Python 3.13 + Poetry로 VSCode·Jupyter 개발환경 만들기

- Mac OS & Windows 두 가지 모두
- 제공된 pyproject.toml 활용
- 그 외 필요한 유틸 Graphviz/Ffmpeg 설치 및 확인

구성

- OS별 Python 3.13 설치/확인
- pipx + Poetry 설치(충돌 최소화)
- 제공된 pyproject.toml로 개발환경 구성
- VSCode Interpreter/터미널 자동 활성화
- Jupyter 커널 등록 → VSCode 갱신 (① Reload Window / ② 재실행) → 커널 (인터프리터) 선택
- Graphviz / FFmpeg / libsndfile 설치 및 인식 확인
- 트러블슈팅 핵심 정리

macOS: Python 3.13 설치/확인

- **VS Code 터미널에서 커맨드 실행 (순서대로 하나씩)**

```
brew install python@3.13

python3.13 --version

which python3.13
```

결과 예시 => /opt/homebrew/bin/python3.13

macOS: pipx + Poetry 설치

- **VS Code 터미널에서 커맨드 실행 (순서대로 하나씩)**

```
brew install pipx

pipx ensurepath

pipx install --python python3.13 poetry

poetry –version
```

macOS: 제공된 pyproject.tom로 환경 복원

metacode라는 폴더를 개발환경 폴더로 가정합니다.
만들어 둔 폴더 없다면 빈 폴더를 하나 만드세요.
해당 폴더내에 제공한 pyproject.toml 가 있어야 합니다.
예시) cd metacode # 해당 폴더로 이동

- **VS Code 터미널에서 커맨드 실행 (순서대로 하나씩)**

  poetry env use python3.13

  poetry lock --no-update

  poetry install

  poetry env info

Windows: Python 3.13 설치/확인

python.org에서 **Python 3.13.6(or 3.13.8)** 설치

- python.org 설치 시 Add python.exe to PATH 체크

- **VS Code 터미널에서 커맨드 실행 (순서대로 하나씩)**

```
python --version

where.exe python
```

Windows: pipx + Poetry 설치

- **VS Code 터미널에서 커맨드 실행 (순서대로 하나씩)**

```
python -m pip install --user pipx

pipx ensurepath

# VSCode 종료 및 재실행 후
pipx install poetry

poetry --version
```

Windows: 제공된 pyproject.tom로 환경 복원

metacode라는 폴더를 개발환경 폴더로 가정합니다.
만들어 둔 폴더 없다면 빈 폴더를 하나 만드세요.
해당 폴더내에 제공한 pyproject.toml 가 있어야 합니다.
예시) cd metacode # 해당 폴더로 이동

- **VS Code 터미널에서 커맨드 실행 (순서대로 하나씩)**

  poetry env use python

  poetry lock --no-update

  poetry install

  poetry env info

VSCode & 주피터 커널 연결 (공통)

1. VSCode 갱신
   ① Developer: Reload Window
   ② VSCode 완전 재실행(가장 확실)

2. VSCode → Python: Select Interpreter
   - mac : (.venv/bin/python)
   - Win : (.venv\Scripts\python.exe)

3. 터미널에서 파이썬 버전 확인
   - mac : `which python`
   - Win : `where.exe python`

3. 커널 등록(터미널에서 실행)

poetry run python -m ipykernel install --user --name metacode --display-name "Python (Poetry: metacode)"

4. VSCode 갱신(한번 더)
   ① Developer: Reload Window.
   ② VSCode 종료 및 재실행

5. VSCode 노트북에서 Kernel: Python (Poetry: metacode) 선택 → sys.executable로 .venv 경로 확인

import sys; print(sys.executable) # .venv 경로면 OK

macOS: Graphviz/ FFmpeg/ libsndfile) 설치 및 확인

- 설치
  brew install graphviz libsndfile ffmpeg

- 확인
  - graphviz 확인
    `which dot`
    `dot -V`
  - ffmpeg 확인
    which ffmpeg
    ffmpeg -hide_banner –version

- 만약에 안보이면
  - Apple CPU (M시리즈 cpu)인 경우, PATH 등록 (아래 한 줄씩 실행)
    echo 'eval "$(/opt/homebrew/bin/brew shellenv)"' >> ~/.zprofile
    echo 'eval "$(/opt/homebrew/bin/brew shellenv)"' >> ~/.zshrc
    eval "$(/opt/homebrew/bin/brew shellenv)"

  - Intel CPU인 경우, PATH 등록 (아래 한 줄씩 실행)
    echo 'eval "$(/opt/local/bin/brew shellenv)"' >> ~/.zprofile
    echo 'eval "$(/opt/local/bin/brew shellenv)"' >> ~/.zshrc
    eval "$(/opt/local/bin/brew shellenv)"

  - VSCode 갱신(⌘⇧P → Developer: Reload Window)
  - VSCode 종료 및 재실행

- 재확인
  - graphviz 확인
    `which dot`
    `dot -V`
  - ffmpeg 확인
    which ffmpeg
    ffmpeg -hide_banner –version

Windows: Graphviz/ FFmpeg/ libsndfile) 설치 및 확인

- 설치
  winget install -e Graphviz.Graphviz
  winget install -e Gyan.Ffmpeg

- 확인
  - graphviz 확인
    where.exe dot
    dot -V
  - ffmpeg 확인
    where.exe ffmpeg
    ffmpeg -hide_banner –version

  - 만약에 안보이면
    [Environment]::SetEnvironmentVariable("Path", $env:Path + ";C:\Program Files\ffmpeg\bin", "User")

  - VSCode 갱신: ⌘⇧P → Developer: Reload Window
  - VSCode 종료 및 재실행

- 재확인
  - graphviz 확인
    where.exe dot
    dot -V
  - ffmpeg 확인
    where.exe ffmpeg
    ffmpeg -hide_banner –version

Poetry 명령어 사용법

- **라이브러리 추가**
  - poetry add <pkg>
  예) poetry add requests
- **설치한 정확한 버전과 하위 의존성을 기록**
  - poetry lock
  * 보통 poetry install 하기전에 한번씩 실행
- **현재 가상환경 정보**
  - poetry env info
- **생성된 가상환경 목록**
  - poetry env list
- **새 프로젝트 생성**
  - poetry new <name>
  예) poetry new myproj
- **특정 파이썬으로 가상환경 생성/전환**
  (Win: python 경로 지정 가능)
  - poetry env use <파이썬버전>
  예) poetry env use python3.13