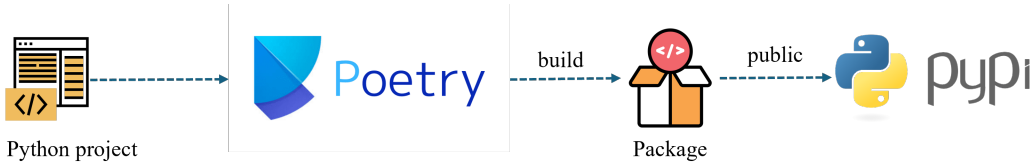
HƯỚNG DẪN TRIỂN KHAI VÀ ĐÓNG GÓI DỰ ÁN PYTHON VỚI POETRY

Dinh-Tiem Nguyen và Quang-Vinh Dinh

1 Mở đầu

Làm thế nào để xây dựng các package, library như math, numpy, tensorflow, pytorch? Làm thế nào để tái sử dụng code trong dự án? Trong bài viết này sẽ hướng dẫn cách triển khai xây dựng một package đơn giản với Poetry, sau đó phát hành nó trên nền tảng chia sẻ package nổi tiếng PyPI.
Yêu cầu:

- Máy tính đã cài đặt python
- Đã biết cơ bản về cách tạo và quản lí môi trường trong python



Hình 1: Pipline xây dựng package

2 Cài đặt thư viện Poetry

Poetry là một công cụ quản lý các package, library phụ thuộc trong môi trường riêng biệt để phát triển các dự án python, đặc biệt là nó cung cấp các chức năng thuận tiện cho việc xây dựng package python và phát hành lên nền tảng PyPI.
Có một số cách khác nhau để cài Poetry, trong hướng dẫn này sử dụng cách cài đặt thông qua pipx - một công cụ để cài đặt và quản lí các package, library python(không phải là các package, library như numpy, pandas mà bạn hay dùng để import khi viết code đâu, mà nó là các package có thể sử dụng từ dòng lệnh như Python, Poetry...) . Để cài đặt pipx, bạn xem hướng dẫn chi tiết tại đây. Đối với hệ điều hành window, các bạn cài đặt qua lệnh sau:

```
py -m pip install --user pipx
```

Tiếp theo chúng ta sẽ cài đặt poetry với lệnh sau:

```
pipx install poetry
```

Sau khi cài đặt hoàn tất, ta có thể xác minh cài đặt thành công không qua lệnh sau:

```
poetry --version
```

Nếu bạn thấy kết quả tương tự như `Poetry (version 1.8.2)` thì tức là đã cài đặt thành công rồi đó.

3 Thiết lập dự án

Để tạo một dự án mới với Poetry ta sẽ sử dụng lệnh poetry new, ví dụ chúng ta tạo dự án xây dựng package tính giai thừa có tên aio2024, ta sẽ sử dụng lệnh sau:

```
poetry new aio2024
```

Sau khi thực thi lệnh trên, ta sẽ nhận được một thư mục dự án có tên là aio2024 với cấu trúc sau:

```
|   pyproject.toml
|   README.md
|
|---aio2024
|       __init__.py
|
|---tests
        __init__.py
```

Trong thư mục aio2024 chứa các tệp và package được sử dụng với các mục đích khác nhau:

- aio2024 : Ta sẽ viết code để tạo package ở trong thư mục này
- tests : Ta sẽ viết code kiểm thử ở đây
- pyproject.toml : Là file chứa thông tin cấu hình package chúng ta đang xây dựng
- README.md : Là file mô tả package mà chúng ta đang xây dựng, hướng dẫn cài đặt... được viết bằng Markdown

Đó chính là cấu trúc tiêu chuẩn của một dự án xây dựng package với Poetry. Nếu ta có một dự án từ trước mà muốn đóng gói lại với Poetry thì cần phải tổ chức lại code theo cấu trúc này.
Vì file pyproject.toml chứa các thông tin quan trọng nên chúng ta sẽ cùng tìm hiểu nội dung của file này:

```
[tool.poetry]
name = "aio2024"
version = "0.1.0"
description = ""
authors = ["NguyenDinhTiem <nguyendinhtiem1999@gmail.com>"]
readme = "README.md"
packages = [{include = "aio2024"}]

[tool.poetry.dependencies]
python = "^3.12"


[build-system]
requires = ["poetry-core"]
build-backend = "poetry.core.masonry.api"
```

Phần đầu tiên trong file là [tool.poetry], trong phần này chứa thông tin:

- name : Tên của thư mục dự án
- version : Phiên bản mà chúng ta đang phát triển
- description : Mô tả ngắn về package
- authors : Thông tin về tác giả, nhóm phát triển package
- readme : Tệp để viết các mô tả, hướng dẫn chi tiết về package

- packages : Vị trí, tên của package, nơi mà chứa các file code của package.

Lưu ý: packages = [{include = "aio2024"}] đây là thông tin vị trí của package, theo mặc định thì poetry sẽ lấy tên theo tool.poetry.name nhưng trong một số trường hợp dự án có tên thư mục dự án và tên package khác nhau, nên chúng ta cần chỉ định lại tên của package. Nếu không sẽ bị lỗi khi build package.
Tiếp theo là phần [tool.poetry.dependencies] chứa thông tin về phiên bản python và các package, library tương thích với package chúng ta xây dựng. Ví dụ package chúng ta đang xây dựng cần sử dụng python, streamlit thì chúng sẽ được liệt kê tại đây.
Phần cuối của file là [build-system], phần này chứa một số thông tin về các phụ thuộc để thực hiện đóng gói package. Ta sẽ để các giá trị theo mặc định.
Theo mặc định phiên bản python Poetry sẽ sử dụng để tạo môi trường cho dự án là phiên bản python trên môi trường chung của máy tính. Ta có thể xem thông tin môi trường bằng cách điều hướng đến thư mục dự án vừa tạo và dùng lệnh sau:

```
===============Input=======================
cd aio2024
poetry env info
```

```
================= Output =================
Virtualenv
Python:         3.12.2
Implementation: CPython
Path:           NA
Executable:     NA

Base
Platform:   win32
OS:         nt
Python:     3.12.2
Path:       C:\Python312
Executable: C:\Python312\python.exe
==========================================
```

Chúng ta có thể kích hoạt môi trường trong Poetry bằng lệnh sau:

```
poetry shell
```

Môi trường này hoạt động tương tự như môi trường trong anaconda. Nhưng điều thú vị là không như Anaconda có thể kích hoạt môi trường ở mọi nơi, môi trường trong Poetry chỉ có thể kích hoạt tại trong chính vị trí thư mục dự án.
Để cài đặt một package hay library cho dự án, ta sẽ sử dụng lệnh sau

```
poetry add name_package
```

Ta cũng có thể cài nhiều package bằng cách thêm nó vào phần [tool.poetry.dependencies] sau đó sử dụng lệnh sau để cài đặt:

```
poetry install
```

4 Xây dựng và public package

4.1 Xây dựng package

Chúng ta sẽ xây dựng package với hai module, module1 là math.py chứa class MyMath với thuộc tính là một số tự nhiên value và một phương thức factorial() để tính giai thừa cho value. Module thứ 2 là cat.py, chúng ta sẽ xây dựng class Cat với thuộc tính name và phương thức describe().

```python
#math.py
class MyMath:
    def __init__(self, value:int) -> None:
        self.value = value
    def factorial(self) -> int:
        if self.value == 0:
            return 1
        else:
            return self.value * MyMath(self.value - 1).factorial()

#cat.py
class Cat:
    def __init__(self, name):
        self.__name = name

    def describe(self):
        print(self.__name)
```

4.2 Đóng gói và Xuất bản package

4.2.1 Đóng gói

Chúng ta đóng gói package để cung cấp cho các nhóm phát triển khác trong dự án hoặc để đăng tải trực tuyến cho cộng đồng python sử dụng. Trong Poetry chúng ta có thể thực hiện điều này rất dễ dàng. Đầu tiên chúng ta đóng gói lại dự án bằng lệnh sau:

```
==============Input======================
poetry build




=========================================
```

```
================= Output =================
Building aio2024 (0.1.0)
  - Building sdist
  - Built aio2024-0.1.0.tar.gz
  - Building wheel
  - Built aio2024-0.1.0-py3-none-any.whl
==========================================
```

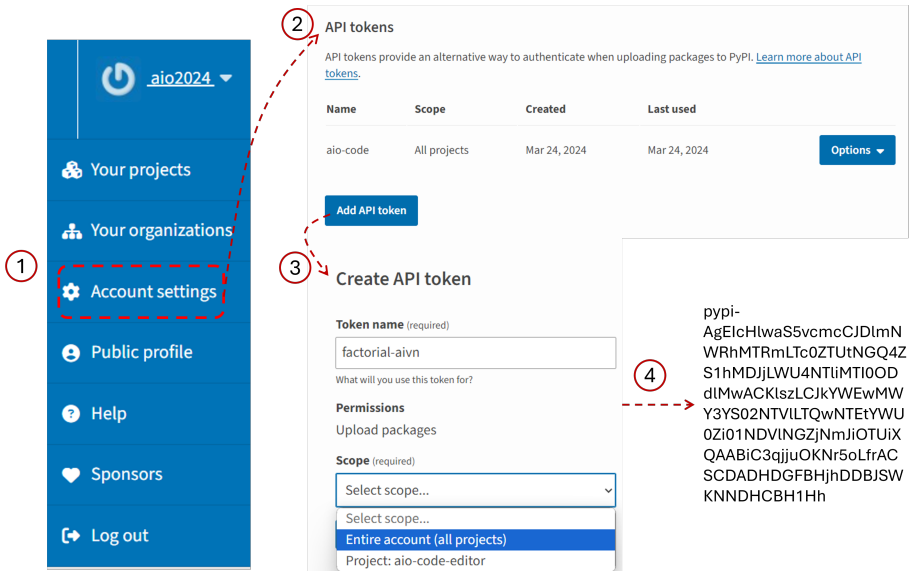
Việc đóng gói này sẽ sử dụng các cấu hình mà chúng ta thiết lập trong file pyproject.toml. Vậy là quá trình đóng gói package của chúng ta đã thành công.

4.2.2 Xuất bản package

Tiếp theo, để đăng tải lên PyPI, ta sẽ truy cập vào trang web pypi.org và tiến hành đăng nhập, nếu chưa có tài khoản chúng ta sẽ cần phải đăng ký một tài khoản mới.
Sau khi đăng nhập thành công, ta sẽ tạo một API token tại phần Account settings (1) hoặc tại link https://pypi.org/manage/account/token/. Sau đó tại phần API tokens chọn Add API tokens(2)phần Scope chúng ta chọn Entire account (all projects) như hình trên(3).
Lưu ý: Trước khi tạo token PyPI sẽ yêu cầu chúng ta xác thực Two factor authentication (2FA). Chúng ta cần làm theo thông báo đó để bật xác thực 2FA, sau đó mới có thể tạo token được.
(4)Cuối cùng chúng ta sẽ nhấn vào button create token, một đoạn mã chứa thông tin về token xuất hiện, ta cần copy lại và lưu vào một tệp nào đó trên máy tính của chúng ta để có thể xem lại khi cần. Vì đoạn mã này chỉ xuất hiện một lần duy nhất khi chúng ta tạo token trên PyPI.
Tiếp theo ta sẽ thêm tài khoản PyPI vào Poetry với lệnh sau:

```
poetry config http-basic.foo <username> <password>
```

Trong đó username là tên đăng nhập, password là mật khẩu của tài khoản PyPI mà chúng ta đã tạo. Tiếp theo chúng ta thêm thông tin token của tài khoản bằng lệnh:



Hình 2: Tạo PyPI API token

```
poetry config pypi-token.pypi <my-token>
```

Trong đó my-token là token ta đã tạo ở phía trên.
Cuối cùng chúng ta đẩy package của chúng ta lên nền tảng PyPI bằng lệnh:

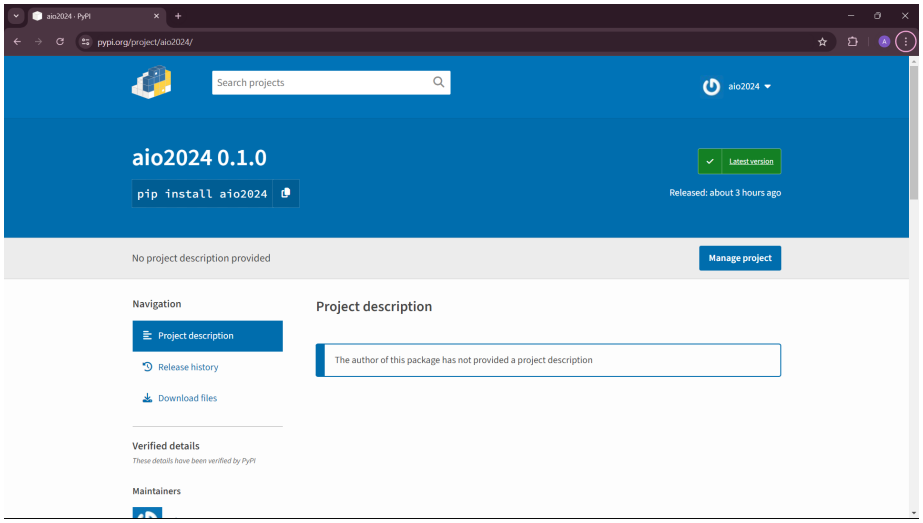
```
==========Input==========
poetry publish




=========================================
```

```
================= Output =================
Publishing aio2024 (0.1.0) to PyPI
 - Uploading aio2024-0.1.0-py3-none-any.
   whl 100%
 - Uploading aio2024-0.1.0.tar.gz 100%
==========================================
```

Vậy là package của chúng ta đã được đăng tải lên PyPI, ngay lúc này cả cộng đồng Python có thể sử dụng packge của chúng ta thông qua cú pháp pip install, thật là tuyệt vời ông mặt trời phải không?



Hình 3: Package được đăng tải lên PyPI