

# Rendre le projet installable

Rendre votre projet installable signifie que vous pouvez créer un fichier de *distribution* et l'installer dans un autre environnement, tout comme vous avez installé Flask dans l'environnement de votre projet. Cela rend le déploiement de votre projet identique à l'installation de n'importe quelle autre bibliothèque, de sorte que vous utilisez tous les outils Python standard pour tout gérer.

L'installation s'accompagne également d'autres avantages qui ne sont peut-être pas évidents à la lecture du tutoriel ou en tant que nouvel utilisateur de Python, notamment :

- Actuellement, Python et Flask comprennent comment utiliser le paquet `flaskr` uniquement parce que vous vous exécutez depuis le répertoire de votre projet. L'installation signifie que vous pouvez l'importer quel que soit l'endroit d'où vous vous exécutez.
- Vous pouvez gérer les dépendances de votre projet comme les autres paquets, pour que `pip install yourproject.whl` les installe.
- Les outils de test peuvent isoler votre environnement de test de votre environnement de développement.

---

## Note:

Cette fonction est introduite tardivement dans le tutoriel, mais dans vos futurs projets, vous devriez toujours commencer par cette fonction.

---


## Décrire le projet

Le fichier `setup.py` décrit votre projet et les fichiers qui lui appartiennent.

`setup.py`

```
from setuptools import find_packages, setup

setup(
    name='flaskr',
    version='1.0.0',
    packages=find_packages(),
    include_package_data=True,
    zip_safe=False,
    install_requires=[
        'flask',
    ],
)
```

`packages` indique à Python les répertoires de paquets (et les fichiers Python qu'ils  `v: latest` ▼ à inclure. `find_packages()` trouve ces répertoires automatiquement pour que vous n'ayez pas à les taper. Pour inclure d'autres fichiers, comme les répertoires static et templates, il faut défi-

nir `include_package_data`. Python a besoin d'un autre fichier nommé `MANIFEST.in` pour indiquer ce que sont ces autres données.

```
MANIFEST.in
```

```
include flaskr/schema.sql
graft flaskr/static
graft flaskr/templates
global-exclude *.pyc
```

Ceci indique à Python de copier tout ce qui se trouve dans les répertoires `static` et `templates`, et le fichier `schema.sql`, mais d'exclure tous les fichiers de bytecode.

Voir le [guide officiel de packaging](#) pour une autre explication des fichiers et options utilisés.

## Installer le projet

Utilisez `pip` pour installer votre projet dans l'environnement virtuel.

```
$ pip install -e .
```


Ceci indique à `pip` de trouver `setup.py` dans le répertoire courant et de l'installer en mode *éritable* ou *développement*. Le mode éritable signifie que lorsque vous apportez des modifications à votre code local, vous n'aurez à réinstaller que si vous changez les métadonnées du projet, comme ses dépendances.

Vous pouvez observer que le projet est maintenant installé avec `pip list`.

```
$ pip list
```

Package	Version	Location
click	6.7	
Flask	1.0	
flaskr	1.0.0	/home/user/Projects/flask-tutorial
itsdangerous	0.24	
Jinja2	2.10	
MarkupSafe	1.0	
pip	9.0.3	
setuptools	39.0.1	
Werkzeug	0.14.1	
wheel	0.30.0	

Rien ne change par rapport à la façon dont vous avez exécuté votre projet jusqu'à présent.

`FLASK_APP` est toujours défini à `flaskr` et `flask run` exécute toujours l'application  `v: latest` ▼ vous pouvez l'appeler de n'importe où, pas seulement du répertoire `flask-tutorial`.

Continuer vers [Couverture des tests.](#)