

Introduction

PyQT를 활용한 GUI 프로그래밍

커리큘럼

- 본 과정에서는 GUI 프로그래밍을 배워서 인하우스 툴 제작 등의 나만의 프로그램을 설계합니다.

주제	내용	시수
PYTHON 기초	<ul style="list-style-type: none">- 파이썬과 메모리- 파이썬의 클래스 기초- 파이썬 메서드- 파이썬 상속	8H
윈도우와 위젯	<ul style="list-style-type: none">- Main 윈도우 꾸미기- 버튼 위젯, 라벨 위젯 사용해 보기- 시그널과 슬롯- 이벤트 제약 사항	2H
쓰레드	<ul style="list-style-type: none">- 쓰레드 사용 이유- 부모 쓰레드와 자식 쓰레드	2H

주제	내용	시수
PyQT의 위젯	<ul style="list-style-type: none">- 다양한 위젯 사용하기- 위젯의 이벤트 처리- Custom 위젯	4H
위젯 레이아웃	<ul style="list-style-type: none">- 절대적 배치 (Absolute positioning)- 박스 레이아웃 (Box layout)- 그리드 레이아웃 (Grid layout)	4H
Mini project	<p>[나만의 업무자동화 GUI 프로그래밍 만들기]</p> <ul style="list-style-type: none">- 기본 주제 제공- 현업에서 필요한 툴 만들기로 대체 가능	4H

미니콘다

- Miniconda
 - 라이브러리를 포함한 파이썬 설치 파일
 - BSD-3를 기반의 라이선스로 무료 사용 가능
 - <https://docs.conda.io/en/latest/miniconda.html>

Miniconda

Miniconda is a free minimal installer for conda. It is a small, bootstrap version of Anaconda that includes only conda, Python, the packages they depend on, and a small number of other useful packages, including pip, zlib and a few others. Use the `conda install` command to install 720+ additional conda packages from the Anaconda repository.

[See if Miniconda is right for you.](#)

개발 환경 (IDE)

- Python In VS Code
 - 설치링크: <https://code.visualstudio.com/>
 - Python Extension 추가 설치 필요

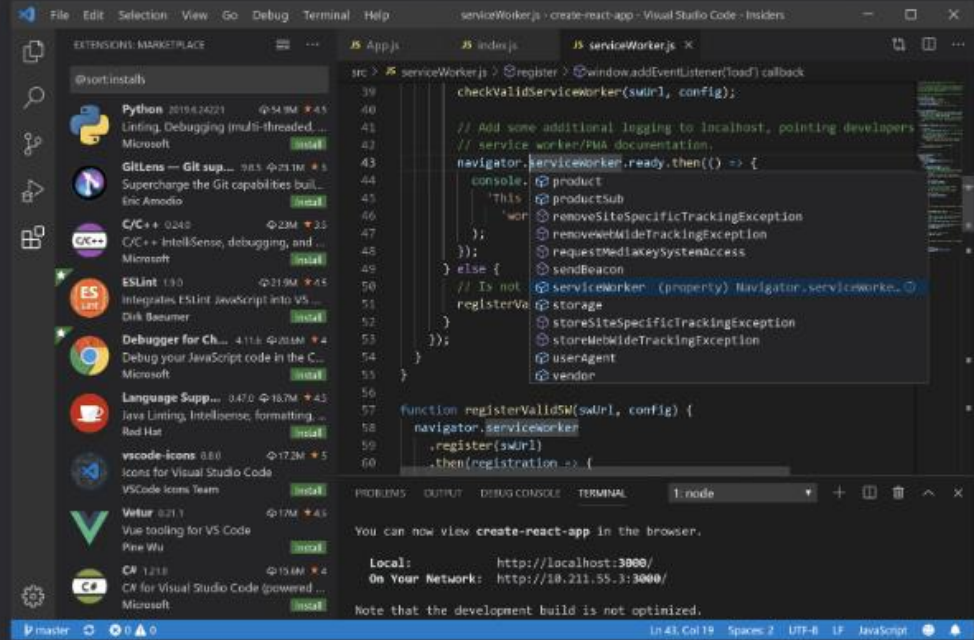
Code editing.
Redefined.

Free. Built on open source. Runs everywhere.

Download for Windows
Stable Build

Web, Insiders edition, or other platforms

By using VS Code, you agree to its
[license and privacy statement.](#)



The screenshot displays the Visual Studio Code IDE interface. On the left, the 'EXTENSIONS: MARKETPLACE' sidebar is open, showing a list of installed and available extensions. The 'Python' extension by Microsoft is highlighted. The main editor area shows a JavaScript file named 'serviceWorker.js' with code for a service worker. The bottom panel contains a 'TERMINAL' window showing the output of a command, indicating that the application is running on localhost:3000.

VScode - Extansion

The diagram illustrates the steps to install the Python extension in Visual Studio Code:

- VS Code Interface:** The left sidebar shows the Extensions icon (four squares) highlighted with a red dashed box.
- Extensions: Market Place:** A search for "python" yields results. The top result is the **Python** extension by **Microsoft**, which includes IntelliSense (Pylance), Linting, and Debugging.
- Python Extension Details:** The detailed view of the Python extension is shown. It includes the Python logo, version **v2022.20.2**, and a rating of 5 stars (524 reviews). The extension is described as providing IntelliSense (Pylance), Linting, Debugging (multi-threaded, remote), Jupyter Notebooks, code formatting, refactoring, unit tests, and more. It is currently **enabled globally**.

Python extension for Visual Studio Code

A Visual Studio Code extension with rich support for the Python language (for all actively supported versions of the language: ≥ 3.7), including features such as IntelliSense (Pylance), linting, debugging, code navigation, code formatting, refactoring, variable explorer, test explorer, and more!

Support for vscode.dev

The Python extension does offer some support when running on vscode.dev (which includes github.dev). This includes partial IntelliSense for open files in the editor.

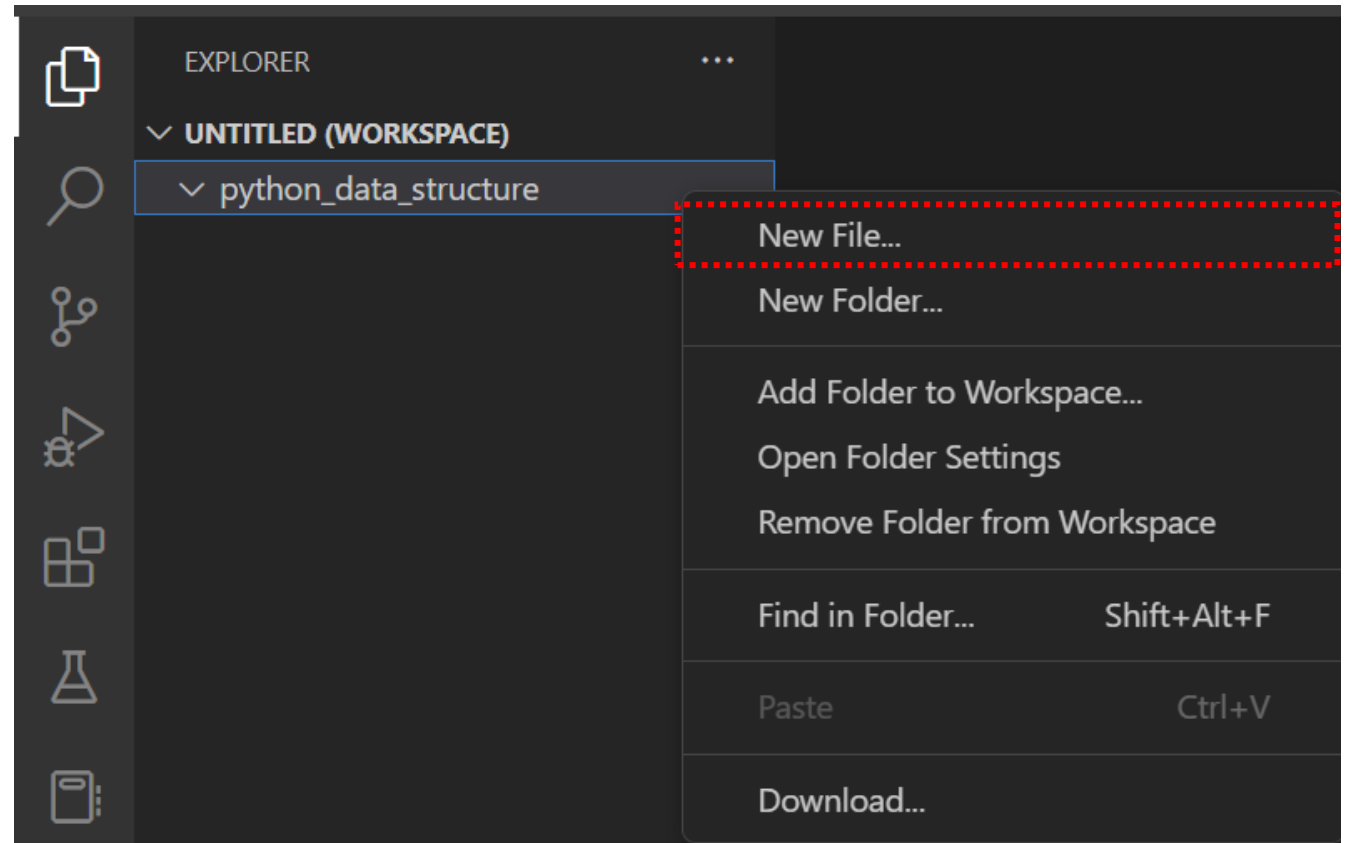
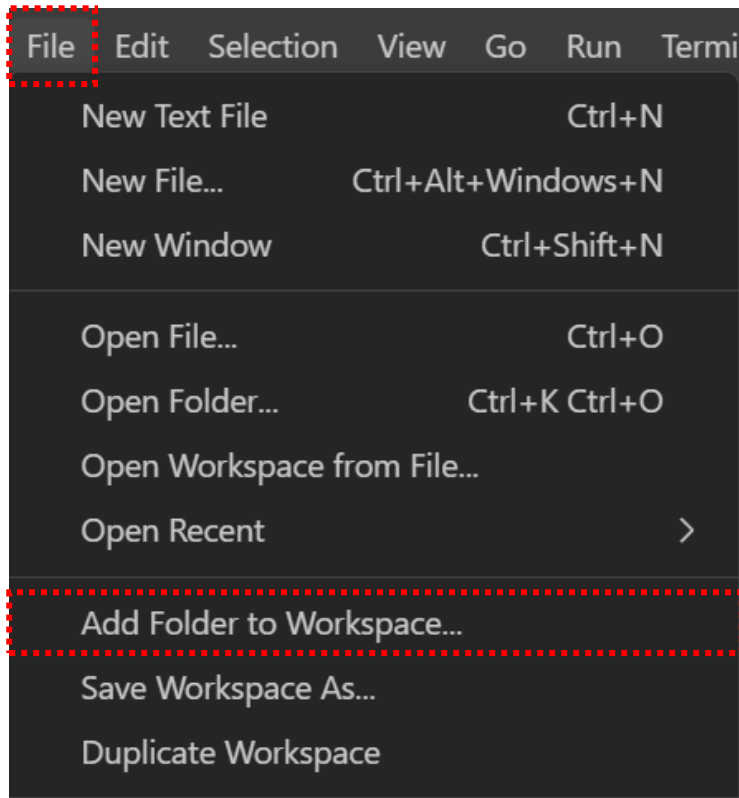
Installed extensions

Categories: Programming Languages, Debuggers, Linters, Formatters, Other, Data Science, Machine Learning, Notebooks.

Extension Resources: Marketplace, Repository, License, Microsoft.

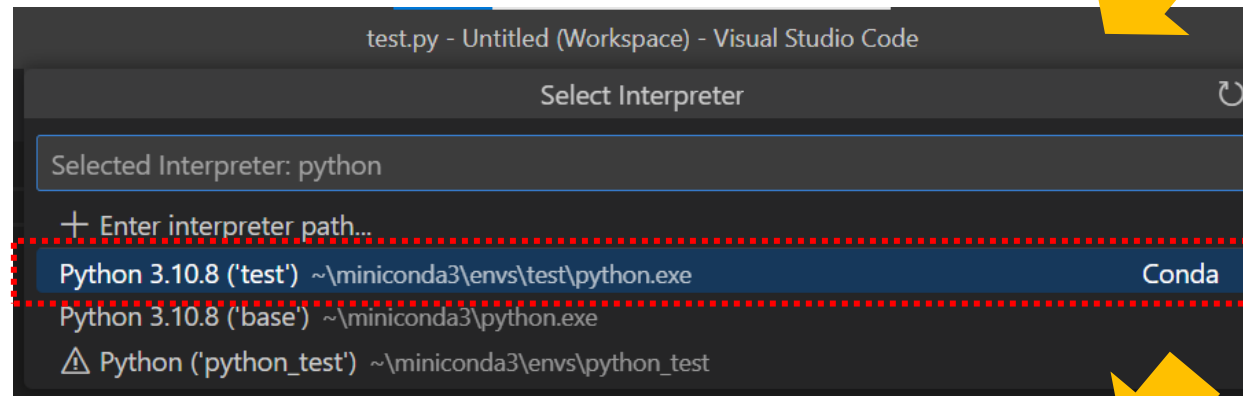
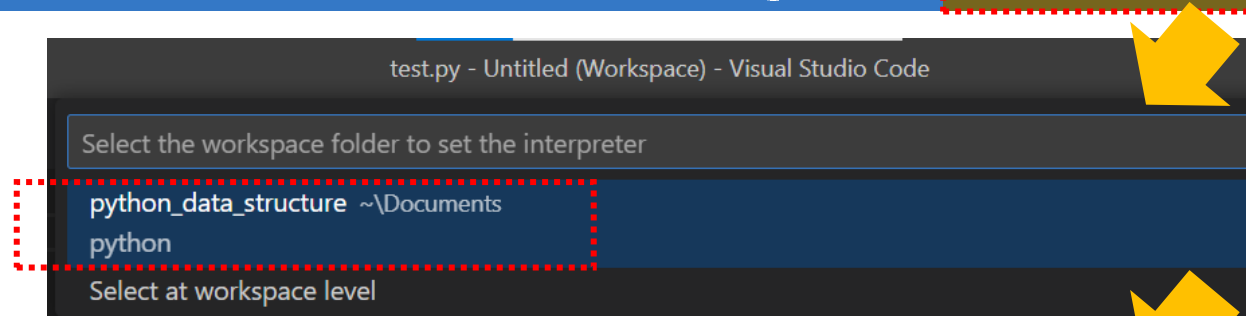
VScode – Workspace

- Workspace - 우리가 사용할 코드들이 모여있는 공간

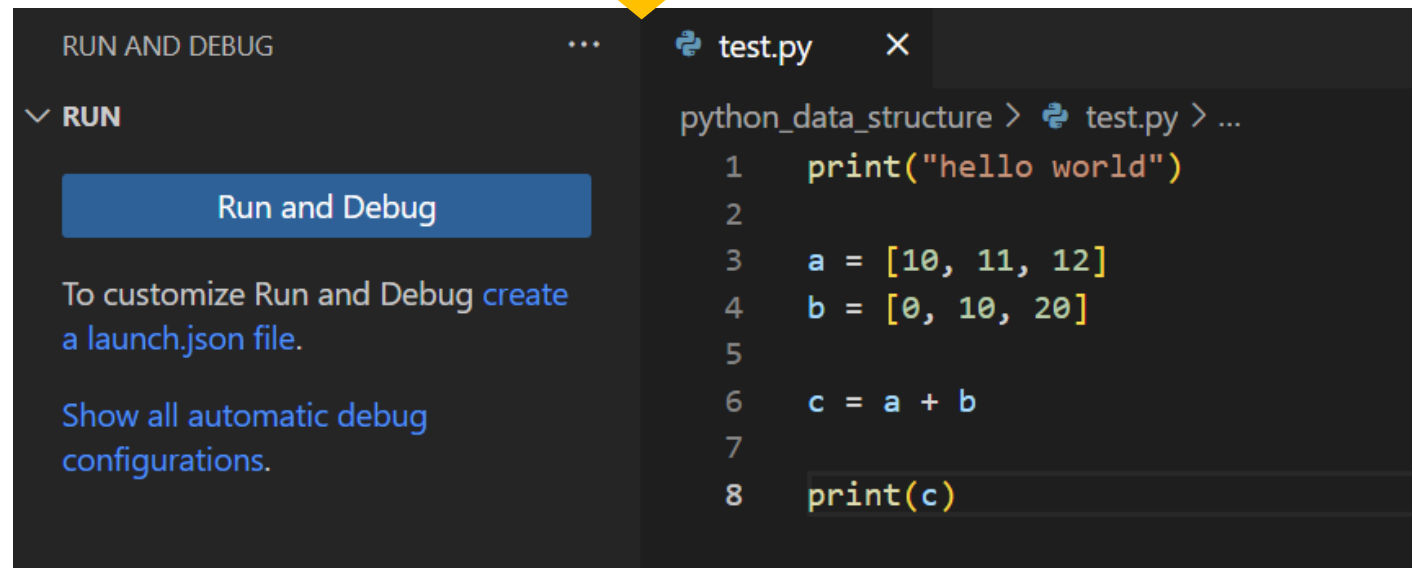
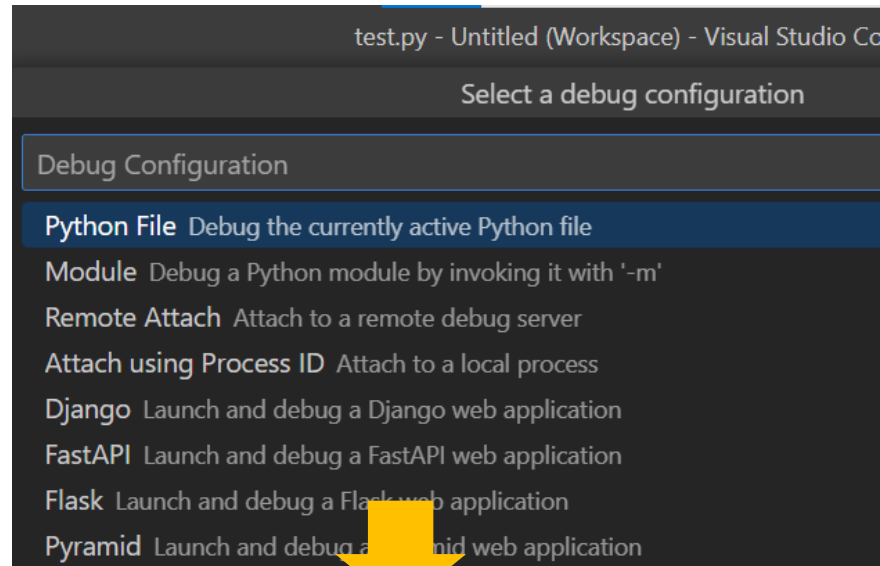


VScode - Interpreter

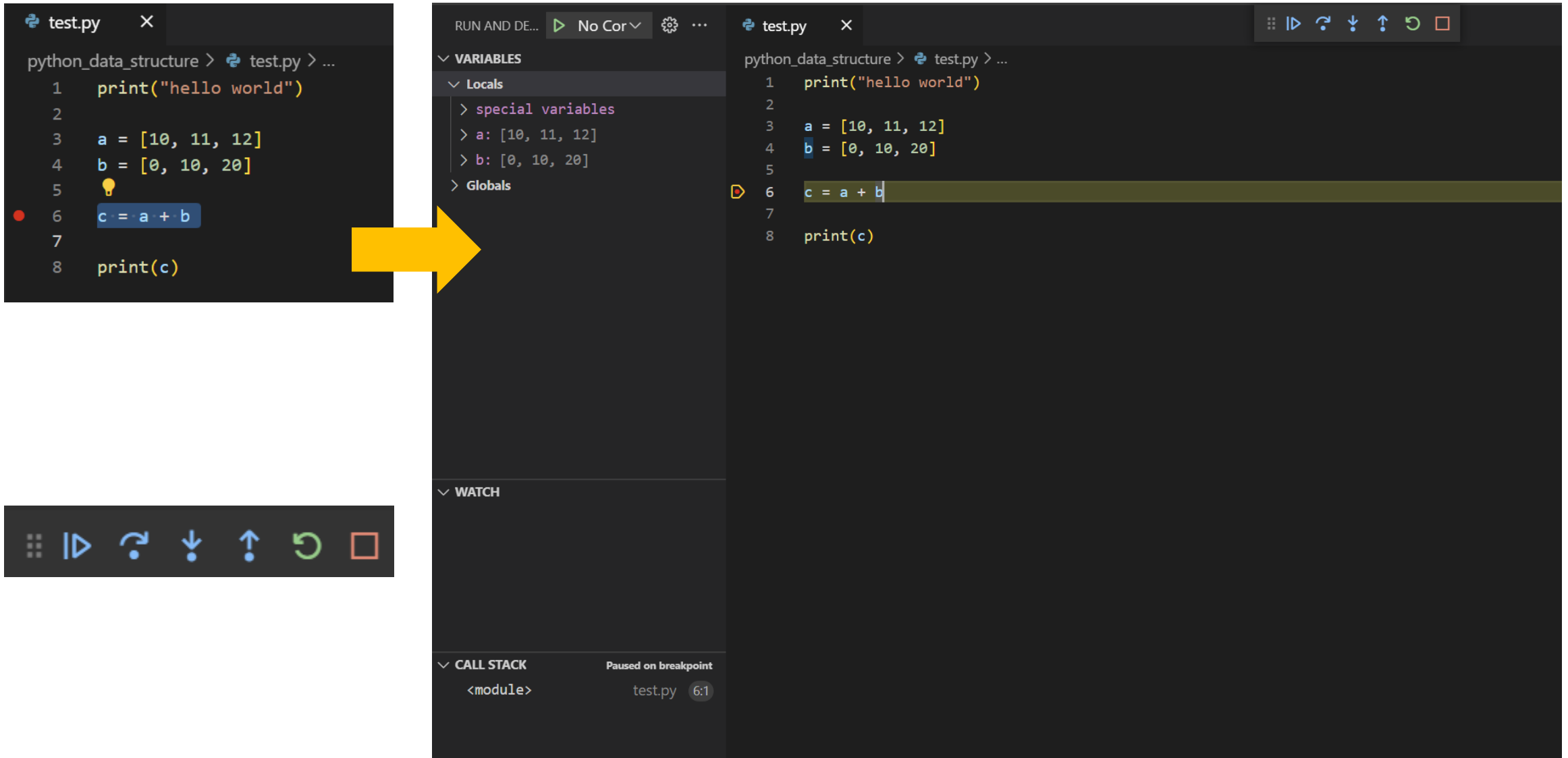
- Vscode 에서 python을 사용하기 위해 python interpreter 를 setting



VScode – Debug mode



VScode - breakpoint



The image illustrates how to set a breakpoint in VS Code and view the state of variables at that point.

Left Panel (Editor): Shows the file `test.py` with the following code:

```
1 print("hello world")
2
3 a = [10, 11, 12]
4 b = [0, 10, 20]
5
6 c = a + b
7
8 print(c)
```

A red dot on line 6 indicates a breakpoint. A yellow arrow points from this breakpoint to the Variables panel.

Right Panel (Debugger): Shows the same file `test.py` with the breakpoint on line 6. The **VARIABLES** panel is expanded, showing the state of variables at the breakpoint:

- Locals:**
 - `special variables`
 - `a`: `[10, 11, 12]`
 - `b`: `[0, 10, 20]`
- Globals:**

The **WATCH** panel is empty. The **CALL STACK** panel shows the current call stack:

- `<module>` (test.py 6:1)

The status bar at the bottom indicates "Paused on breakpoint".