

FEBRUARY 2, 2023/[#DATA SCIENCE](#)

How to Install Apache Airflow on Windows without Docker



[Aviator Ifeanyichukwu](#)



Apache Airflow is a tool that helps you manage and schedule data pipelines. According to the [documentation](#), it lets you "programmatically author, schedule, and monitor workflows."

Airflow is a crucial tool for data engineers and scientists. In this article, I'll show you how to install it on Windows without Docker.

Although it's recommended to run Airflow with Docker, this method works for low-memory machines that are unable to run Docker.

Prerequisites:

This article assumes that you're familiar with using the command line and can set up your development environment as directed.

Requirements:

You need Python 3.8 or higher, Windows 10 or higher, and the Windows Subsystem for Linux (WSL2) to follow this tutorial.

What is Windows Subsystem for Linux (WSL2)?

WSL2 allows you to run Linux commands and programs on a Windows operating system.

It provides a Linux-compatible environment that runs natively on Windows, enabling users to use Linux command-line tools and utilities on a Windows machine.

You can read more [here to install WSL2](#) on your machine. With Python and WSL2 installed and activated on your machine, launch the terminal by searching for Ubuntu from the start menu.

Step 1: Set Up the Virtual Environment

To work with Airflow on Windows, you need to set up a virtual environment. To do this, you'll need to install the virtualenv package.

Note: Make sure you are at the root of the terminal by typing:

```
cd ~  
pip install virtualenv
```

Create the virtual environment like this:

```
virtualenv airflow_env
```

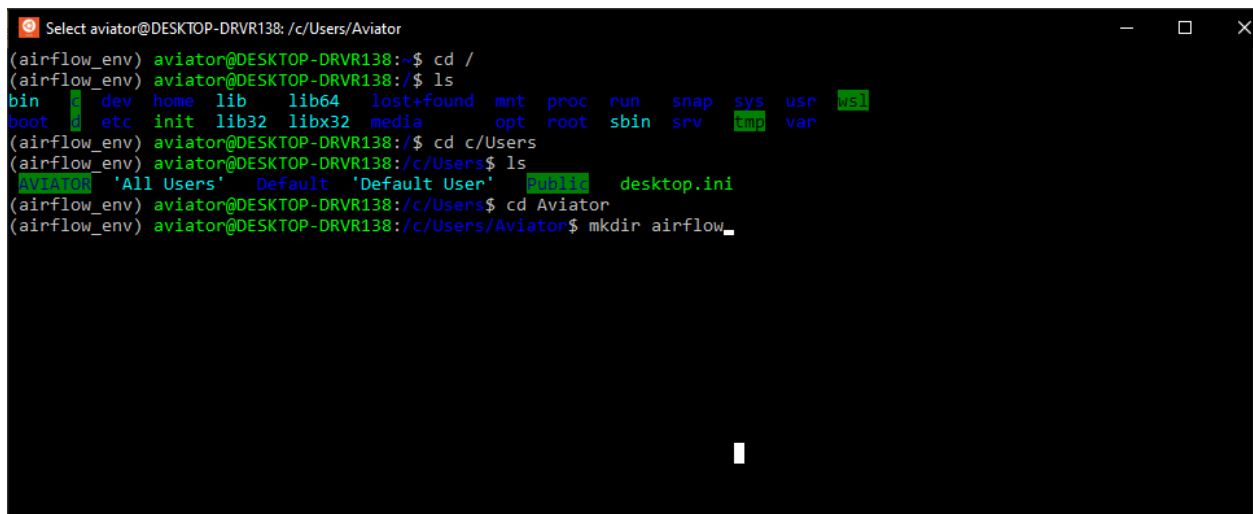
And then activate the environment:

```
source airflow_env/bin/activate
```

Step 2: Set Up the Airflow Directory

Create a folder named airflow. Mine will be located at c/Users/[Username]. You can put yours wherever you prefer.

If you do not know how to navigate the terminal, you can follow the steps in the image below:

A terminal window titled "Select aviator@DESKTOP-DRVR138: /c/Users/Aviator" with standard window controls. The terminal shows a sequence of commands and their outputs: 1. Command: `cd /`, Output: `(airflow_env) aviator@DESKTOP-DRVR138:~$ cd /`. 2. Command: `ls`, Output: `(airflow_env) aviator@DESKTOP-DRVR138:/$ ls` followed by a directory listing: `bin dev home lib lib64 lost+found mnt proc run snap sys usr wsl` and `boot etc init lib32 libx32 media opt root sbin srv tmp var`. 3. Command: `cd c/Users`, Output: `(airflow_env) aviator@DESKTOP-DRVR138:/$ cd c/Users`. 4. Command: `ls`, Output: `(airflow_env) aviator@DESKTOP-DRVR138:/c/Users$ ls` followed by a directory listing: `AVIATOR 'All Users' Default 'Default User' Desktop desktop.ini`. 5. Command: `cd Aviator`, Output: `(airflow_env) aviator@DESKTOP-DRVR138:/c/Users$ cd Aviator`. 6. Command: `mkdir airflow_`, Output: `(airflow_env) aviator@DESKTOP-DRVR138:/c/Users/Aviator$ mkdir airflow_`.

```
Select aviator@DESKTOP-DRVR138: /c/Users/Aviator
(airflow_env) aviator@DESKTOP-DRVR138:~$ cd /
(airflow_env) aviator@DESKTOP-DRVR138:/$ ls
bin dev home lib lib64 lost+found mnt proc run snap sys usr wsl
boot etc init lib32 libx32 media opt root sbin srv tmp var
(airflow_env) aviator@DESKTOP-DRVR138:/$ cd c/Users
(airflow_env) aviator@DESKTOP-DRVR138:/c/Users$ ls
AVIATOR 'All Users' Default 'Default User' Desktop desktop.ini
(airflow_env) aviator@DESKTOP-DRVR138:/c/Users$ cd Aviator
(airflow_env) aviator@DESKTOP-DRVR138:/c/Users/Aviator$ mkdir airflow_
```

Create an Airflow directory from the terminal

Now that you have created this folder, you have to set it as an environment variable. Open a .bashrc script from the terminal with the command:

```
nano ~/.bashrc
```

Then write the following:

```
AIRFLOW_HOME=/home/[YourUsername]/airflow
```

```
aviator@DESKTOP-DRVR138: /c/Users/Aviator
GNU nano 4.8 /home/aviator/.bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

AIRFLOW_HOME=/c/Users/AVIATOR/airflow

# If not running interactively, don't do anything
case $- in
  *i*) ;;
  *) return;;
esac
```

Setup Airflow directory path as an environment variable
Press ctrl s and ctrl x to exit the nano editor.

This part of the Airflow directory will be permanently saved as an environment variable. Anytime you open a new terminal, you can recover the value of the variable by typing:

```
cd $AIRFLOW_HOME
```

```
aviator@DESKTOP-DRVR138: /c/Users/AVIATOR/airflow
(airflow_env) aviator@DESKTOP-DRVR138:/c/Users/Aviator$ nano ~/.bashrc
(airflow_env) aviator@DESKTOP-DRVR138:/c/Users/Aviator$ cd ~
(airflow_env) aviator@DESKTOP-DRVR138:~$ cd $AIRFLOW_HOME
(airflow_env) aviator@DESKTOP-DRVR138:/c/Users/AVIATOR/airflow$ _
```

Navigate to Airflow directory using the environment variable

Step 3: Install Apache Airflow

With the virtual environment still active and the current directory pointing to the created Airflow folder, install Apache Airflow:

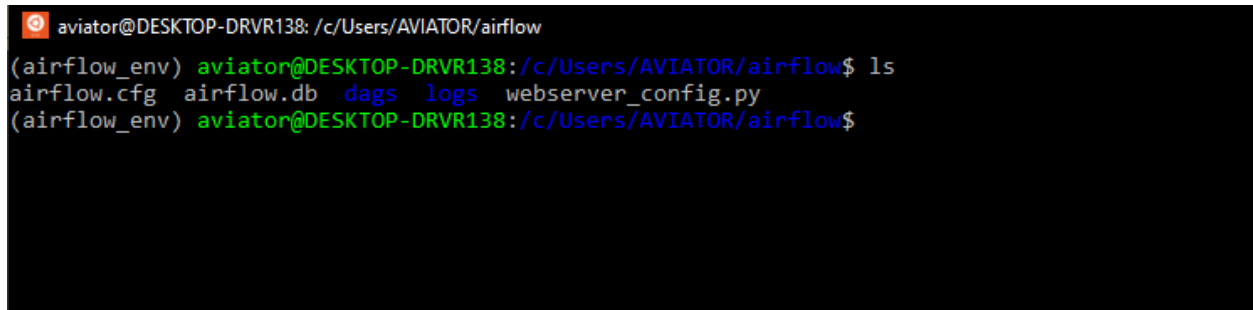
```
pip install apache-airflow
```

Initialize the database:

```
pip install Flask-Session==0.5.0
```

```
airflow db migrate
```

Create a folder named dags inside the airflow folder. This will be used to store all Airflow scripts.

A terminal window with a black background and green text. The prompt is 'aviator@DESKTOP-DRV138: /c/Users/AVIATOR/airflow'. The command 'ls' has been executed, and the output is 'airflow.cfg airflow.db dags logs webserver_config.py'.

```
aviator@DESKTOP-DRV138: /c/Users/AVIATOR/airflow
(airflow_env) aviator@DESKTOP-DRV138:/c/Users/AVIATOR/airflow$ ls
airflow.cfg airflow.db dags logs webserver_config.py
(airflow_env) aviator@DESKTOP-DRV138:/c/Users/AVIATOR/airflow$
```

View files and folders generated by Airflow db init

Step 4: Create an Airflow User

When airflow is newly installed, you'll need to create a user. This user will be used to login into the Airflow UI and perform some admin functions.

```
airflow users create --username admin --password admin --firstname admin --
lastname admin --role Admin --email youremail@email.com
```

Check the created user:

```
airflow users list
```

```
aviator@DESKTOP-DRV138: /c/Users/AVIATOR/airflow
(airflow_env) aviator@DESKTOP-DRV138:/c/Users/AVIATOR/airflow$ airflow users create --username admin --password admin
--firstname admin --lastname admin --role Admin --email [REDACTED]@gmail.com
[2023-01-28 22:55:35,781] {dynamodb.py:10} WARNING - no boto3 module found

Please confirm database initialize (or wait 4 seconds to skip it). Are you sure? [y/N]
[2023-01-28 22:55:41,684] {manager.py:824} WARNING - No user yet created, use flask fab command to do it.
[2023-01-28 22:55:45,738] {manager.py:212} INFO - Added user admin
User "admin" created with role "Admin"
(airflow_env) aviator@DESKTOP-DRV138:/c/Users/AVIATOR/airflow$ airflow users list
[2023-01-28 22:55:57,109] {dynamodb.py:10} WARNING - no boto3 module found
id | username | email | first_name | last_name | roles
-----+-----+-----+-----+-----+-----
1 | admin | [REDACTED]@gmail.com | admin | admin | Admin

(airflow_env) aviator@DESKTOP-DRV138:/c/Users/AVIATOR/airflow$
```

Create an Airflow user and list the created user

Step 5: Run the Webserver

Run the scheduler with this command:

```
airflow scheduler
```

Launch another terminal, activate the airflow virtual environment, cd to \$AIRFLOW_HOME, and run the webserver:

```
airflow webserver
```

If the default port 8080 is in use, change the port by typing:

```
airflow webserver -port <port number>
```

Log in to the UI using the username created earlier with "airflow users create".

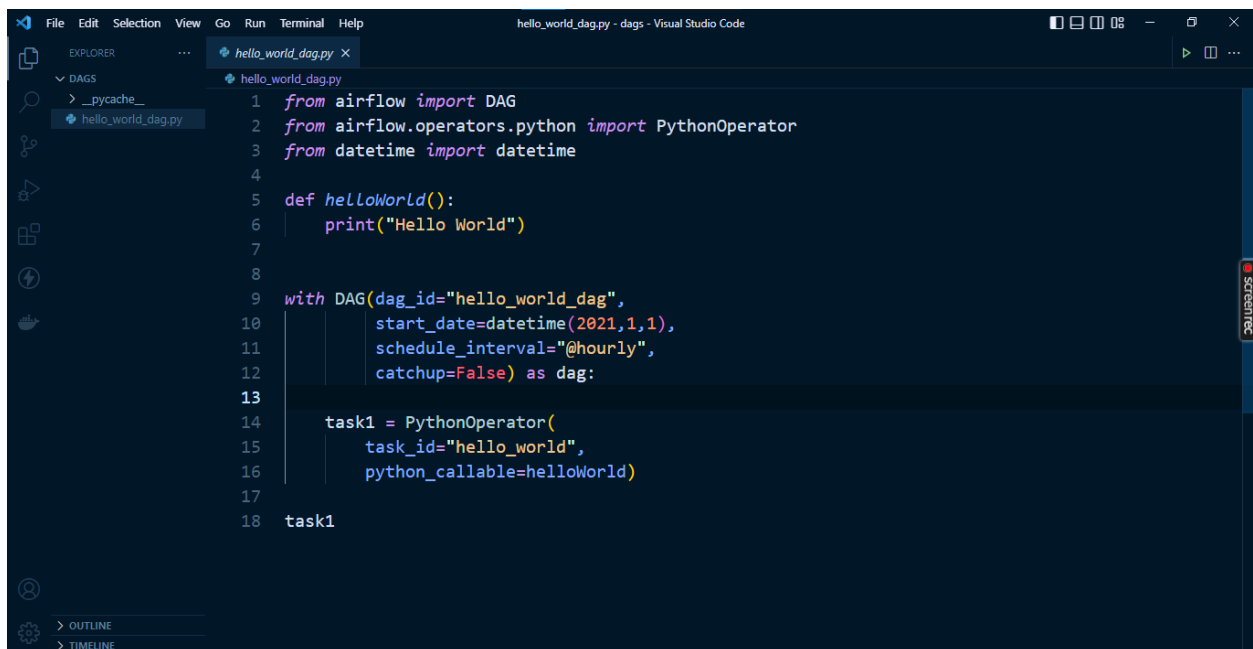
In the UI, you can view pre-created DAGs that come with Airflow by default.

How to Create the first DAG

A DAG is a Python script for organizing and managing tasks in a workflow.

To create a DAG, navigate into the dags folder created inside the \$AIRFLOW_HOME directory. Create a file named "hello_world_dag.py". Use VS Code if it's available.

Enter the code from the image below, and save it:

A screenshot of the Visual Studio Code editor interface. The title bar at the top reads "hello_world_dag.py - dags - Visual Studio Code". The Explorer sidebar on the left shows a file tree with "DAGS" expanded, containing "_pycache_" and "hello_world_dag.py". The main editor area displays the following Python code:

```
1 from airflow import DAG
2 from airflow.operators.python import PythonOperator
3 from datetime import datetime
4
5 def helloWorld():
6     print("Hello World")
7
8
9 with DAG(dag_id="hello_world_dag",
10         start_date=datetime(2021,1,1),
11         schedule_interval="@hourly",
12         catchup=False) as dag:
13
14     task1 = PythonOperator(
15         task_id="hello_world",
16         python_callable=helloWorld)
17
18 task1
```

Example DAG script in VS Code editor

Go to the Airflow UI and search for hello_world_dag. If it does not show up, try refreshing your browser.

That's it. This completes the installation of Apache Airflow on Windows.