

Le client, le noyau, et où ils sont installés

Objectif : installer de quoi ouvrir un notebook de deux façons, le client et le noyau au même endroit ou séparés

`cours1/1i_noyaux/` $\approx 20'$

Ce que l'environnement contient, et ce qui manque

Un notebook demande deux choses : un **client**, qui affiche la page, et un **noyau**, qui exécute les cellules.

	Ce qu'il faut faire
1	<code>conda list -n base jupyterlab ipykernel</code>
2	<code>conda activate recette, celui du TD 1h, puis jupyter lab</code>
3	dans VSCode, ouvrir <code>altitudes.ipynb</code> et choisir <code>recette</code> comme noyau

Le TD 1g marchait sans rien installer parce que les postes de la salle ont Anaconda, dont l'environnement `base` porte les deux. Une installation Miniforge part d'un `base` minimal.

Tout dans le même environnement

La façon la plus simple : poser le client dans l'environnement du projet. Il entraîne le noyau avec lui, et les deux moitiés sont au même endroit.

	Ce qu'il faut faire
4	<code>conda install -n recette -c conda-forge jupyterlab</code>
5	<code>conda activate recette</code> , puis <code>jupyter lab</code>
6	ouvrir <code>altitudes.ipynb</code> , et lire le noyau en haut à droite
7	fermer l'onglet, puis <code>Ctrl + C</code> deux fois dans le terminal

C'est la configuration du TD 1g, refaite à la main : `jupyterlab` déclare `ipykernel` dans ses dépendances, donc l'installer une fois suffit.

Le client d'un côté, le noyau de l'autre

L'autre façon : un client installé une fois, et un environnement par projet qui se déclare auprès de lui. C'est celle qu'on trouve en stage.

	Ce qu'il faut faire	Ce que vous constatez
8	<code>conda create -n altitudes -c conda-forge python=3.12 ipykernel numpy</code>	
9	<code>conda run -n altitudes python -m ipykernel install --user --name altitudes</code>	
10	<code>conda activate recette, puis jupyter kernelspec list</code>	
11	ouvrir <code>altitudes.ipynb</code> , choisir le noyau <code>altitudes</code> , taper <code>import numpy</code>	

`altitudes` n'a pas `jupyterlab`, et n'en a pas besoin : le `kernel.json` écrit à l'étape 9 ne contient qu'un chemin, `.../envs/altitudes/bin/python`.

Le notebook du projet recette

`recette.ipynb` reprend le programme du TD 1h, une fonction par cellule et rien d'importé. Il lit les données du projet, et c'est à vous de lui dire où elles sont.

	Ce qu'il faut faire
12	ouvrir <code>recette.ipynb</code> , lire la première cellule
13	commenter chaque ligne de cette cellule, puis l'exécuter
14	Noyau → Redémarrer et tout exécuter
15	changer <code>PERSONNES</code> , puis tout réexécuter

Le noyau à choisir est celui de `recette` : c'est lui qui porte `markdown` et `tabulate`. Avec celui d'`altitudes`, les dernières cellules échoueraient.