

Un programme installable

TD d'application : transformer le script du TD 4a en projet Python installable, avec une fonction main, deux modules, un environnement et une commande

`cours4/4b_paquet/` `≈ 55'`

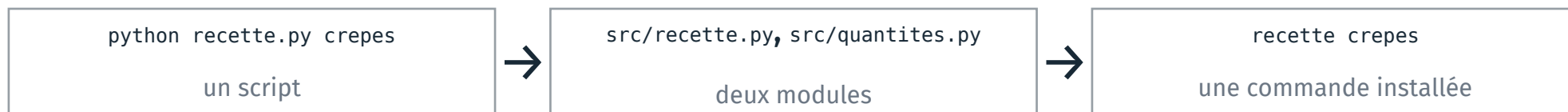
Le guide et le matériel de départ

Le TD part du programme de l'étape B6 du TD 4a, dans le même dossier. Le guide détaille chaque étape.

Fichier	Emplacement	Usage
guide_4b_paquet.pdf, .html, guide.ipynb	cours4/4b_paquet/	le détail des étapes, le code à copier
recette.py, recettes/	cours4/4a_recette/travail/recette/	le projet du TD 4a, à l'étape B6
environment.yml, pyproject.toml	cours4/4b_paquet/depart/modeles/	l'environnement (C3) et la fiche du projet (C4), à copier et compléter

Les étapes du TD

Le script du TD 4a devient un programme installé, lancé par la commande `recette`, dans un environnement créé depuis un fichier.



Étape	Objectif	Résultat
C1	isoler le programme principal	<code>import recette</code> n'exécute rien
C2	répartir le code en deux modules	<code>quantites.py</code> , importé
C3	décrire l'environnement dans un fichier	l'environnement <code>recette</code>
C4	installer une commande	<code>recette crepes -p 6</code>
C5 (bonus)	lancer pandoc depuis Python	<code>sortie/crepes_6_SI.html</code>

Étape C1 · Une fonction main

Objectif : le même script, dont les fonctions peuvent être importées par un autre fichier sans lancer le programme.

Attendu à la fin de l'étape

- ✓ `python recette.py crepes -p 6` affiche la même recette qu'à l'étape B6
- ✓ `python -c "import recette"` n'affiche rien
- ✓ un commit

Pistes

1. écrire `def main()` : au-dessus de la première ligne du programme
2. sélectionner les lignes du programme, puis `Tab` : elles passent dans `main`
3. appeler `main()` en bas du fichier, sous `if __name__ == "__main__":`

Guide, étape C1.

Étape C2 · Deux modules

Objectif : séparer ce qui calcule de ce qui dépend du programme, dans deux fichiers.

Attendu à la fin de l'étape

- ✓ `quantites.py` contient les deux tables et les cinq fonctions
- ✓ `recette.py` ne contient plus que les chemins, `main` et son appel
- ✓ `python recette.py crepes -p 6` affiche la même recette ; un commit

Pistes

1. créer `quantites.py`, y couper-coller les tables et les fonctions, avec `import csv` en tête
2. dans `recette.py`, importer ce qu'il emploie : `from quantites import ...`
3. ajouter le nouveau fichier : `git add quantites.py`, puis le commit

Guide, étape C2.

Étape C3 · Un environnement pour le projet

Objectif : un environnement qu'une autre personne recrée à l'identique, à partir d'un fichier versionné avec le code.

Attendu à la fin de l'étape

- ✓ `environment.yml` versionné dans le dépôt
- ✓ l'invite commence par `(recette)`
- ✓ `which python` affiche un chemin qui contient `envs/recette` ; le programme fonctionne

Pistes

1. copier le modèle : `cp ../../../../4b_paquet/depart/modeles/environment.yml .`
2. créer l'environnement : `conda env create -f environment.yml` (plusieurs minutes)
3. l'activer : `conda activate recette`
4. faire le commit du fichier

Guide, étape C3.

Étape C4 · Une commande installée

Objectif : lancer le programme par son nom, comme `git` ou `pandoc`.

Attendu à la fin de l'étape

- ✓ `recette crepes -p 6` affiche la recette, sans `python` ni nom de fichier
- ✓ `recette --help` affiche l'aide
- ✓ un commit

Pistes

1. déplacer les modules : `mkdir src`, puis `git mv recette.py quantites.py src/`
2. remonter la racine d'un dossier : `RACINE = Path(__file__).parent.parent`
3. copier `pyproject.toml`, compléter sa ligne `description`
4. installer : `pip install -e . --no-build-isolation`

Guide, étape C4.

```
recette/  
├─ src/  
│   ├── recette.py  
│   └─ quantites.py  
├─ recettes/  
├─ environment.yml  
├─ pyproject.toml  
└─ .gitignore
```

Étape C5 (bonus) · La page HTML, par pandoc

Objectif : lancer un programme extérieur depuis Python, comme au cours 3.

Attendu à la fin de l'étape

- ✓ `recette crepes -p 6 --page écrit sortie/crepes_6_SI.html`
- ✓ la page s'ouvre dans le navigateur, avec le titre et le tableau des ingrédients

Pistes

1. écrire un tableau Markdown dans `sortie/`
2. le convertir en page HTML : `subprocess.run(["pandoc", ...], check=True)`
3. ajouter l'option `--page : action="store_true"`

Guide, étape C5. pandoc est dans l'environnement `recette`.