Everything is better with friends: Using SAS in Python applications with SASPy and open-source tools

[OPTIONAL] JOIN GITHUB

Visit https://github.com/ and create an account for free, if you don't already have one.

If you're new to GitHub, we highly recommend completing the hands-on "Hello World" exercise at https://guides.github.com/activities/hello-world/.

[OPTIONAL] JOIN OUR GITTER COMMUNITY

Visit https://gitter.im/saspy-bffs/community, sign in using your GitHub account credentials, and click the orange **JOIN ROOM** button.

Then chat privately with **ilankham** or **mtslaugh**, and tell us you'd like to be added to our Private Room for SAS Global Forum 2020.

Note. We also recommend installing the dedicated desktop and mobile Gitter apps available at https://gitter.im/apps.

REPLICATE TUTORIAL DEMO

These instructions use SASPy inside the Integrated Development Environment (IDE) PyCharm[1] and were developed under Windows 10 with SAS 9.4, Java SE version 8 update 201, Python 3.7.3, Git 2.21.0, and PyCharm Professional Edition 2019.1.1.

All default installation options are recommended.

Step 1: Setup development environment.

a. Download and install Java SE (to be used by SASPy to invoke SAS) from https://www.java.com/
b. Download and install Python[2] from https://www.python.org/

[1] You can choose between PyCharm Community Edition, which is free and open-source, and PyCharm Professional Edition, which has many additional features but requires a commercial license.

[2] Alternatively, the Anaconda distribution can be used, but please note that it has its own separate form of virtual environment called a *conda environment*, which will need to be used instead.

c. Download and install Git (to be used by PyCharm's GitHub integration) from https://git-scm.com/

d. Download and install PyCharm from https://www.jetbrains.com/pycharm/

Step 2: Setup project in PyCharm.

a. Start PyCharm, and when prompted, choose **Check out from Version Control** → **Git**, enter URL https://github.com/saspy-bffs/dataset-explorer, choose a directory to copy the files to, and click **Clone**.

b. Use the menu commands **File** → **Settings** → **Project: dataset-explorer** → **Project Interpreter**. Then click the **Gear Icon** (⚙) in the upper-right corner of the dialog box and select **Add…** This should prompt you to create a new virtual environment[3] as a subfolder named *venv* in your project folder. Once setup, click **OK** and, once processing has finished, click **OK** again to exit the dialog box.

c. Use the menu command **View** → **Tool Windows** → **Terminal** to open a terminal window. (Alternatively, click **Terminal** at the bottom of the PyCharm window.) Then type the following at the command prompt and press Enter: **`pip install -r requirements.txt`**

d. Use the project-navigation area in the left-hand panel to open the file *sascfg_personal-example.py* (i.e., double-click its name), and copy its contents to the system clipboard. Then use the menu command **File** → **New** → **Python File** to create a new file named *sascfg_personal.py*, paste the contents of the system clipboard into it, and update to match your SAS installation setup per the instructions at https://sassoftware.github.io/saspy/install.html.

Warning: Step 2(d) is not straightforward.

Step 3: Run the example file.

a. Use the menu command **Run** → **Run …** → **app.py**, and then watch the output scroll by in the Run portion of bottom panel. (Alternatively, open the file *app.py* by double-clicking its name in the project-navigation area in the left-hand panel, right-click anywhere inside the code editor window, and choose the command **Run 'app'**.)

Step 4: Access the Python application.

a. Visit the URL http://127.0.0.1:8000/ in a web browser, and follow the instructions.

If SAS was installed with default options, try the directory **C:\Program Files\SASHome\SASFoundation\9.4\core\sashelp**

[3] A *virtual environment* (aka *venv*) is essentially a completely separate installation of Python, which is cloned from the version of Python installed as Step 1(b). It's considered best practice to create a new venv (or conda environment, if using the Anaconda distribution of Python) for each project in order to keep its dependencies isolated from other projects and their dependencies.