**Airflow with docker**

[**https://airflow.apache.org/docs/apache-airflow/stable/tutorial/pipeline.html**](https://airflow.apache.org/docs/apache-airflow/stable/tutorial/pipeline.html)

**Detailed tricks & usage**

[**https://www.youtube.com/watch?v=K9AnJ9\_ZAXE&list=PLwFJcsJ61oujFW8pTo9S8\_b6wujg5NgGW**](https://www.youtube.com/watch?v=K9AnJ9_ZAXE&list=PLwFJcsJ61oujFW8pTo9S8_b6wujg5NgGW)

**to start airflow with docker**

*# Download the docker-compose.yaml file*

curl -LfO 'https://airflow.apache.org/docs/apache-airflow/stable/docker-compose.yaml'

*# Make expected directories and set an expected environment variable*

mkdir -p ./dags ./logs ./plugins

echo -e "AIRFLOW\_UID=**$(**id -u**)**" > .env

*# Initialize the database*

docker-compose up airflow-init

*# Start up all services*

docker-compose up

**turn on just airflow without init db: init db will reset all settings in db**

docker-compose up

**turn down docker and release volume assigned to docker**

docker-compose down -v

**to change port  
Text

Description automatically generated**

**8080 is the docker container**

**5050 will be the airflow link** [**http://localhost:5050**](http://localhost:5050)

**Db**

**Text

Description automatically generated**

**To access postgres outside of airflow**

**And**

**To connect airflow to the db**

**Note that this postgres is created inside a docker image so it needs docker to be up for access .**

**Graphical user interface, application

Description automatically generated**

**Graphical user interface, application

Description automatically generated**

**Change to port** other than 5432 **so dbeaver can connect eg 6432:5432**

**DAGS**

1. [task1, task2]>>task3

**# means task1 and task2 will run at the same time, once both complete, task 3 will run**

1. **Use bash command to execute python files/packages**
2. **Preferred way to use for python functions: taskflow api**
3. **Dag using python operator with parameter:**
4. **To backfill after setting catchup = False**

In terminal run:

docker ps

**this will give you instance of docker running**

Text

Description automatically generated

docker exec -it c4bc….. bash

**to backfill**

airflow dags backfill -s 2022-01-01 -e 2022-02-22

exit

**refresh UI to see refreshed in the past dates**

**Graphical user interface, text, application, chat or text message

Description automatically generated**

1. **a good DAG example:**

**https://docs.astronomer.io/learn/dags**

**Text

Description automatically generated**

**To manage python dependencies (library) on airflow,**

**Create**

requirements.txt

Dockerfile

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text

Description automatically generated

Then restart the airflow using docker-compose up

Try docker image extending

**To run existing env**

python3 -m venv /path/to/new/virtual/environment

**to create env**

python3 -m venv [name of your env eg: env\_airflow]

source env\_ariflow/bin/activate

pip install apache-airflow