VINCENT LE GOFF

APPRENEZ À PROGRAMMER EN PYTHON

DÉVELOPPER EN PYTHON N'A JAMAIS ÉTÉ AUSSI FACILE !

le Livre du Zéro
www.siteduzero.com

Simple IT 2011 - ISBN : 979-10-90085-03-9

Chapitre 33

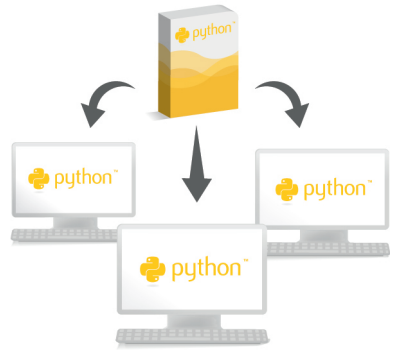
Distribuer facilement nos programmes Python avec cx_Freeze

Difficulté :

Comme nous l'avons vu, Python nous permet de générer des exécutables d'une façon assez simple. Mais, si vous en venez à vouloir distribuer votre programme, vous risquez de vous heurter au problème suivant : pour lancer votre code, votre destinaire doit installer Python ; qui plus est, la bonne version. Et si vous commencez à utiliser des bibliothèques tierces, il doit aussi les installer !

Heureusement, il existe plusieurs moyens pour produire des fichiers exécutables que vous pouvez distribuer et qui incluent tout le nécessaire.

Sur Windows, il faut enfermer vos fichiers à l'extension `.py` dans un `.exe` accompagné de fichiers `.dll`. **Cx_freeze** est un des outils qui permet d'atteindre cet objectif.



En théorie

L'objectif de ce chapitre est de vous montrer comment faire des programmes dits *standalone*[1]. Comme vous le savez, pour que vos fichiers `.py` s'exécutent, il faut que Python soit installé sur votre machine. Mais vous pourriez vouloir transmettre votre programme sans obliger vos utilisateurs à installer Python sur leur ordinateur.

Une version *standalone* de votre programme contient, en plus de votre code, l'exécutable Python et les dépendances dont il a besoin.

Sur Windows, vous vous retrouverez avec un fichier `.exe` et plusieurs fichiers compagnons, bien plus faciles à distribuer et, pour vos utilisateurs, à exécuter.

Le programme résultant ne sera pas sensiblement plus rapide ou plus lent. Il ne s'agit pas de compilation, Python reste un langage interprété et l'interpréteur sera appelé pour lire votre code, même si celui-ci se trouvera dans une forme un peu plus compressée.

Avantages de cx_Freeze

- Portabilité : **cx_Freeze** est fait pour fonctionner aussi bien sur Windows que sur Linux ou Mac OS ;
- Compatibilité : **cx_Freeze** fonctionne sur des projets Python de la branche 2.X ou 3.X ;
- Simplicité : créer son programme *standalone* avec **cx_Freeze** est simple et rapide ;
- Souplesse : vous pouvez aisément personnaliser votre programme *standalone* avant de le construire.

Il existe d'autres outils similaires, dont le plus célèbre est **py2exe**. Il est accessible via le code web suivant :

▷ Télécharger py2exe
Code web : 603896

Il a toutefois l'inconvénient de ne fonctionner que sur Windows et, à l'heure où j'écris ces lignes du moins, de ne pas proposer de version compatible avec Python 3.X.

Nous allons à présent voir comment installer **cx_Freeze** et comment construire nos programmes *standalone*.

En pratique

Il existe plusieurs façons d'utiliser **cx_Freeze**. Il nous faut dans tous les cas commencer par l'installer.

1. Que l'on peut traduire très littéralement par "se tenir seul".

Installation

Sur Windows

Rendez-vous sur le site `sourceforge`, où est hébergé le projet **cx_Freeze**, en utilisant le code web suivant :

▷ Télécharger cx_Freeze
Code web : 429982

et téléchargez le fichier correspondant à votre version de Python.

Après l'avoir téléchargé, lancez l'exécutable et laissez-vous guider. Rien de trop technique jusqu'ici !

Sur Linux

Je vous conseille d'installer **cx_Freeze** depuis les sources.

Commencez par vous rendre sur le site de téléchargement via le code web ci-dessus et sélectionnez la dernière version de **cx_Freeze** (*Source Code only*).

Téléchargez et décompressez les sources :

```
tar -xvf cx\_Freeze_version.tar.gz
```

Rendez-vous dans le dossier décompressé puis lancez l'installation en tant qu'utilisateur `root` :

```
$ cd cx\_Freeze_version
$ sudo python3.2 setup.py build
$ sudo python3.2 setup.py install
```

Si ces deux commandes s'exécutent convenablement, vous disposerez de la commande `cxfreeze` :

```
$ cxfreeze
cxfreeze: error: script or a list of modules must be specified
```

Utiliser le script `cxfreeze`

Pour les utilisateurs de Windows, je vous invite à vous rendre dans la ligne de commande (`Démarrer` > `Exécuter...` > `cmd`).

Rendez-vous dans le sous-dossier `scripts` de votre installation Python (chez moi, `C:\python32\scripts`).

```
cd \
cd C:\python32\scripts
```

Sur Linux, vous devez avoir accès au script directement. Vous pouvez le vérifier en tapant `cxfreeze` dans la console. Si cette commande n'est pas disponible mais que vous avez installé **cxfreeze**, vous pourrez trouver le script dans le répertoire `bin` de votre version de Python.

Sur Windows ou Linux, la syntaxe du script est la même : `cxfreeze fichier.py`.

Faisons un petit programme pour le tester. Créez un fichier `salut.py` (sur Windows, mettez-le dans le même dossier que le script `cxfreeze`, ce sera plus simple pour le test). Vous pouvez y placer le code suivant :

```
1 """Ce fichier affiche simplement une ligne grâce à la fonction
      print."""
2
3 import os
4
5 print("Salut le monde !")
6
7 # Sous Windows il faut mettre ce programme en pause (inutile
      sous Linux)
8 os.system("pause")
```

N'oubliez pas la ligne spécifiant l'encodage.

À présent, lancez le script **cxfreeze** en lui passant en paramètre le nom de votre fichier : `cxfreeze salut.py`.

Si tout se passe bien, vous vous retrouvez avec un sous-dossier `dist` qui contient les bibliothèques dont votre programme a besoin pour s'exécuter... et votre programme lui-même.

Sur Windows, ce sera `salut.exe`. Sur Linux, ce sera simplement `salut`.

Vous pouvez lancer cet exécutable : comme vous le voyez, votre message s'affiche bien à l'écran.

Formidable ! Ou pas...

Au fond, vous ne voyez sans doute pas de différence avec votre programme `salut.py`. Vous pouvez l'exécuter, lui aussi, il n'y a aucune différence.

Sauf que l'exécutable que vous trouvez dans le sous-dossier `dist` n'a pas besoin de Python pour s'exécuter : il contient lui-même l'interpréteur Python.

Vous pouvez donc distribuer ce programme à vos amis ou le mettre en téléchargement sur votre site, si vous le désirez.

Une chose importante à noter, cependant : veillez à copier, en même temps que votre programme, tout ce qui se trouve dans le dossier `dist`. Sans quoi, votre exécutable pourrait ne pas se lancer convenablement.

Le script **cxfreeze** est très pratique et suffit bien pour de petits programmes. Il com-

porte certaines options utiles que vous pouvez retrouver dans la documentation de **cx_Freeze**, accessible avec le code web suivant :

▷ Documentation cx_Freeze
Code web : 713172

Nous allons à présent voir une seconde méthode pour utiliser **cx_Freeze**.

Le fichier `setup.py`

La seconde méthode n'est pas bien plus difficile mais elle peut se révéler plus puissante à l'usage. Cette fois, nous allons créer un fichier `setup.py` qui se charge de créer l'exécutable de notre programme.

Un fichier `setup.py` basique contient ce code :

```
1  """Fichier d'installation de notre script salut.py."""
2
3  from cx\_Freeze import setup, Executable
4
5  # On appelle la fonction setup
6  setup(
7      name = "salut",
8      version = "0.1",
9      description = "Ce programme vous dit bonjour",
10     executables = [Executable("salut.py")],
11 )
```

Tout tient dans l'appel à la fonction `setup`. Elle possède plusieurs arguments nommés :

- `name` : le nom de notre futur programme.
- `version` : sa version.
- `description` : sa description.
- `executables` : une liste contenant des objets de type `Executable`, type que vous importez de `cx_Freeze`. Pour se construire, celui-ci prend en paramètre le chemin du fichier `.py` (ici, c'est notre fichier `salut.py`).

Maintenant, pour créer votre exécutable, vous lancez `setup.py` en lui passant en paramètre la commande `build`.

Sur Windows, dans la ligne de commande : `C:\python32\python.exe setup.py build`.

Et sur Linux : `$ python3.2 setup.py build`.

Une fois l'opération terminée, vous aurez dans votre dossier un sous-répertoire `build`. Ce répertoire contient d'autres sous-répertoires portant différents noms en fonction de votre système.

Sur Windows, je trouve par exemple un dossier appelé `exe.win32-3.2`.

Dans ce dossier se trouve l'exécutable de votre programme et les fichiers dont il a besoin.

Pour conclure

Ceci n'est qu'un survol de **cx_Freeze**. Vous trouverez plus d'informations dans la documentation indiquée plus haut, si vous voulez connaître les différentes façons d'utiliser **cx_Freeze**.

En résumé

- **cx_Freeze** est un outil permettant de créer des programmes Python *standalone*.
- Un programme *standalone* signifie qu'il contient lui-même les dépendances dont il peut avoir besoin, ce qui rend sa distribution plus simple.
- **cx_Freeze** installe un script qui permet de créer nos programmes *standalone* très rapidement.
- On peut arriver à un résultat analogue en créant un fichier appelé traditionnellement `setup.py`.