

Play with your DAG

Before moving forward, it's important to know how to run your DAGs, retry tasks, clean the meta data, access the logs, etc.

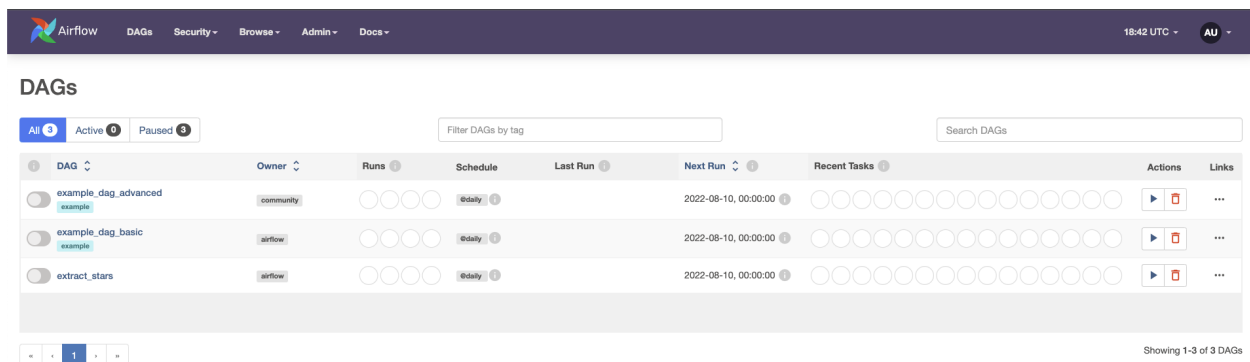
That's exactly what you are going to discover in this activity

Prerequisites

You should have the DAG `extract_stars.py` in your folder `days/`. If not, take a look at the previous activity `Start coding your first DAG...`

Play with your DAG

If you go on `localhost:8080`, you should land on the following page:



The screenshot shows the Apache Airflow web interface. At the top, there's a navigation bar with links for DAGs, Security, Browse, Admin, and Docs. The main heading is "DAGs". Below it, there are tabs for "All", "Active", and "Paused". A search bar "Filter DAGs by tag" and a "Search DAGs" input field are present. The main table lists DAGs with columns: DAG, Owner, Runs, Schedule, Last Run, Next Run, Recent Tasks, Actions, and Links. Three DAGs are listed: "example_dag_advanced" (owner: community), "example_dag_basic" (owner: airflow), and "extract_stars" (owner: airflow). Each DAG has a toggle switch, a "Runs" column with four circles, a "Schedule" column with "daily", a "Last Run" column with "2022-08-10, 00:00:00", a "Next Run" column with "2022-08-10, 00:00:00", and a "Recent Tasks" column with ten circles. The "extract_stars" DAG has its toggle switch turned on. At the bottom, there's a pagination bar showing "1" and a status "Showing 1-3 of 3 DAGs".

DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions	Links
<input checked="" type="checkbox"/> example_dag_advanced	community	○ ○ ○ ○	daily	2022-08-10, 00:00:00	2022-08-10, 00:00:00	○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ 🗑️ ...	
<input type="checkbox"/> example_dag_basic	airflow	○ ○ ○ ○	daily	2022-08-10, 00:00:00	2022-08-10, 00:00:00	○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ 🗑️ ...	
<input checked="" type="checkbox"/> extract_stars	airflow	○ ○ ○ ○	daily	2022-08-10, 00:00:00	2022-08-10, 00:00:00	○ ○ ○ ○ ○ ○ ○ ○ ○ ○	▶ 🗑️ ...	

First, turn on the toggle next to the DAG name `extract_stars`. That allows to pause or unpause your DAG. In other words, start scheduling it or not.

Refresh the page many times until you see something like that

DAG: extract_stars Schedule: @daily Next Run: 2022-08-11, 00:00:00

Grid Graph Calendar Task Duration Task Tries Landing Times Gantt Details Code Audit Log

11/08/2022, 18:45:54 25 All Run Types All Run States Clear Filters

Auto-refresh ☐

Duration: 00:00:02 00:00:01 00:00:00

get_date

DAG Run Details Graph Mark Failed Mark Success

Re-run Clear existing tasks Queue up new tasks

Status: success

Run Id: scheduled_2022-08-10T00:00:00+00:00

Run Type: scheduled

Duration: 00:00:02

Last Scheduling Decision: 2022-08-11, 18:44:28 UTC

Started: 2022-08-11, 18:44:26 UTC

Ended: 2022-08-11, 18:44:28 UTC

Data Interval:

Start: 2022-08-10, 00:00:00 UTC

End: 2022-08-11, 00:00:00 UTC

Click **Confirm**. That's how you can rerun a DAG Run.

DAG: extract_stars Schedule: @daily Next Run: 2022-08-11, 00:00:00

Grid Graph Calendar Task Duration Task Tries Landing Times Gantt Details Code Audit Log

11/08/2022, 18:45:54 25 All Run Types All Run States Clear Filters

Auto-refresh ☒

Duration: 00:00:02 00:00:01 00:00:00

get_date

DAG Run Details Graph Mark Failed Mark Success

Re-run Clear existing tasks Queue up new tasks

Status: running

Run Id: scheduled_2022-08-10T00:00:00+00:00

Run Type: scheduled

Duration: 00:00:02

Last Scheduling Decision: 2022-08-11, 18:48:44 UTC

Started: 2022-08-11, 18:48:44 UTC

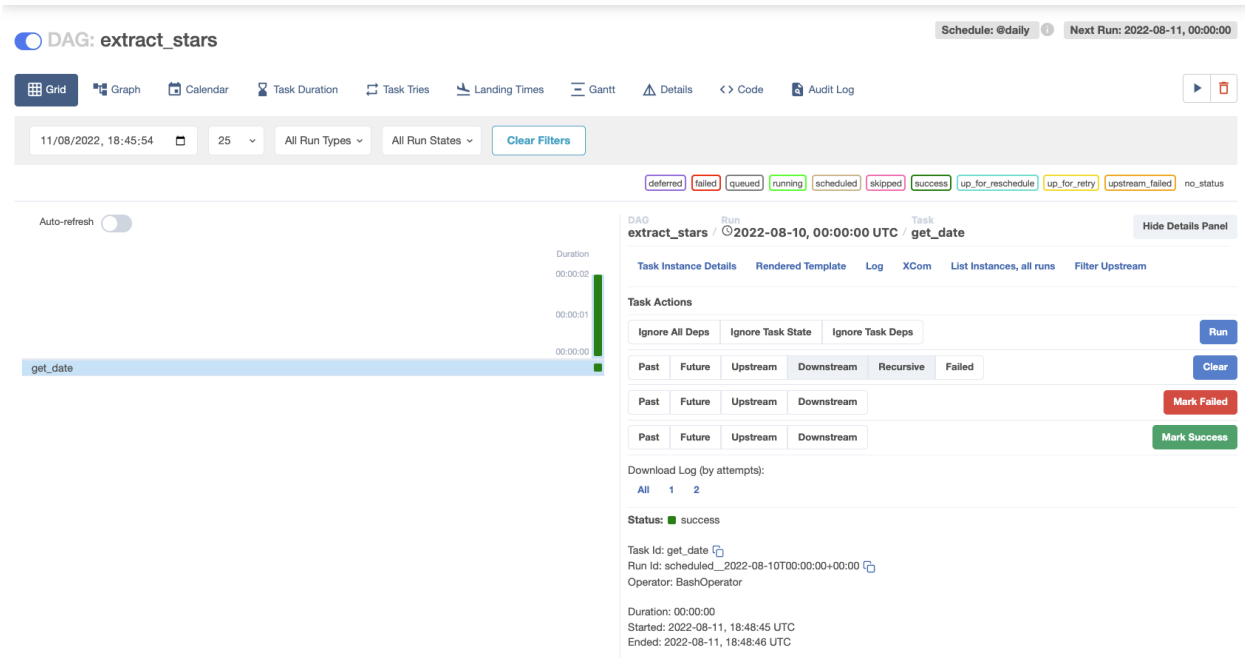
Data Interval:

Start: 2022-08-10, 00:00:00 UTC

End: 2022-08-11, 00:00:00 UTC

Click on the task (square) just under the DAG Run bar.

From there, you can interact with the task



Access the logs, details, XCOM, and more.

For now, click **Clear** to rerun that task only and **Confirm**

Great, now let's simulate an error.

Open the dag file `extract_stars.py` in your code editor.

In the `bash_command` parameter, set the value `exit 1` instead of `date`.

```
get_date = BashOperator(
    task_id="get_date",
    bash_command="exit 1"
)
```

Save the file, and rerun the task like you did just before.

As you can see, the task fails:

Airflow DAGs Security Browse Admin Docs 18:53 UTC AU

DAG: extract_stars Schedule: @daily Next Run: 2022-08-11, 00:00:00

Grid Graph Calendar Task Duration Task Tries Landing Times Gantt Details Code Audit Log

11/08/2022, 18:45:54 25 All Run Types All Run States Clear Filters

Auto-refresh

Duration 00:00:01 00:00:00 00:00:00

get_date

deferred failed queued running scheduled skipped success up_for_reschedule up_for_retry upstream_failed no_status

DAG extract_stars / 2022-08-10, 00:00:00 UTC / Task get_date Hide Details Panel

Task Instance Details Rendered Template Log XCom List Instances, all runs Filter Upstream

Task Actions

Ignore All Deps Ignore Task State Ignore Task Deps Run

Past Future Upstream Downstream Recursive Failed Clear

Past Future Upstream Downstream Mark Failed

Past Future Upstream Downstream Mark Success

Download Log (by attempts): All 1 2 3 4

Status: failed

Task Id: get_date
Run Id: scheduled__2022-08-10T00:00:00+00:00
Operator: BashOperator

Duration: 00:00:00
Started: 2022-08-11, 18:53:05 UTC
Ended: 2022-08-11, 18:53:06 UTC

If that happens, the first step is to look at the logs.

Click on **Log**

Airflow DAGs Security Browse Admin Docs 18:54 UTC AU

Task Instance Details <> Rendered Template Log XCom

Log by attempts 1 2 3 4 Jump To End Toggle Wrap Download

```

*** Reading local file: /usr/local/airflow/logs/dag_id=extract_stars/run_id=scheduled__2022-08-10T00:00:00+00:00/task_id=get_date/attempt=4.log
[2022-08-11, 18:53:05 UTC] {taskinstance.py:1159} INFO - Dependencies all met for <TaskInstance: extract_stars.get_date scheduled__2022-08-10T00:00:00+00:00 [queued]>
[2022-08-11, 18:53:05 UTC] {taskinstance.py:1159} INFO - Dependencies all met for <TaskInstance: extract_stars.get_date scheduled__2022-08-10T00:00:00+00:00 [queued]>
[2022-08-11, 18:53:05 UTC] {taskinstance.py:1356} INFO -

-----
[2022-08-11, 18:53:05 UTC] {taskinstance.py:1357} INFO - Starting attempt 4 of 4
[2022-08-11, 18:53:05 UTC] {taskinstance.py:1358} INFO -

-----
[2022-08-11, 18:53:05 UTC] {taskinstance.py:1377} INFO - Executing <Task(BashOperator): get_date> on 2022-08-10 00:00:00+00:00
[2022-08-11, 18:53:05 UTC] {standard_task_runner.py:52} INFO - Started process 1022 to run task
[2022-08-11, 18:53:05 UTC] {standard_task_runner.py:79} INFO - Running: ['airflow', 'tasks', 'run', 'extract_stars', 'get_date', 'scheduled__2022-08-10T00:00:00+00:00', '--job-id', '14', '--r
[2022-08-11, 18:53:05 UTC] {standard_task_runner.py:80} INFO - Job 14: Subtask get_date
[2022-08-11, 18:53:05 UTC] {logging_mixin.py:115} WARNING - /usr/local/lib/python3.9/site-packages/airflow/configuration.py:525 DeprecationWarning: The sqlalchemy_conn option in [core] has b
[2022-08-11, 18:53:06 UTC] {task_command.py:378} INFO - Running <TaskInstance: extract_stars.get_date scheduled__2022-08-10T00:00:00+00:00 [running]> on host 29e20c4214a2
[2022-08-11, 18:53:06 UTC] {taskinstance.py:1569} INFO - Exporting the following env vars:
AIRFLOW_CTX_DAG_OWNER=airflow
AIRFLOW_CTX_DAG_ID=extract_stars
AIRFLOW_CTX_TASK_ID=get_date
AIRFLOW_CTX_EXECUTION_DATE=2022-08-10T00:00:00+00:00
AIRFLOW_CTX_TRY_NUMBER=4
AIRFLOW_CTX_DAG_RUN_ID=scheduled__2022-08-10T00:00:00+00:00
[2022-08-11, 18:53:06 UTC] {subprocess.py:62} INFO - Tmp dir root location:
/tmp
[2022-08-11, 18:53:06 UTC] {subprocess.py:74} INFO - Running command: ['bash', '-c', 'exit 1']
[2022-08-11, 18:53:06 UTC] {subprocess.py:85} INFO - Output:
[2022-08-11, 18:53:06 UTC] {subprocess.py:96} INFO - Command exited with return code 1
[2022-08-11, 18:53:06 UTC] {taskinstance.py:1889} ERROR - Task failed with exception
Traceback (most recent call last):
  File "/usr/local/lib/python3.9/site-packages/airflow/operators/bash.py", line 194, in execute
    raise AirflowException(
airflow.exceptions.AirflowException: Bash command failed. The command returned a non-zero exit code 1.
[2022-08-11, 18:53:06 UTC] {taskinstance.py:1395} INFO - Marking task as FAILED. dag_id=extract_stars, task_id=get_date, execution_date=20220810T000000, start_date=20220811T185305, end_date=2
[2022-08-11, 18:53:06 UTC] {standard_task_runner.py:92} ERROR - Failed to execute job 14 for task get_date (Bash command failed. The command returned a non-zero exit code 1.; 1022)
[2022-08-11, 18:53:06 UTC] {local_task_job.py:156} INFO - Task exited with return code 1
[2022-08-11, 18:53:06 UTC] {local_task_job.py:273} INFO - 0 downstream tasks scheduled from follow-on schedule check

```

The logs help you to understand what's the issue with your task.

Once fixed in the code, you just have to rerun it.

Change the BashOperator to succeed by giving it a different command - `date` - and rerun it to see success.

```
get_date = BashOperator(  
    task_id="get_date",  
    bash_command="date"  
)
```

Checking the logs, you should see that the task output the current date

Well done! You are now able to schedule and monitor your DAG! 🕶️

Additional resources

Airflow UI: <https://airflow.apache.org/docs/apache-airflow/stable/ui.html>