## [Apache Airflow](https://airflow.apache.org/docs/apache-airflow/stable/index.html)

Apache Airflow Core, which includes web server, scheduler, CLI and other components that are needed for minimal Airflow installation.

# What is Airflow?

[Apache Airflow](https://github.com/apache/airflow) is an open-source platform for developing, scheduling, and monitoring batch-oriented workflows. Airflow’s extensible Python framework enables you to build workflows connecting with virtually any technology. A web interface helps manage the state of your workflows. Airflow is deployable in many ways, varying from a single process on your laptop to a distributed setup to support even the biggest workflows.

# Workflows as code

The main characteristic of Airflow workflows is that all workflows are defined in Python code. “Workflows as code” serves several purposes:

* **Dynamic**: Airflow pipelines are configured as Python code, allowing for dynamic pipeline generation.
* **Extensible**: The Airflow framework contains operators to connect with numerous technologies. All Airflow components are extensible to easily adjust to your environment.
* **Flexible**: Workflow parameterization is built-in leveraging the [Jinja](https://jinja.palletsprojects.com/) templating engine.

**Here you see:**

* A DAG named “demo”, starting on Jan 1st 2022 and running once a day. A DAG is Airflow’s representation of a workflow.
* Two tasks, a BashOperator running a Bash script and a Python function defined using the @task decorator
* >> between the tasks defines a dependency and controls in which order the tasks will be executed

Airflow evaluates this script and executes the tasks at the set interval and in the defined order.

**Each column represents one DAG run. These are two of the most used views in Airflow, but there are several other views which allow you to deep dive into the state of your workflows.**

# Why Airflow?

Airflow is a batch workflow orchestration platform. The Airflow framework contains operators to connect with many technologies and is easily extensible to connect with a new technology. If your workflows have a clear start and end, and run at regular intervals, they can be programmed as an Airflow DAG.

If you prefer coding over clicking, Airflow is the tool for you. Workflows are defined as Python code which means:

* **Workflows can be stored in version control so that you can roll back to previous versions**
* **Workflows can be developed by multiple people simultaneously**
* **Tests can be written to validate functionality**
* **Components are extensible and you can build on a wide collection of existing components**

Running Airflow on Mac os Locally

Before starting **make sure you set your airflow home variable to the working directory**. Validate this by typing echo $AIRFLOW\_HOME .

—------------------------------------------------------------------------------------------------------------------

**Installation of airflow**

<https://dlcdn.apache.org/airflow/2.4.3/apache-airflow-2.4.3-source.tar.gz>

Or

## **Install Apache Airflow**

I've created an environment based on Python 3.9, so the constraints file path looks like this:

https://raw.githubusercontent.com/apache/airflow/constraints-2.2.3/constraints-no-providers-3.9.txt

Assuming everything works, copy the following command to install Apache Airflow:

pip install "apache-airflow==2.2.3" --constraint "https://raw.githubusercontent.com/apache/airflow/constraints-2.2.3/constraints-no-providers-3.9.txt"

## **Setup Airflow Database and User**

Once you have Airflow installed, initialize the database with the following Terminal command:

airflow db init

It will create the airflow folder in your root directory, so navigate to it:

cd ~/airflow

ls

Here are the files:

Image 4 - Airflow root directory (image by author)

The airflow.db is the *Metastore* Airflow uses, and you'll see how to access it at the end of the article. You'll also see how to edit airflow.cfg, and why should you do it.

But first, let's create an Airflow user:

airflow users create \

--username admin \

--password admin \

--firstname <FirstName> \

--lastname <LastName> \

--role Admin \

--email <YourEmail>

## **Start Airflow Webserver and Scheduler**

Apache Airflow consists of two core parts - *Webserver* and *Scheduler*. You'll have to run both to inspect and run your DAGs.

First, start the Webserver in the daemon mode (as a background process):

airflow webserver -D

Once it's running, use a similar command to run the Scheduler:

airflow scheduler -D

Airflow runs on port 8080 by default, so open the following URL in your browser:

http://localhost:8080

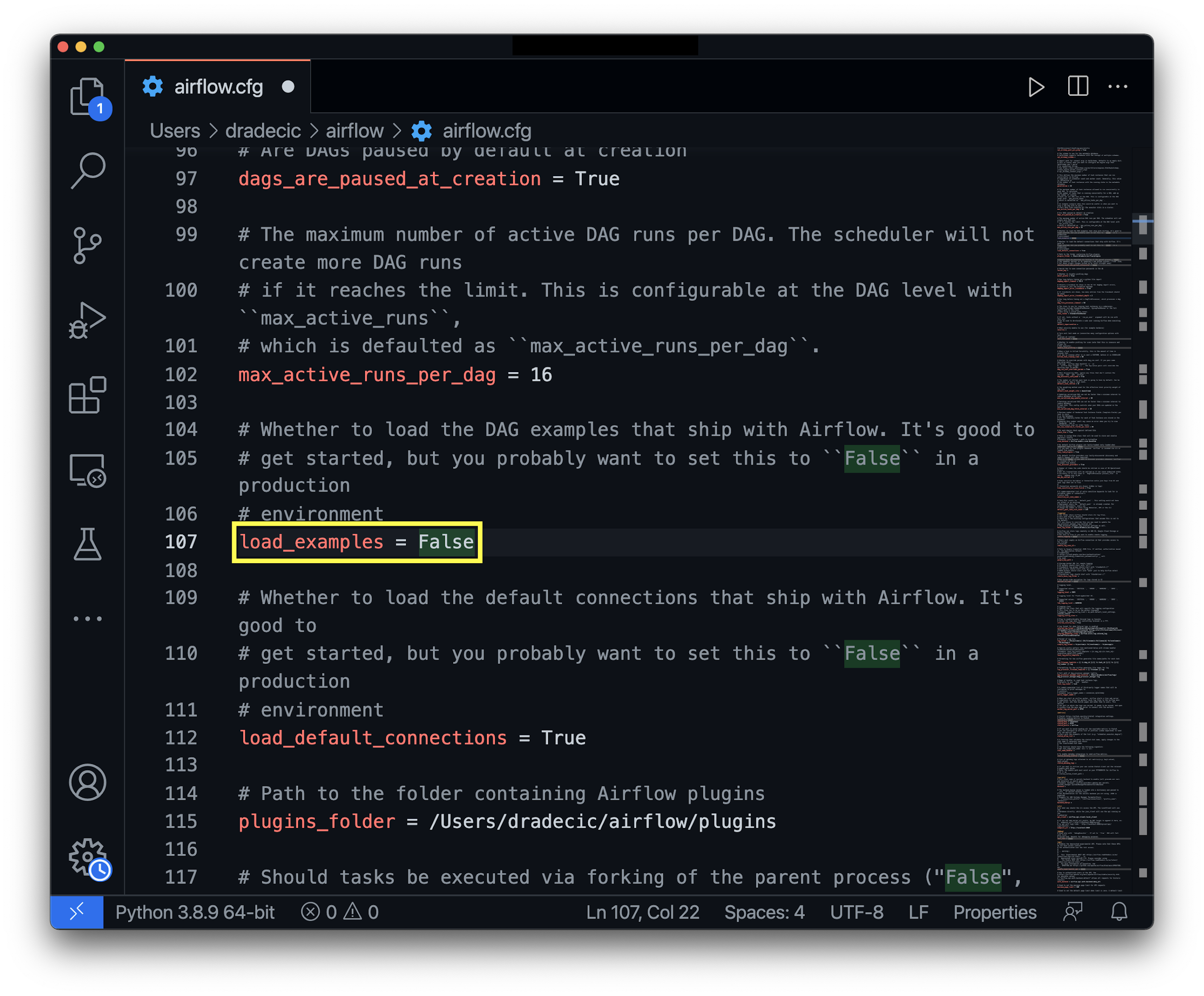
Use credentials specified when creating the Airflow user (admin/admin) and hit the *Sign in* button:

## 

## 

## **Bonus: How to Remove Airflow Example DAGs**

You could delete the DAGs one by one, but there's a better approach. Open the ~/airflow/airflow.cfg file and change the load\_examples value to False:

Image 11 - Editing airflow.cfg file (image by author)

Save the file and reopen the terminal. You'll have to reset the Airflow database:

airflow db reset

Once done, start the Airflow webserver and scheduler once again:

airflow webserver -D

airflow scheduler -D

Hint: If Terminal tells you Airflow is already running, get the process ID of the task running on port 8080 (lsof -i tcp:8080) and then use kill <pid> to terminate the process. Now you should be able to run both Webserver and Scheduler.

Or

1. **Installing just Airflow:**

Note: Only pip installation is currently officially supported.

While it is possible to install Airflow with tools like [Poetry](https://python-poetry.org/) or [pip-tools](https://pypi.org/project/pip-tools), they do not share the same workflow as pip - especially when it comes to constraint vs. requirements management. Installing via Poetry or pip-tools is not currently supported.

If you wish to install Airflow using those tools, you should use the constraint files and convert them to the appropriate format and workflow that your tool requires.

pip install 'apache-airflow==2.3.3' \

--constraint "https://raw.githubusercontent.com/apache/airflow/constraints-2.3.3/constraints-3.7.txt"

1. Installing with extras (i.e., postgres, google)

pip install 'apache-airflow[postgres,google]==2.3.3' \

--constraint "https://raw.githubusercontent.com/apache/airflow/constraints-2.3.3/constraints-3.7.txt"