

# Use variables to your DAG

Airflow Variables can be mixed with Templating. Templating is a super powerful feature to inject data at runtime. It is particularly useful when you process data based on dates. As you may want rerun past DAG Runs, templating allows you to inject DAG Run logical date at runtime otherwise you would always process the same chunk of data. Let's discover how to do it!

## Prerequisites

You should have create the `endpoint` variable and completed the previous activities.

The DAG `extract_stars.py` under the folder  `dags`  should look like that:

```
from airflow import DAG
from airflow.operators.bash import BashOperator
from airflow.providers.http.operators.http import SimpleHttpOperator

from datetime import datetime

with DAG('extract_stars', schedule_interval='@daily', start_date=datetime(2022, 1, 1), cat
chup=False) as dag:

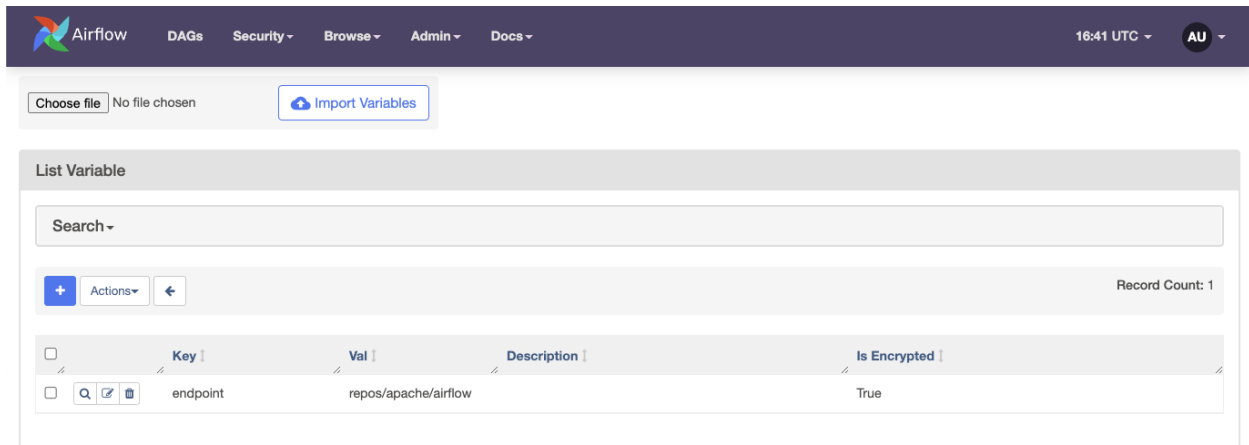
    get_date = BashOperator(
        task_id="get_date",
        bash_command="date"
    )

    query_github_stats = SimpleHttpOperator(
        task_id="query_github_stats",
        endpoint="repos/apache/airflow",
        method="GET",
        http_conn_id="github_api",
        log_response=True
    )
```

## Use a variable to your DAG

Go to `localhost:8080` , click `Admin` , `Variables`

Make sure you have the following variable



Go to your code editor and open `extract_stars.py` under the folder `dags`

In the `query_github_stats` task, change the `endpoint` value by:

```
{{ var.value.endpoint }}
```

The two pairs of curly brackets indicate a templated placeholder.

That means `var.value.endpoint` must be replaced by the corresponding value at runtime.

In this case `repos/apache/airflow`

That's it.

```
query_github_stats = SimpleHttpOperator(  
    task_id="query_github_stats",  
    endpoint="{{ var.value.endpoint }}",  
    method="GET",  
    http_conn_id="github_api",  
    log_response=True  
)
```

Save the file, go back on the Airflow UI and select the task from the Grid View.

The screenshot shows the Apache Airflow web interface for the DAG 'extract\_stars'. The top navigation bar includes links for DAGs, Security, Browse, Admin, and Docs. The right side shows the current time as 16:43 UTC and the user 'AU'. Below the navigation bar, the DAG name 'extract\_stars' is displayed along with its schedule '@daily' and the next run time '2022-08-16, 00:00:00'. A toolbar offers various views: Grid, Graph, Calendar, Task Duration, Task Tries, Landing Times, Gantt, Details, and Code. An 'Audit Log' button is also present. A filter bar allows selecting a specific run (16/08/2022, 16:43:17) and filtering by run type and state. A status bar shows various task states like deferred, failed, queued, running, scheduled, skipped, success, up\_for\_reschedule, up\_for\_retry, upstream\_failed, and no\_status. On the left, a Gantt chart shows the duration of tasks, with 'get\_date' and 'query\_github\_stats' visible. The main panel shows the 'Task Instance Details' for 'extract\_stars' at '2022-08-15, 00:00:00 UTC' for the task 'query\_github\_stats'. It includes tabs for 'Task Instance Details', 'Rendered Template', 'Log', 'XCom', 'List Instances, all runs', and 'Filter Upstream'. Under 'Task Actions', there are buttons for 'Ignore All Deps', 'Ignore Task State', 'Ignore Task Deps', 'Run', 'Clear', 'Mark Failed', and 'Mark Success'. A table shows task actions for 'Past', 'Future', 'Upstream', and 'Downstream' states.

Click **Rendered Template**

You should see the endpoint:

The screenshot shows the 'Rendered Template' view for the task 'query\_github\_stats' at '2022-08-15, 00:00:00'. The top navigation bar is the same as the previous screenshot. The main panel shows the 'Task Instance Details' for 'query\_github\_stats' at '2022-08-15, 00:00:00'. It includes tabs for 'Task Instance Details', 'Rendered Template', 'Log', and 'XCom'. The 'Rendered Template' section shows the endpoint 'repos/apache/airflow' and the data '1 {}' for the 'data' field. The 'headers' field also shows '1 {}'.

That view gives the rendered template. What you will end up with once the DAG runs and data got injected.

# Use the DAG Run date

The DAG pulls Github stars out of the Airflow repository. That number of stars changes everyday.

If you look at the task `get_date`:

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

We execute the bash command `date`

That means we always get the current date.

What if we want to rerun past DAG Runs?

In this case, past DAG Runs would run with the current date and not the dates *at which they got executed*.

To fix that, you have to use templating!

```
get_date = BashOperator(  
    task_id="get_date",  
    bash_command="echo {{ data_interval_start }}"  
)
```

Now, every time the DAG runs it uses the current date of the DAG Run (`data_interval_start`) and not the current date (`date`).

Don't forget to add `echo` just before.

Well done! You are able to use variables and mix them with templating! 😎

## Additional resources

Airflow template ref: <https://airflow.apache.org/docs/apache-airflow/stable/templates-ref.html>

Astronomer Template guide: <https://www.astronomer.io/guides/templating/>

