

Working environment: VMware workstation 12.1.1

Operating System: Ubuntu 18.04.1 LTS

Python version: Python 2.7

Airflow

Introduction

Use Airflow to author workflows as directed acyclic graphs (DAGs) of tasks. The Airflow scheduler executes your tasks on an array of workers while following the specified dependencies. Rich command lines utilities makes performing complex surgeries on DAGs a snap. The rich user interface makes it easy to visualize pipelines running in production, monitor progress and troubleshoot issues when needed.

Version

1.10.0

Installation

Airflow needs a home, ~/airflow is the default, but you can lay foundation somewhere else if you prefer

(optional)

```
export AIRFLOW_HOME=~/airflow
```

install from pypi using pip

```
pip install apache-airflow
```

start the web server, default port is 8080

```
airflow webserver -p 8080
```

start the scheduler. This can make the server run as daemon.

```
airflow scheduler
```

visit localhost:8080 in the browser and enable the example dag in the home page

run task instance

```
airflow run example_bash_operator runme_0 2015-01-01
```

Tutorial application

"""

Code that goes along with the Airflow Located at:

<http://airflow.readthedocs.org/en/latest/tutorial.html>

"""

```
from airflow import DAG
```

```
from airflow.operators.bash_operator import BashOperator
```

```
from datetime import datetime, timedelta
```

```

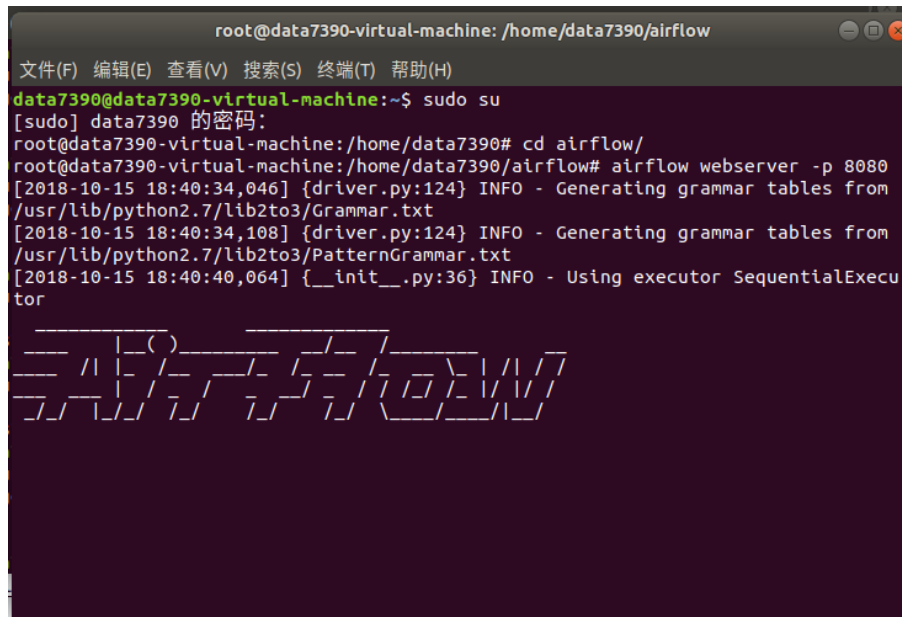
default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'start_date': datetime(2015, 6, 1),
    'email': ['airflow@example.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
    # 'queue': 'bash_queue',
    # 'pool': 'backfill',
    # 'priority_weight': 10,
    # 'end_date': datetime(2016, 1, 1),
}

dag = DAG(
    'tutorial', default_args=default_args, schedule_interval=timedelta(1))
# t1, t2 and t3 are examples of tasks created by instantiating operators
t1 = BashOperator(
    task_id='print_date',
    bash_command='date',
    dag=dag)
t2 = BashOperator(
    task_id='sleep',
    bash_command='sleep 5',
    retries=3,
    dag=dag)
templated_command = """
    {% for i in range(5) %}
        echo "{{ ds }}"
        echo "{{ macros.ds_add(ds, 7) }}"
        echo "{{ params.my_param }}"
    {% endfor %}
"""
t3 = BashOperator(
    task_id='templated',
    bash_command=templated_command,
    params={'my_param': 'Parameter I passed in'},
    dag=dag)
t2.set_upstream(t1)
t3.set_upstream(t1)

```

When I run start serve syntax, I need to get into root model, input
sudo su

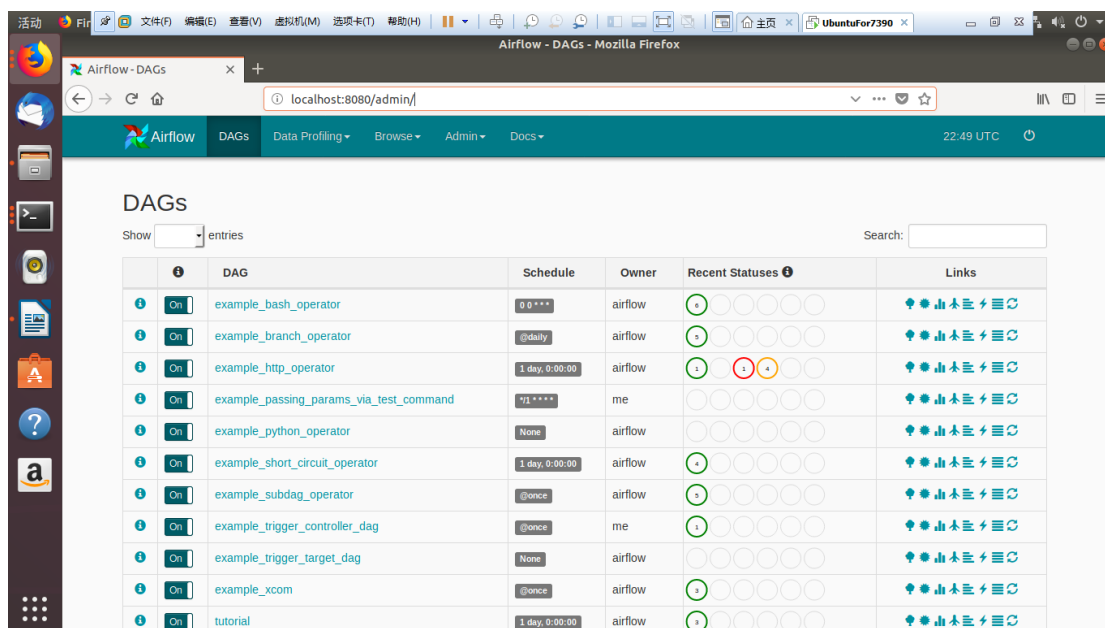
```
airflow# airflow webserver -p 8080
```



Then open your browser and input

<http://localhost:8080>

Airflow page will appear. Our tasks will show in different columns.



To run the tutorial script, we need to open a new terminal, find our task's path. Here I need to start root model. Input

```
sudo su
```

And then input

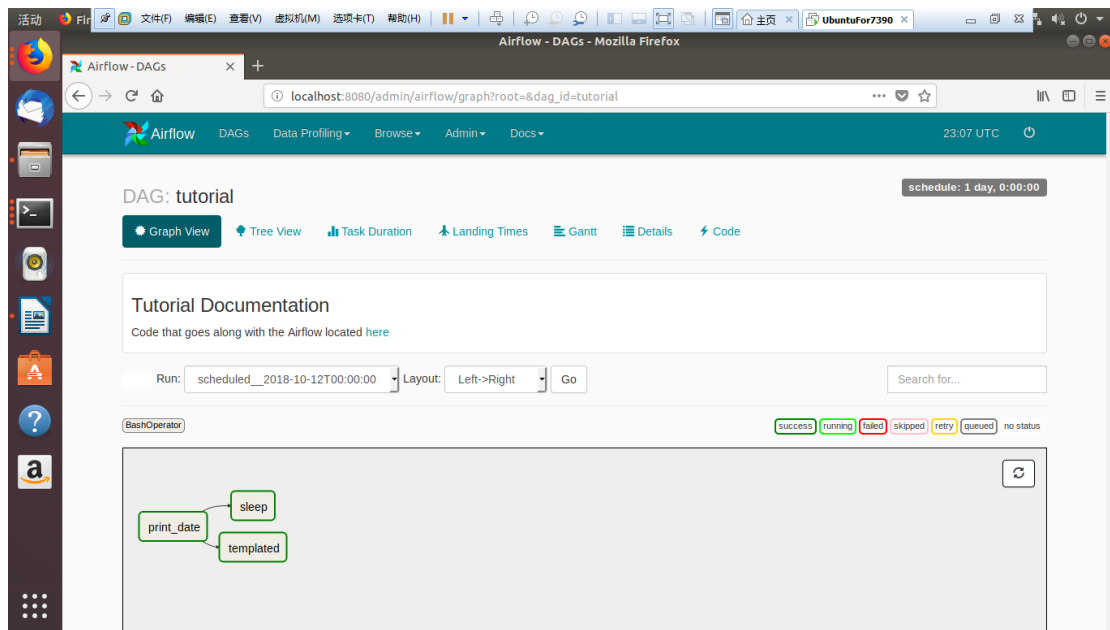
python tutorial.py

After that, we can test that task. Input

```
airflow test tutorial print_date 2018-10-12
```

Then refresh airflow page. Click tutorial DAG and we can get see tutorial task's running

record..



Celery

Introduction

Celery is an asynchronous task queue/job queue based on distributed message passing. It is focused on real-time operation, but supports scheduling as well.

Version

Celery 4.2

Choosing a Broker

Celery requires a solution to send and receive messages; usually this comes in the form of a separate service called a *message broker*. In my presentation, I chose RabbitMQ.

Installing RabbitMQ:

```
sudo apt-get install rabbitmq-server
```

Installing Celery:

```
pip install celery
```

Application:

Create the file tasks.py:

```
from celery import Celery
```

```
app = Celery('tasks', broker='pyamqp://guest@localhost/')
```

```
@app.task
```

```
def add(x, y):
```

```
    return x + y
```

Running the Celery worker server

```
celery -A tasks worker --loglevel=info
```

```
data7390@data7390-virtual-machine:~/celery$ celery -A tasks worker --loglevel=info
----- celery@data7390-virtual-machine v4.2.1 (windowlicker)
-----
-- * * * * *
-- * * * * * Linux-4.15.0-36-generic-x86_64-with-Ubuntu-18.04-bionic 2018-10-15 18:32:20
-- * * * * *
-- * * * * *
-- * * * * * [config]
-- * * * * * .> app: tasks:0x7f9c0e91f0d0
-- * * * * * .> transport: amqp://guest:**@localhost:5672//
-- * * * * * .> results: disabled://
-- * * * * * .> concurrency: 1 (prefork)
-- * * * * * .> task events: OFF (enable -E to monitor tasks in this worker)
-- * * * * *
-- * * * * * [queues]
-- * * * * * .> celery exchange=celery(direct) key=celery

[tasks]
. tasks.add

[2018-10-15 18:32:22,611: INFO/MainProcess] Connected to amqp://guest:**@127.0.0.1:5672//
[2018-10-15 18:32:23,328: INFO/MainProcess] mingle: searching for neighbors
[2018-10-15 18:32:38,530: INFO/MainProcess] mingle: all alone
[2018-10-15 18:32:38,644: INFO/MainProcess] celery@data7390-virtual-machine ready.
```

Calling the task:

After running celery server, you can open a new terminal and call that task:

```
>>> from tasks import add
```

```
>>> add.delay(4, 4)
```

Then, the result will be printed at the previous terminal.

```
data7390@data7390-virtual-machine: ~/celery
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H) How to Enable SSH in Ubuntu 16.04 LTS | UbuntuH

-----
-- * * * * -- Linux-4.15.0-36-generic-x86_64-with-Ubuntu-18.04-bionic 2018-10-15 18:32:20
-- * - * * * --
- ** ----- [config]
- ** ----- .> app: tasks:0x7f9c0e91f0d0
- ** ----- .> transport: amqp://guest:**@localhost:5672//
- ** ----- .> results: disabled://
- *** --- * --- .> concurrency: 1 (prefork)
-- ***** --- .> task events: OFF (enable -E to monitor tasks in this worker)
-- ***** ---

data7390@data7390-virtual-machine: ~/celery
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)

data7390@data7390-virtual-machine:~$ cd celery/
data7390@data7390-virtual-machine:~/celery$ python
Python 2.7.15rc1 (default, Apr 15 2018, 21:51:34)
[GCC 7.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> from tasks import add
>>> add.delay(4, 4)
<AsyncResult: 7e3f0f3e-8211-4c43-b052-4ef04027cc73>
>>>

[2018-10-15 18:35:35,062: INFO/MainProcess] Received task: tasks.add[7e3f0f3e-8211-4c43-b052-4ef04027cc73]
[2018-10-15 18:35:35,066: INFO/ForkPoolWorker-1] Task tasks.add[7e3f0f3e-8211-4c43-b052-4ef04027cc73] succeeded in 0.00170980700022s: 8
```

I zipped and uploaded my operation system in Google Drive. It contains all tools and configurations. The link is in rar.txt.

Reference

<https://airflow.apache.org/start.html>

<http://docs.celeryproject.org/en/latest/getting-started/first-steps-with-celery.html>