Airflow That Remembers: The Dag Versioning Era is here!

ASTRONOMER

Jed Cunningham

Principal Software Engineer @ astronomer.io

Committer

PMC Member

ASTRONOMER

Ephraim Anierobi

Senior Software Engineer @ astronomer.io

Committer

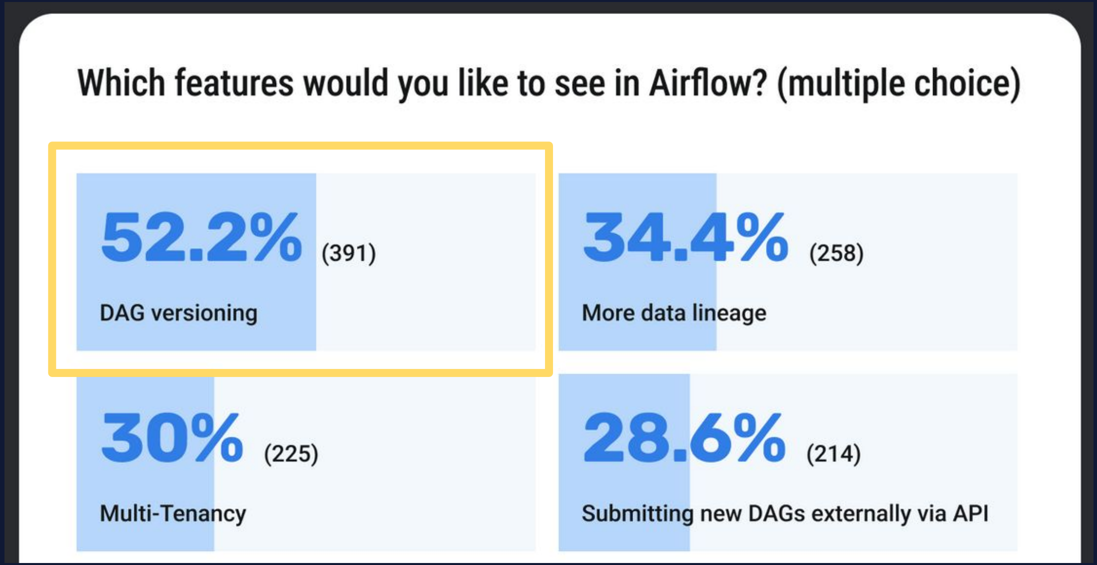
PMC Member

ASTRONOMER

Slido for questions

OR

Join link:
https://app.sli.do/event/tFD2BRhuH4jT3rD2nBwntg



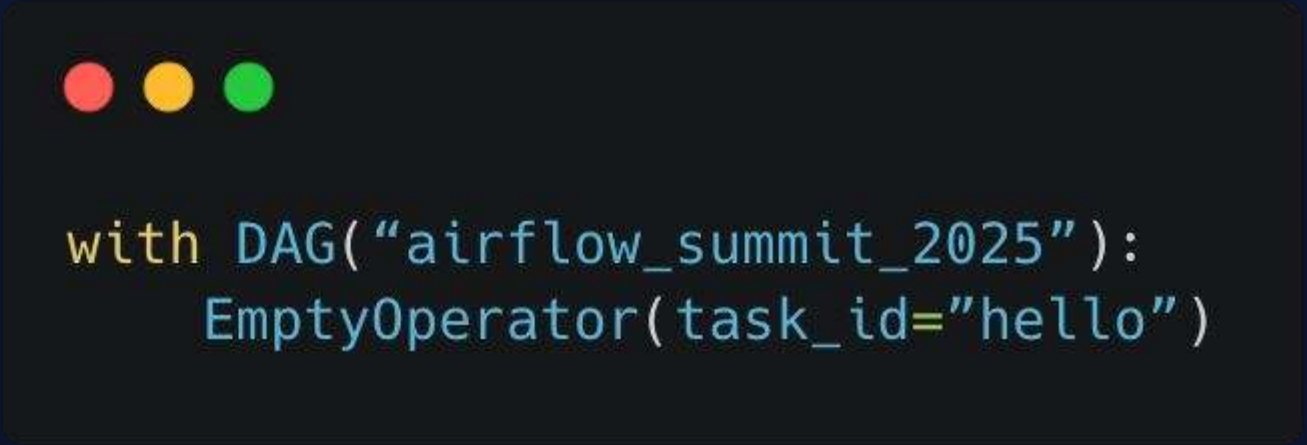
Which features would you like to see in Airflow? (multiple choice)

52.2% (391)
DAG versioning

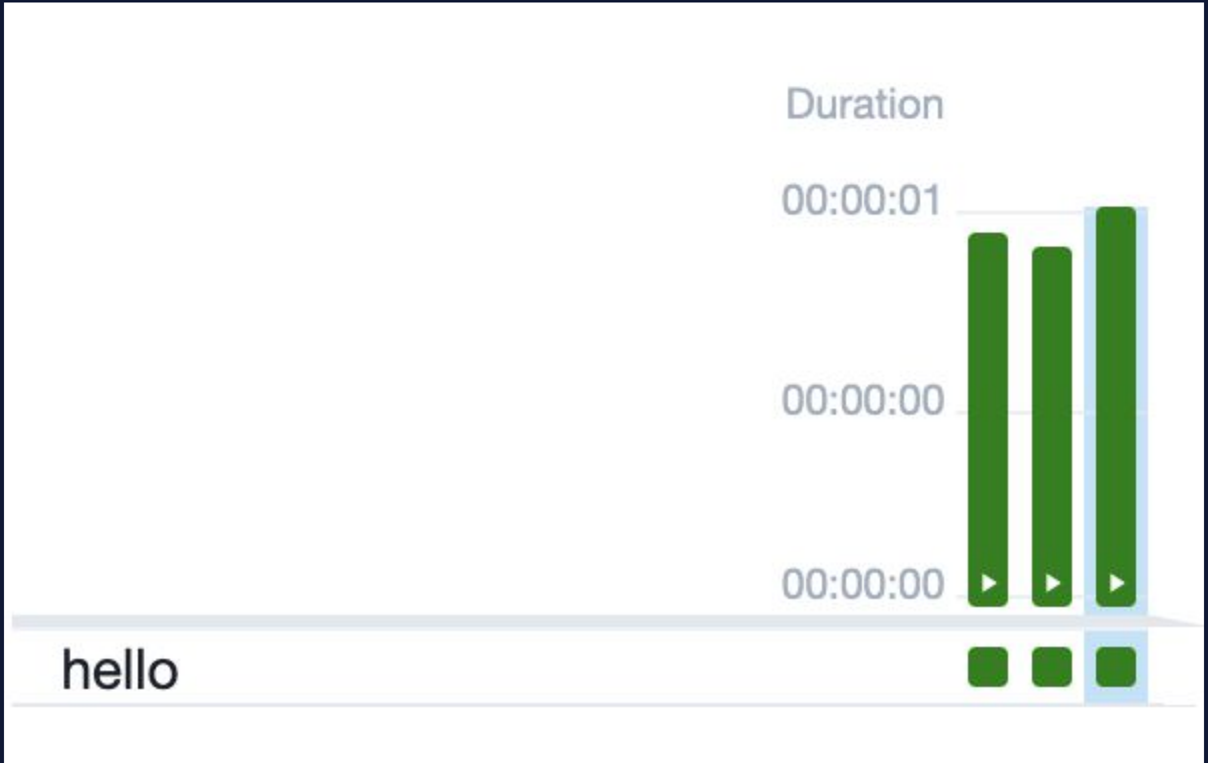
34.4% (258)
More data lineage

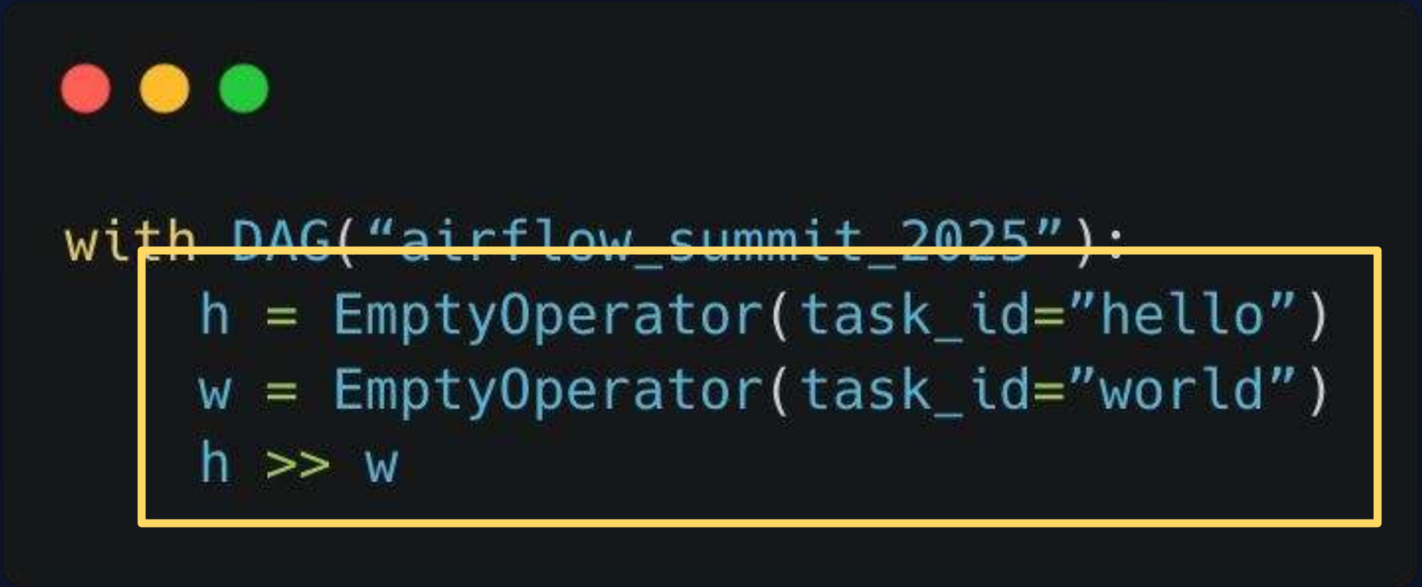
30% (225)
Multi-Tenancy

28.6% (214)
Submitting new DAGs externally via API

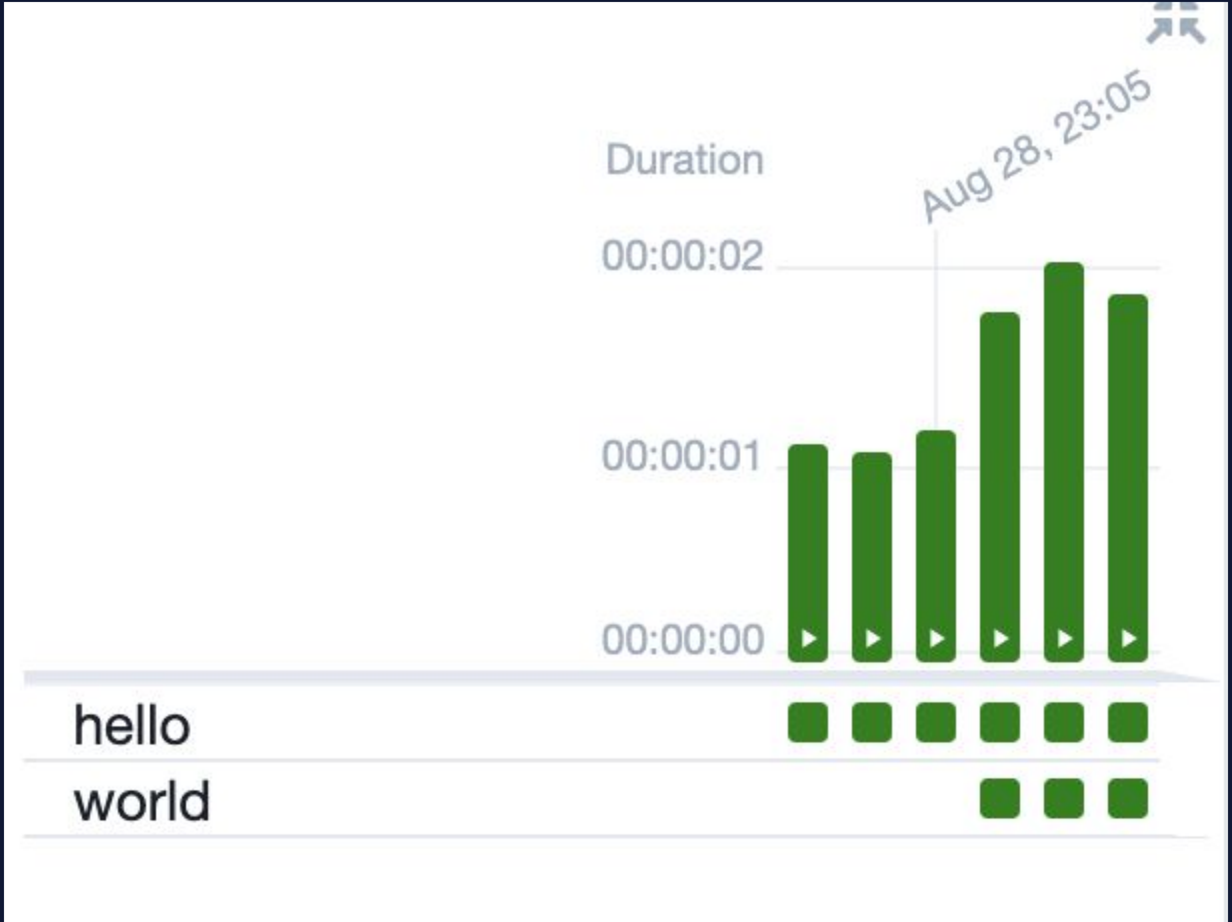


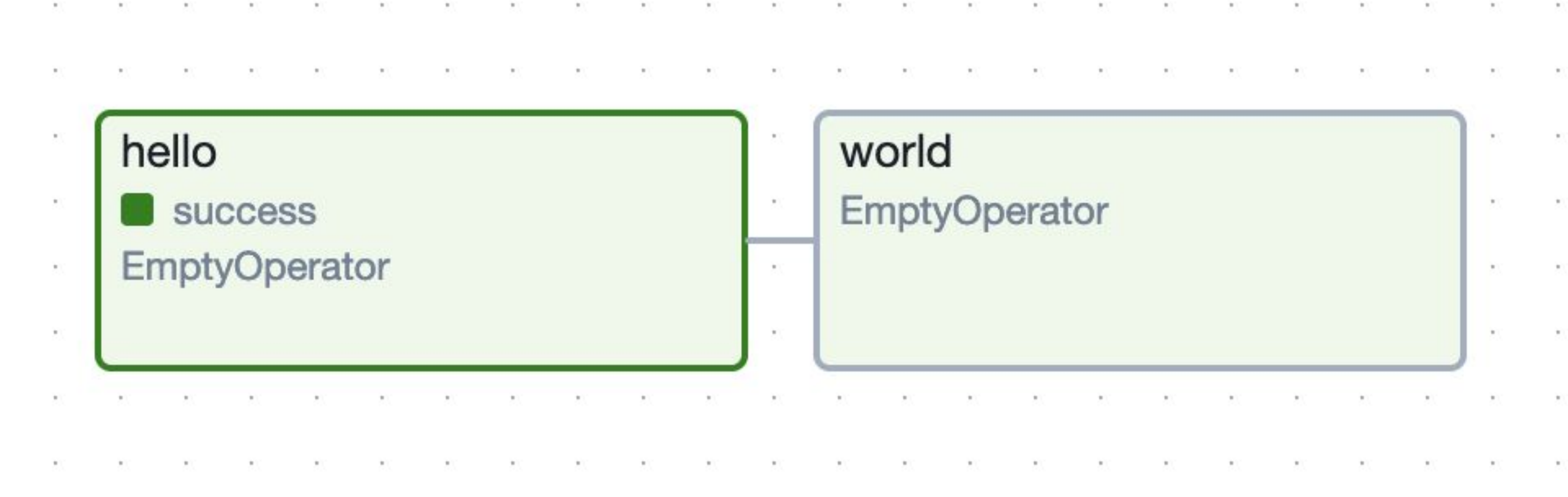
```
with DAG(“airflow_summit_2025”):
    EmptyOperator(task_id=”hello”)
```

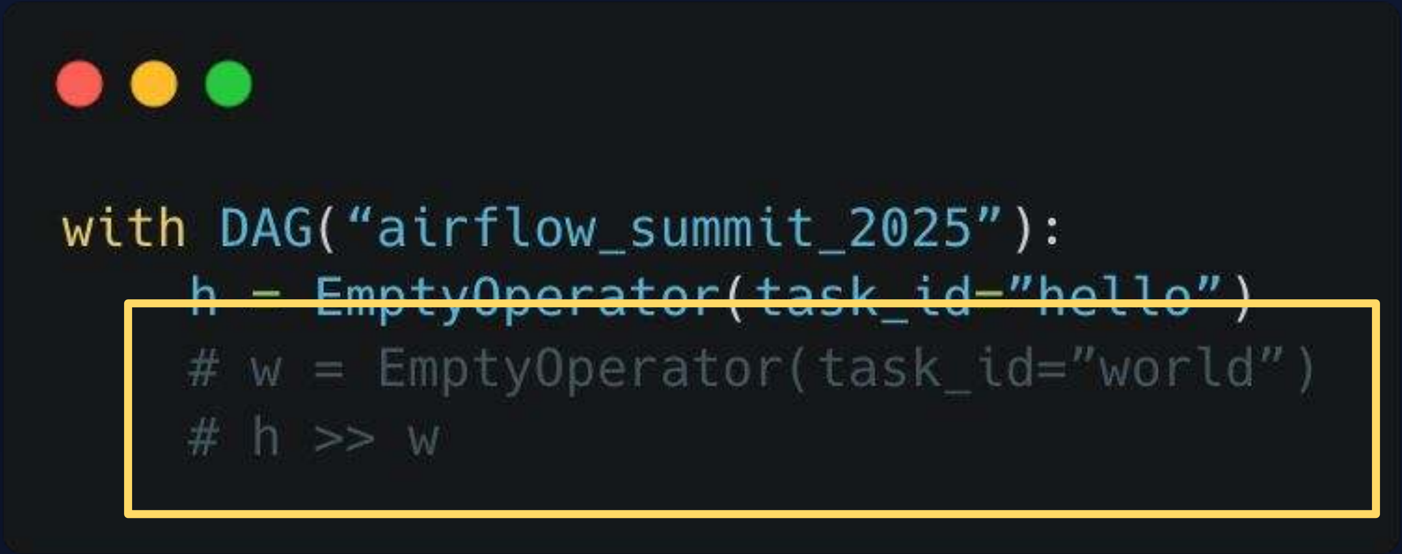

Duration
00:00:01
00:00:00
00:00:00
hello



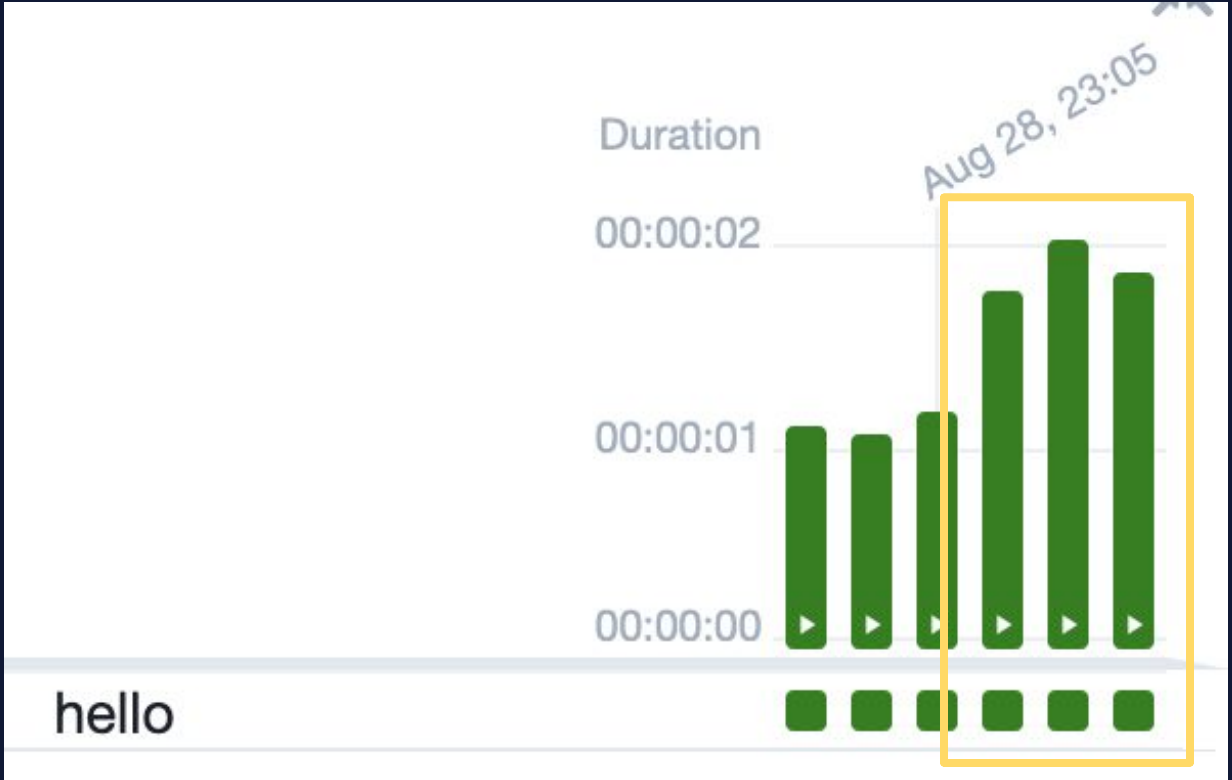
```
with DAG(“airflow_summit_2025”):
    h = EmptyOperator(task_id=”hello”)
    w = EmptyOperator(task_id=”world”)
    h >> w
```

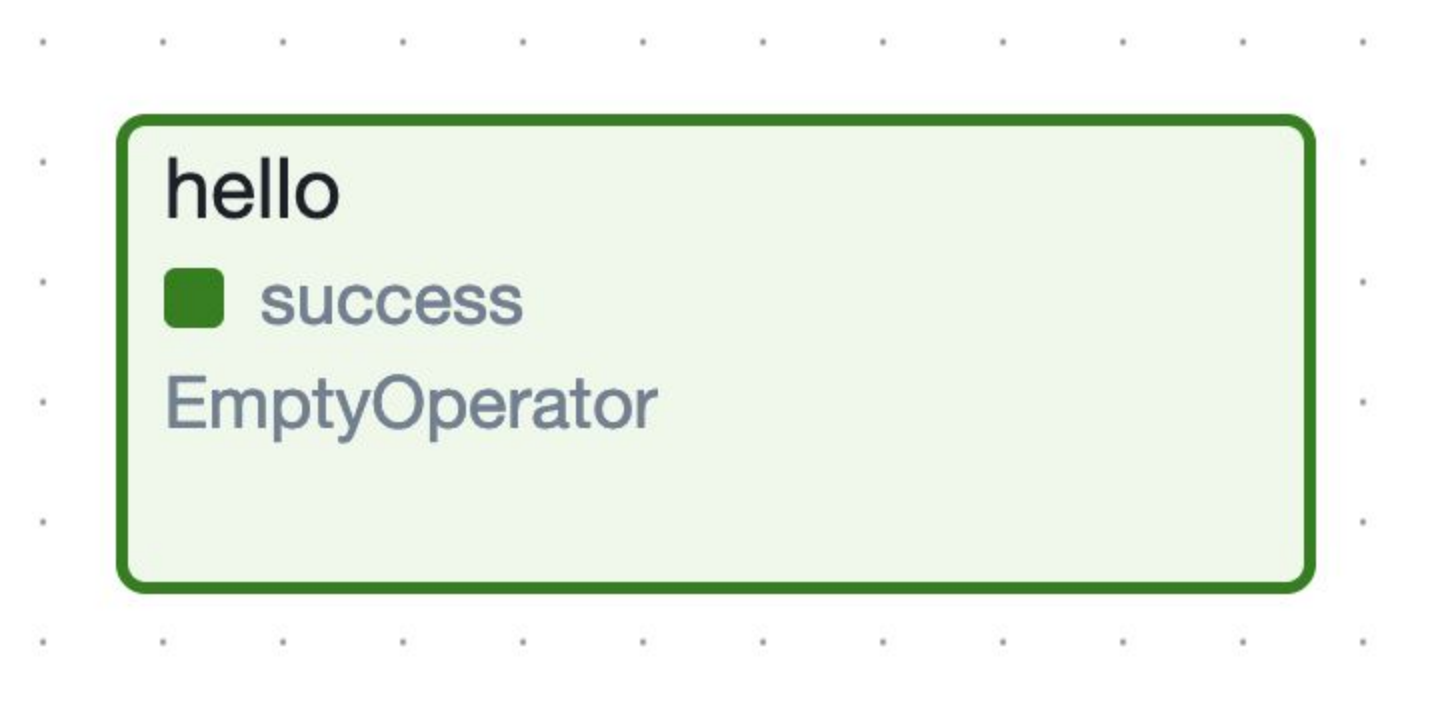

Duration
Aug 28, 23:05
00:00:02
00:00:01
00:00:00
hello
world


hello
🟩 success
EmptyOperator
world
EmptyOperator



```
with DAG(“airflow_summit_2025”):
    h = EmptyOperator(task_id=”hello”)
    # w = EmptyOperator(task_id=”world”)
    # h >> w
```

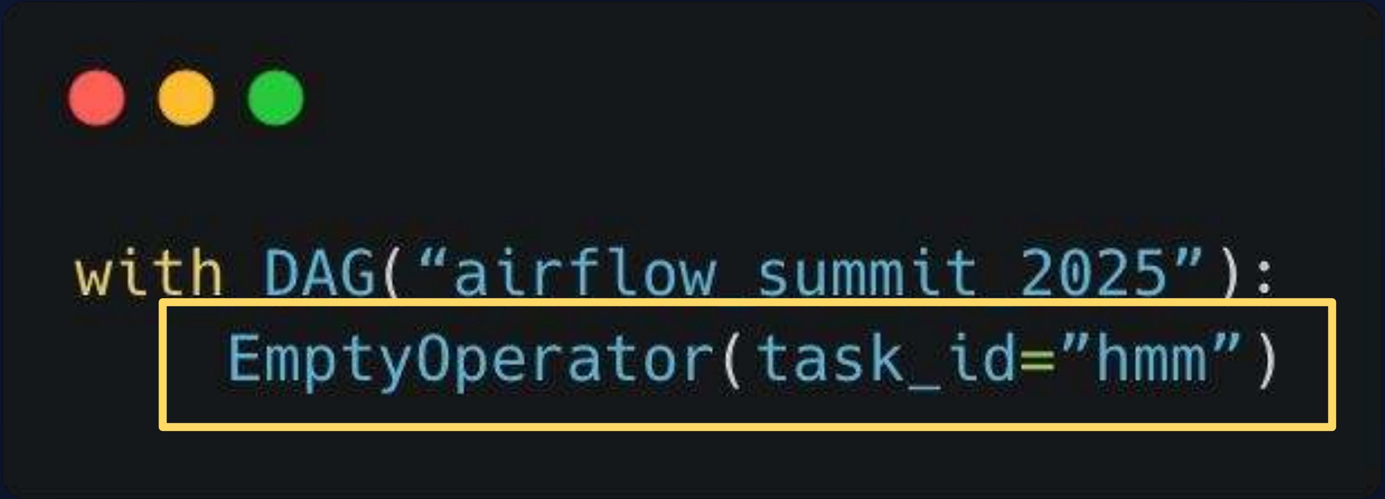

Duration
Aug 28, 23:05
00:00:02
00:00:01
00:00:00
hello



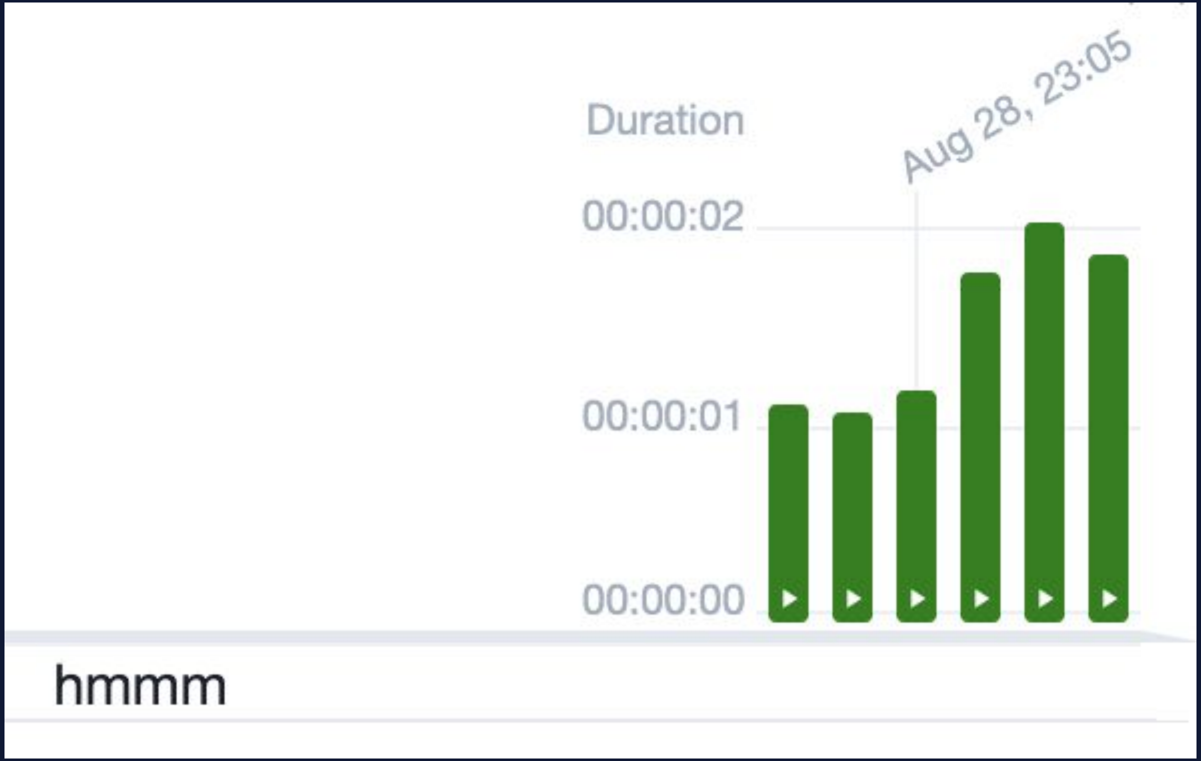
hello

🟩 success

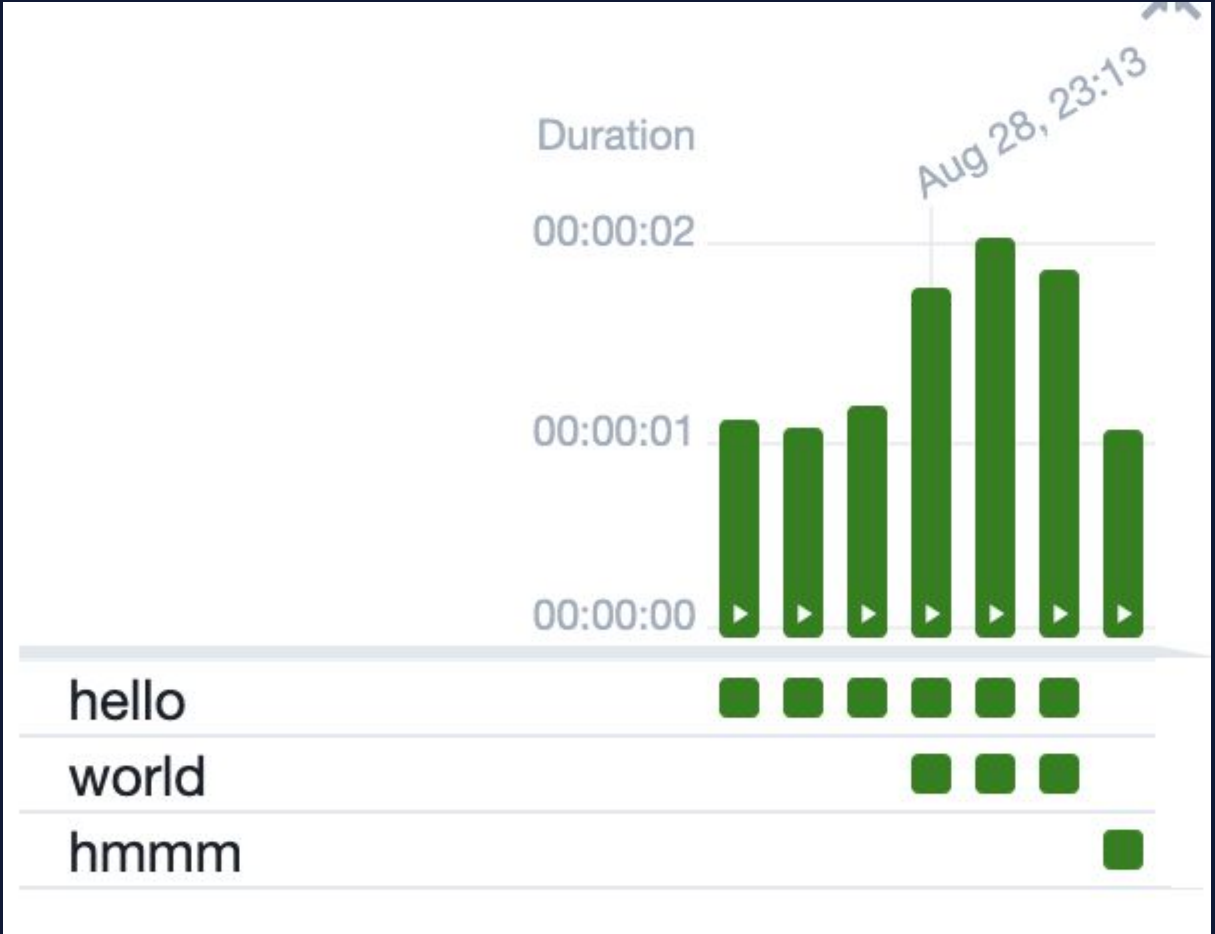
EmptyOperator

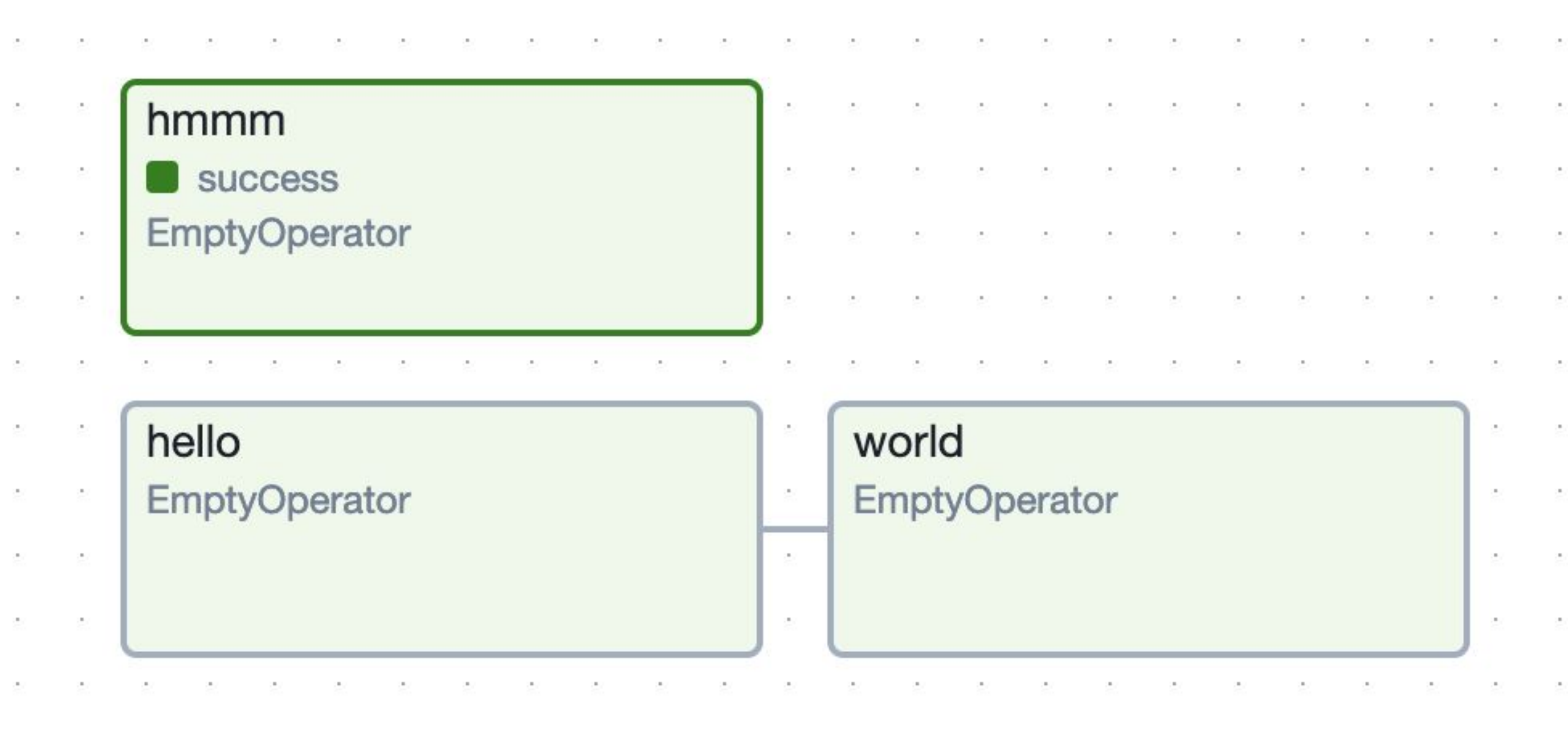


```
with DAG(“airflow summit 2025”):
    EmptyOperator(task_id=”hmm”)
```



hmmm


Aug 28, 23:13
Duration
00:00:02
00:00:01
00:00:00
hello
world
hmmm


hmmm
🟩 success
EmptyOperator
hello
EmptyOperator
world
EmptyOperator

Dag Versioning

Execution
(requires config)

UI
(automatic)

Dag Bundle

collection of Dags and other files

Types of Dag Bundles

Versioned vs not

ASTRONOMER

Run task using a specific version of

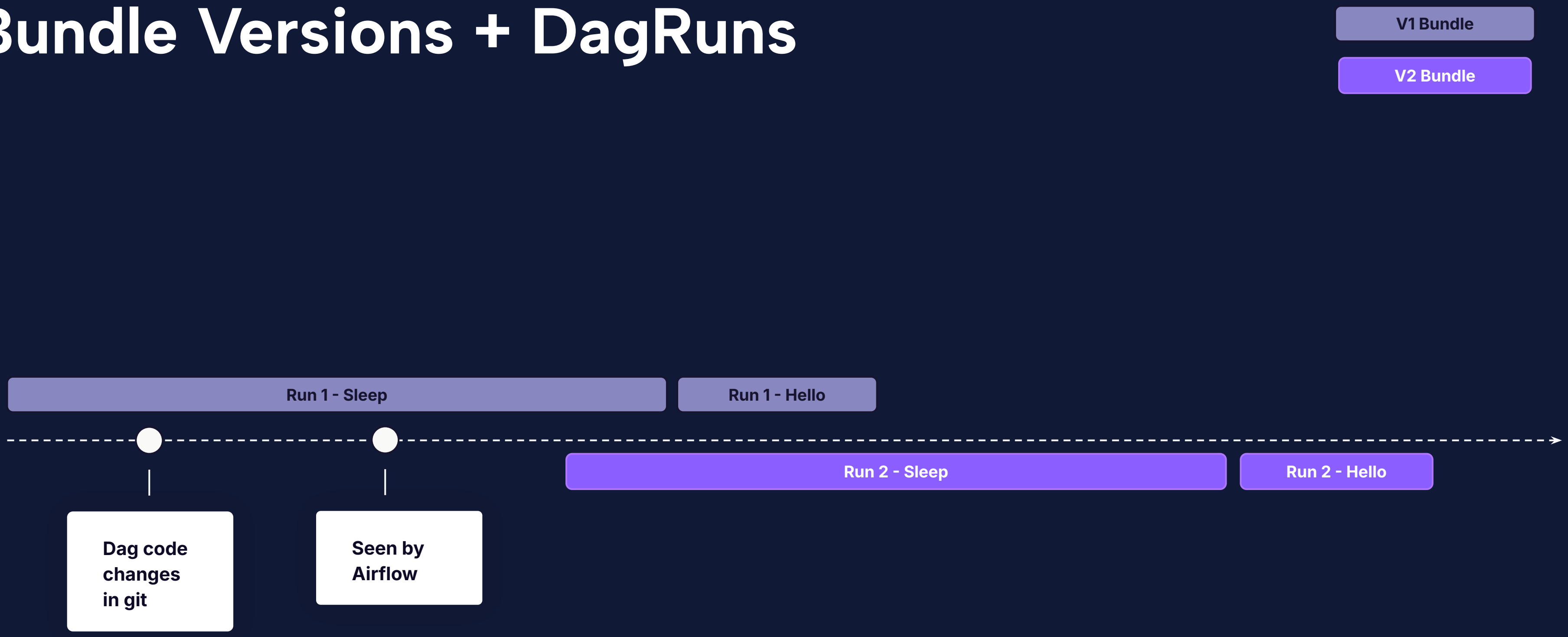
Dag code

ASTRONOMER

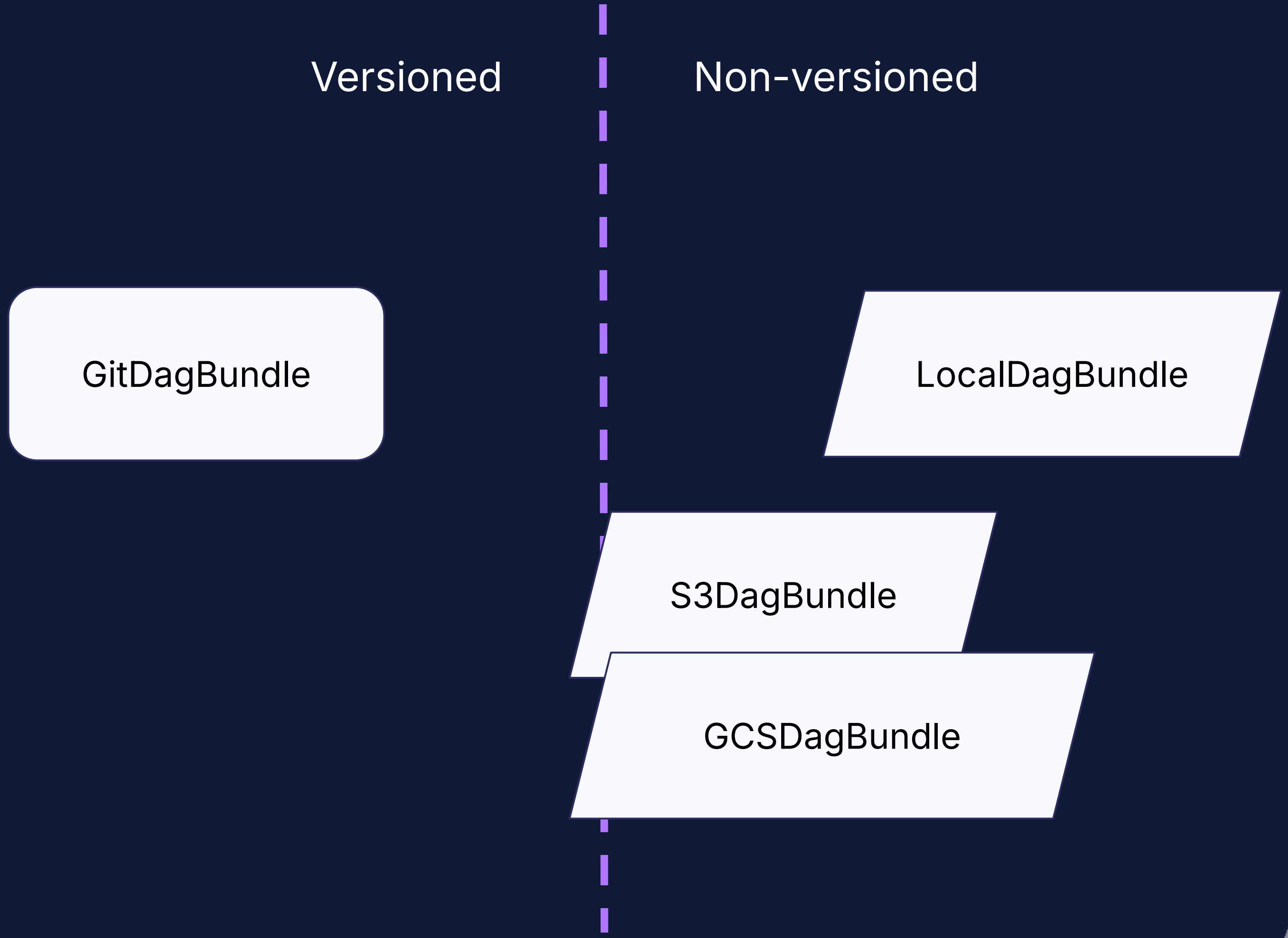
Dag code from
many sources

ASTRONOMER

Bundle Versions + DagRuns



ASTRONOMER



Versioned

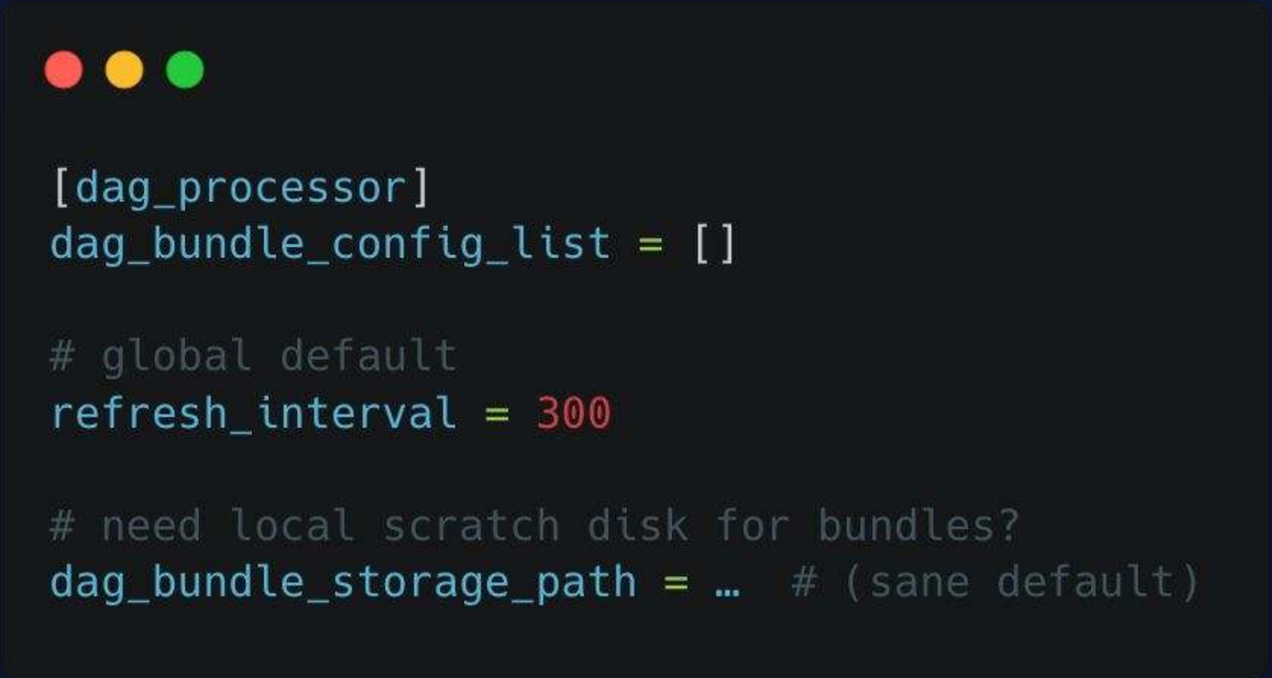
Non-versioned

GitDagBundle

LocalDagBundle

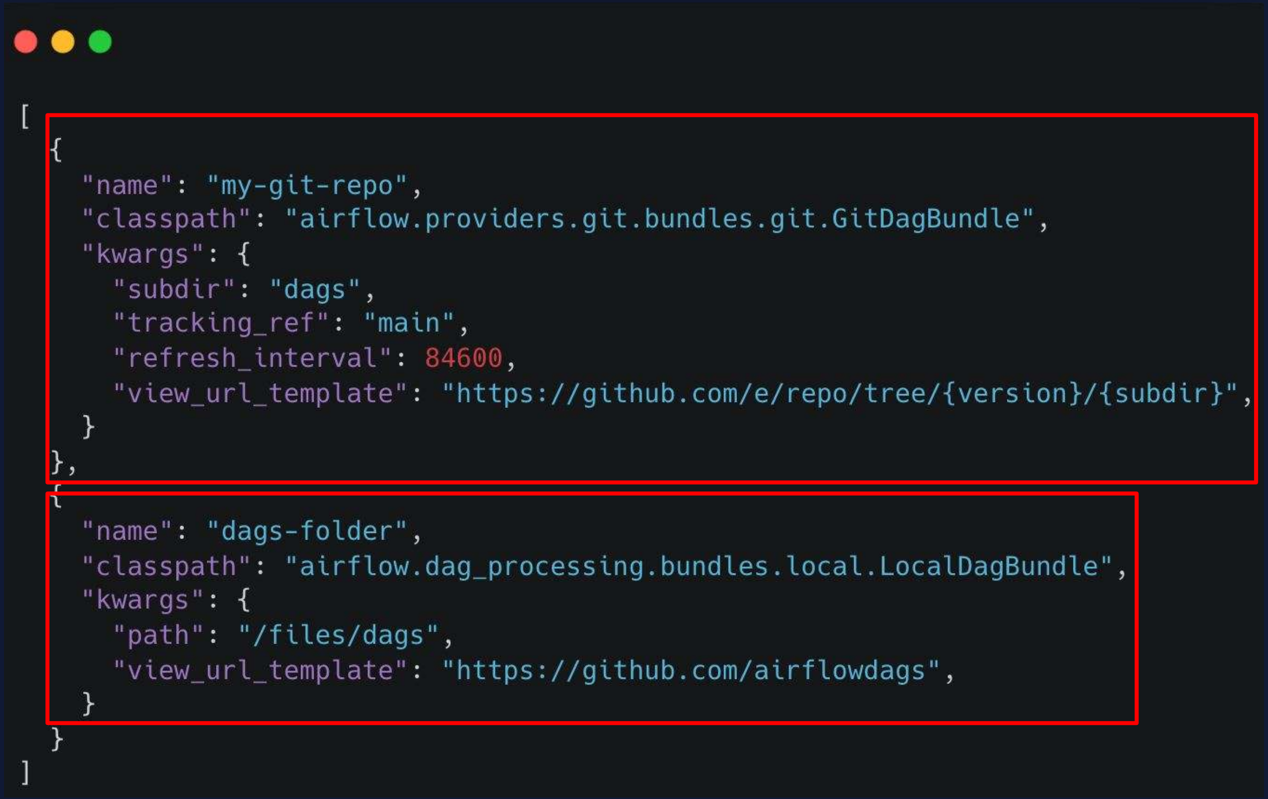
S3DagBundle

GCSDagBundle



```
[dag_processor]
dag_bundle_config_list = []

# global default
refresh_interval = 300

# need local scratch disk for bundles?
dag_bundle_storage_path = …  # (sane default)
```

[dag_processor] dag_bundle_config_list

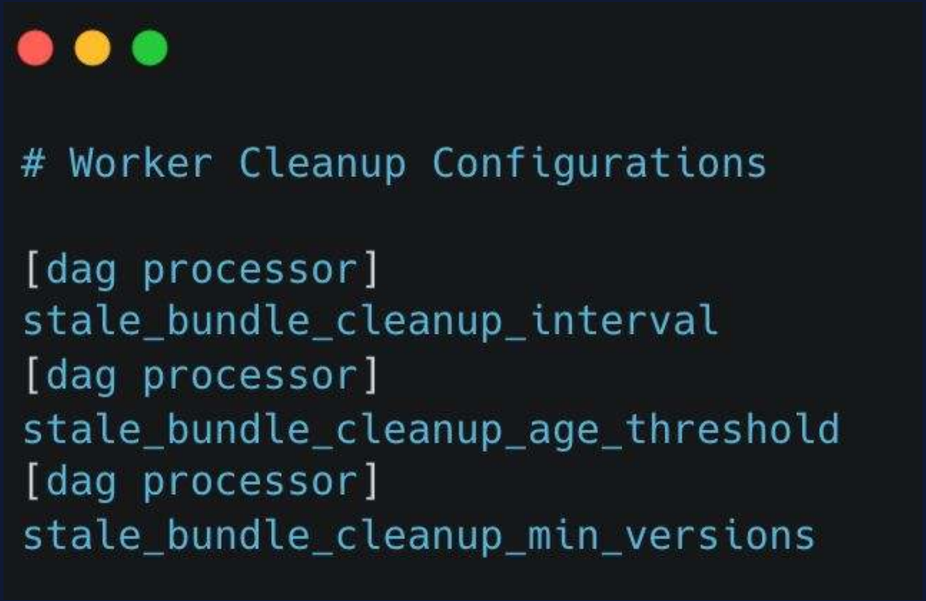


```
[
  {
    "name": "my-git-repo",
    "classpath": "airflow.providers.git.bundles.git.GitDagBundle",
    "kwargs": {
      "subdir": "dags",
      "tracking_ref": "main",
      "refresh_interval": 84600,
      "view_url_template": "https://github.com/e/repo/tree/{version}/{subdir}",
    }
  },
  {
    "name": "dags-folder",
    "classpath": "airflow.dag_processing.bundles.local.LocalDagBundle",
    "kwargs": {
      "path": "/files/dags",
      "view_url_template": "https://github.com/airflowdags",
    }
  }
]
```

Dag bundles lifecycle

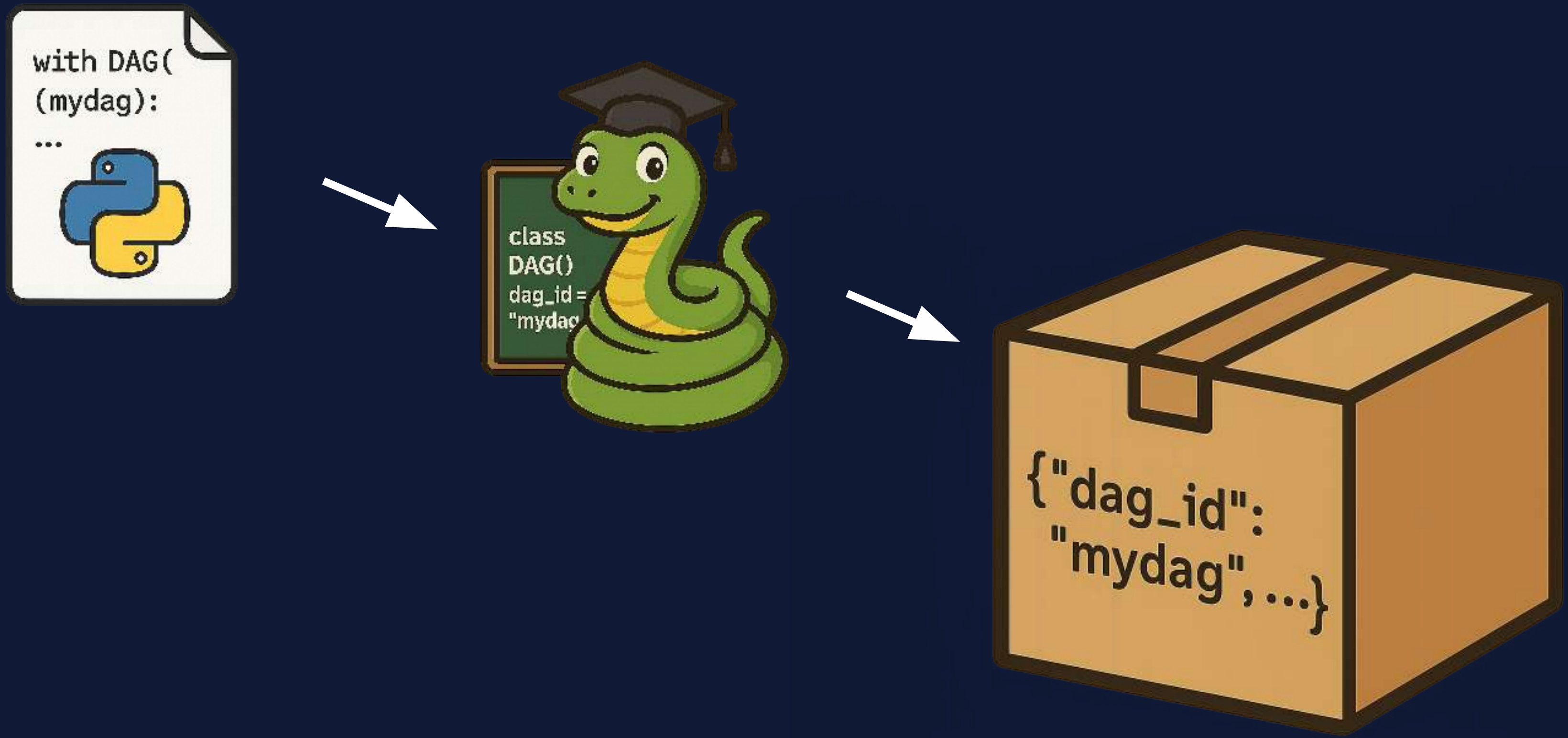
Discovery

- refresh_interval



```
# Worker Cleanup Configurations

[dag processor]
stale_bundle_cleanup_interval
[dag processor]
stale_bundle_cleanup_age_threshold
[dag processor]
stale_bundle_cleanup_min_versions
```

Observed dag version

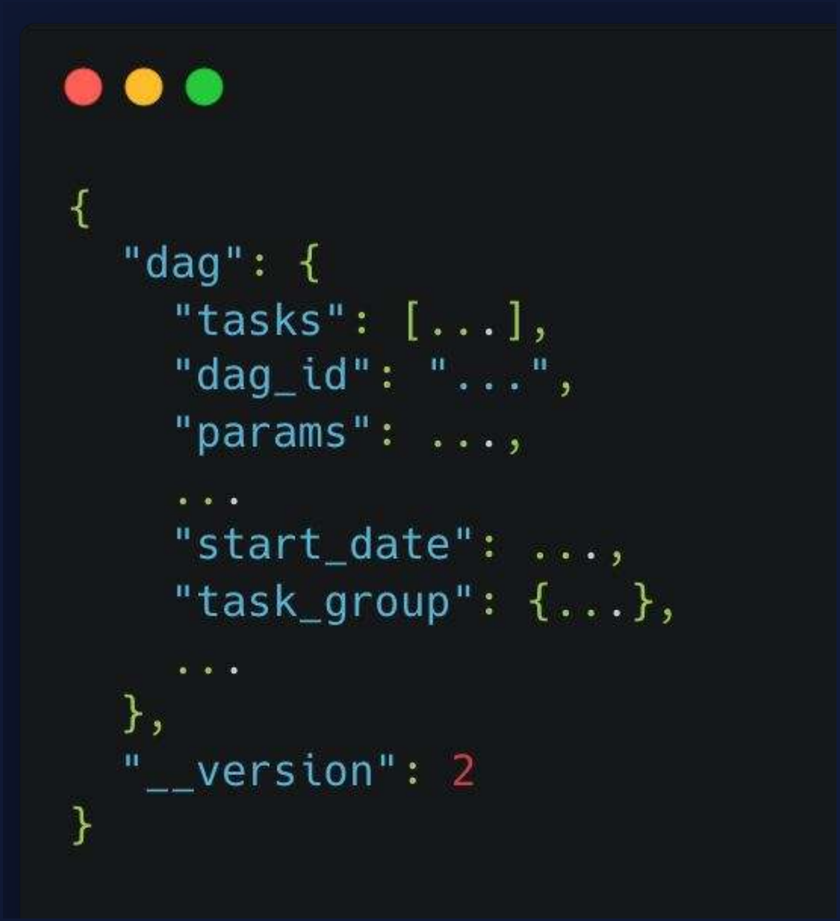


```
with DAG(
(mydag):
...
```

```
class
DAG()
dag_id =
"mydag
```

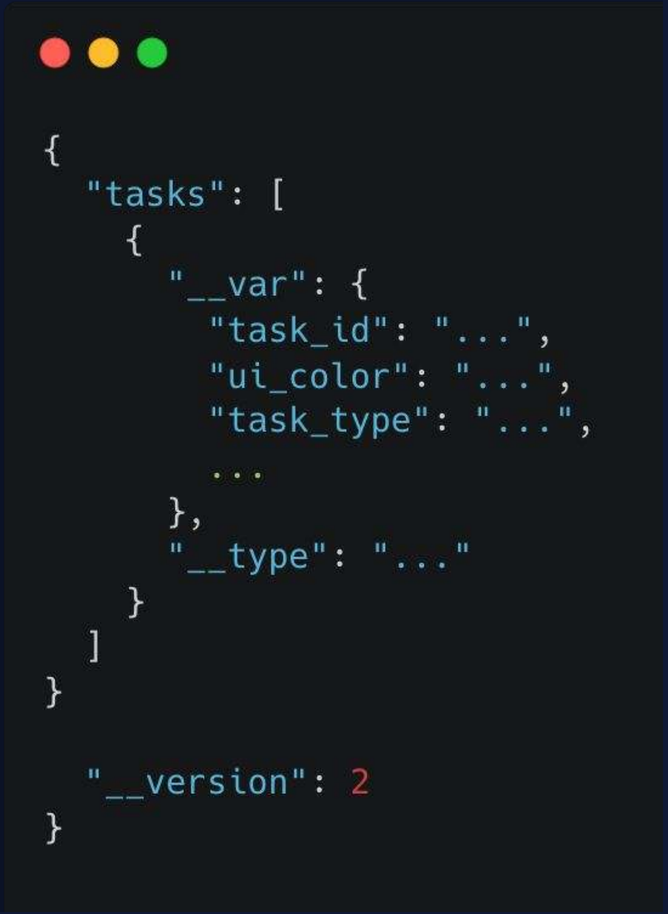
```
{"dag_id":
 "mydag",...}
```

Sample JSON Serialization



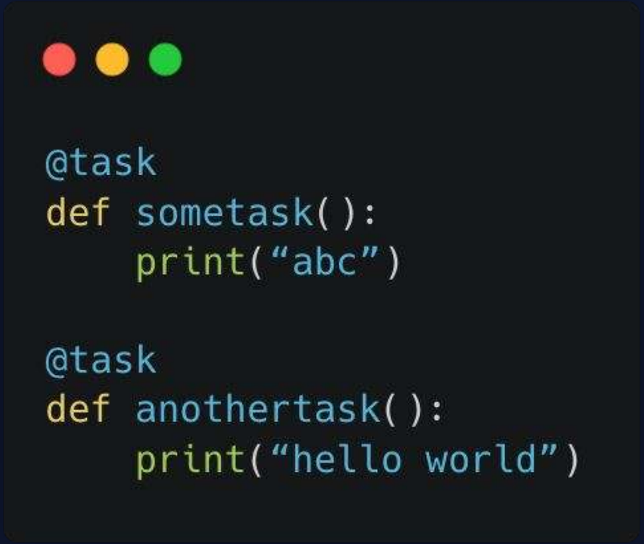
```
{
  "dag": {
    "tasks": [...],
    "dag_id": "...",
    "params": ...,
    ...
    "start_date": ...,
    "task_group": {...},
    ...
  },
  "__version": 2
}
```

Sample JSON Serialization

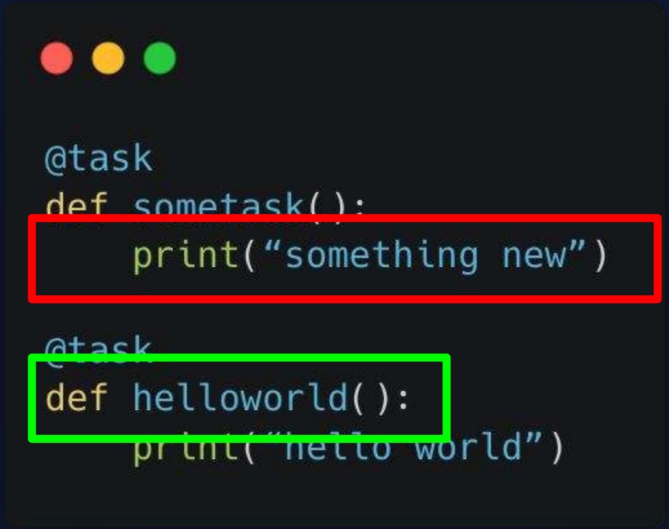


```
{
  "tasks": [
    {
      "__var": {
        "task_id": "...",
        "ui_color": "...",
        "task_type": "...",
        ...
      },
      "__type": "..."
    }
  ]
}

  "__version": 2
}
```

Observed dag version

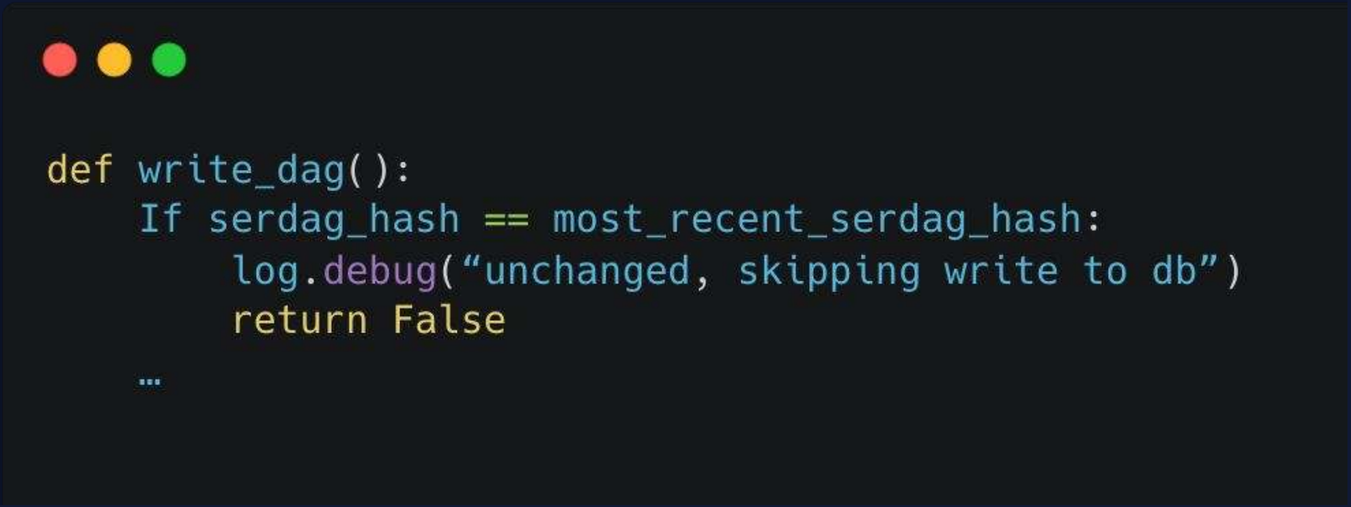


```
@task
def sometask():
    print(“abc”)

@task
def anothertask():
    print(“hello world”)
```

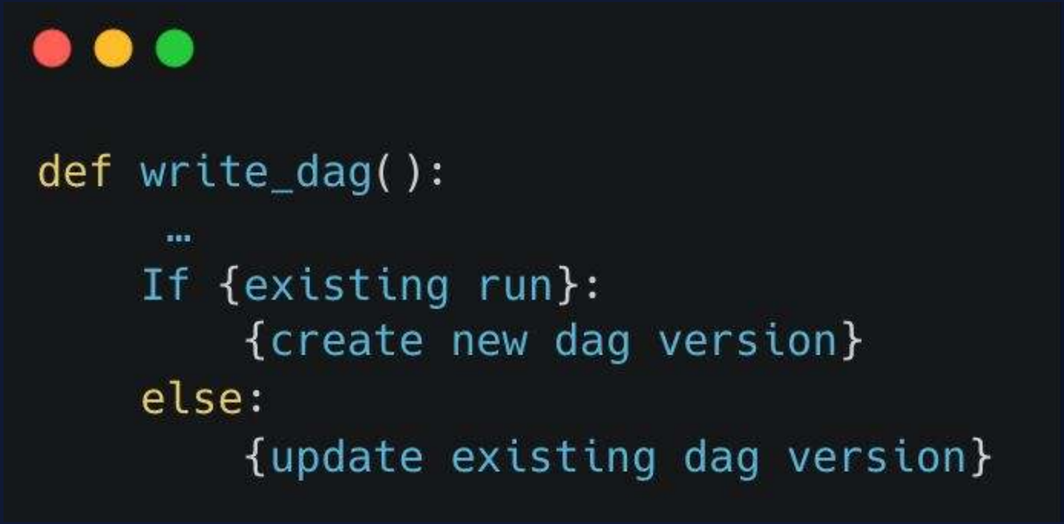


```
@task
def sometask():
    print(“something new”)

@task
def helloworld():
    print(“hello world”)
```

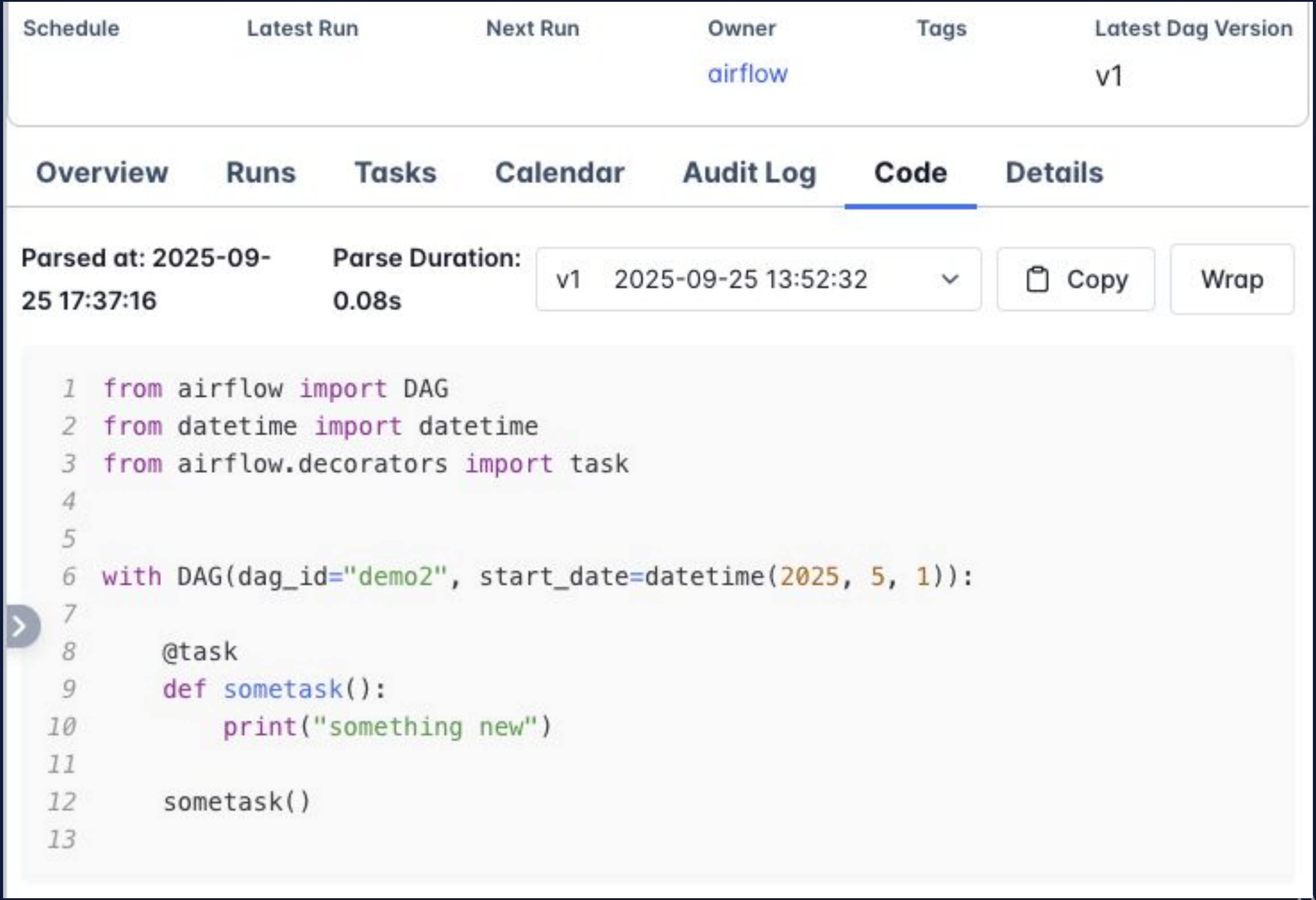
How do we create new observed dag versions?



```
def write_dag():
    If serdag_hash == most_recent_serdag_hash:
        log.debug(“unchanged, skipping write to db”)
        return False
    …
```



```
def write_dag():
     …
    If {existing run}:
        {create new dag version}
    else:
        {update existing dag version}
```



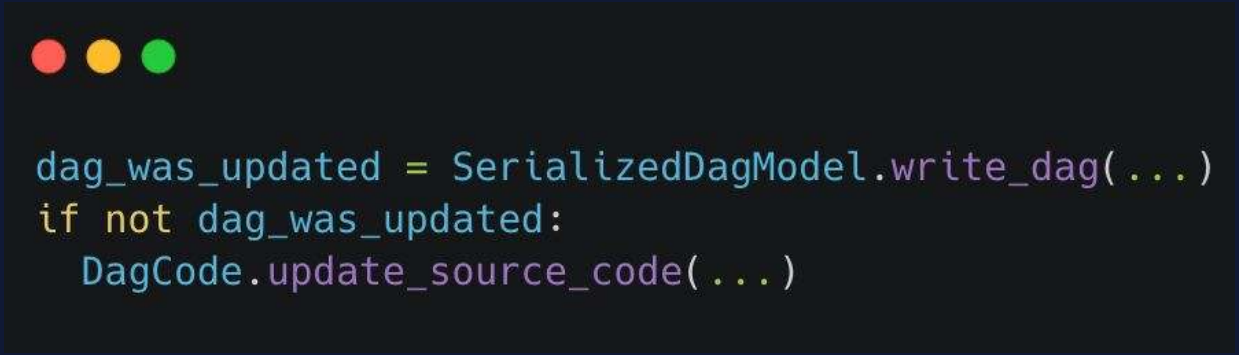
Schedule	Latest Run	Next Run	Owner	Tags	Latest Dag Version
			airflow		v1

Parsed at: 2025-09-25 17:37:16

Parse Duration: 0.08s

v1 2025-09-25 13:52:32

```python
from airflow import DAG
from datetime import datetime
from airflow.decorators import task


with DAG(dag_id="demo2", start_date=datetime(2025, 5, 1)):

    @task
    def sometask():
        print("something new")

    sometask()

```



```
dag_was_updated = SerializedDagModel.write_dag(...)
if not dag_was_updated:
  DagCode.update_source_code(...)
```

How are bundles versioned?

Bundle and Observed dag version, what is the difference?

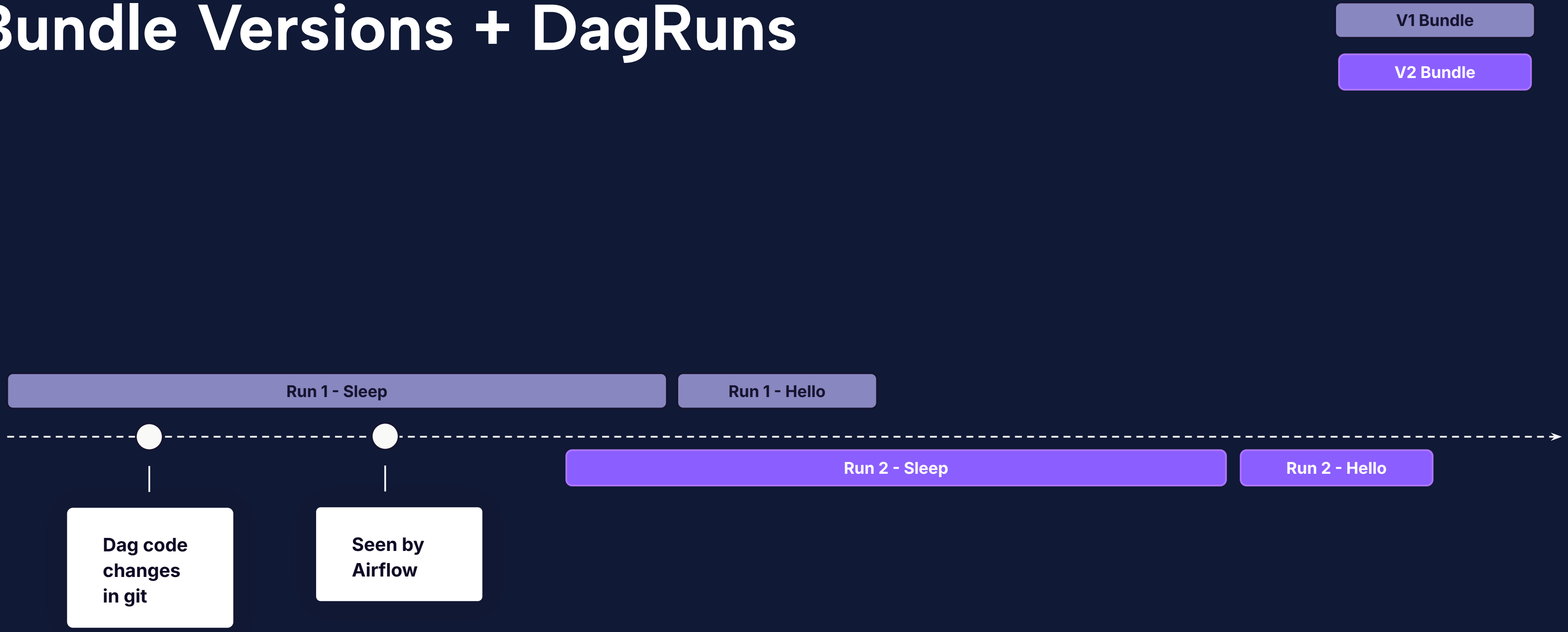
Bundle version → The version of a Dag bundle (the whole code snapshot)

DAG version → The version of a single Dag definition as observed after parsing/serialization.

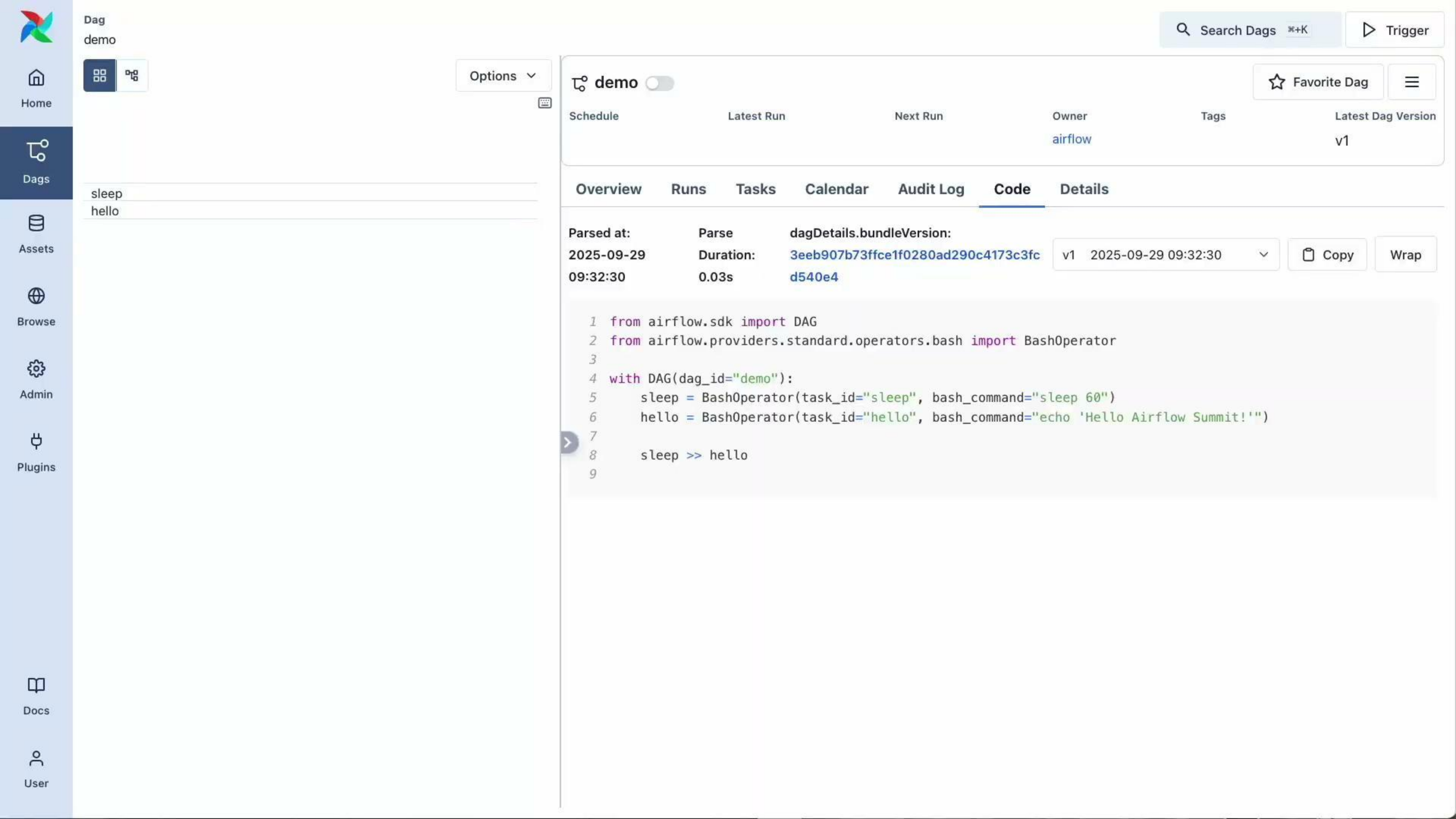
A look into dag parsing loop

- **Version-aware refresh**
- **Per-bundle discovery**
- **Reconciliation after refresh:**
  - Removed files & import errors
- **Isolation & parallelism by file**

Bundle Versions + DagRuns



ASTRONOMER



Owner: airflow

Latest Dag Version: v1

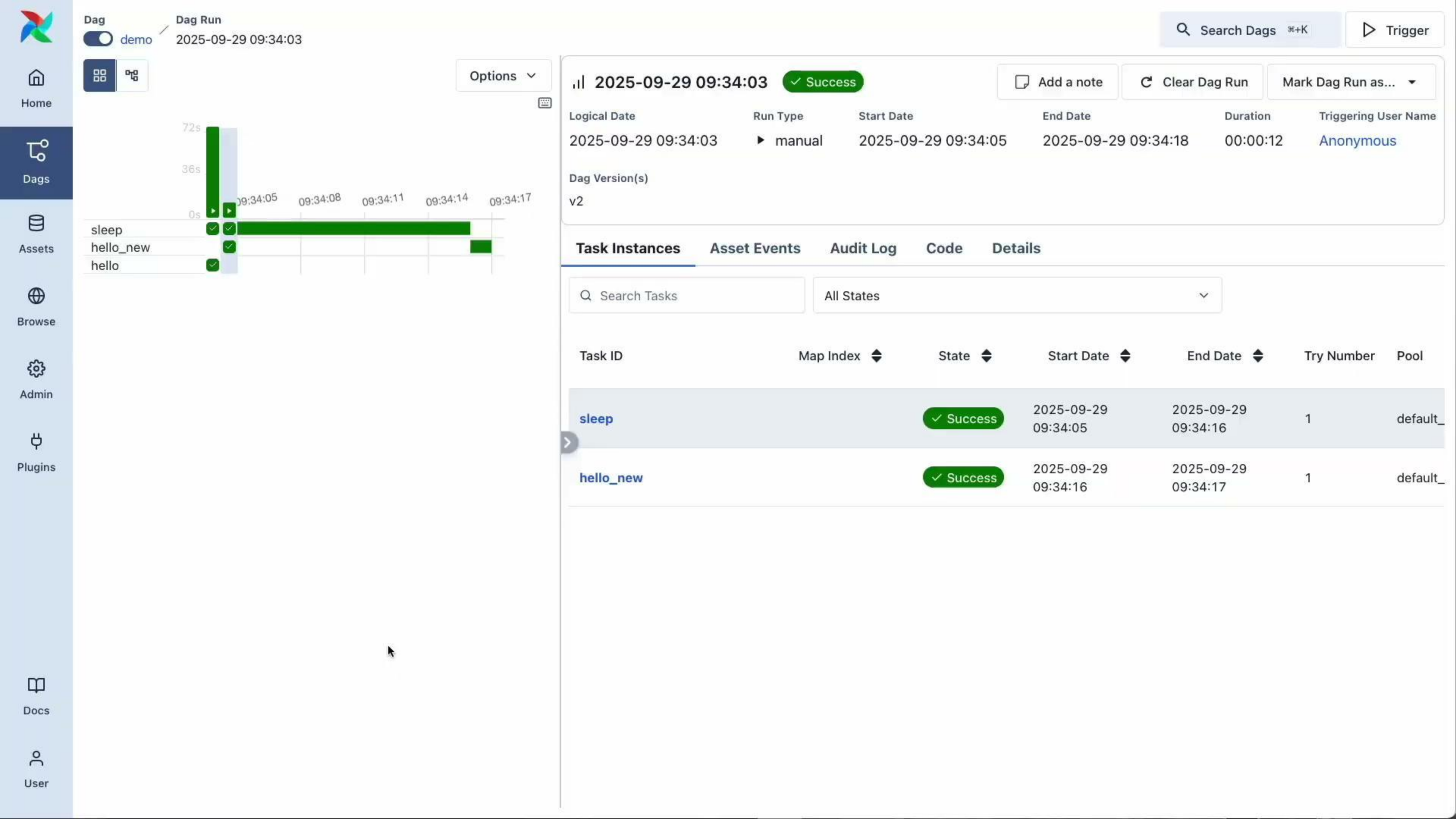
Parsed at: 2025-09-29 09:32:30

Parse Duration: 0.03s

dagDetails.bundleVersion: 3eeb907b73ffce1f0280ad290c4173c3fcd540e4

v1 2025-09-29 09:32:30

```python
from airflow.sdk import DAG
from airflow.providers.standard.operators.bash import BashOperator

with DAG(dag_id="demo"):
    sleep = BashOperator(task_id="sleep", bash_command="sleep 60")
    hello = BashOperator(task_id="hello", bash_command="echo 'Hello Airflow Summit!'")

    sleep >> hello

```



Dag: demo / Dag Run: 2025-09-29 09:34:03

2025-09-29 09:34:03 ✓ Success

Logical Date	Run Type	Start Date	End Date	Duration	Triggering User Name
2025-09-29 09:34:03	manual	2025-09-29 09:34:05	2025-09-29 09:34:18	00:00:12	Anonymous

Dag Version(s)
v2

Task ID	Map Index	State	Start Date	End Date	Try Number	Pool
sleep		✓ Success	2025-09-29 09:34:05	2025-09-29 09:34:16	1	default_
hello_new		✓ Success	2025-09-29 09:34:16	2025-09-29 09:34:17	1	default_

3.1+ only

Rerunning dags/taskinstance: versioning behaviour

3.1+ only

When you clear a dag/task instances

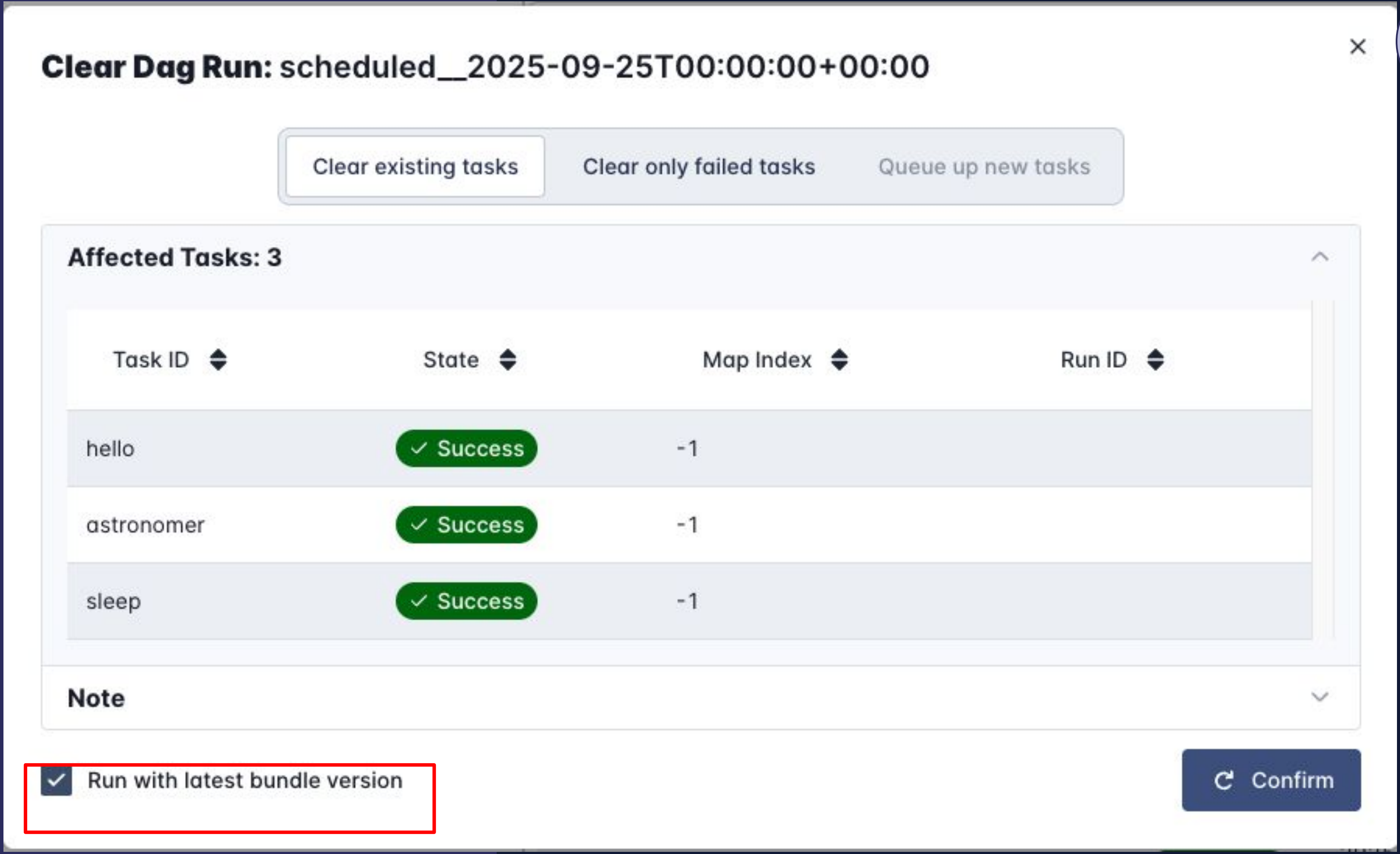
You have two options:

1. **Original serialized dag/bundle version (default)**

2. **Latest dag/bundle version**

∗Only applicable to versioned bundles*

3.1+ only



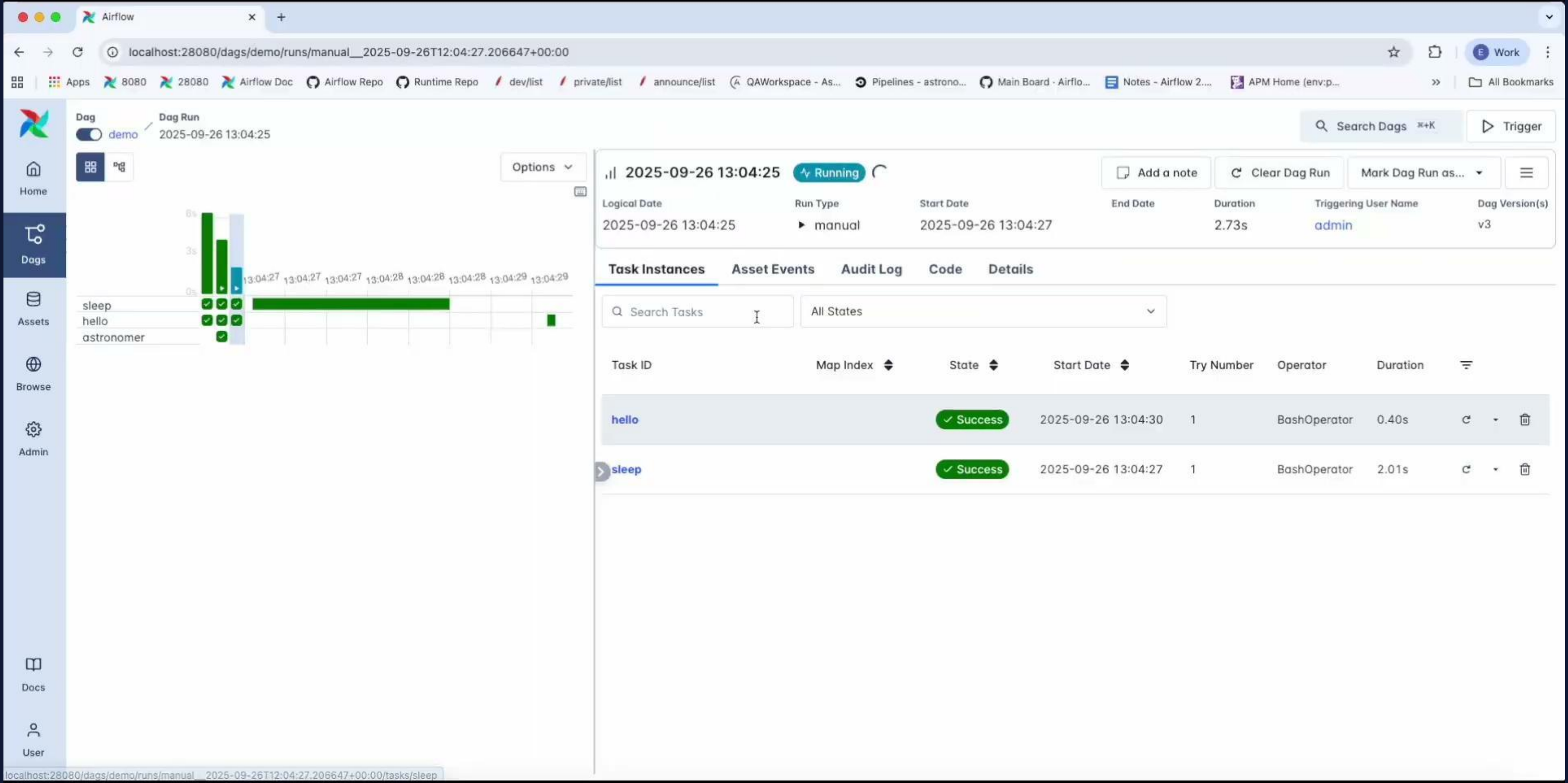
Clear Dag Run: scheduled__2025-09-25T00:00:00+00:00

Affected Tasks: 3

Task ID	State	Map Index	Run ID
hello	✓ Success	-1	
astronomer	✓ Success	-1	
sleep	✓ Success	-1	

Note

☑ Run with latest bundle version



Dag: demo / Dag Run: 2025-09-26 13:04:25

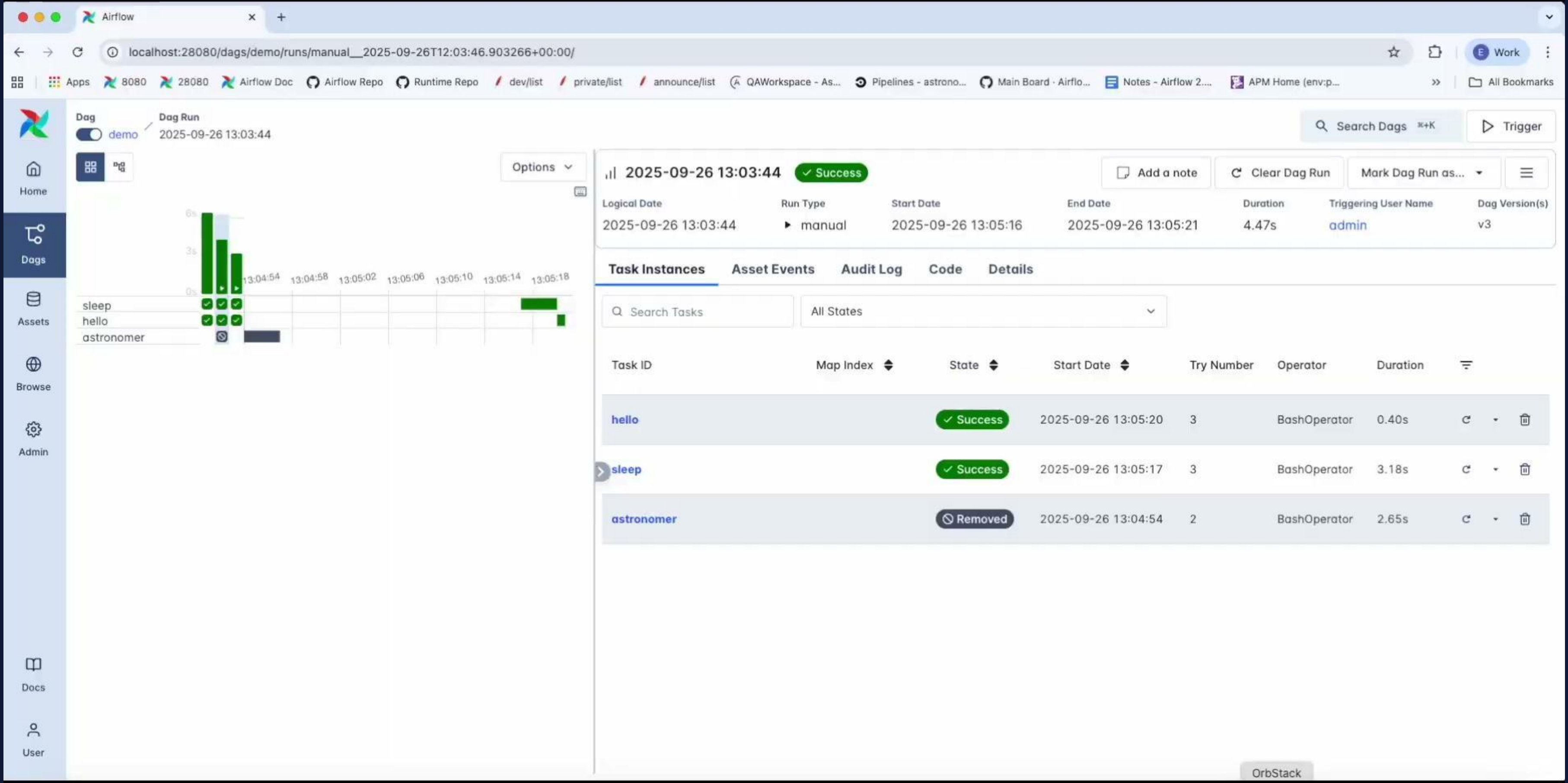
2025-09-26 13:04:25 Running

Logical Date	Run Type	Start Date	End Date	Duration	Triggering User Name	Dag Version(s)
2025-09-26 13:04:25	manual	2025-09-26 13:04:27		2.73s	admin	v3

Task ID	Map Index	State	Start Date	Try Number	Operator	Duration
hello		✓ Success	2025-09-26 13:04:30	1	BashOperator	0.40s
sleep		✓ Success	2025-09-26 13:04:27	1	BashOperator	2.01s

localhost:28080/dags/demo/runs/manual__2025-09-26T12:04:27.206647+00:00/tasks/sleep

MER

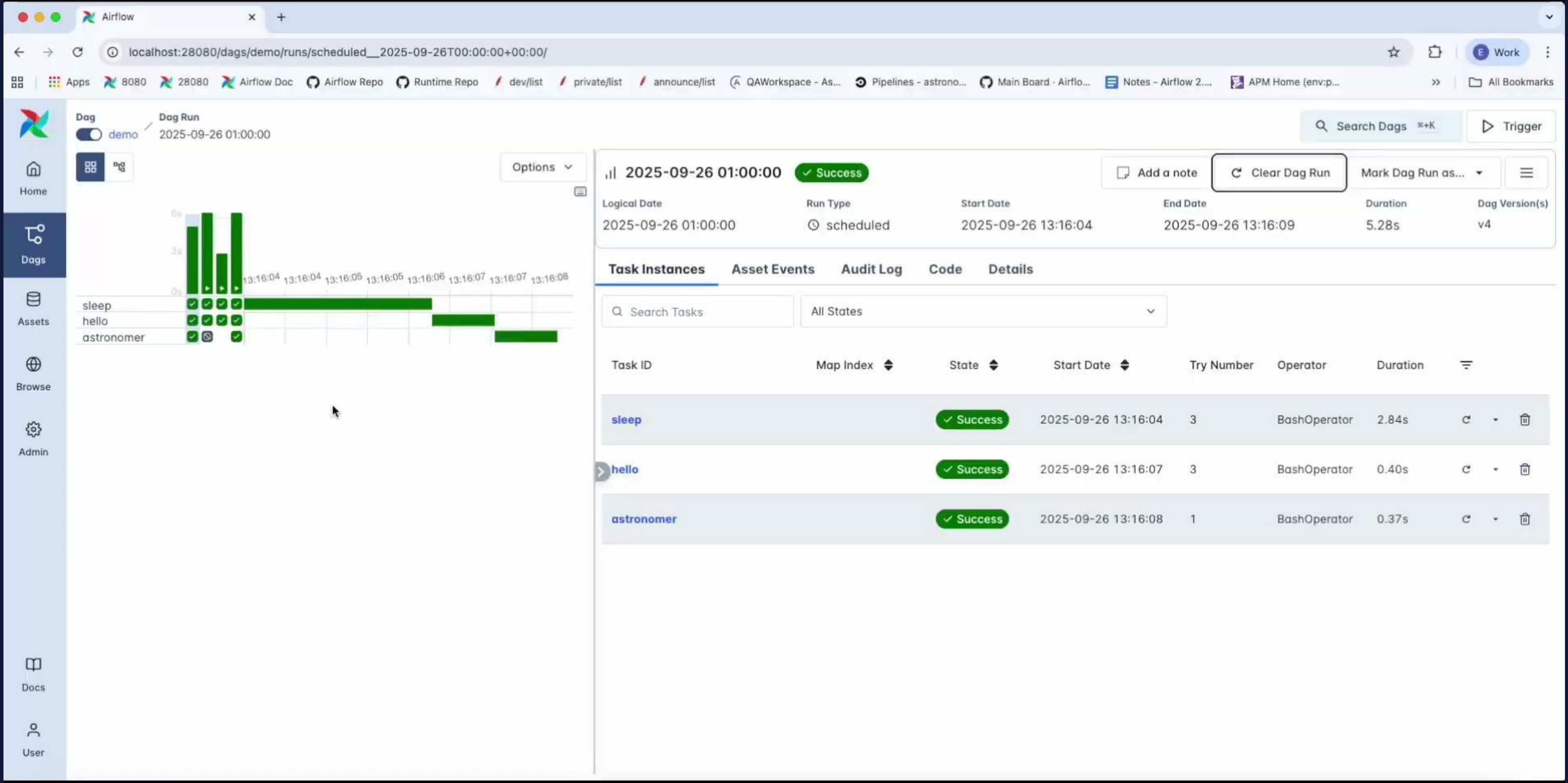


Dag: demo / Dag Run: 2025-09-26 13:03:44

2025-09-26 13:03:44 ✓ Success

Logical Date	Run Type	Start Date	End Date	Duration	Triggering User Name	Dag Version(s)
2025-09-26 13:03:44	manual	2025-09-26 13:05:16	2025-09-26 13:05:21	4.47s	admin	v3

Task ID	Map Index	State	Start Date	Try Number	Operator	Duration
hello		✓ Success	2025-09-26 13:05:20	3	BashOperator	0.40s
sleep		✓ Success	2025-09-26 13:05:17	3	BashOperator	3.18s
astronomer		Removed	2025-09-26 13:04:54	2	BashOperator	2.65s



Dag
demo

Dag Run
2025-09-26 01:00:00

2025-09-26 01:00:00 ✓ Success

Logical Date	Run Type	Start Date	End Date	Duration	Dag Version(s)
2025-09-26 01:00:00	scheduled	2025-09-26 13:16:04	2025-09-26 13:16:09	5.28s	v4

Task Instances | Asset Events | Audit Log | Code | Details

Search Tasks
All States

Task ID	Map Index	State	Start Date	Try Number	Operator	Duration
sleep		✓ Success	2025-09-26 13:16:04	3	BashOperator	2.84s
hello		✓ Success	2025-09-26 13:16:07	3	BashOperator	0.40s
astronomer		✓ Success	2025-09-26 13:16:08	1	BashOperator	0.37s

MER

3.1+ only

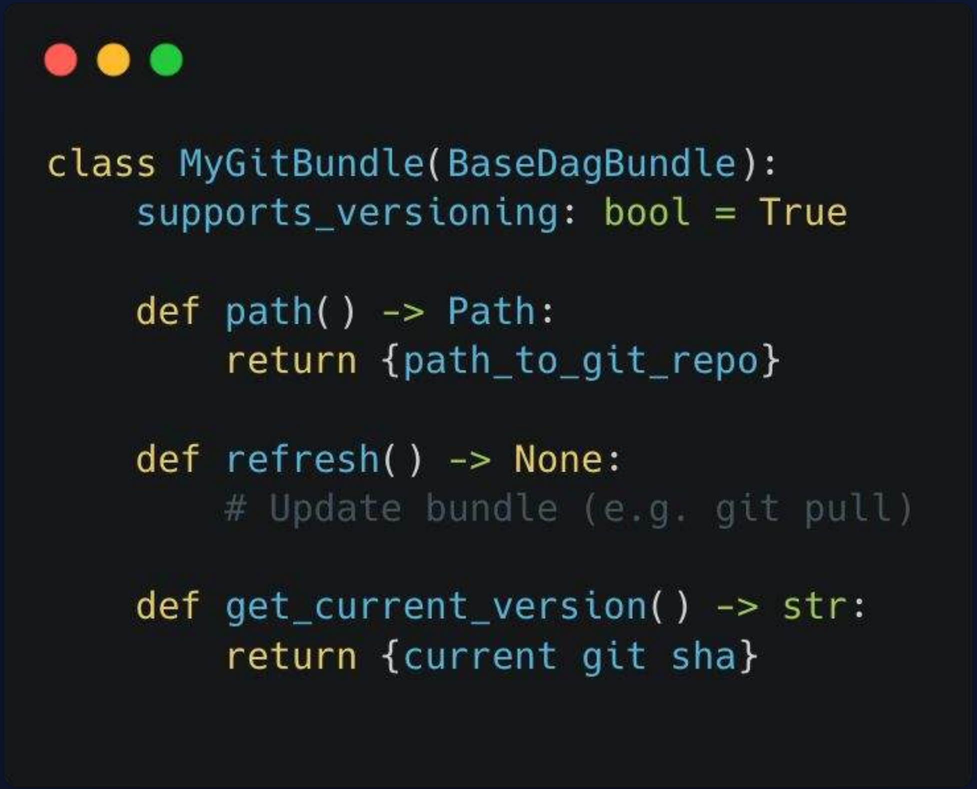
Clearing DAGs / Task Instances; What Changes vs. What Doesn’t

Only versions change: Updates Serialized Dag Version and Bundle Version on your request.

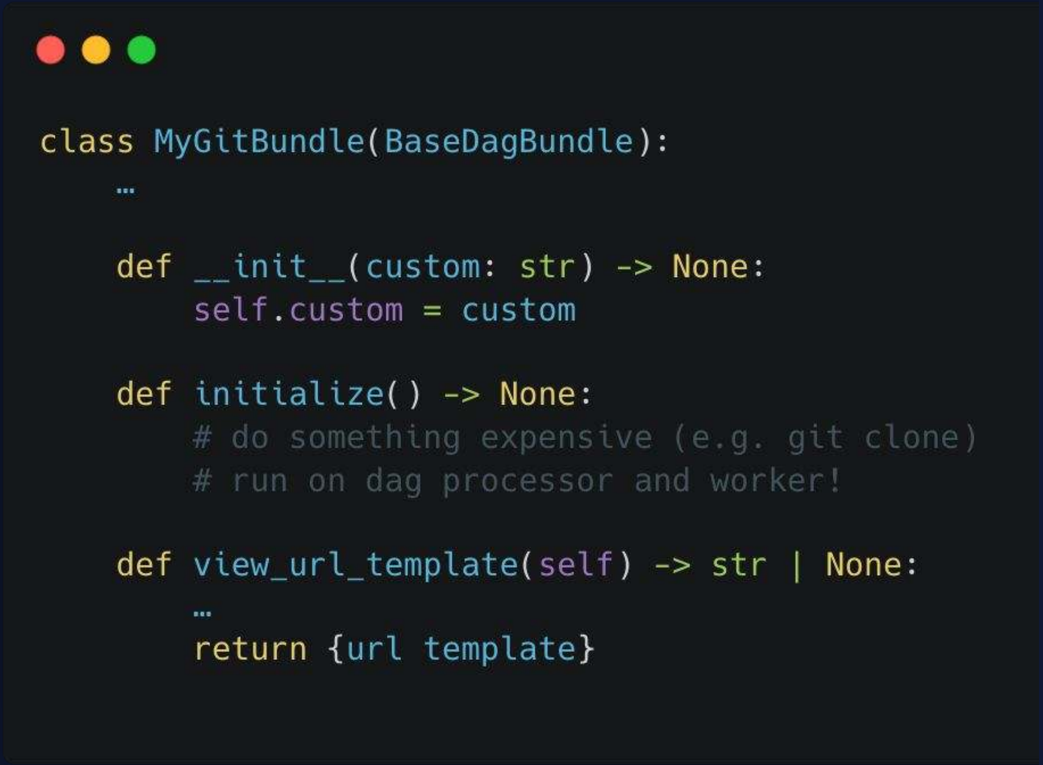
Topology: Because of the above request, a new task can be added or tasks marked removed but the TIs on the dagrun are never removed

Non-versioned bundle: No option to clear with the latest bundle version since it runs always with the latest version change

Custom Bundles 📦?
Of course!



```
class MyGitBundle(BaseDagBundle):
    supports_versioning: bool = True

    def path() -> Path:
        return {path_to_git_repo}

    def refresh() -> None:
        # Update bundle (e.g. git pull)

    def get_current_version() -> str:
        return {current git sha}
```



```python
class MyGitBundle(BaseDagBundle):
    …

    def __init__(custom: str) -> None:
        self.custom = custom

    def initialize() -> None:
        # do something expensive (e.g. git clone)
        # run on dag processor and worker!

    def view_url_template(self) -> str | None:
        …
        return {url template}
```

Slido for questions

OR

Join link:
https://app.sli.do/event/tFD2BRhuH4jT3rD2nBwntg

Recap 🍻

Dag Versioning

Execution
(requires config)

UI
(automatic)

Bundles for

code versioning

ASTRONOMER

Dag versions for history:

UI + scheduler

The 2025 Apache Airflow® Survey is here!

Fill it out to for a free Airflow 3 Fundamentals or DAG Authoring in Airflow 3 certification code

Questions!
slido?

ASTRONOMER