進階指南：如何編寫可重用程式

這篇進階指南從 Tutorial 7 結尾的地方繼續講起。我們將會把我們的 Web-poll 放進一個獨立的 Python 套件中，以便你在新的專案中重用它或將它與他人分享。

如果你尚未完成教學 1-7，我們推薦你先瀏覽一遍教學，這樣你的樣例工程會和下面的一致。

可重用性很重要

設計，構建，測試以及維護一個 web 應用要做很多的工作。很多 Python 以及 Django 專案都有一些常見問題。如果我們能儲存並利用這些重復的工作豈不是更好?

可重用性是 Python 的根本。The Python Package Index (PyPI) 有許大量的套件，都可被用在你自己的 Python 專案中。同樣可以在 Django Packages 中查找已發布的可重用應用，也可將其引入到你的專案中。Django 本身也是一個 Python 套件，也就是說你可以將已有的 Python 套件或 Django 應用並入你的專案。你只需要編寫屬於你的那部分即可。

假設你現在建立了一個新的專案，並且需要一個類似我們之前做的投票應用。你該如何復用這個應用呢？慶幸的是，其實你已經知道了一些。在 教學 1，我們使用過 **`include`** 從專案級別的 URLconf 分割出 polls。在本教學中，我們將進一步使這個應用易用於新的專案中，並發布給其他人安裝使用。

套件? 應用?

一個 package 提供了一組關聯的 Python 程式的簡單復用方式。一個套件（“模組”）套件含了一個或多個 Python 程式文件。

一個套件透過 **`import foo.bar`** 或 **`from foo import bar`** 的形式導入。一個目錄（例如 **`polls`**）要成為一個套件，它必須套件含一個特定的文件 **`__init__.py`**，即便這個文件是空的。

Django *應用* 僅僅是專用於 Django 專案的 Python 套件。應用會按照 Django 規則，建立好 **`models`**, **`tests`**, **`urls`**, 以及 **`views`** 等子模組。

稍後，我們將解釋術語 *打套件* ——為了方便其它人安裝 Python 套件的處理流程。我知道，這可能會使你感到一點點迷惑。

你的專案和可復用應用

透過前面的教學，我們的工程應該看起來像這樣:

```
mysite/
    manage.py
    mysite/
        __init__.py
        settings.py
        urls.py
        asgi.py
        wsgi.py
    polls/
        __init__.py
        admin.py
        apps.py
        migrations/
            __init__.py
            0001_initial.py
        models.py
        static/
            polls/
                images/
                    background.gif
```

```
            style.css
    templates/
        polls/
            detail.html
            index.html
            results.html
    tests.py
    urls.py
    views.py
templates/
    admin/
        base_site.html
```

1

目錄 **polls** 現在可以被拷貝至一個新的 Django 工程，且立刻被復用。不過現在還不是發布它的時候。為了這樣做，我們需要打套件這個應用，便於其他人安裝它。

安裝必須環境

目前，打套件 Python 程式需要工具，有許多工具可以完成此項工作。在此教學中，我們將使用 setuptools 來打套件我們的程式。這是推薦的打套件工具（與 發布 分支合並）。我們仍舊使用 pip 來安裝和卸載這個工具。現在，你需要安裝這兩個套件。如果你需要協助，你可以參考如何透過 pip 安裝 Django，你可以透過相同的方式安裝 **setuptools**。

打套件你的應用

Python 的 *打套件* 將以一種特殊的格式組織你的應用，意在方便安裝和使用這個應用。Django 本身就被打套件成類似的形式。對於一個小應用，例如 polls，這不會太難。

1. 首先，在你的 Django 專案目錄外建立一個名為 **django-polls** 的文件夾，用於盛放 **polls**。

   > **為你的應用選擇一個名字**
   >
   > 當為你的套件選一個名字時，避免使用像 PyPI 這樣已存在的套件名，否則會導致衝突。當你建立你的發布套件時，可以在模組名前增加 **django-** 前綴，這是一個很常用也很有用的避免套件名衝突的方法。同時也有助於他人在尋找 Django 應用時確認你的 app 是 Django 獨有的。
   >
   > 應用標籤（指用點分隔的套件名的最後一部分）在 **INSTALLED_APPS** 中 *必須* 是獨一無二的。避免使用任何與 Django contrib packages 文件中相同的標籤名，比如 **auth**，**admin**，**messages**。

2. 將 **polls** 目錄移入 **django-polls** 目錄。

3. 建立一個名為 **django-polls/README.rst** 的文件，套件含以下內容：

   django-polls/README.rst

```
=====
Polls
=====

Polls is a Django app to conduct Web-based polls. For each question,
visitors can choose between a fixed number of answers.

Detailed documentation is in the "docs" directory.

Quick start
-----------
```

```
1. Add "polls" to your INSTALLED_APPS setting like this::

    INSTALLED_APPS = [
        ...
        'polls',
    ]

2. Include the polls URLconf in your project urls.py like this::

    path('polls/', include('polls.urls')),

3. Run ``python manage.py migrate`` to create the polls models.

4. Start the development server and visit http://127.0.0.1:8000/admin/
   to create a poll (you'll need the Admin app enabled).

5. Visit http://127.0.0.1:8000/polls/ to participate in the poll.
```

4. 建立一個 **`django-polls/LICENSE`** 文件。選擇一個非本教學使用的授權協議，但是要足以說明發布程式沒有授權證書是 *不可能的*。Django 和很多相容 Django 的應用是以 BSD 授權協議發布的；不過，你可以自己選擇一個授權協議。只要確定你選擇的協議能夠限制未來會使用你的程式的人。

5. 下一步我們將建立 **`setup.cfg`**``和``**`setup.py`** 文件用於說明如何構建和安裝應用的細節。關於此文件的完整介紹超出了此教學的範圍，但是 setuptools docs 有詳細的介紹。建立文件 **`django-polls/setup.py`** 套件含以下內容：

django-polls/setup.cfg

```
[metadata]
name = django-polls
version = 0.1
description = A Django app to conduct Web-based polls.
long_description = file: README.rst
url = https://www.example.com/
author = Your Name
author_email = yourname@example.com
license = BSD-3-Clause  # Example license
classifiers =
    Environment :: Web Environment
    Framework :: Django
    Framework :: Django :: X.Y  # Replace "X.Y" as appropriate
    Intended Audience :: Developers
    License :: OSI Approved :: BSD License
    Operating System :: OS Independent
    Programming Language :: Python
    Programming Language :: Python :: 3
    Programming Language :: Python :: 3 :: Only
    Programming Language :: Python :: 3.6
    Programming Language :: Python :: 3.7
    Programming Language :: Python :: 3.8
    Topic :: Internet :: WWW/HTTP
    Topic :: Internet :: WWW/HTTP :: Dynamic Content

[options]
include_package_data = true
packages = find:
```

django-polls/setup.py

```
from setuptools import setup

setup()
```

6. 預設套件中只套件含 Python 模組和套件。為了套件含額外文件，我們需要建立一個名為 **MANIFEST.in** 的文件。上一步中關於 setuptools 的文件詳細介紹了這個文件。為了套件含模板、**README.rst** 和我們的 **LICENSE** 文件，建立文件 **django-polls/MANIFEST.in** 套件含以下內容：

   django-polls/MANIFEST.in

```
include LICENSE
include README.rst
recursive-include polls/static *
recursive-include polls/templates *
```

7. 在應用中套件含詳細文件是可選的，但我們推薦你這樣做。建立一個空目錄 **django-polls/docs** 用於未來編寫文件。額外增加一行至 **django-polls/MANIFEST.in**

```
recursive-include docs *
```

   注意，現在 **docs** 目錄不會被加入你的應用套件，除非你往這個目錄加幾個文件。許多 Django 應用也提供他們的在線文件透過類似 readthedocs.org 這樣的網站。

8. 試著構建你自己的應用套件透過 **ptyhon setup.py sdist** （在 **django-polls``目錄內**）。這將建立一個名為 **``dist** 的目錄並構建你自己的應用套件， **django-polls-0.1.tar.gz**。

更多關於打套件的資訊，見 Python 的 關於打套件和發布專案的教學。

使用你自己的套件名

由於我們把 **polls** 目錄移出了專案，所以它無法工作了。我們現在要透過安裝我們的新 **django-polls** 應用來修正這個問題。

> **作為用戶庫安裝**
>
> 以下步驟將 **django-polls** 以用戶庫的形式安裝。與安裝整個系統的軟件套件相比，用戶安裝具有許多優點，例如可在沒有管理員開啟權的系統上使用，以及防止應用套件影響系統服務和其他用戶。
>
> Note that per-user installations can still affect the behavior of system tools that run as that user, so using a virtual environment is a more robust solution (see below).

1. 為了安裝這個套件，使用 pip (你早已 安裝 pip, 對嗎？):

```
python -m pip install --user django-polls/dist/django-polls-0.1.tar.gz
```

2. 幸運的話，你的 Django 專案應該再一次正確執行。啟動伺服器確認這一點。

3. 透過 pip 卸載套件:

```
python -m pip uninstall django-polls
```

發布你的應用

現在，你已經對 **`django-polls`** 完成了打套件和測試，準備好向世界分享它！如果這不是一個例子應用，你現在就可以這樣做。

- 透過郵件將你的套件發送給朋友。
- 將這個套件上傳至你的網站。
- 將你的套件發布至公共倉庫，比如 the Python Package Index (PyPI)。 packaging.python.org 有一個不錯的 教學 說明如何發布至公共倉庫。

Installing Python packages with a virtual environment

早些時候，我們以用戶庫的形式安裝了投票應用。這樣做有一些缺點。

- 修改用戶庫會影響你系統上的其他 Python 軟件。
- 你將不能執行此套件的多個版本（或者其它用有相同套件名的套件）。

Typically, these situations only arise once you're maintaining several Django projects. When they do, the best solution is to use venv. This tool allows you to maintain multiple isolated Python environments, each with its own copy of the libraries and package namespace.

編寫你的第一個 Django 應用，第 7 部分

下一步看什麼