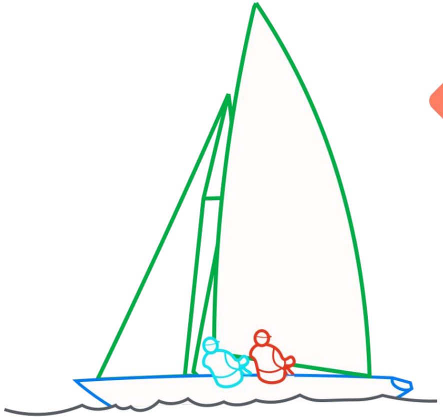
Mastering dependencies

The Airflow Way



Airflow Summit

Let's flow together

September 19-21, 2023,
Toronto, Canada

Hi!

Jarek Potiuk

- Independent Open-Source Contributor and advisor
- Airflow Summit Organizer
- PMC Member & Committer Apache Airflow
- Member of the Apache Software Foundation

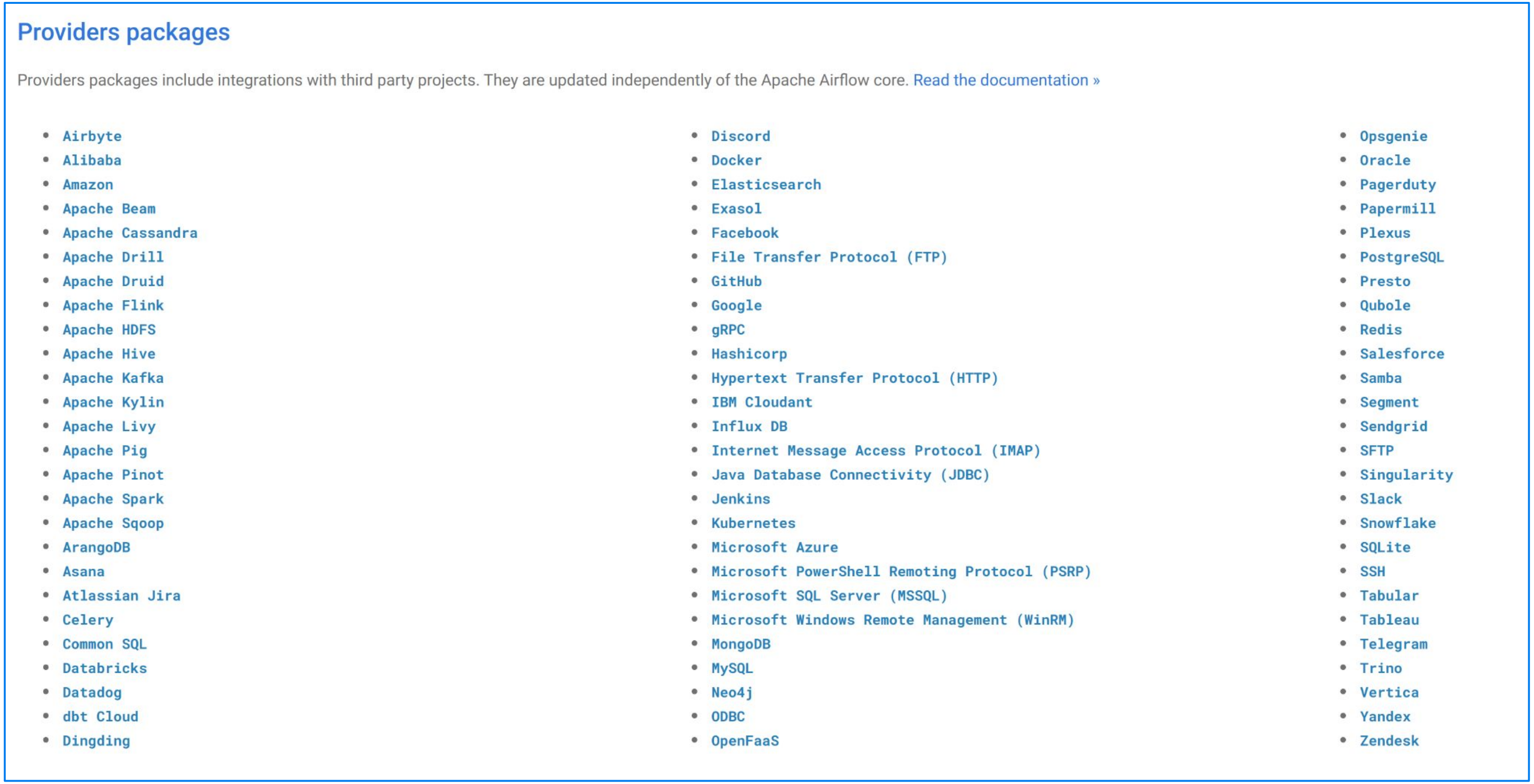
𝕏 @jarekpotiuk

@potiuk

in @jarekpotiuk

Dependency Problem

Airflow ~80(!) PyPI packages

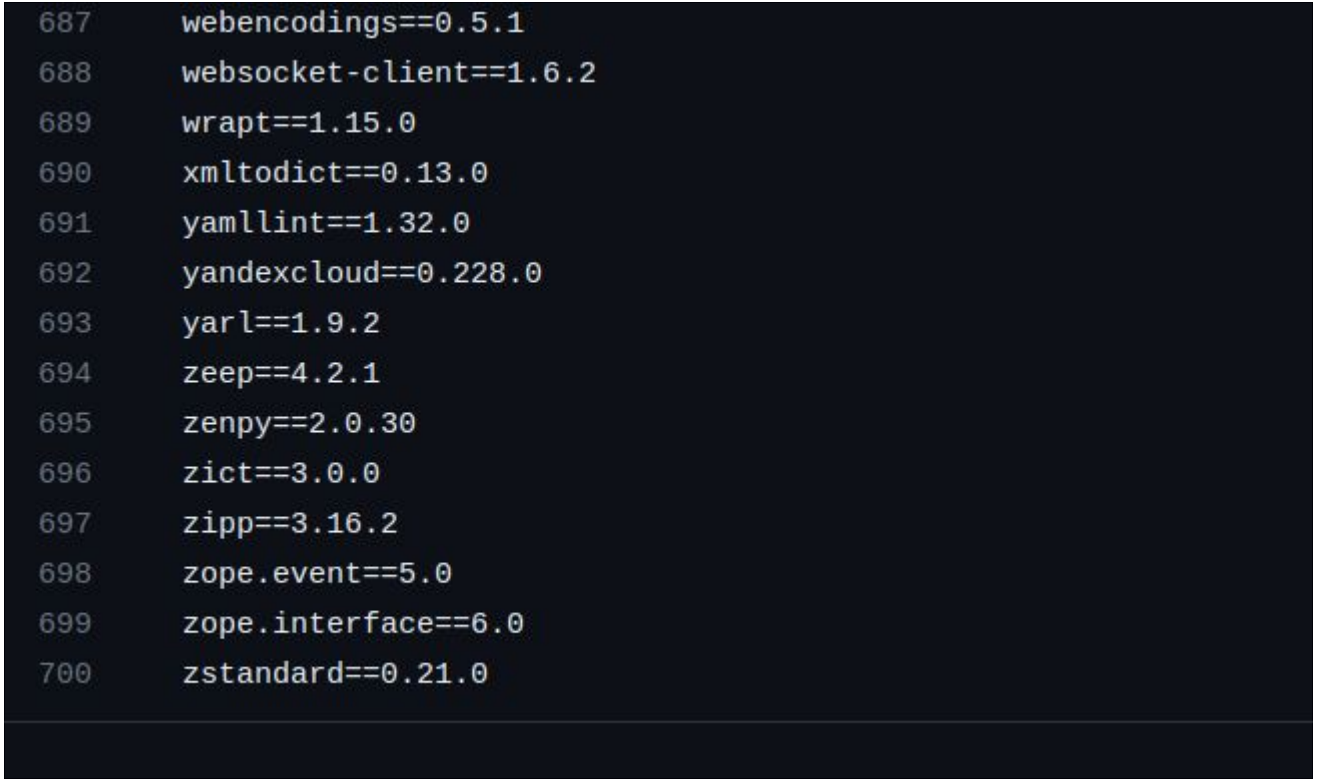


Providers packages

Providers packages include integrations with third party projects. They are updated independently of the Apache Airflow core. Read the documentation »

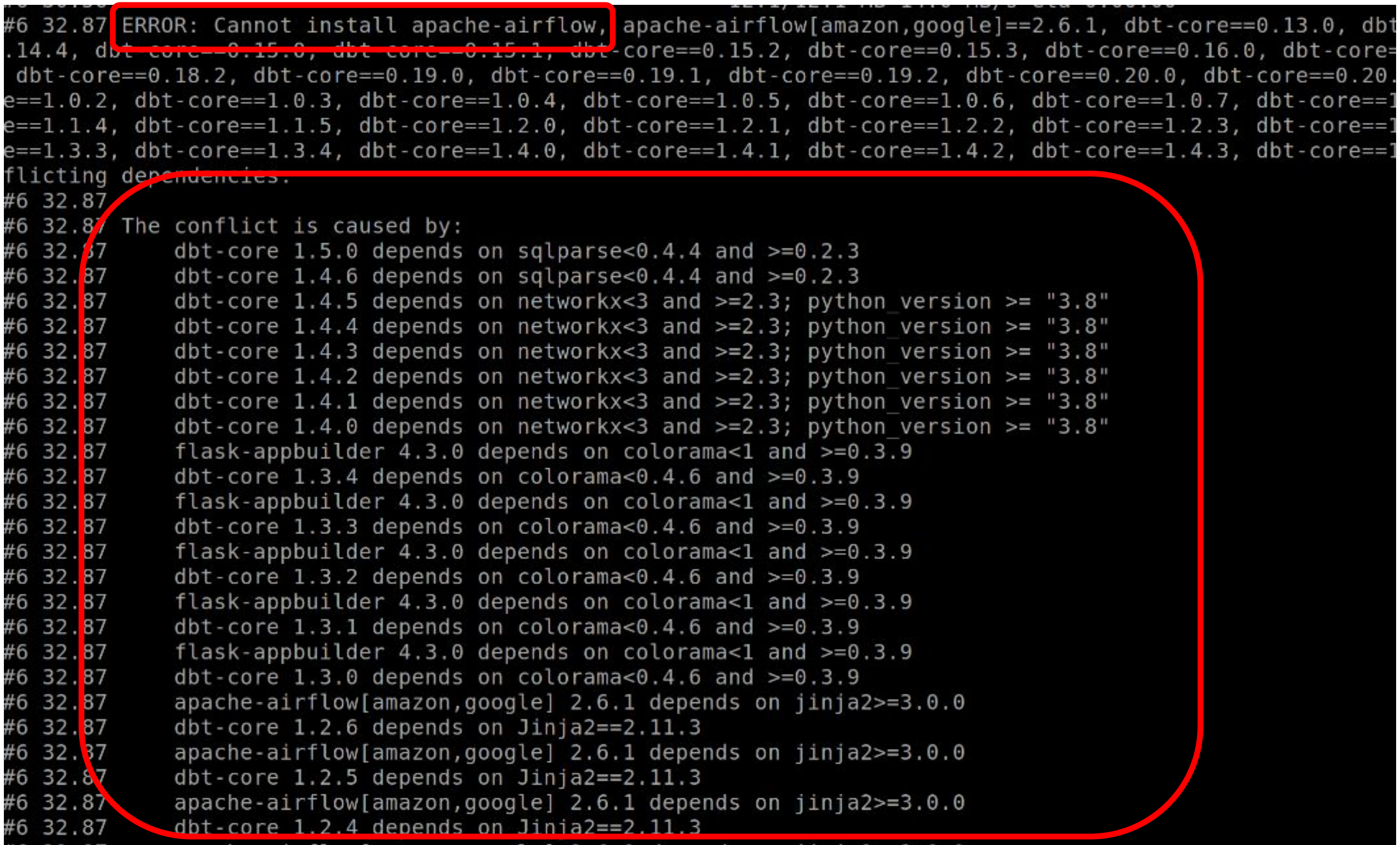
- Airbyte
- Alibaba
- Amazon
- Apache Beam
- Apache Cassandra
- Apache Drill
- Apache Druid
- Apache Flink
- Apache HDFS
- Apache Hive
- Apache Kafka
- Apache Kylin
- Apache Livy
- Apache Pig
- Apache Pinot
- Apache Spark
- Apache Sqoop
- ArangoDB
- Asana
- Atlassian Jira
- Celery
- Common SQL
- Databricks
- Datadog
- dbt Cloud
- Dingding
- Discord
- Docker
- Elasticsearch
- Exasol
- Facebook
- File Transfer Protocol (FTP)
- GitHub
- Google
- gRPC
- Hashicorp
- Hypertext Transfer Protocol (HTTP)
- IBM Cloudant
- Influx DB
- Internet Message Access Protocol (IMAP)
- Java Database Connectivity (JDBC)
- Jenkins
- Kubernetes
- Microsoft Azure
- Microsoft PowerShell Remoting Protocol (PSRP)
- Microsoft SQL Server (MSSQL)
- Microsoft Windows Remote Management (WinRM)
- MongoDB
- MySQL
- Neo4j
- ODBC
- OpenFaaS
- Opsgenie
- Oracle
- Pagerduty
- Papermill
- Plexus
- PostgreSQL
- Presto
- Qubole
- Redis
- Salesforce
- Samba
- Segment
- Sendgrid
- SFTP
- Singularity
- Slack
- Snowflake
- SQLite
- SSH
- Tabular
- Tableau
- Telegram
- Trino
- Vertica
- Yandex
- Zendesk

Airflow + Providers: 700 (!) dependencies



```
687    webencodings==0.5.1
688    websocket-client==1.6.2
689    wrapt==1.15.0
690    xmltodict==0.13.0
691    yamllint==1.32.0
692    yandexcloud==0.228.0
693    yarl==1.9.2
694    zeep==4.2.1
695    zenpy==2.0.30
696    zict==3.0.0
697    zipp==3.16.2
698    zope.event==5.0
699    zope.interface==6.0
700    zstandard==0.21.0
```

Installing problematic packages



```
#6 32.87 ERROR: Cannot install apache-airflow, apache-airflow[amazon,google]==2.6.1, dbt-core==0.13.0, dbt
.14.4, dbt-core==0.15.0, dbt-core==0.15.1, dbt-core==0.15.2, dbt-core==0.15.3, dbt-core==0.16.0, dbt-core=
 dbt-core==0.18.2, dbt-core==0.19.0, dbt-core==0.19.1, dbt-core==0.19.2, dbt-core==0.20.0, dbt-core==0.20.
e==1.0.2, dbt-core==1.0.3, dbt-core==1.0.4, dbt-core==1.0.5, dbt-core==1.0.6, dbt-core==1.0.7, dbt-core==1
e==1.1.4, dbt-core==1.1.5, dbt-core==1.2.0, dbt-core==1.2.1, dbt-core==1.2.2, dbt-core==1.2.3, dbt-core==1
e==1.3.3, dbt-core==1.3.4, dbt-core==1.4.0, dbt-core==1.4.1, dbt-core==1.4.2, dbt-core==1.4.3, dbt-core==1
flicting dependencies.
#6 32.87
#6 32.87 The conflict is caused by:
#6 32.87     dbt-core 1.5.0 depends on sqlparse<0.4.4 and >=0.2.3
#6 32.87     dbt-core 1.4.6 depends on sqlparse<0.4.4 and >=0.2.3
#6 32.87     dbt-core 1.4.5 depends on networkx<3 and >=2.3; python_version >= "3.8"
#6 32.87     dbt-core 1.4.4 depends on networkx<3 and >=2.3; python_version >= "3.8"
#6 32.87     dbt-core 1.4.3 depends on networkx<3 and >=2.3; python_version >= "3.8"
#6 32.87     dbt-core 1.4.2 depends on networkx<3 and >=2.3; python_version >= "3.8"
#6 32.87     dbt-core 1.4.1 depends on networkx<3 and >=2.3; python_version >= "3.8"
#6 32.87     dbt-core 1.4.0 depends on networkx<3 and >=2.3; python_version >= "3.8"
#6 32.87     flask-appbuilder 4.3.0 depends on colorama<1 and >=0.3.9
#6 32.87     dbt-core 1.3.4 depends on colorama<0.4.6 and >=0.3.9
#6 32.87     flask-appbuilder 4.3.0 depends on colorama<1 and >=0.3.9
#6 32.87     dbt-core 1.3.3 depends on colorama<0.4.6 and >=0.3.9
#6 32.87     flask-appbuilder 4.3.0 depends on colorama<1 and >=0.3.9
#6 32.87     dbt-core 1.3.2 depends on colorama<0.4.6 and >=0.3.9
#6 32.87     flask-appbuilder 4.3.0 depends on colorama<1 and >=0.3.9
#6 32.87     dbt-core 1.3.1 depends on colorama<0.4.6 and >=0.3.9
#6 32.87     flask-appbuilder 4.3.0 depends on colorama<1 and >=0.3.9
#6 32.87     dbt-core 1.3.0 depends on colorama<0.4.6 and >=0.3.9
#6 32.87     apache-airflow[amazon,google] 2.6.1 depends on jinja2>=3.0.0
#6 32.87     dbt-core 1.2.6 depends on Jinja2==2.11.3
#6 32.87     apache-airflow[amazon,google] 2.6.1 depends on jinja2>=3.0.0
#6 32.87     dbt-core 1.2.5 depends on Jinja2==2.11.3
#6 32.87     apache-airflow[amazon,google] 2.6.1 depends on jinja2>=3.0.0
#6 32.87     dbt-core 1.2.4 depends on Jinja2==2.11.3
```

Tooling

Dependency tools landscape

- pip
- pipx
- pipenv
- pip-tools
- poetry
- micropipenv
- tox
- conda
- bazel
- hatch
- flit
- ....

- setup.py
- setup.cfg
- requirements.txt
- Pipfile
- MANIFEST.in
- *.lock

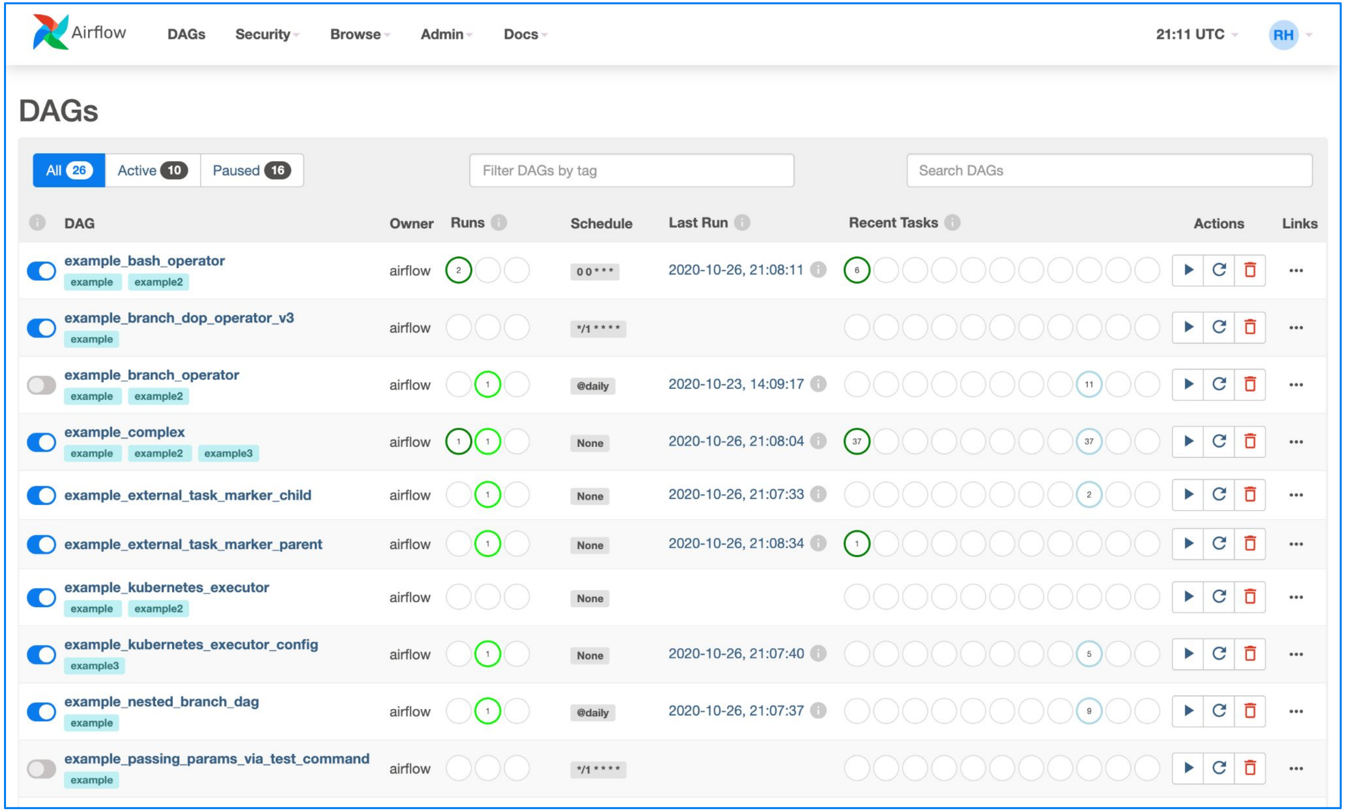
One tool to rule them all (for Airflow at least)

pip

Don’t use anything else
(you’ve been warned)

Application or library?

Apache Airflow is an application



DAGs

All 26 | Active 10 | Paused 16

Filter DAGs by tag

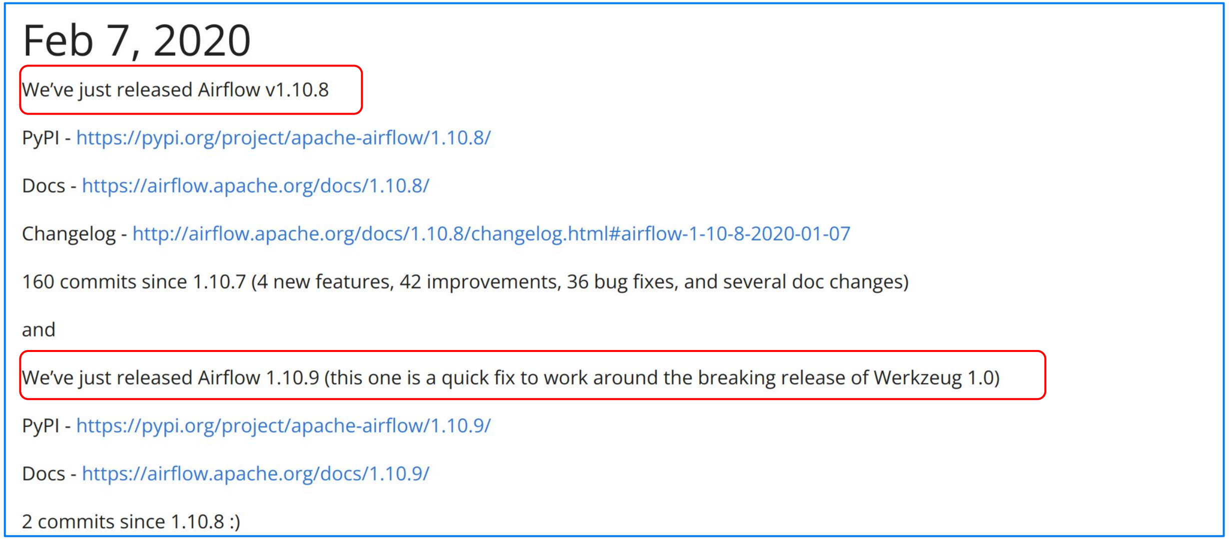
Search DAGs

DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links
example_bash_operator example example2	airflow	2	0 0 * * *	2020-10-26, 21:08:11	6	▶ ↻ 🗑	···
example_branch_dop_operator_v3 example	airflow		*/1 * * * *			▶ ↻ 🗑	···
example_branch_operator example example2	airflow	1	@daily	2020-10-23, 14:09:17	11	▶ ↻ 🗑	···
example_complex example example2 example3	airflow	1 1	None	2020-10-26, 21:08:04	37 37	▶ ↻ 🗑	···
example_external_task_marker_child	airflow	1	None	2020-10-26, 21:07:33	2	▶ ↻ 🗑	···
example_external_task_marker_parent	airflow	1	None	2020-10-26, 21:08:34	1	▶ ↻ 🗑	···
example_kubernetes_executor example example2	airflow		None			▶ ↻ 🗑	···
example_kubernetes_executor_config example3	airflow	1	None	2020-10-26, 21:07:40	5	▶ ↻ 🗑	···
example_nested_branch_dag example	airflow	1	@daily	2020-10-26, 21:07:37	9	▶ ↻ 🗑	···
example_passing_params_via_test_command example	airflow		*/1 * * * *			▶ ↻ 🗑	···

Reproducible installs

- Airflow installation should work the same tomorrow and a year from now

  -
    ```
    pip install apache-airflow[amazon, google,kerberos]==2.6.1 ...
    ```
- February 7th, 2020 - the date to remember



Feb 7, 2020

We've just released Airflow v1.10.8

PyPI - https://pypi.org/project/apache-airflow/1.10.8/

Docs - https://airflow.apache.org/docs/1.10.8/

Changelog - http://airflow.apache.org/docs/1.10.8/changelog.html#airflow-1-10-8-2020-01-07

160 commits since 1.10.7 (4 new features, 42 improvements, 36 bug fixes, and several doc changes)

and

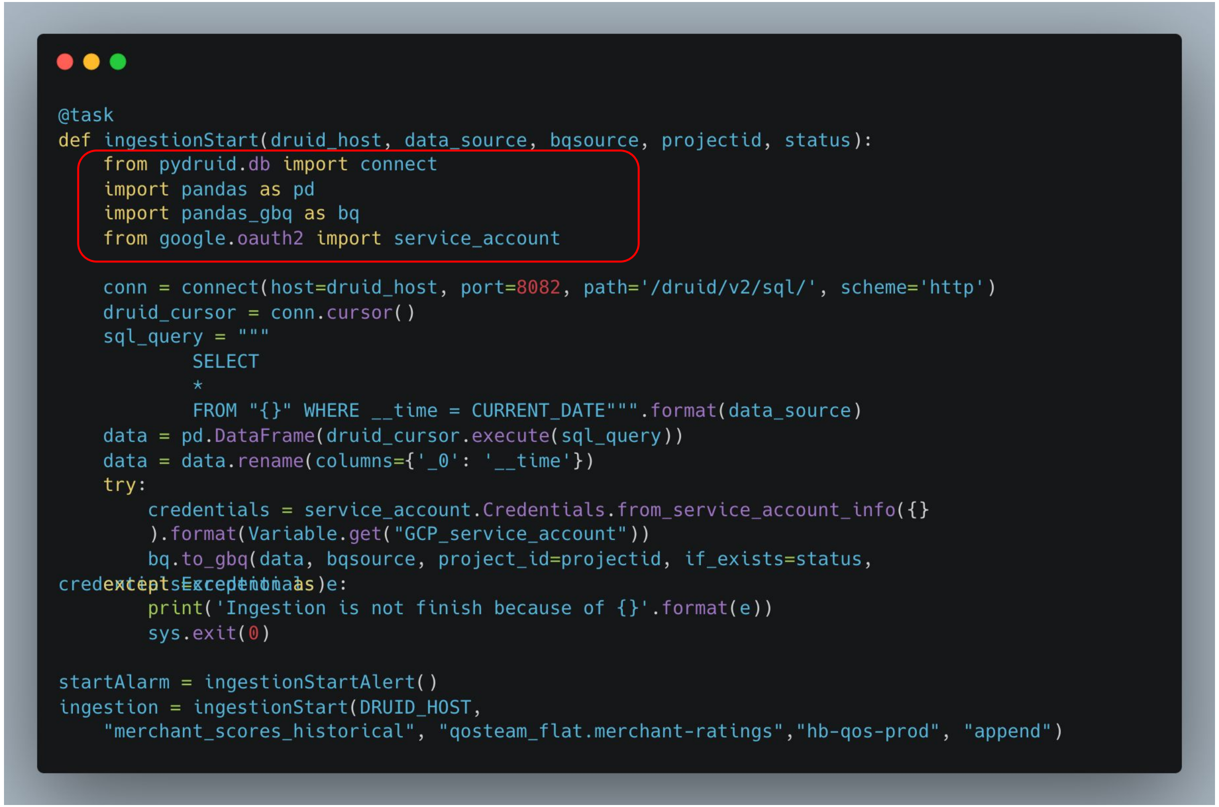
We've just released Airflow 1.10.9 (this one is a quick fix to work around the breaking release of Werkzeug 1.0)

PyPI - https://pypi.org/project/apache-airflow/1.10.9/

Docs - https://airflow.apache.org/docs/1.10.9/

2 commits since 1.10.8 :)

Airflow is a library too



```
@task
def ingestionStart(druid_host, data_source, bqsource, projectid, status):
    from pydruid.db import connect
    import pandas as pd
    import pandas_gbq as bq
    from google.oauth2 import service_account

    conn = connect(host=druid_host, port=8082, path='/druid/v2/sql/', scheme='http')
    druid_cursor = conn.cursor()
    sql_query = """
            SELECT
            *
            FROM "{}" WHERE __time = CURRENT_DATE""".format(data_source)
    data = pd.DataFrame(druid_cursor.execute(sql_query))
    data = data.rename(columns={'_0': '__time'})
    try:
        credentials = service_account.Credentials.from_service_account_info({}
        ).format(Variable.get("GCP_service_account"))
        bq.to_gbq(data, bqsource, project_id=projectid, if_exists=status,
credentials=credentials)
    except Exception as e:
        print('Ingestion is not finish because of {}'.format(e))
        sys.exit(0)

startAlarm = ingestionStartAlert()
ingestion = ingestionStart(DRUID_HOST,
    "merchant_scores_historical", "qosteam_flat.merchant-ratings","hb-qos-prod", "append")
```

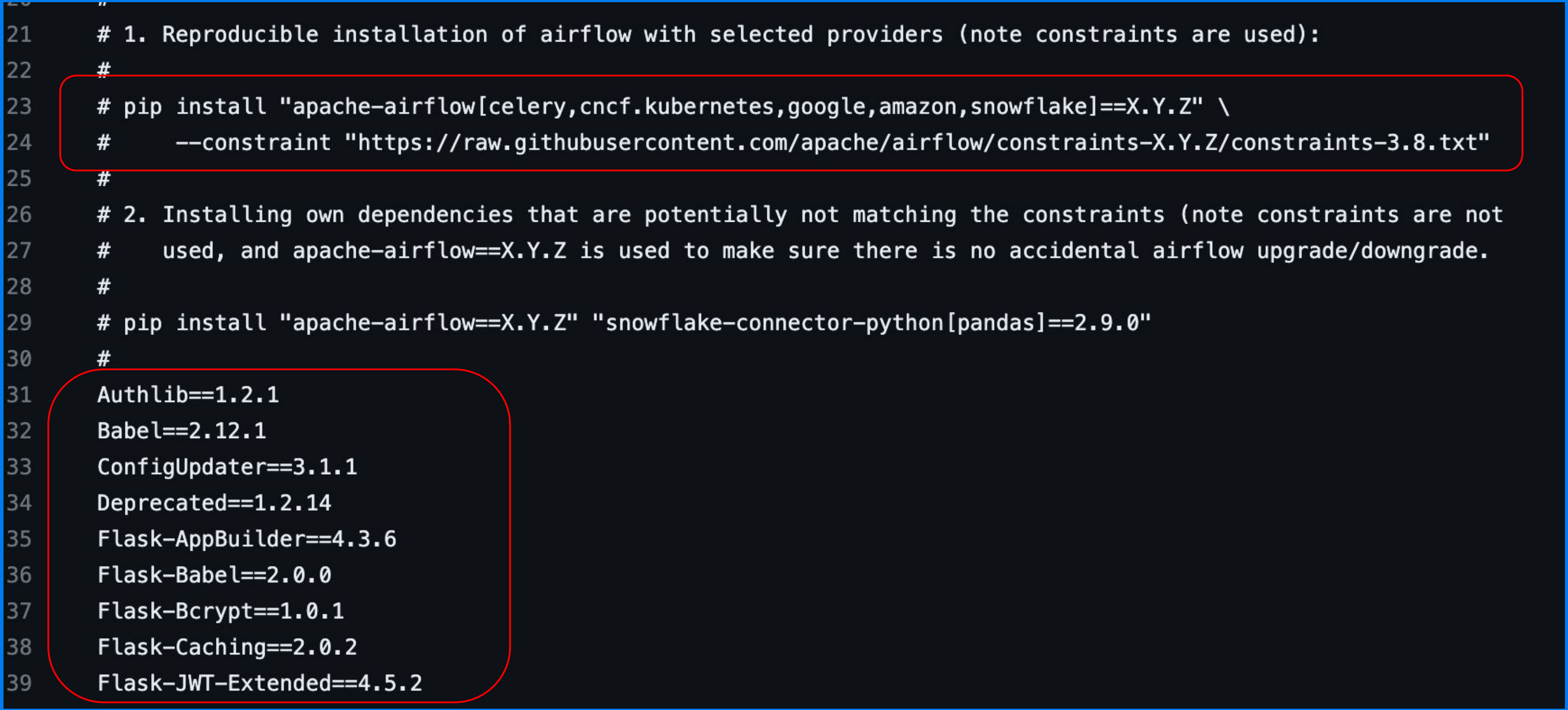
You can’t have cake and eat it too

Creator: Wallpaper Flare

Creator: ADRIANNAELKPHOTOGRAPHY

Constraints

Airflow constraints

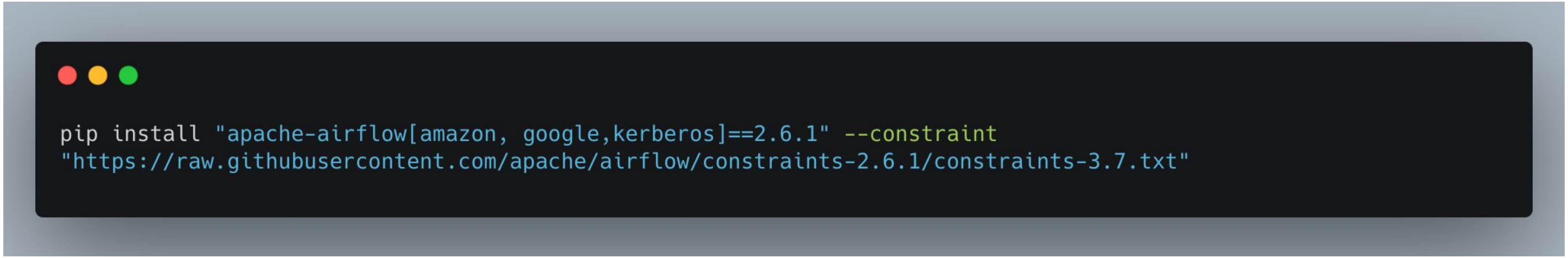


```
#
# 1. Reproducible installation of airflow with selected providers (note constraints are used):
#
# pip install "apache-airflow[celery,cncf.kubernetes,google,amazon,snowflake]==X.Y.Z" \
#     --constraint "https://raw.githubusercontent.com/apache/airflow/constraints-X.Y.Z/constraints-3.8.txt"
#
# 2. Installing own dependencies that are potentially not matching the constraints (note constraints are not
#    used, and apache-airflow==X.Y.Z is used to make sure there is no accidental airflow upgrade/downgrade.
#
# pip install "apache-airflow==X.Y.Z" "snowflake-connector-python[pandas]==2.9.0"
#
Authlib==1.2.1
Babel==2.12.1
ConfigUpdater==3.1.1
Deprecated==1.2.14
Flask-AppBuilder==4.3.6
Flask-Babel==2.0.0
Flask-Bcrypt==1.0.1
Flask-Caching==2.0.2
Flask-JWT-Extended==4.5.2
```

PIP constraints

- Constraints are NOT requirements
- Only supported by pip (so far)
- They allow to HAVE cake and EAT it too
  - Reproducible installation
  - AND ability to upgrade to different dependencies

Reproducible installation



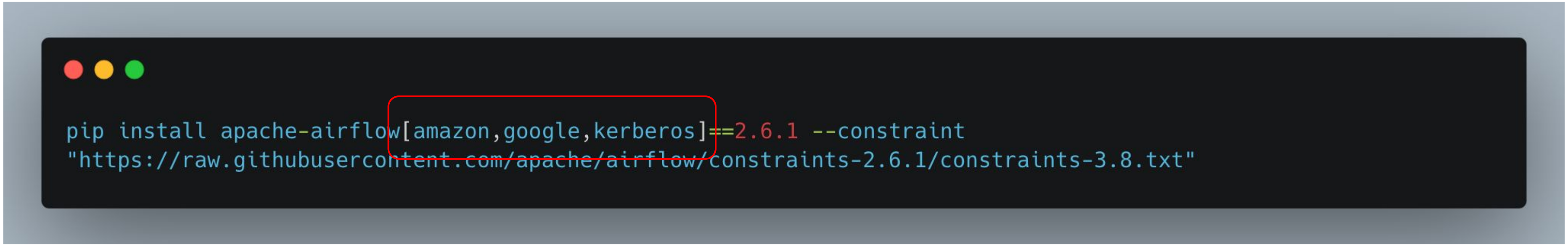
```
pip install "apache-airflow[amazon, google,kerberos]==2.6.1" --constraint
"https://raw.githubusercontent.com/apache/airflow/constraints-2.6.1/constraints-3.7.txt"
```

- Hosted on Github
- Separate constraint set per Airflow version / Python version
- Can be updated after release (in exceptional cases)

Installing Airflow + Providers

Installing Airflow from scratch (venv/containers)

- Primary use case for constraints
- Reproducible installation of Airflow in specific version (with Providers)
- Suitable for CI/CD pipeline
- DON’T install your own specific dependencies together

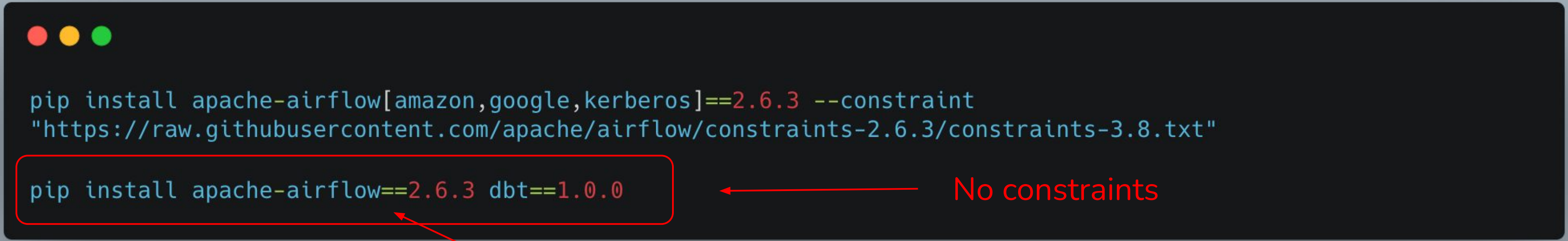


```
pip install apache-airflow[amazon,google,kerberos]==2.6.1 --constraint
"https://raw.githubusercontent.com/apache/airflow/constraints-2.6.1/constraints-3.8.txt"
```

Installing other dependencies

Adding new / update dependencies

- Separate step from installing Airflow
- DON’t use constraints
- It MAY upgrade or downgrade dependencies
- Use `**apache-airflow==<version>**` to keep it from accidental up/downgrade



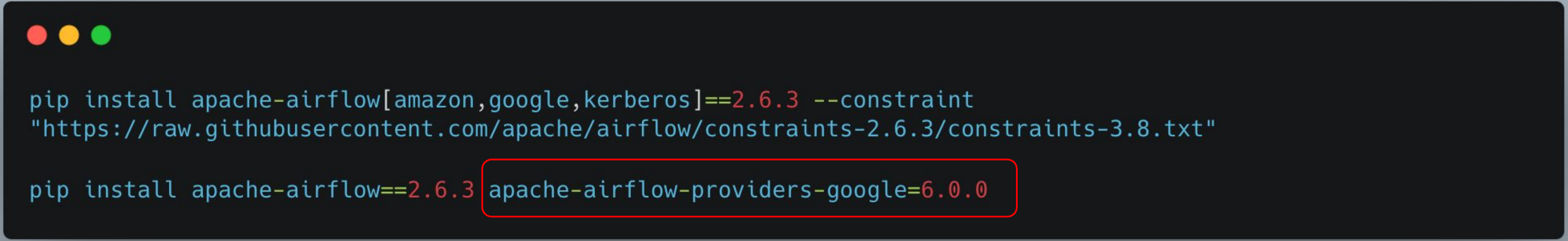
```
pip install apache-airflow[amazon,google,kerberos]==2.6.3 --constraint
"https://raw.githubusercontent.com/apache/airflow/constraints-2.6.3/constraints-3.8.txt"

pip install apache-airflow==2.6.3 dbt==1.0.0
```

No constraints

Keep your airflow version

Upgrading/downgrading providers?

- Same as other dependencies
- DON’t use constraints when you downgrade/upgrade providers

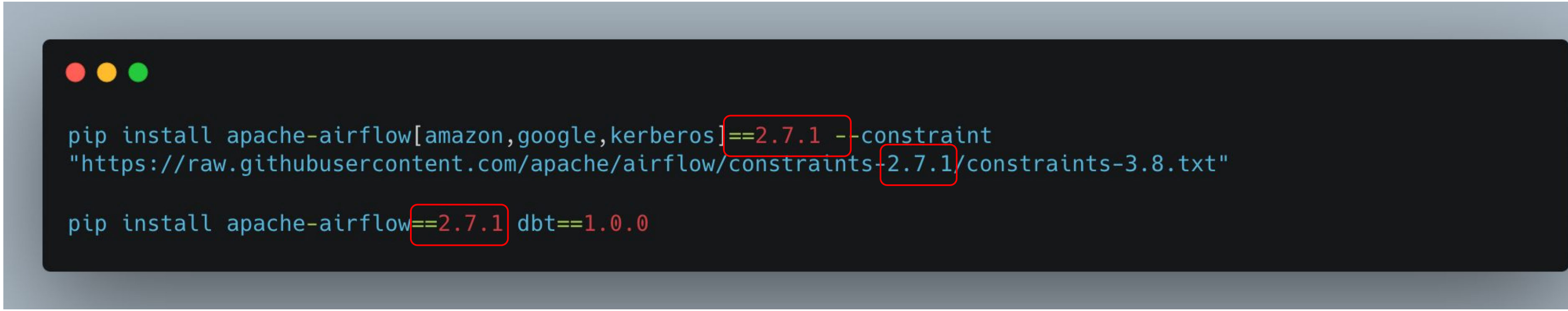


```
pip install apache-airflow[amazon,google,kerberos]==2.6.3 --constraint
"https://raw.githubusercontent.com/apache/airflow/constraints-2.6.3/constraints-3.8.txt"

pip install apache-airflow==2.6.3 apache-airflow-providers-google=6.0.0
```

Upgrading Airflow + Providers

Upgrading Airflow installation

- Handle full Upgrade scenarios
- Reproducible upgrade of Airflow WITH providers in specific version
- But you can add other dependencies after that

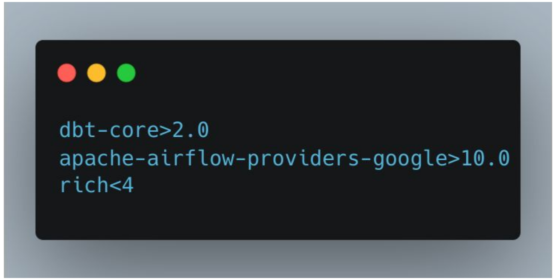


```
pip install apache-airflow[amazon,google,kerberos]==2.7.1 --constraint
"https://raw.githubusercontent.com/apache/airflow/constraints-2.7.1/constraints-3.8.txt"

pip install apache-airflow==2.7.1 dbt==1.0.0
```

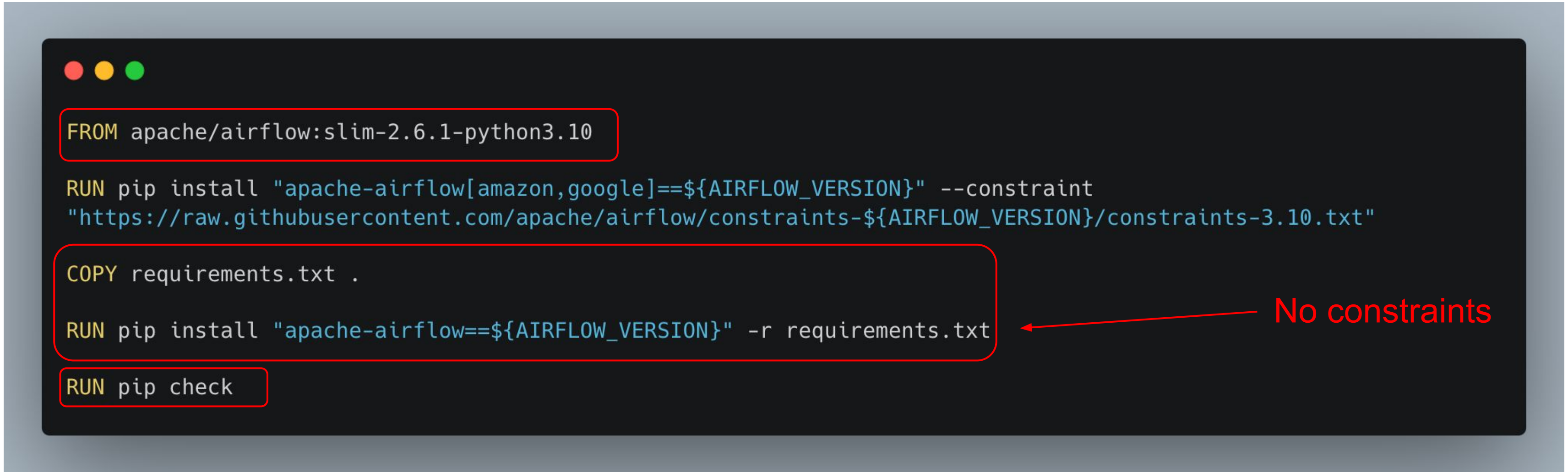
Custom
Docker image

Custom docker image

- slim image
- install providers
- add extra requirements



```
dbt-core>2.0
apache-airflow-providers-google>10.0
rich<4
```



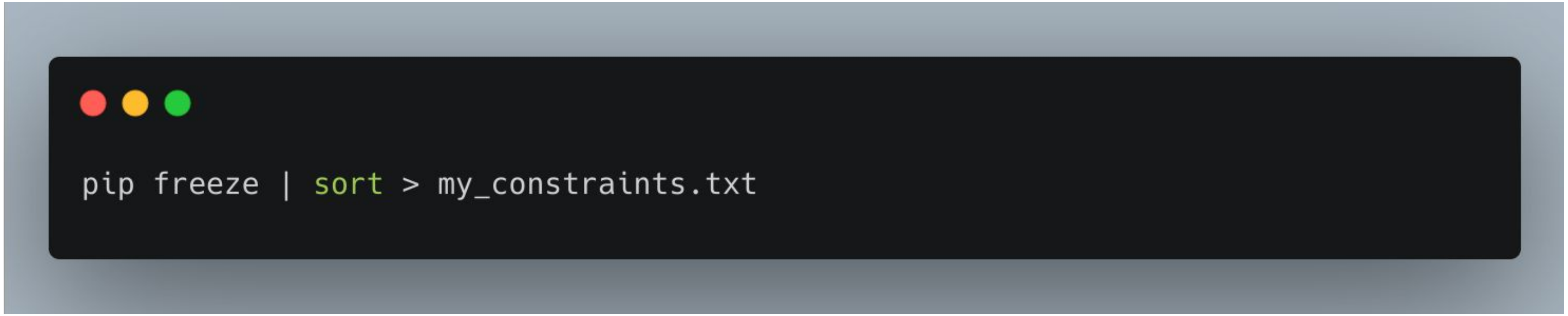
```
FROM apache/airflow:slim-2.6.1-python3.10

RUN pip install "apache-airflow[amazon,google]==${AIRFLOW_VERSION}" --constraint
"https://raw.githubusercontent.com/apache/airflow/constraints-${AIRFLOW_VERSION}/constraints-3.10.txt"

COPY requirements.txt .

RUN pip install "apache-airflow==${AIRFLOW_VERSION}" -r requirements.txt

RUN pip check
```

No constraints

Using your own constraints

How to build your own constraints

- Install airflow + dependencies you need
- Run `pip check`
- and ...



```
pip freeze | sort > my_constraints.txt
```

When all else fails

Using different Python interpreters

- Launch new interpreter with different dependencies
- Choices:
  - **PythonVirtualenvOperator** - virtual environment created on the flight
  - **ExternalPythonOperator** - virtual environment pre-created (in the image for example)

Using Docker And Kubernetes Operators

- When conflict are at system dependencies level
- Cannot pass Python objects
- Might be able to use Airflow Public Interface (Same Airflow Version)
- Provides nice isolation

Q&A

Jarek Potiuk

Independent Open-Source Contributor and advisor

Airflow Summit Organizer

PMC Member & Committer Apache Airflow

Member of the Apache Software Foundation

@jarekpotiuk

@potiuk

@jarekpotiuk

Q&A

Mastering Dependencies: The Airflow Way