Gestion de version et d’environnements virtuels pour Python avec pyenv

`pyenv` permet d'installer différentes versions de python et de créer des environnements "virtuels" dans lesquels il est possible d'installer des modules, d'utiliser des version spécifiques de python, sans "corrompre" l'installation de python sur votre système. Un nouvel environnement est créé à l'intérieur de chaque projet.

Liens GitHub

- pyenv;
- pyenv-virtualenv;
- installation automatique.

Installation

- Installer les modules `pyenv` et `pyenv-virtualenv` grâce à l'installation automatique. L'installation automatique installe pyenv et pyenv-virtualenv :

```
# (!!! Si vous n'AVEZ PAS curl installé sur votre système !!!)
$ sudo apt install curl

$ curl https://pyenv.run | bash
```

Lisez bien la sortie du terminal, il vous demande d'ajouter les lignes suivantes dans votre fichier **.bashrc**

```
export PATH="~/.pyenv/bin:$PATH"
eval "$(pyenv init -)"
eval "$(pyenv virtualenv-init -)"
```

Gérer les versions de python

- Liste des commandes :

```
$ pyenv commands
```

- Lister les versions de python disponibles (et autres)

```
$ pyenv install -l
```

- Installer une version de python

```
$ pyenv install PYTHON_VERSION
```

- Désinstaller une version de python

```
$ pyenv uninstall PYTHON_VERSION
```

- Lister les versions installées

```
$ pyenv versions
```

- Afficher la version active en cours

```
$ pyenv version
```

- Définir une version par défaut pour le système

```
$ pyenv global PYTHON_VERSION
```

- Revenir à la version du système

```
$ pyenv global system
```

- Définir une version par défaut pour un répertoire (crée un fichier .python-version)

```
$ pyenv local PYTHON_VERSION
```

- Annuler la version par défaut pour un répertoire

```
$ pyenv local --unset
```

- Définir une version par défaut pour le shell en cours

```
$ pyenv shell PYTHON_VERSION
```

- Revenir à la version du système

```
$ pyenv shell --unset
```

Gérer un environnement virtuel

- Création :

```
$ pyenv virtualenv PYTHON_VERSION ENV_NAME
```

- Liste des environnements créés :

```
$ pyenv virtualenvs
```

- Démarrer l'environnement virtuel :

```
$ pyenv activate ENV_NAME
```

- Installer des modules dans l'environnement virtuel :

```
(ENV_NAME)$ pip install MODULE
```

- Quitter l'environnement virtuel :

```
(ENV_NAME)$ pyenv deactivate
```

- Supprimer l'environnement virtuel :

```
$ pyenv virtualenv-delete ENV_NAME

$ pyenv uninstall ENV_NAME
```

Utiliser jupyter dans virtualenv

- Se connecter à votre environnement virtuel (Le créer et le configurer le cas échéant) :

```
$ pyenv activate ENV_NAME
```

Installer jupyter :

```
(ENV_NAME) $ pip install jupyter
(ENV_NAME) $ pip install ipykernel
```

- Configurer le kernel

```
(ENV_NAME) $ python -m ipykernel install --user --name ENV_NAME --display-name
ENV_NAME
```

- Lancer jupyter, un navigateur web devrait se lancer…

```
(ENV_NAME) $ jupyter notebook
```

- Vérifier la version du kernel : Menu Kernel → Change kernel