**Running Airflow on Windows 10 & WSL**

Windows has come a long way in the last few years by catering to open source software and developers in an increasing capacity. Nothing illustrates this more than their development of WSL (Windows Subsystem for Linux), which allows you to install a Linux distribution on your PC alongside Windows without having to worry about VMs or containers. This is great for developers that work with tools that only run in Linux, such as Apache Airflow.

Airflow is top-level Apache project used for orchestrating workflows and data pipelines. It is quickly becoming a popular choice for organizations of all sizes and industries. Airflow is built in Python but contains some libraries that will only work in Linux, so workarounds using virtual machines or Docker are required for fully-functional usage. While both VMs and Docker are great options, this post will talk about setting up Airflow in WSL for very simple access to Airflow with little overhead.

**Installing WSL**

WSL is a relatively new and feature of Windows. It's frequently updated and its functionality is rapidly improving. Instead of walking through all the steps to install here (since they may change), follow [this doc from Microsoft](https://docs.microsoft.com/en-us/windows/wsl/install-win10) for the latest steps. This post will focus on using Ubuntu for WSL, which you can [download here](https://www.microsoft.com/en-us/p/ubuntu/9nblggh4msv6?activetab=pivot:overviewtab).

Make sure you follow the link to initialize your new distro instance. This will create a Linux username and password. Remember this password- it is needed to use sudo going forward. Also run sudo apt update && sudo apt upgrade to make sure everything is up to date.

Warning

Files in the Linux file system should not be accessed from Windows, as they can end up getting corrupted. Do not open or edit any files from processes running in Windows such as explorer, notepad, atom, or another IDE.

However, Linux can access files in the Windows file system. Because of this capability, I like to organize all my project files in the Windows file system. For example, you can create a folder in C:/Users/your\_username/airflow and use it as your AIRFLOW\_HOME directory. You can edit files (DAGs, plugins, etc.) in here with your favorite Windows editors (notepad, atom, PyCharm, etc.) and then use them with Airflow from WSL.

Future Windows updates may change this behavior and allow Linux files to be edited by Windows processes, so keep an eye out for that.

Additional Setup

Once you have WSL installed, launch Ubuntu to land on the bash command prompt in your Linux home directory. If you type pwd (present working directory), you should see /home/user\_name, indicating that you are in the Linux file system in your Linux home directory. Using ~ will reference this path.

Type ls -a to view the contents of this directory, including hidden objects. You’ll see .bashrc. This file is useful for changing settings such as environment variables that you want to apply to your bash sessions every time you start WSL.

The Windows file system is available to you in Linux and is located at /mnt/c. If you type ls /mnt/c you will see the contents of the C:/ in windows. Note: you will see some permissions errors regarding symbolic links, you can ignore those.

It can be a bit burdensome to have to look in /mnt/c to get to your windows files. In fact, if you end up wanting to use Docker in WSL (which you probably will), this actually won’t work at all. So let’s change how the Windows file system is mounted in Linux.

Type sudo nano /etc/wsl.conf. This will open the Linux nano text editor for editing files. Change the file structure to match the following:

[automount]

root = /

options = "metadata"

Enter ctrl + s, ctrl + x to save the changes and exit nano. Now sign out, then back in to windows for these changes to take affect. Open Ubuntu and type ls / and you should see that c is now mounted on root / instead of /mnt/.

Python 3

Confirm you have Python 3 installed with python3 --version.

Now let's install pip: sudo apt update, sudo apt install python3-pip should do the trick.

Run pip3 --version to make sure it's installed correctly.

Installing Airflow

This step should be no different than installing Airflow in any normal Ubuntu environment. If you want, you can include other Airflow modules such as postrges or s3. Some of these may require dependencies to be installed on Ubuntu using sudo apt install [your\_dependency].

Run pip3 install apache-airflow.

Now let’s set AIRFLOW\_HOME (Airflow looks for this environment variable whenever Airflow CLI commands are run). When you run airflow init it will create all the Airflow stuff in this directory. As mentioned earlier, we want this to be in the Windows file system so you can edit all the files from Windows based tools.

Running nano ~/.bashrc will open up the file mentioned earlier for setting environment variables in your bash session. On a new line add the following statement export AIRFLOW\_HOME=/c/Users/user\_name/AirflowHome statement replacing user\_name with your actual windows home folder name. It should look something like this:

# ~/.bashrc: executed by bash(1) for non-login shells.

# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)

# for examples

export AIRFLOW\_HOME=/c/Users/user\_name/AirflowHome

Use ctrl + s, ctrl + x to save and exit nano. Now, anytime you open a bash session in Ubuntu, the AIRFLOW\_HOME environment variable will be set to AirflowHome folder in your Windows home directory.

Close and reopen Ubuntu. airflow version should now show you the version of airflow you installed with out any errors and running airflow initdb should populate your AirflowHome folder with a clean setup for Airflow. You can run airflow webserver or airflow scheduler to start those services.

And that’s it- happy Airflowing!

Note: You *could* use a python virtual environment to install Airflow. Below I’ll talk about using PyCharm IDE with WSL and, at the time this post was written, PyCharm doesn’t support virtual environments with WSL.

Using Astronomer

If you are using the Astronomer platform, you can install the Astro CLI in WSL to easily create local development environments and push deployments to clusters. The CLI requires Docker installed in WSL. Astro CLI for Windows is coming soon!

Docker

Docker is required to use Astronomer's CLI. [Here](https://nickjanetakis.com/blog/setting-up-docker-for-windows-and-wsl-to-work-flawlessly) is a great guide on how to install docker for use in WSL.

Astro CLI

Install the CLI via curl -sSL https://install.astronomer.io | sudo bash.

Running astro version will test if it's installed correctly. You can now navigate to your AIRFLOW\_HOME directory and run astro airflow init, creating an initialized Astronomer project in that directory. Running astro airlfow start will spin up some docker containers and launch a local environment with a webserver and scheduler, which you can view in your browser at localhost:8080. There, you will also see your Airflow UI and your example\_dag. Running astro airlfow stop will take down the containers. [Here](https://www.astronomer.io/docs/cli-quickstart) is the quickstart guide for the Astro CLI if you'd like a more detailed rundown of how to get started with it.

Note: If you get a Postgres error, try running Docker pull postgres:10.1-alpine

Using PyCharm

Pycharm is a great IDE that I enjoy using to develop my Airflow DAGs, plugins and all other python projects. The Pro edition isn't free and is required to interact with WSL, but if you want to make the purchase, here are a few tips for setting it up to work with WSL and Airflow. There is a free 30 day trial if you want to give this a try.

Setup an Interpreter for WSL

PyCharm has as issue that it expects the Windows files system to be mounted at /mnt/c. However, earlier we changed the settings so it would mount to /c. For the sake of PyCharm, we can create a symbolic link at /mnt/c to point to /c in the Linux files system.

sudo rm -r /mnt/c

cd /mnt

sudo ln -s /c c

This will first remove the old /mnt/c directory, then create a link named c that points to /c. Now when Pycharm looks for /mnt/c it will be directed to /c

Start PyCharm, if a project opens, go to File -> Settings, if it's the welcome screen, go to Configure -> Settings.

Select Project Interpreter and click the cog icon in the top right.

Click Add and select WSL.

Select Ubuntu for your Linux Distribution and set the Python Interpreter path to /usr/bin/python3

Now when you use the interpreter with a Run Config, it will execute your python code in the WSL environment which has Airflow installed. (Even though PyCharm and the files you're debugging are in Windows)

Debug Airflow DAGs, Hooks and Operators (With Breakpoints!)

Now you can put breakpoints in your DAGs and plugins, run the code in debug mode, and step through line by line. You will be using airflow test for this. This will run a task that exists in a DAG. So you will need to create a DAG and add tasks to it. Put breakpoints in the Operator of the task you want to debug.

Create a Run/Debug Configuration by clicking the drop down in the upper right and select Edit Configurations.

Click the + to add a new configuration and give it a name.

in the script path you will point this to where airflow was install in WSL which should be in your user home directory, so use ~\.local\bin\airflow.

For parameters enter test dag\_id task\_id date. You are using Airlflow's native [test functionality](https://airflow.apache.org/cli.html#test).

For the Python interpreter, choose the interpreter you created in the previous step.

Make sure your working directory points to the dags folder in your Windows file system.

Click OK, then click the debug icon next to the play icon. The code will begin to run in WSL and PyCharm will attach itself wherever you enter your breakpoints.

Using Cmder

Cmder is console emulator for windows. It is great for interacting with both WSL and Windows. I use it instead of the Ubuntu console or the windows command prompt/powershell

Click [here](https://cmder.net/) to read more and download

In Cmder, click the dropdown by the green plus in the bottom right corner and select setup tasks. There should be a WSL::bash task. In the commands box, I like to add a line that makes Cmder open to my Windows home folder instead of the Ubuntu home folder. It will look something like this.

set "PATH=%ConEmuBaseDirShort%\wsl;%PATH%" & %ConEmuBaseDirShort%\conemu-cyg-64.exe --wsl -cur\_console:pm:/mnt -C "/c/Users/user\_name" -e

And that's just about it! Please do reach out if you have any questions or feedback on this guide- happy Airflowing!