パッケージを配ろう

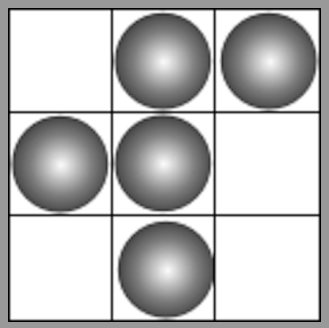
Atsushi Odagiri

2024-04-25

Outline

お前誰よ

- Atsushi Odagiri
- Open Collector
- Python は 1.5 くらいのころから



Open Collector

ケンオール

パッケージエコシステム

- 作る
- 配る
- 使う

パッケージを配るということ

- 広く一般に向けて配る
- 狭い範囲で限られた利用のために配る

広く一般に向けて pypi で配る

- PyPA ツールのデフォルト
- `tween` でアップロード
- `pip` がダウンロードしてインストール

狭い範囲で限られた利用のために配る

- マイクロサービスのそれぞれて使うようなライブラリ
- 特殊なパッチをあてたローカルフレーバーライブラリ

狭い範囲で配る

- 社内ネットワークや VPN の中で
- k8s や vpc の中で
- 範囲内の IP アドレスにだけ
- 認証をつけたい

httplib.server でのお手軽 repository

- ダウンロードできるリンクがあればいいので `http` モジュールでサーバーを起動するだけ
- wheel ファイルのあるディレクトリで実行

```
python3 -m pip download pyramid
python3 -m http.server
```

URL 指定でインストール

- pip は URL 指定で直接インストールできる
- 正確なファイル名を知らないといけない
- wheel はプラットフォームなどの情報を含んでいる

```
pip install \
    http://localhost:8000/pyramid-2.0.2-py3-none-any.whl
```

複雑な wheel ファイル名

- oh…

```
zope.interface-6.4
-cp311
-cp311
-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.
.whl
```

find-links

- `find-links` で指定した場所から探しだしてもらう

```
pip install -f http://localhost:8000 zope.interface
```

no-index

- 場合によっては pypi への接続も制限される環境
- 全てをお手軽 repository から取得するなら `no-index` も使うようにしてみよう
- `no-index` pypi などの index を見にいかない
- `find-url` 指定したページからダウンロード URL をスクレーピング

index は必要？

- pip を直接使うなら `find-url` でもいいかも？
- メタデータを取得するのに配布物をダウンロードするという効率の悪さはある
- `poetry source add` で使えるのは simple repository
  - pip だと `--index-url` で指定するものに相当

独自の pypi を立てたい!

- PyPI 自体のソースコードは公開されている
  - `https://github.com/pypi/warehouse`
  - インフラ構築保守など手間もかかる
- devpi
  - `https://github.com/devpi/devpi`
  - PyPI へのプロキシやプロジェクトごとの名前空間設定など多機能
  - それなりにインフラ構築保守の手間がかかる
- `http.server` くらいに簡単に立ち上って欲しいところ

パッケージを配るための PEP

- PEP 458 – Secure PyPI downloads with signed repository metadata
- PEP 480 – Surviving a Compromise of PyPI: End-to-end signing of packages
- PEP 503 – Simple Repository API
- PEP 592 – Adding “Yank” Support to the Simple API
- PEP 629 – Versioning PyPI’s Simple API
- PEP 658 – Serve Distribution Metadata in the Simple Repository API
- PEP 691 – JSON-based Simple API for Python Package Indexes
- PEP 700 – Additional Fields for the Simple API for Package Indexes
- PEP 714 – Rename dist-info-metadata in the Simple API

Simple Repository

representation

- HTML PEP503
- JSON PEP691

バージョン

- 1.0 PEP503/PEP691
- 1.1 PEP700
- PEP714 メタデータフィールドの取り扱いについての修正
  - warehouse の実装で間違えがあったらしい

PyPI の Simple Repository

- `https://pypi.org/simple/` とても大きいのでアクセス注意！

実装方針

- 標準ライブラリでいこう
  - Batteries Included!
- 1 ファイルデプロイ
- DB などを使わず起動するだけで使える

標準ライブラリで web アプリケーションを書く

- json
- wsgiref

simple repository の機能

- 2 つだけ!
  - / project list
  - /{project} project detail

project list

- ホストしているプロジェクト (ほぼパッケージの意味) を一覧で出すだけ
- v1.0 のプロジェクトに関する情報は `name` のみ

project list の typing

```
Project = TypedDict("Project", {"name": str})
ProjectList = TypedDict(
    "ProjectList",
    {
        "meta": Meta,
        "projects": list[Project],
    },
)
```

project detail

- プロジェクト (パッケージ) ごとのダウンロード可能なファイル一覧
- ファイルの URL やパッケージメタデータなど

project detail の typing

```
ProjectDetail = TypedDict(
    "ProjectDetail",
    {
        "name": str,
        "files": list[ProjectFile],
        "meta": Meta,
    },
)
```

project file の typing

```
ProjectFile = TypedDict(
    "ProjectFile",
    {
        "filename": str,
        "url": str,
        "hashes": dict[str, str],
        "requires-python": NotRequired[str],
        "dist-info-metadata": NotRequired[str],
        "gpg-sig": NotRequired[bool],
        "yanked": NotRequired[bool],
    },
)
```

pypi version

- 今回は v1.0 の範囲でやってみます

```
{
  "meta": {
    "api-version": "1.0"
  }
}
```

wheel ファイルを探しだす

- pathlib でできちゃうね!
- wheel ファイルのファイル名は形式が決まっている
  - PEP 491 The Wheel Binary Package Format 1.9
  - `{distribution}-{version}(-{build tag})?-{python tag}-{abi tag}-{platform tag}.whl`.

metadata

- METADATA を wheel から取り出す
- wheel は zip ファイル
- METADATA の場所は決まっている
  - PEP 491 The Wheel Binary Package Format 1.9
  - `{distribution}-{version}.dist-info/` contains metadata.

```
def get_metadata(whl: pathlib.Path):
    parts = whl.name.split("-")
    dist_name, version = parts[0], parts[1]
    metadata_path = f"{dist_name}-{version}.dist-info/METADATA
    with zipfile.ZipFile(whl) as zf:
        with zf.open(metadata_path) as metadata:
            return metadata.read()
```

ダウンロード

- wheel ファイルの中身をレスポンスボディにする
- wheel の content-type は特に決まってないので application/octet-stream にする
- ブラウザでアクセスしたときにダウンロードになるよう Content-Disposition をつける

```
def get_data(self, whl: str) -> bytes:
    with self.wheels[whl]["data"].open("br") as f:
        return f.read()
```

```
def __call__(self, environ, start_response) -> Iterable[bytes]
    whl = environ["wsgiorg.routing_args"][1]["wheel_name"]
    start_response("200 OK", [("Content-type", "application/oc
stream")])
    return [self.repo.get_data(whl)]
```

pip から使う

- project list 呼ばれてないかも？

```
$ pip install pyramid --index-url=http://localhost:8000/
```

The Update Framework

- TUF

参考文献

- PyPA Simple Repository API, https://packaging.python.org/en/latest/specifications/simple-repository-api/
- The Update Framework, https://theupdateframework.io/