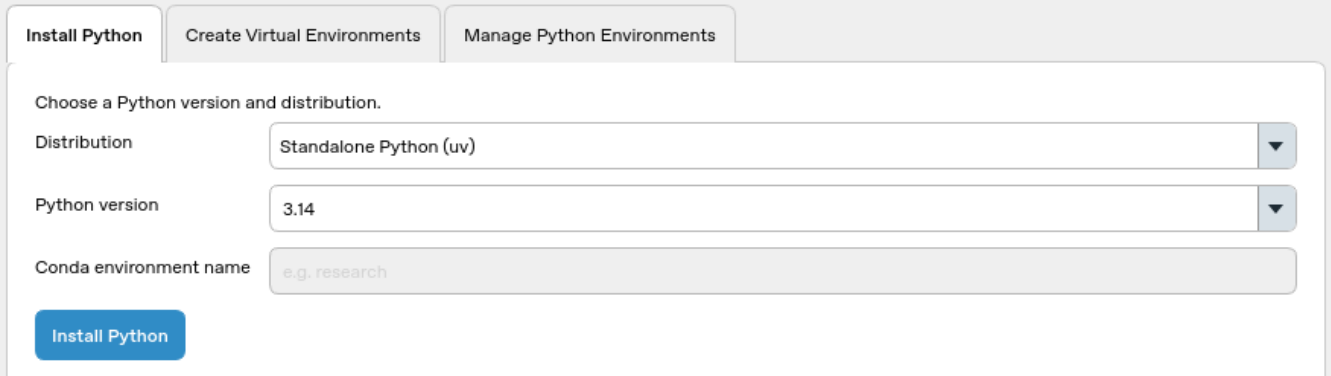
Install Python

PyYK is a portable Python environment manager for 64-bit Windows. Double-click **PyYK.exe** to open it. The app itself needs no installer or preinstalled Python.



Install Python tab. Standalone Python is selected in this example.

Install a standalone Python interpreter

01 In **Install Python**, choose **Standalone Python (uv)** under **Distribution**.

02 Choose or type a **Python version**. A minor version such as **3.13** requests an available 3.13 patch release. An exact version such as **3.13.2** requests that release.

03 Click **Install Python**. Watch the **Activity log**. After PyYK locates the installed interpreter, it selects it in **Create Virtual Environments**.

Create a conda-forge environment

01 Install Miniforge if needed. Start PyYK with Conda on **PATH** or **CONDA_EXE** set.

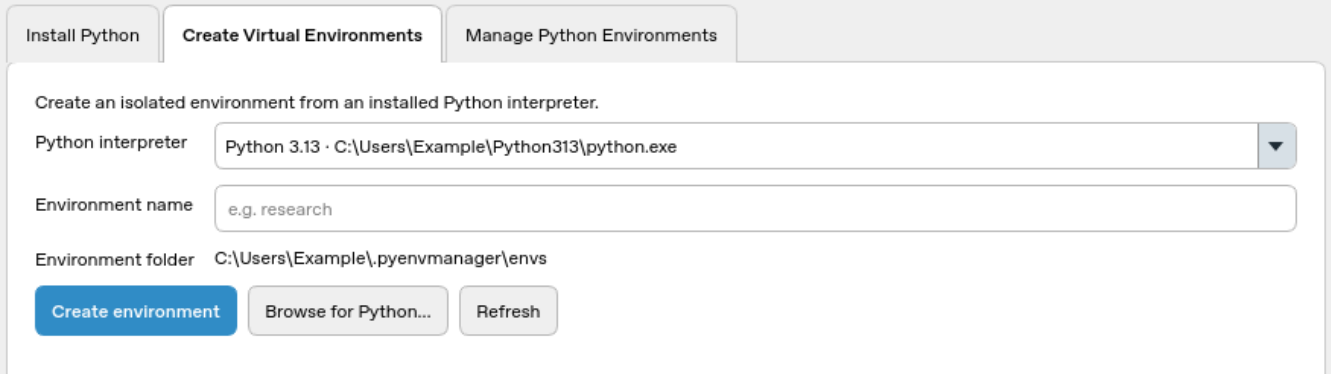
02 Set **Distribution** to **Conda-forge Python**, choose the Python version, and fill in **Conda environment name**. This field is enabled only for Conda.

03 Click **Install Python**. This creates the named Conda environment with Python and pip in the configured folder. PyYK selects it in **Manage Python Environments**.

Older versions You can type a version even if it is absent from the presets. The bundled uv offers CPython 3.8.20 for Windows x64 and has no CPython 3.7 download. You can browse to a separately installed 3.7 interpreter. Both are end-of-life.

Create an environment

A virtual environment gives each project its own interpreter entry point and installed packages.



Create Virtual Environments tab. The paths shown are examples.

Create a Python virtual environment

01 Open **Create Virtual Environments** and select a **Python interpreter**. Match it to your project's required version.

02 Enter an **Environment name** such as **research**. Start with a letter or digit, then use letters, digits, hyphens, or underscores. Use a distinct name and avoid Windows reserved names such as **CON**.

03 Check the **Environment folder** shown below the name. Click **Create environment**. The environment is created in a subfolder named after the environment.

04 In **Manage Python Environments**, confirm that the new environment is selected.

Interpreter and folder controls

Python interpreter — Selects the installed Python executable used to create the virtual environment.

Browse for Python… — Opens a file picker. Select the real **python.exe** from an installed Python distribution. PyYK registers that path and selects it.

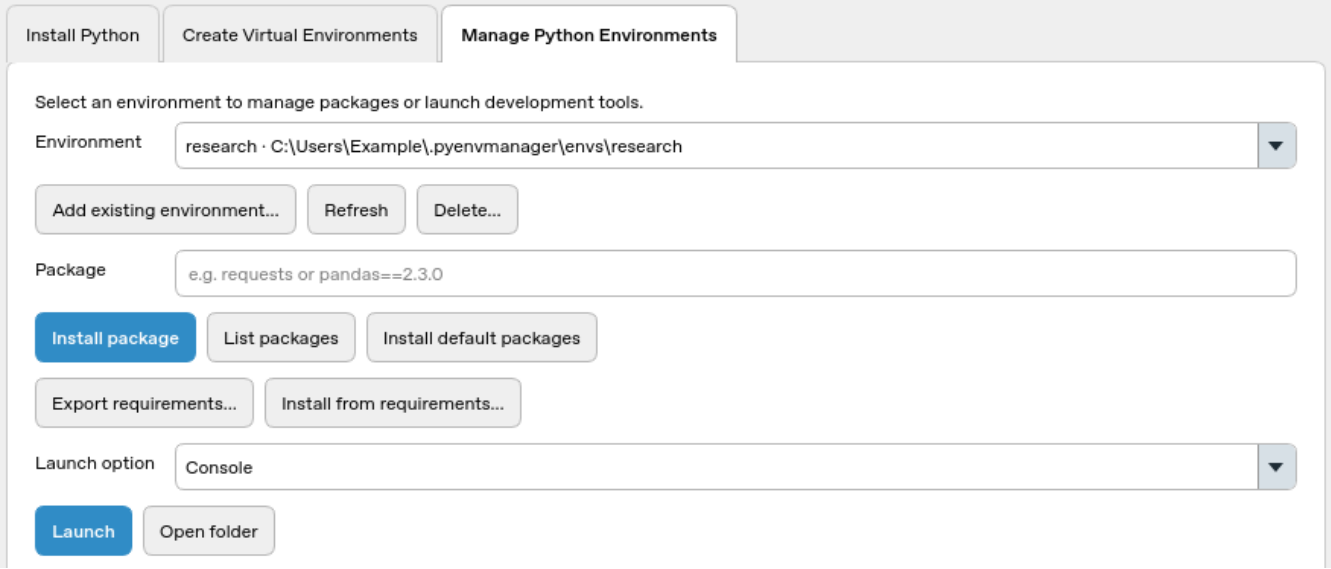
Refresh — Rescans interpreters and the environment list. Use it after an external installation or folder change.

Environment folder — Displays the configured parent folder. The Windows default is **%USERPROFILE%\.pyenvmanager\envs**. Change it through **File → Change environment folder…**.

Retrying If creation failed and left an empty folder, you can retry with the same name. If the folder has contents, inspect it or choose another name.

Manage packages

Check the environment name and path before making changes. Package operations apply to the selected environment.



Manage Python Environments tab. The paths shown are examples.

Install and verify a package

01 In **Manage Python Environments**, choose the destination in the **Environment** dropdown.

02 Enter one package specification in **Package**, such as **requests** or **pandas==2.3.0**. Click **Install package**.

03 When the command finishes, click **List packages**. The **Activity log** shows the packages and their installed versions.

Install default packages — Installs NumPy, SciPy, pandas, Matplotlib, PyVista, and scikit-learn into the selected environment.

Export and reuse a package list

01 Select the source environment. Click **Export requirements...**, choose a save location, and save the file. PyYK writes the output of **pip freeze** and reports the saved path.

02 Create or select the destination environment, ideally with a compatible Python version. Click **Install from requirements...**, select the file, and check the activity log.

03 Click **List packages** to verify the result. The requirements file records Python packages. Local package paths, private sources, or platform-specific packages may need adjustments on another computer.

Manage environments and launch tools

Use the Environment dropdown to select a virtual environment or Conda environment before choosing an action.

Register, refresh, open, or delete an environment

Add existing environment… — Choose the environment's root folder. For a Python virtual environment, select the folder containing **pyvenv.cfg**. For Conda, select the folder containing **conda-meta**. PyYK registers the folder in place.

Refresh — Rescans the configured folder, registered environments, and installed Python interpreters.

Open folder — Opens the selected environment's root in File Explorer.

Delete… — Asks before permanently deleting the selected environment. Deletion is available for eligible environments directly inside the configured folder. Conda must be available to remove a Conda environment.

Open a development tool

01 Select the target **Environment**.

02 Choose a tool from **Launch option**, then click **Launch**. Follow any installation prompt.

Console — Opens Windows Command Prompt in your home folder. The banner shows the selected path and the prompt shows the active environment name. Run **python --version** or **python -m pip list** to check it.

VS Code — Opens the selected environment folder in a new VS Code window. If VS Code is missing, PyYK offers a per-user WinGet installation and then opens it. If WinGet is unavailable, PyYK offers the official download page. In VS Code, use **Python: Select Interpreter** to select the intended interpreter if needed. The command is provided by the Python extension.

JupyterLab — Opens JupyterLab using the selected environment. If missing, PyYK offers to install **jupyterlab** into that environment first.

Jupyter Notebook — Opens Jupyter Notebook using the selected environment. If missing, PyYK offers to install the **notebook** package.

Spyder — Opens the Spyder IDE using the selected environment. If missing, PyYK offers to install **spyder** into that environment.

> **Interpreter location on Windows** A regular virtual environment uses **Scripts\python.exe** inside its root folder. A Conda environment uses **python.exe** at the root.

Menus, log, and troubleshooting

The menus are available from every tab. The activity log and status line report what PyYK is doing.

File menu

Change environment folder… — Opens a folder picker. Select the parent folder for future environments and automatic discovery. Existing folders stay in place. Register an older environment with **Add existing environment…** if it is outside the new location.

Quit — Closes PyYK. If an operation is running, let it finish before closing.

Help menu

Python documentation — Opens the official Python documentation.

Miniforge downloads — Opens the Miniforge release page.

VS Code downloads — Opens the official VS Code download page.

About PyYK — Shows the app description and credits Ansys Python Manager as an inspiration, with a link to the original PyAnsys project.

Activity log and window controls

Activity log — Shows command output, install progress, package lists, and failures. Scroll to review earlier messages or select text to copy it. Drag the divider above the log to give it more space.

Status line — Shows Ready, the active operation, or a failure message. Tabs may be disabled while a command is running.

Title bar — Use the standard Windows controls to minimize, maximize or restore, and close the app.

If something goes wrong

Python is missing from the interpreter list — Click **Refresh**. If it is still absent, browse to the real **python.exe**. A terminal PATH warning from uv can coexist with a successful install, and PyYK can use the interpreter by its full path.

A package or tool fails to install — Read the last error in the **Activity log**. Verify the selected environment, Python version, and package availability. An existing environment also needs pip for package actions.

An environment is missing — Use **Refresh** for folders under the configured location, or **Add existing environment…** for an environment elsewhere.