は`poetry shell`コマンドを使って仮想環境の中でshellを立ち上げることも出来る.

Frequent commands

show

```
# 現在インストールされているパッケージの一覧を表示。
```

```
$ poetry show
```

```
# パッケージの依存関係をツリーとして表示。
```

```
$ poetry show --tree
```

```
# パッケージの最新のバージョンを表示。
```

```
$ poetry show --latest
```

- Poetryでインストールしたパッケージの様々な情報を表示する.
- 特に`poetry show --latest`は`poetry update`と組み合わせて使用すると便利.

Off-topic

Python module / package / library

- モジュール (module) とは `.py` ファイルのこと.
- パッケージ (package) とはモジュールを構造化する手段で, `__init__.py` ファイルと `.py` ファイルを含んだディレクトリのこと. パッケージはその中に下位のパッケージを含むこともある.
- ライブラリ (library) の定義についてはPythonの公式ドキュメントには記載が無いと思われるが, PyPI等に公開されているパッケージ, もしくはパッケージの集合を意味することが多い.

Off-topic

Semantic versioning

- ソフトウェアのバージョンを`x.y.z`の形で指定する.
- `x`はメジャーバージョンと呼ばれ, APIの変更に互換性のない場合にインクリメントする.
- `y`はマイナーバージョンと呼ばれ, 後方互換性があり機能性を追加した場合にインクリメントする.
- `z`はパッチバージョンと呼ばれ, 後方互換性を伴うバグ修正をした場合にインクリメントする.

Dependency specification

Caret requirements

- `^`を用いてバージョン制約を記述する.
- ゼロでない再左の数字を変更しない範囲を表す.

例	表す範囲
<code>^1.2.3`</code>	<code>`>=1.2.3,<2.0.0`</code>
<code>^1.2`</code>	<code>`>=1.2.0,<2.0.0`</code>
<code>^1`</code>	<code>`>=1.0.0,<2.0.0`</code>
<code>^0.2.3`</code>	<code>`>=0.2.3,<0.3.0`</code>
<code>^0.0.3`</code>	<code>`>=0.0.3,<0.0.4`</code>

Dependency specification

Tilde requirements

- `~``を用いてバージョン制約を記述する.
- 形式によって意味が異なる.
 - `~x.y.z``または`~x.y``の形式のときパッチバージョンの変更の範囲を表す.
 - `~x``の形式のときマイナーバージョンの変更の範囲を表す.

例

表す範囲

`~1.2.3``

`>=1.2.3,<1.3.0``

`~1.2``

`>=1.2.0,<1.3.0``

`~1``

`>=1.0.0,<2.0.0``

Frequently faced error

SolverProblemError

StumblingPoints

```
# AWSに関連する2つのパッケージのインストールを試みる
$ poetry add "boto3==1.16.43"
$ poetry add "s3fs^2022.5.0"
```

```
Updating dependencies
Resolving dependencies... (0.4s)
```

SolverProblemError

(途中略)

```
Thus, s3fs (>=2022.5.0,<2023.0.0) requires botocore (>=1.2
And because boto3 (1.16.43) depends on botocore (>=1.19.43
So, because ascender depends on both boto3 (1.16.43) and s
```

- `s3fs`は`botocore(>=1.24.21,<1.24.22)`に,
`boto3`は`botocore (>=1.19.43,<1.20.0)`にそれぞれ依存しており, 依存パッケージが競合している. これは, 更新可能要件の2を満たしていないため
`SolverProblemError`が発生する.
- `s3fs`と`boto3`を両方インストールするにはバージョン制約を調整して, `botocore`の競合が発生しないようにする必要がある.

Frequently faced error

SolverProblemError

StumblingPoints

```
# AWSに関連する2つのパッケージのインストールを試みる
$ poetry add "boto3==1.16.43"
$ poetry add "s3fs<=2022.5.0" # s3fsの制約を緩める

Updating dependencies
Resolving dependencies... (8.4s)

Writing lock file

Package operations: 2 installs, 0 updates, 0 removals

  • Installing fsspec (2022.5.0)
  • Installing s3fs (0.4.2)
```


- 例えば、`s3fs`のバージョン制約を緩めることで、`boto3`の依存している`botocore`のバージョンと競合しないバージョンの`s3fs`のバージョンを探索するようになる。
- 上のように`s3fs`バージョン制約を緩めても競合しないバージョンがもし見つからない場合は同様に`SolverProblemError`が発生する。このような場合は`boto3`側の制約も緩めることを考える必要がある。


Yoshihiro Fukuhara


ML / MLOps engineer.

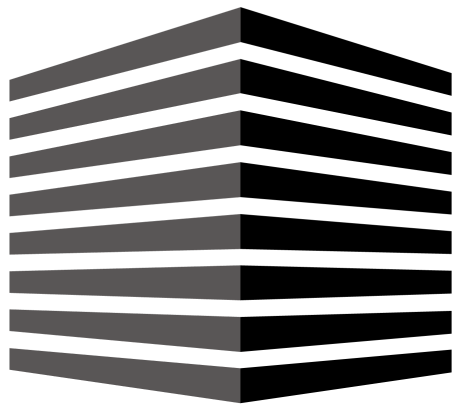
HQ member and XCCV group lead of cvpaper.challenge.



 [gatheluck](#)

 [gatheluck](#)

 [gatheluck.net](#)



cvpaper.challenge

Visit our page

END