Python in Posit Workbench

Table of contents

Users can write and execute Python code in all four IDEs available in Posit Workbench.

1. VS Code
2. JupyterLab
3. Jupyter Notebook
4. RStudio Pro

In this section, we will cover:

- Python best practices for using Python in any of the above IDEs.
- Additional resources for using Python.

Python best practices

Default Python interpreter

Typically your Workbench Server Administrator will choose a default version of Python for the server. To identify your default version of Python, run the following command:

```
which python3
```

If you wish to override this setting, you can do so with an alias that points towards the Python interpreter you want to use as the default. For example, if you have Python 3.10.7 installed at `/opt/python/3.10.7/bin/python3`, you could use the following alias:

Listing 1 `/.bashrc`

```
alias python="/opt/python/3.10.7/bin/python3"
```

Using multiple versions of Python

Posit Workbench users can switch between multiple versions of Python. Assuming that your Server Administrator installed Python following the instructions from the Installing Python Admin Guide page, users can identify all available versions of Python by running the following command:

```
$ ls -1d /opt/python/*
/opt/python/3.7.14
/opt/python/3.8.14
/opt/python/3.9.14
/opt/python/3.10.7
```

Users can choose their desired interpreter by directly invoking the absolute path to the interpreter (as opposed to invoking `python3` only). For example:

```
$ /opt/python/3.10.7/bin/python3
```

Virtual environments

We recommend using virtual environments for all Python projects. Virtual environments create isolated Python environments, which are helpful when projects have different dependency requirements. For example, you may have done some analysis last year using an older version of pandas. However, starting a new project today requires the latest version. Creating a virtual environment for each project ensures the old project continues using the old version of pandas and the new project uses the latest version.

To create and activate a virtual environment, run the following commands:

```
python3 -m venv .venv
source .venv/bin/activate
```

To create a virtual environment with a specific version of Python, invoke `venv` using the Python interpreter's absolute path:

```
/opt/python/3.10.4/bin/python3 -m venv .venv
source .venv/bin/activate
```

Invoking the `python3` command will use Python in your virtual environment instead of the default interpreter:

```
which python3
# /usr/home/<username>/my-project/.venv/bin/python3
```

URL Prefixes

When running in remote environments, many Python applications require users to specify the URL prefix of where the application is running. How to set this prefix varies by application, but the `rserver-url` binary can be used to generate the value for a given port. The `rserver-url` binary is located at `/usr/lib/rstudio-server/bin`, and we recommend adding this location to your `$PATH` for easy retrieval. Retrieving the prefix for a session running at `https://workbench-server/s/4566a3c9ab5a7ad01e1a7/` for port 8050 would look like the following:

```
$ rserver-url -l 8050
https://workbench-server/s/4566a3c9ab5a7ad01e1a7/p/30507931/
```

Because this value changes per session, rather than hard-coding this value in your code, we recommend you define a variable that calls `rserver-url` each time your code is run. This code can be added to an `if` block that checks if the environment variable `$RS_SERVER_URL` has been set so that it only executes inside of Workbench and not when your code is running elsewhere. For example, the following code sets `path` to the value generated by `rserver-url` and passes it as the `root_path` variable to the Uvicorn ASGI server:

```
if __name__ == '__main__':
    path, port = '', 8050

    # check if we're running in Posit Workbench
    if 'RS_SERVER_URL' in os.environ and os.environ['RS_SERVER_URL']:
        import subprocess
        # call rserver-url to retrieve path prefix
        path = subprocess.run(f'echo $(rserver-url -l {port})',
                              stdout=subprocess.PIPE, shell=True).stdout.decode().strip()
    # pass prefix to ASGI server
    uvicorn.run(app, port = port, root_path = path)
```

Alternatively, `rserver-url -l` can be passed to your application by wrapping it in `$()` and passing it from the command line. For example, starting a Uvicorn server from the command line looks like the following:

```
$ uvicorn main:app --port=8050 --root-path=$(rserver-url -l 8050)
```

Additional Python resources

Many good resources are available on the web for learning Python and seeking answers to questions about how to accomplish various tasks. Check out the resources below for more information on learning and using Python.

Learning Python

The Python Software Foundation has developed many good resources for starting with learning Python:

- The recommended starting place is with the tutorial.
- You could also view the Beginner's Guide to Python.
- Other training resources are listed at Getting Started. You can find many resources developed for different backgrounds ranging from expert programmers to those without programming experience, and even some developed for young children.
- There is an extensive list of books written on various Python topics.
- The r/learnpython Reddit maintains an extensive wiki for resources for getting started with Python.

Asking questions

If you have a question about a particular Python code issue, we'd recommend starting by creating a reproducible example (or reprex). This will often help as the process of creating an example will help you think about your problem in a way that helps solve it, and will help to capture the problem in a way that will allow others to understand the issue clearly. See here for more on creating a reprex.

You may also want to check out the following resources:

- The Posit Community is our discussion board for our community and covers topics ranging across multiple programming languages and workflows.
- Stack Overflow is an important resource for seeking answers to questions about Python
  - in particular, we'd recommend making sure your question is tagged as "python".

News and information

Python developers are an active community, and many new things are happening all the time. If you want to stay on top of what's happening, we recommend keeping up with the following sites:

- Follow the Posit blog to hear about our latest features, packages, and workshops.
- Python-bloggers is a blog aggregator that reposts Python-related articles from across the web. Python Bloggers is a good place to find Python tutorials, announcements, and other random happenings.
- The Python Software Foundation has several different newsgroups and mailing lists devoted to Python.

Other resources

- The r/Python and r/learnpython subreddits are active communities with a breadth of knowledge that welcome new members.