Squelette de projet C++ à la demande

Créer et instancier un modèle de projet C++ grâce à Cookiecutter

Sébastien Dinot, 4 novembre 2023

Pourquoi un modèle de projet ?

- Pour être efficace quand on amorce un projet
- Dans quelles circonstances ?
  - Quand on instancie souvent des projets
  - Quand on instancie rarement des projets
  - Quand on veut faire gagner du temps aux autres
  - Quand on veut diffuser et homogénéiser des pratiques

Pourquoi un modèle de projet paramétré ?

- Adaptation manuelle d'un modèle « statique » possible :

  https://gitlab.com/sdinot/static-cpp-template

  Clonez ce dépôt, il va être notre base de travail

- C'est vite fastidieux !

- Il nous faut un modèle paramétré, qui adapte le squelette à la volée, avec les informations qu'on lui donne

Cookiecutter

- Instanciation d’un squelette de projet à partir d’un modèle paramétré
- Pour Python… et à peu près tout ce que vous voulez !
- https://www.cookiecutter.io

Éléments de base

- Modèle de projet (arborescence de sources)
- « Moteur de template » Jinja
  - https://jinja.palletsprojects.com
- Fichier de paramètres (format JSON)
- Python
  - Scripts de prologue et d'épilogue (« hooks »)
  - Extensions du moteur Jinja

Fichier de paramètres

- Format JSON : les paramètres et leur valeur par défaut

```
{
  "project_name": "My Amazing Project",
  "author": "John Doe",
  "cpp_standard": [ "C++20", "C++17", "C++14", "C++11", "Unspecified" ]
}
```

- On les retrouve sous forme de variables Jinja

```
cookiecutter.project_name
cookiecutter.author
cookiecutter.cpp_standard
```

- Et dans les hooks Python (traités eux aussi par Jinja)

```
name = "{{ cookiecutter.project_name }}"
files_to_be_removed = [
  '{% if cookiecutter.packaging == "no" %} conanfile.txt {% endif %}'
]
```

Structure d’un projet Cookiecutter

```
cookiecutter-something/
├── {{ cookiecutter.slug }}/  <--- Racine du squelette de projet à créer
│   ├── CmakeLists.txt  <--------- Fichiers composant le squelette
│   ├── README.md
│   └── ...
│
├── hooks     <------------------- Répertoire des « hooks » Python
│   ├── pre_gen_project.py   <---- Prologue
│   ├── post_gen_project.py  <---- Épilogue
│   └── pre_prompt.py        <---- Avant affichage de chaque question
│
├── blah.txt  <------------------- Les ressources ne faisant pas partie
│                                  du squelette sont créées en dehors
│
└── cookiecutter.json  <---------- Fichier de paramètres de Cookiecutter
```

Deux mots de Jinja

- Délimiteurs

```
{{ ... }} pour les variables
{% ... %} pour les instructions
{# ... #} pour les commentaires
```

- Variables

```
{{ cookiecutter.project_name }}
```

- Filtres

```
{{ cookiecutter.project_name | lower | replace(' ','-') }}
{{ authors | join(‘, ‘) }}
```

- Instructions

```
{% if cookiecutter.packaging == "no" %} conanfile.txt {% endif %}
```

Extensions Jinja fournies par Cookiecutter

- Jsonify

  `{{ cookiecutter | jsonify }}` *=> Structure de données au format JSON*

- Random string

  `{{ random_ascii_string(12, punctuation=True) }}` *=> chaine aléatoire : Af@TrI3b.Que*

- Slugify

  ```
  {{ cookiecutter.project_name | slugify }}
  ```
  => Plus sûr que :
  ```
  {{ cookiecutter.project_name | lower | replace(' ','-') }}
  ```

- UUID (version 4, valeur générée aléatoire)

  `{{ uuid4() }}` *=> 83b5de62-31b4-4a1e-83fa-8c548de65a11*

Installation de Cookiecutter

- Paquet Debian (version 1.7.3 contre 2.4.0 à date)

  ```
  sudo apt install cookiecutter
  ```

- Pip

  ```
  pip install --user cookiecutter
  ```

- Pipx

  ```
  pipx install cookiecutter
  ```

- Possible aussi via conda, brew…

Exécution de Cookiecutter

- Paquet Debian

  ```
  cookiecutter /path/to/cookiecutter-template
  ```

- Pip

  ```
  cookiecutter /path/to/cookiecutter-template
  ```

- Pipx

  ```
  cookiecutter /path/to/cookiecutter-template
  ```