Controlling APx500 Using Python

by Ben Kilada

When writing a script to control the APx500 software, the APx500 API must be referenced so that API commands can successfully compile at run-time. Because the API libraries are packaged into .NET assemblies (dll files) that can only be executed in a .NET runtime environment, the standard Python implementation cannot be used without modification. Support for .NET libraries must be added using a custom implementation (IronPython) or by installing a Python.NET module.

While both IronPython and Python.NET are workable solutions, Python.NET option may be preferred in situations where using standard Python or a newer version of Python is required. Python.NET supports the latest versions of Python (3.12 as of March 1st, 2024). These instructions cover the Python.NET solution.

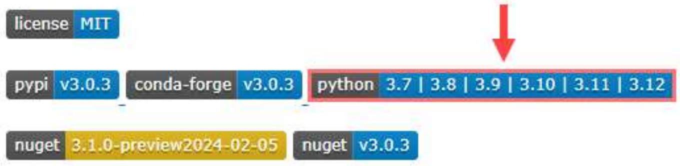
Note that although Python is a cross-platform language, API control of APx500 is only available on operating systems supported by the APx500 application. A list of supported operating systems can be found on our website at the link below.

[AP Software Windows Version Compatibility]

To get started controlling APx500 using Python and Python.NET, follow the steps below.

Step 1: Install Python

Before installing Python, it is important to check which versions are currently supported by Python.NET using the [Python.NET GitHub repository.] Scroll down and look for a section which indicates the list of supported Python versions. As of March 1st, 2024, the supported versions are: 3.7, 3.8, 3.9, 3.10, 3.11, and 3.12.

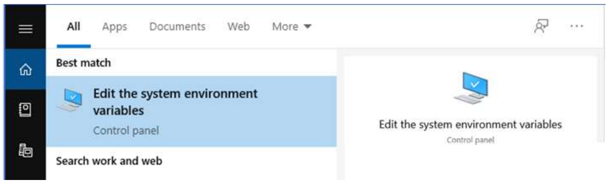

license MIT
pypi v3.0.3
conda-forge v3.0.3
python 3.7 | 3.8 | 3.9 | 3.10 | 3.11 | 3.12
nuget 3.1.0-preview2024-02-05
nuget v3.0.3

Once compatibility has been confirmed, download and install the desired version from [Python.org.]

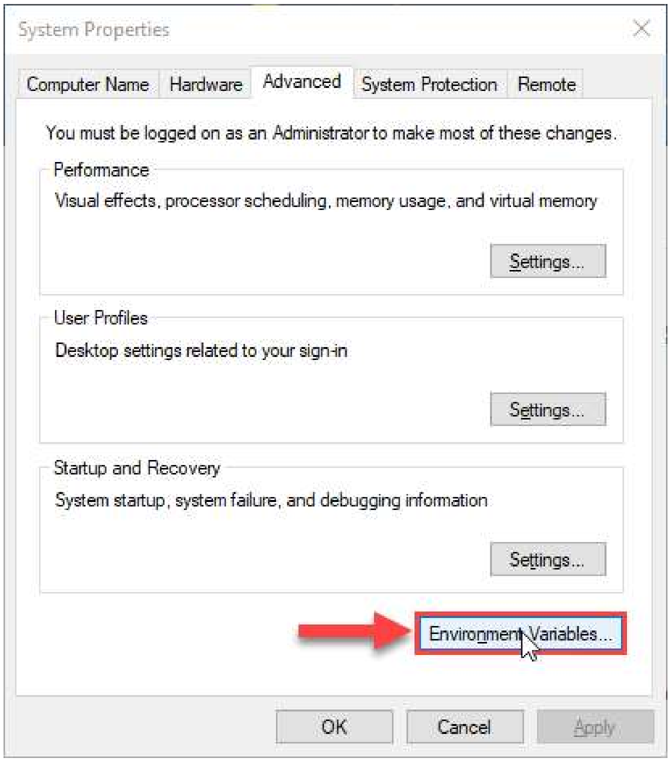
Step 2: Add Python to the Windows Path

Once Python has been installed, the Windows Path should be modified to allow Python commands to be executed from the Windows command prompt. To do so (in Windows 10):

1. Use the Windows taskbar to search “Edit the system environment variables”. Press Enter to open the System Properties dialog.



2. Click the “Environment Variables...” button.



You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

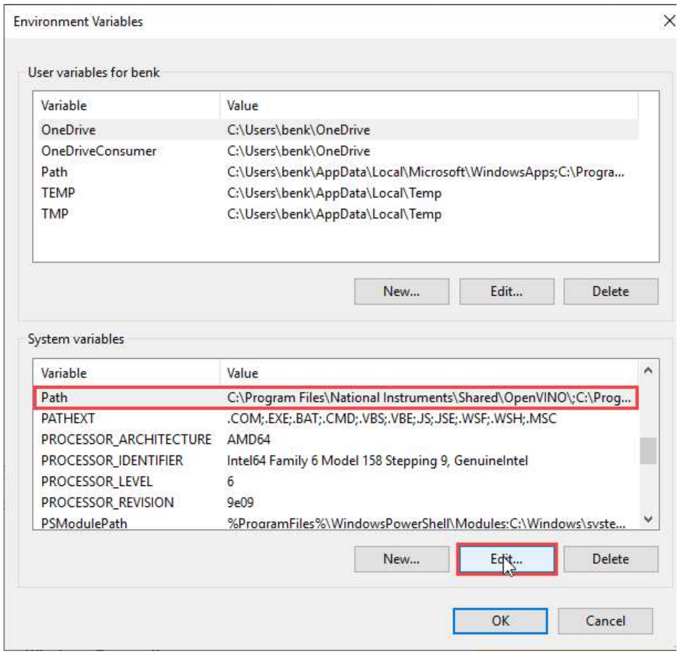
User Profiles

Desktop settings related to your sign-in

Startup and Recovery

System startup, system failure, and debugging information

3. Under “System Variables”, find “Path” and click Edit...”.



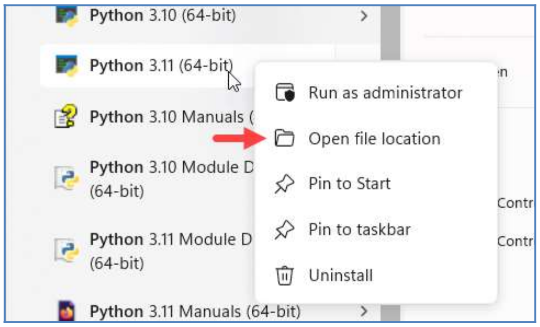
User variables for benk

Variable	Value
OneDrive	C:\Users\benk\OneDrive
OneDriveConsumer	C:\Users\benk\OneDrive
Path	C:\Users\benk\AppData\Local\Microsoft\WindowsApps;C:\Progra...
TEMP	C:\Users\benk\AppData\Local\Temp
TMP	C:\Users\benk\AppData\Local\Temp

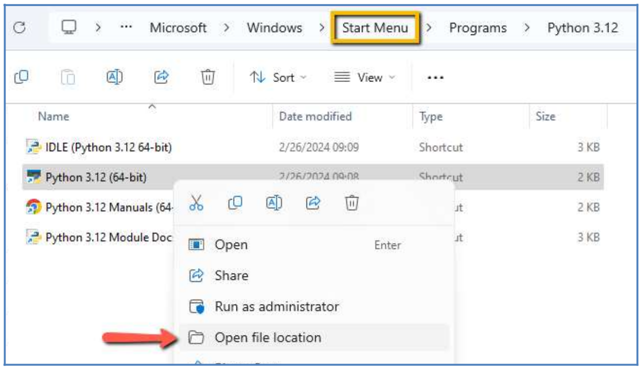
System variables

Variable	Value
Path	C:\Program Files\National Instruments\Shared\OpenVINO\;C:\Prog...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 Model 158 Stepping 9, GenuineIntel
PROCESSOR_LEVEL	6
PROCESSOR_REVISION	9e09
PSModulePath	%ProgramFiles%\WindowsPowerShell\Modules;C:\Windows\syste...

4. To find the Python installation directory that will be added to the Path: using the Windows taskbar, search for “Python”. Right-click the version of Python you intend to use and select “Open file location”.

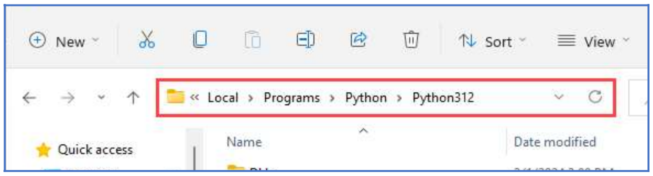


5. If, in the Explorer window that appears, the file path points to the Start Menu directory or to another location other than the python install directory, right-click the shortcut to Python and select “Open file location”. Repeat as necessary until the explorer window displays the python installation directory.



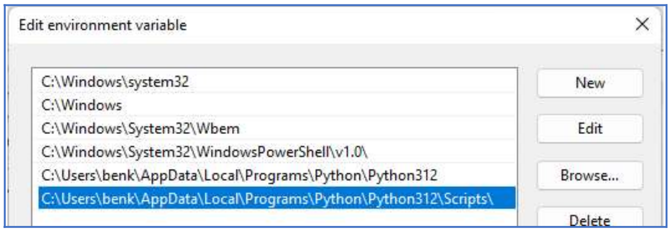
Name	Date modified	Type	Size
IDLE (Python 3.12 64-bit)	2/26/2024 09:09	Shortcut	3 KB
Python 3.12 (64-bit)	2/26/2024 09:08	Shortcut	2 KB
Python 3.12 Manuals (64-			2 KB
Python 3.12 Module Doc			3 KB

6. Select and copy the full file path from the Explorer address bar.



« Local › Programs › Python › Python312

7. Go back to the “Edit environment variable” dialog and click “New”.



C:\Windows\system32
C:\Windows
C:\Windows\System32\Wbem
C:\Windows\System32\WindowsPowerShell\v1.0\
C:\Users\benk\AppData\Local\Programs\Python\Python312
C:\Users\benk\AppData\Local\Programs\Python\Python312\Scripts\

8. Paste the copied Python path into the new path entry.

9. Click “New” again, paste the copied path and append “\Scripts\” to the end.

10. Click OK to confirm changes in both the “Edit environment variable” and “Environment Variables” windows.

To confirm that Python was successfully added to the path, open a new Windows Command Prompt and type “python --version” (two dashes). The Command Prompt should print out the version of Python that was just added to the Windows path (e.g., Python 3.12.2).

Step 3: Install Python.NET using a package manager

Once python has been installed and added to the Windows path, the Python.NET package can be installed using pip, a package manager for Python. Pip is automatically installed with newer versions of Python.

To install Python.NET using pip:

1. Open a Windows Command Prompt.
2. Type “`pip install pythonnet`”.

And that’s it! You are now ready to write code to control APx500 using Python and Python.NET.

Example Script

Included with the guide is an example script (and project file) that shows how to control APx500 using a simple graphical user interface (tested with version 3.12 of Python and 3.0.3 of Python.NET). The example demonstrates how to reference the Audio Precision API, instantiate the Apx500 software, load a project file, run a sequence, and get data acquired in the sequence (for demonstration purposes, the values are simply retrieved and stored in variables). It also shows how to configure signal path settings and add and configure a measurement and derived result. Each function has been separated into a separate block of code to make it easier to read.

If a command line interface is preferred, delete the System.Drawing and System.Windows.Forms “From” statements, as well as all the code after and including “class Container(Form)”, then write your API calls directly below the From statements.

For more examples, please send your request to the Global Support team at techsupport@ap.com.