

Programmation Python

Installation et outils

SAÏDI MASSINISSA



Table des matières



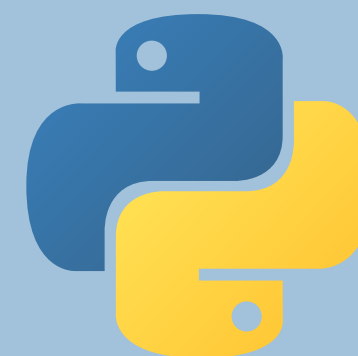
- Installation Anaconda
- IDE
- Environnement virtuel avec conda
- Git
- Bonnes pratiques Python



Installation Anaconda



IDE / Notebook



Présentation

- IDE : Integrated Development Environment
- Ensemble d'outil regroupé pour faciliter le développement
- Spyder déjà intégré dans Anaconda
- Les plus connus : VSCode et PyCharm
- **Notebooks** : page interactif permettant d'exécuter du code python avec de la mise en forme (markdown)



Notebook : commandes magiques

- % en début de ligne
- %% opèrent sur la cellule
- %run myscript.py
- %timeit L = [n ** 2 for n in range(1000)] pour mesurer le temps d'exécution du code
- %lsmagic



Environnement virtuel avec conda



Présentation de conda

- Gestionnaire de libraries et d'environnements.
- Fonctionne avec Python, R et d'autres langages.
- Cross-platform: Windows, macOS, Linux.



Motivations

- Gestion des dépendances et isolation des projets : évite les conflit entre les différentes versions de libraries
- Reproductibilité : évite le “ça marchait sur mon ordi”
- Performance : installation rapide et efficace des libraries

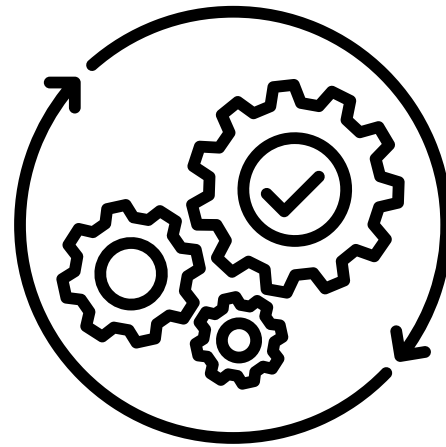


</> Explication du conflit



Projet A

installation des liibraries

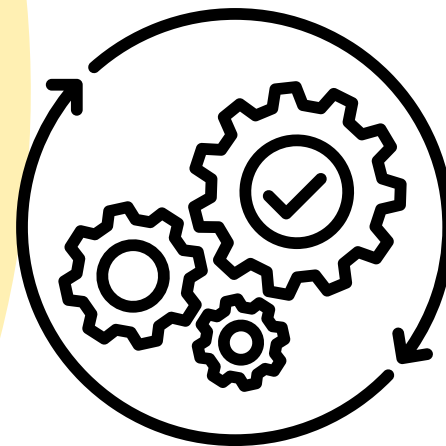


numpy==1.9.5
pandas==2.0.1



Projet B

installation des liibraries



numpy==1.0.0
pandas==1.4.2



Création d'un environnement Conda

Création d'un nouvel environnement

```
conda create -n mon_env --yes python=3.10
```

Activer l'environnement

```
conda activate mon_env
```

A noter que l'environnement par défaut est *base*



Gestion des libraries

Installation d'une library

*conda install numpy **ou** pip install numpy==1.9.5*

Supprimer d'une library

*conda remove numpy **ou** pip uninstall numpy*

Lister les libraries

*conda list **ou** pip freeze **ou** pip list*

Exporter les libraries

*conda list --export > requirements.txt **ou** pip freeze > requirements.txt*



Gestion des environnements

Lister les environnements

conda env list

Supprimer un environnement

conda remove -n mon_env --all



Conclusion

- Simplifie la gestion de projets
- Facilite la collaboration et le déploiement
- Offre une solution robuste pour la gestion des dépendances et versions



GIT / GITLAB

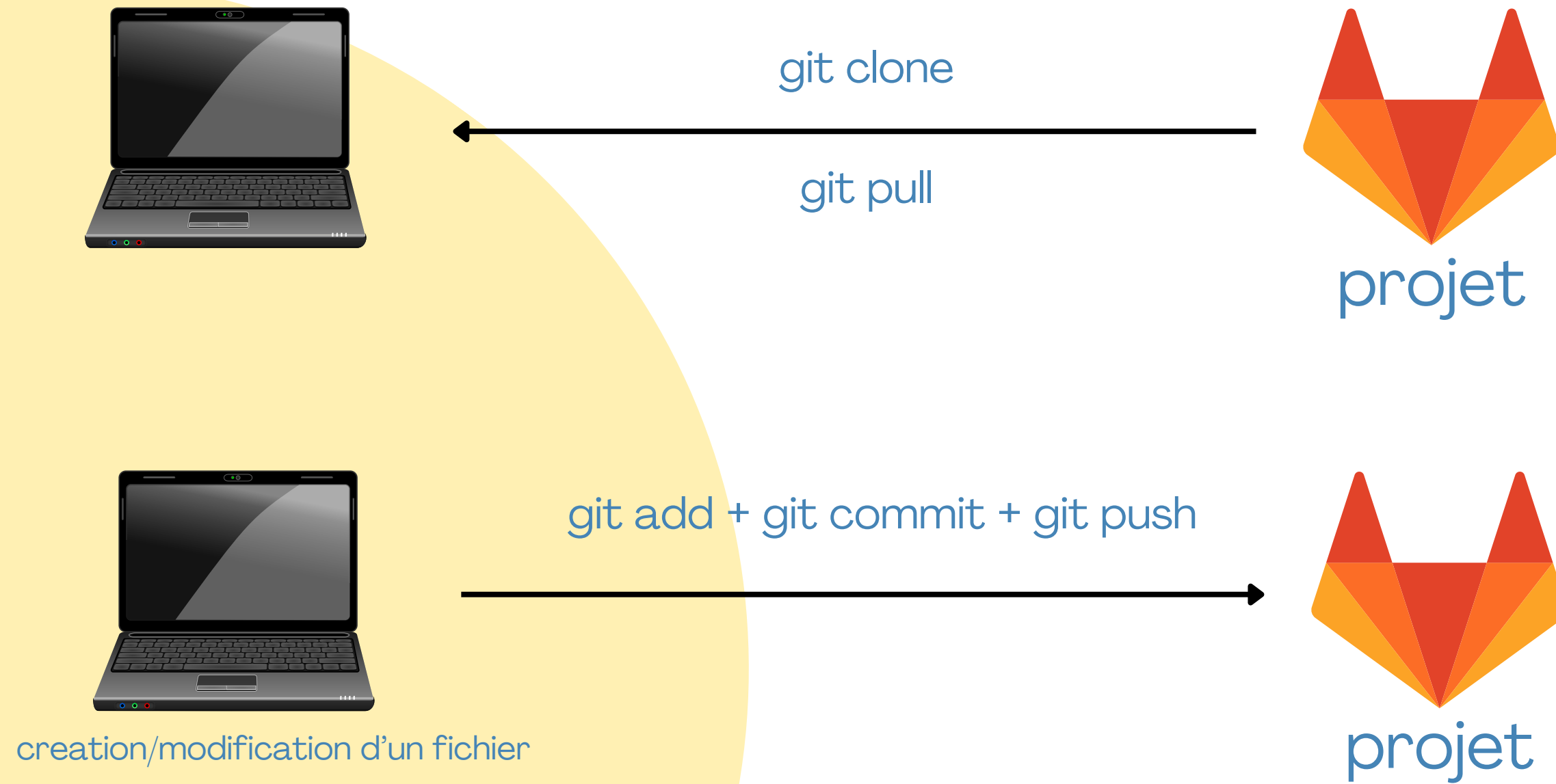


Présentation

- Outil permettant de faciliter la collaboration
 - système de branching et merging
 - suivi des modifications
- Versionne les fichiers
 - Réversibilité : Possibilité de revenir à une version précédente.
- Stock le code source



Fonctionnement



Gitlab : créer un projet

- création d'un compte gitlab
- création d'un projet



Git : commandes

Cloner un projet

git clone [url]

Créer un fichier à la racine du projet

Visualiser les changements

Ajouter le fichier dans le projet git

git add [fichier]

Commit le fichier

git commit -m "Ajout du fichier XXX"

Envoyer les modifications

git push



Bonnes pratiques



Structuration d'un projet

- Pas de nomenclature officielle mais très important
- Division scripts, données brutes, données traitées, notebooks, documents.
- Des IDE comme VSCode, PyCharm permettent de créer des projets avec une localisation, un environnement virtuel.
- Packages structuration projet : Cookiecutter DS, Kedro
- Cours OPC : Écrivez du code Python maintenable
- **PEP 8** : guide de bonnes pratiques Python
 - Être consistant.
 - Toujours préférer l'explicite à l'implicite.
 - Commenter le code.
 - Penser à la docstring.
 - Bien organiser le code (respecter l'indentation, éviter les lignes trop longues (<79), sauts de ligne, etc...)

