1. **파이썬(Python) 설치**
2. **Git 및 Git Bash 설치**
3. **VS Code 설치 및 기본 설정**
4. **VS Code 기본 터미널을 Git Bash로 변경**
5. **기본 Bash 명령어 학습**
6. **파이썬 패키지 개념과 설치/제거**
7. **다양한 파이썬 실행 방법**

윈도우용 파이썬 개발 환경 구축 가이드

본 가이드는 윈도우(Windows) PC에서 **파이썬 개발 환경**을 구축하는 방법을 단계별로 설명합니다. 각 섹션을 차례대로 따라 하면 개발에 필요한 환경을 모두 준비할 수 있습니다.

---

1. 파이썬(Python) 설치

1.1 파이썬 설치 파일 다운로드

1. **공식 웹사이트 방문:** 웹 브라우저로 파이썬 공식 사이트(https://www.python.org)에 접속합니다.
2. **다운로드 페이지 이동:** 상단 메뉴의 *Downloads* 페이지 → *Windows*를 선택합니다. **최신 안정 버전 (Python 3.x.x)**을 찾아 다운로드합니다.
3. **설치 프로그램 실행:** 다운로드한 설치 파일 (예: `python-3.xx.x-amd64.exe`)을 더블 클릭해 실행합니다.

1.2 파이썬 설치 과정과 환경 변수 설정

- **설치 옵션:** “Install Now” 버튼을 클릭해 기본 경로로 설치 진행.
  - *중요*: 하단의 “Add Python 3.x to PATH” 옵션을 **반드시 체크**.
- **설치 완료 후 확인:**

```
python --version
pip --version
```

버전 정보가 출력되면 정상 설치. 만약 `python`을 찾지 못하면, 환경 변수를 수동 설정해야 합니다.

1.3 설치 확인 및 Python 쉘 테스트

- **파이썬 인터프리터 진입:**

```
python
>>> print("Hello Python!")
>>> exit()
```

---

2. Git 및 Git Bash 설치

2.1 Git Bash 설치 파일 다운로드 및 설치

1. **Git 공식 사이트:** https://git-scm.com 방문 → “Download for Windows”.
2. **설치 프로그램 실행:** “Git-2.x.x-64-bit.exe” 등 실행.
3. **기본 옵션**(권장)으로 진행. “Adjusting your PATH environment” 스텝에서 Git Bash 전용으로 Git 사용을 선택하거나, 필요 시 모든 터미널에서 Git 사용하도록 설정.

2.2 Git Bash 실행 및 확인

- **Git 버전 확인:**

```
git --version
```

- **Git Bash 쉘 특징:** Bash 명령어(`ls`, `pwd` 등) 사용 가능. Windows 경로는 `/c/Users/...` 형태로 표기.

---

3. VS Code 설치 및 기본 설정

3.1 VS Code 설치

1. **공식 웹사이트:** https://code.visualstudio.com → “Download for Windows”.
2. **설치 프로그램**(`VSCodeUserSetup-x64-*.exe`) 실행.
3. **옵션:** “Add to PATH” 체크 시 VS Code를 `code` 명령으로 실행 가능.

3.2 Python 개발을 위한 필수 확장

- **Python (by Microsoft)** : 필수.
- **Jupyter (by Microsoft)** : .ipynb 노트북 사용 시.

3.3 VS Code에서 Python 인터프리터 설정

- **Command Palette**(Ctrl+Shift+P) → “Python: Select Interpreter” → 설치한 Python 3.x 선택.

---

4. VS Code 기본 터미널을 Git Bash로 변경

1. **VS Code 메뉴 → Terminal → Select Default Profile → Git Bash** 선택.
2. 새 터미널 열면 Bash 쉘이 실행됨. (파워셸 종료하고 필요 시 휴지통 아이콘으로 제거)

---

5. 기본 Bash 명령어 학습

- `pwd` : 현재 경로 출력
- `ls` : 현재 디렉토리 목록 표시
- `cd` : 디렉토리 이동
- `mkdir project` : 새 폴더 생성
- `rm file.txt` : 파일 삭제
- `touch test.py` : 빈 파일 생성 (또는 수정시간 갱신)
- `echo "print('Hello')" > hello.py` : 파일에 내용 쓰기

- `cat hello.py` : 파일 내용 출력

---

6. 파이썬 패키지 개념과 설치/제거

파이썬에서 **패키지(package)**는 특정 기능을 담고 있는 모듈 모음입니다. 예컨대 수학연산에 특화된 `numpy`, 웹 프레임워크 `django`, 데이터 분석 라이브러리 `pandas` 등이 대표적입니다. 패키지는 **`pip`** 명령어를 사용하여 쉽게 설치하거나 삭제할 수 있으며, PyPI(Python Package Index)라는 저장소(https://pypi.org)에서 수많은 오픈소스 패키지를 검색하고 내려받아 쓸 수 있습니다.

6.1 PyPI와 pip 개념

- **PyPI (파이파이):** 전 세계 파이썬 개발자들이 만들어 공유하는 수많은 라이브러리(패키지)가 모여 있는 저장소.
- **pip:** PyPI에서 원하는 패키지를 다운로드/설치하는 **패키지 관리자**.

6.2 pip를 이용한 패키지 설치/삭제

패키지 설치

```
# 일반 설치
pip install 패키지이름

# 특정 버전 설치
pip install 패키지이름==버전번호

# 예: numpy 최신 버전 설치
pip install numpy
```

- `pip install numpy==1.19.3` 처럼 특정 버전을 명시해 설치할 수 있음.
- 가상환경이 아닌 전역에서 설치 시, 다른 프로젝트와의 버전 충돌이 생길 수 있으니 주의. (초급 과정에서는 일단 전역 설치 위주로 진행)

패키지 제거

```
pip uninstall 패키지이름
```

- 예: `pip uninstall numpy`
- 제거 시, “Proceed (y/n)?” 질문에 **y**를 입력하면 삭제 완료.

패키지 업그레이드

```
pip install --upgrade 패키지이름
```

- 이미 설치된 패키지를 최신 버전으로 업데이트.

6.3 설치된 패키지 목록 확인

```
pip list
```

- 현재 파이썬 환경에 설치된 패키지와 버전을 보여줌.
- `pip freeze` 명령도 사용 가능하며, 이는 `requirements.txt` 파일을 생성할 때 자주 쓰임.

6.4 requirements.txt 활용 (옵션)

프로젝트에서 의존하는 패키지 목록과 버전을 `requirements.txt` 파일에 기록해두면, 협업 시 다른 사람도 동일한 패키지 버전으로 설치할 수 있습니다.

- **requirements.txt 작성:**

  ```
  numpy==1.21.0
  pandas==1.3.0
  matplotlib==3.4.2
  ```

- **설치:**

  ```
  pip install -r requirements.txt
  ```

> **참고:** 복잡한 프로젝트일수록 **가상환경(venv)**을 만들어 필요한 패키지를 설치하는 것이 좋습니다. 그러나 초급 단계에서는 전역(Environment) 설치로 간단히 시작해볼 수 있습니다. 가상환경 사용법은 추후 학습하면 됩니다.

---

7. 다양한 파이썬 실행 방법

7.1 Git Bash 터미널에서 파이썬 CLI 실행

- **대화형 모드:**

  ```
  python
  >>> 2 + 3
  >>> exit()
  ```

- **스크립트 실행:**

  ```
  python hello.py
  ```

7.2 VS Code에서 `.py` 파일 실행

- **터미널에서 수동 실행:**
  VS Code 하단 터미널(=Git Bash)에서 `python hello.py`.
- **Run Python File in Terminal 버튼:**
  VS Code 우상단 혹은 좌측 Run 패널에서 ▶ 버튼(또는 Ctrl+F5).

7.3 VS Code에서 `.ipynb` (Jupyter Notebook) 인터랙티브 실행

- **Notebook 파일 열기:**
  VS Code에서 `*.ipynb` 파일을 열면 Jupyter 인터페이스 적용.
- **셀 실행:**
  각 셀 왼쪽의 ▶ 버튼, 또는 Shift+Enter.
- **커널 선택:**
  우상단 Kernel Picker에서 Python 3.x 인터프리터 선택.
- **결과 확인:**
  셀 바로 아래에 출력 표시. 그래프 등 시각화 가능.
