pyenv+poetry

～ Python 環境構築～

目次

pyenv, Poetry とは

pyenv	Poetry
Python の バージョン管理ツール	Python の パッケージ管理ツール

二つのツールを併用することで

- Python と依存パッケージの管理ができる
- 仮想環境を構築できる
- プロジェクト構成を一貫して定義できる

インストールする環境

windows11

pyenv インストール ①

1. ユーザーローカルに `.pyenv` ディレクトリを作成する

```
> mkdir $env:USERPROFILE\.pyenv
```

2. Git clone する

```
> cd $env:USERPROFILE\.pyenv; git clone https://github.com/pyenv-win/pyenv-win.git .
```

3. 環境変数へ PYENV と PYENV_HOME を追加

```
> [System.Environment]::SetEnvironmentVariable('PYENV',$env:USERPROFILE + "\.pyenv\pyenv-win\","User")
> [System.Environment]::SetEnvironmentVariable('PYENV_HOME',$env:USERPROFILE + "\.pyenv\pyenv-win\","User")
```

4. 環境変数に Path を追加する

```
> [System.Environment]::SetEnvironmentVariable('path', $env:USERPROFILE + "\.pyenv\pyenv-win\bin;" + $env:USERPROFILE + "\.pyenv\pyenv-win\shims;" + [System.Environment]::GetEnvironmentVariable('path', "User"),"User")
```

pyenv インストール ②

5. インストール確認

```
> pyenv --version
```

6. ポリシーエラーが出たら管理者権限 PowerShell で下記コマンドを実行

```
> Set-ExecutionPolicy -ExecutionPolicy RemoteSigned
```

7. 利用可能な Python バージョンの確認

```
> pyenv versions
```

8. インストール可能な Python バージョンの確認

```
> pyenv install --list
```

pyenv インストール ③

9. Python のインストール

```
> pyenv install [バージョン]
# pyenv install 3.12
```

10. 再ハッシュ、同期

```
> pyenv rehash
```

11. インストールした Python のセット

```
> pyenv local [バージョン] #3.12.1など
> pyenv global [バージョン]
```

> local：現在のディレクトリに対してセット
> global：システム全体にセット

Poetry インストール ①

1. インストール、うまくいかないときは管理者権限で

```
> (Invoke-WebRequest -Uri https://install.python-poetry.org -UseBasicParsing).Content | python -
```

2. 環境変数に設定

```
> [System.Environment]::SetEnvironmentVariable('path', $env:APPDATA + "\Python\Scripts;" + [System.Environment]::GetEnvironmentVariable('path', "User"),"User")
```

3. インストールの確認

```
> poetry --version
```

Poetry インストール ②

4. 仮想環境ディレクトリをプロジェクト直下に変更

```
> poetry config virtualenvs.in-project true
```

5. pyenv で指定したバージョンを Poetry で使用するように変更

```
> poetry config virtualenvs.prefer-active-python true
```

仮想環境の構築手順 ①

1. プロジェクトルートに移動

```
> cd path/to/projectroot
```

2. プロジェクトで使用する Python のバージョンを指定

```
> pyenv versions
> pyenv local [バージョン]
```

3. プロジェクトの作成

```
> poetry new [プロジェクト名]
```

仮想環境の構築手順 ②

4. Python バージョンを再指定して `.python-version` ファイルを作成する

```
> pyenv local [バージョン]
```

5. 仮想環境ディレクトリのパスを指定して `poetry.toml` ファイルを作成する

```
> poetry config virtualenvs.in-project true --local
```

6. 仮想環境のセットアップ（プロジェクト配下にフォルダを作成し cd しておく）

```
> poetry install
```

コマンド ①

- Python パッケージの追加

```
> poetry add [モジュール名]
```

- Python の実行方法

```
> poetry run python [Pythonスクリプトのパス]
```

pytest や pyinstaller の場合

```
> poetry run pyinstaller app.py
> poetry run pytest test.py
```

- pyenv のアップデート

```
> cd $env:USERPROFILE\.pyenv\pyenv-win; git pull
```

コマンド ②

- Poetry のアップデート

```
> poetry self update
```

- `requirements.txt` のエクスポート

```
> poetry export -f requirements.txt --output requirements.txt
```

- インストールされているパッケージの確認

```
> poetry show
```

- インストールされているパッケージの削除

```
> poetry remove [モジュール名]
```