AIRFLOW SUMMIT

Multi-instance Asset Synchronization

-

Push or Pull ?

Sébastien Crocquevieille

Othmane EL Metioui
Head of Data & AI

Numberly

What we do

We generate value and growth for our clients by turning their digital marketing expenses into impactful and profitable data-driven investments.

CRM & Loyalty Management

- 1st Party Data Collection
- Customer Journey design
- Omnichannel campaign management
- **Clean Room management**
- Incrementality & CLV

Numberly Martech Platform

- Content builder
- Marketing Automation tool
- Messaging APIs
- Impactly

Digital Media

- Media Strategy
- Full funnel activation
- Data Collaboration & Retail Media
- Experimentation & measurement

Data Strategy & Insights

- Digital Consulting
- Insights & Analytics
- AI-driven solutions
- Numberly Academy

Numberly the Marketing Technologist

- **We are around since 2000, HQ in Paris France.**

Paris Brussels London Amsterdam Dubai Tel Aviv New-York Montréal

- **95% of Our Infrastructure is Self-Hosted on Bare Metal Servers**
- **We are our own Internet Service Provider.**
- **We commit to Open Source & Developer Community**
  - Numberly Github, **OSS** financing, **Europython** Sponsor **11 years** straight
- **Self hosted Airflow users since 2017 - version 1.x**

numberly 1000mercis group

Numberly the Marketing Technologist

Airflow Key figures in Production:

- **1817** DAGs on 2 instances with **341** Datasets
- **16 500** daily DagRuns with **44 500** daily TaskInstances

WhoAmI

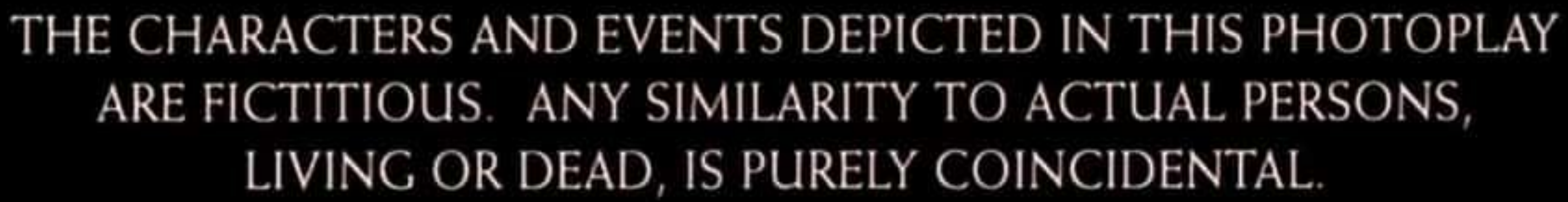
Who Am I ?

- Sébastien Crocquevieille
- Data Engineer @ Numberly
- French & Mexican
- Airflow user since 2018
- EuroPython 2023 Speaker
  - [Orchestrating Python Workflows in Apache Airflow]
- Pycon TW 2025 Speaker

You can call me **SEB**

Events vs Assets Scheduling

Disclaimer



THE CHARACTERS AND EVENTS DEPICTED IN THIS PHOTOPLAY ARE FICTITIOUS. ANY SIMILARITY TO ACTUAL PERSONS, LIVING OR DEAD, IS PURELY COINCIDENTAL.

External Events

It could be anything that happens outside Airflow

- Any message/post
- Any change in a value
- Any detectable action or signal

Event Driven Scheduling (Airflow 3.0)

Design :

- **Execution that strictly follows external events**
- **Individual action based on payload ?**
- **1 to 1 execution**

In practice :

- **A little complex to set up**
    - AssetWatcher: monitors external event source via triggers
- **Requires guarantees concerning:**
    - Ensuring deliverability
    - Avoiding duplicate events
    - Ordering ??

Asset-Aware Scheduling (Airflow 2.4)

Design :

- **Event that indicates a change in Data Source**
- **Data source changes are indistinguishable**

Constraints:

- **Similar to Event Driven but with lower constraints**
  - 1 to 1 execution **not** guaranteed
  - Order of events **shouldn't** be important
- **No triggers needed, just Airflow Assets**

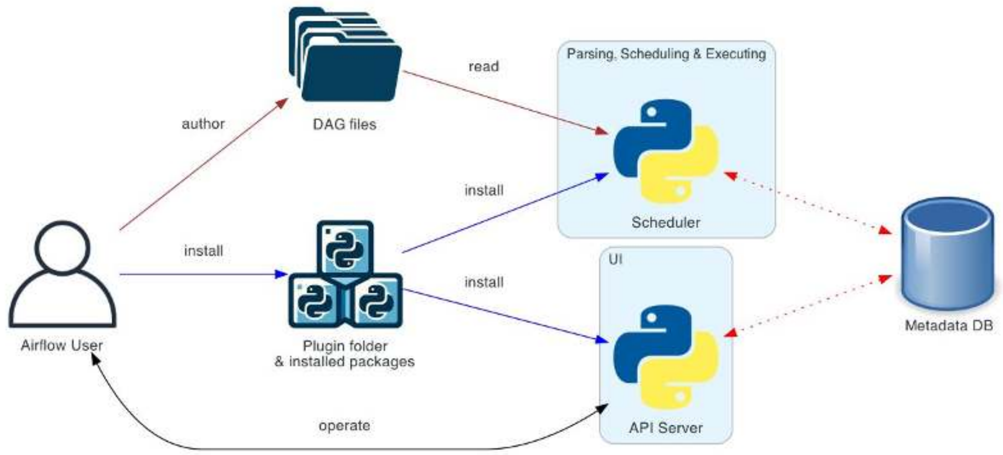
Airflow (Data) Assets

```
from airflow.sdk import Asset

example_asset = Asset("s3://asset-bucket/example.csv")
```

- **It is just a string → representing a Data Source**
- **Stored in Airflow DB**
- **Maintaining link is YOUR responsibility**
- **Generated by DAG parsing**

Upscaling our Scheduling

Airflow Scheduler



Performance issues

- **Machine performance**
  - High CPU and RAM usage
  - Scheduler heartbeat failures
  - dag_processing.total_parse_time is high

Performance issues

- **Machine performance**
  - High CPU and RAM usage
  - Scheduler heartbeat failures
  - dag_processing.total_parse_time is high

- **Slow scheduling**
  - Many tasks in “queued” or ”scheduled” state
  - Delayed DAG runs

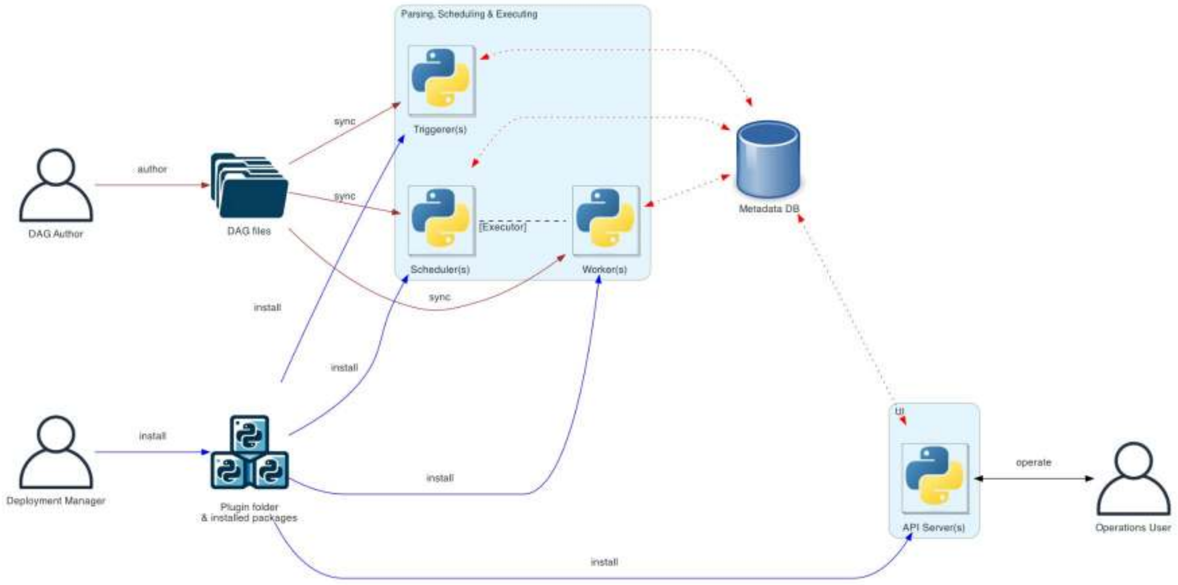
Performance issues

- **Machine performance**
  - High CPU and RAM usage
  - Scheduler heartbeat failures
  - dag_processing.total_parse_time is high

- **Slow scheduling**
  - Many tasks in “queued” or ”scheduled” state
  - Delayed DAG runs

- **Database limitations**
  - Too many connections
  - Slow queries

Airflow Remote Executors



Airflow Remote Executors

Pros:

- Robust: decoupling workers from scheduler process
- Effective: Worker parallelization
- Available: Low latency workers always running

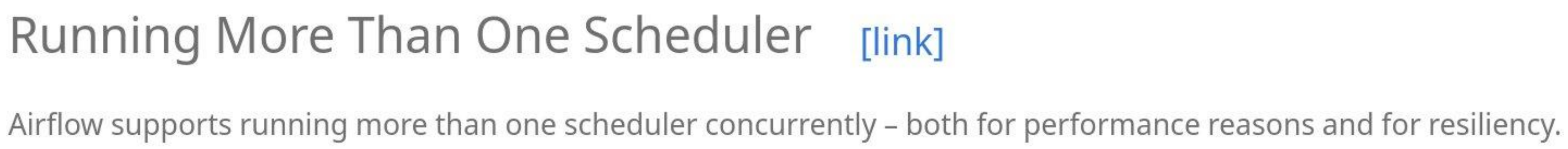
Airflow Remote Executors

Cons:

- Requires Infrastructure setup
- Can be Expensive: Cloud cost

Still having issues?

Scheduler Replication



Running More Than One Scheduler [link]

Airflow supports running more than one scheduler concurrently – both for performance reasons and for resiliency.

Scheduler Replication

Pros:
- Split scheduling load
- More resilient
- Not much work

Cons:
- More Database operations

Need more ?

Separate Instances

Just have 2 Airflows

Pros:

- Maximum scalability
- Separation of Concerns
- Better Access Control

Separate Instances

Just have 2 Airflows ?!

Cons:

- Even more maintenance
- Even more resources
- Redundant operations

Assets

+

Multiple instances

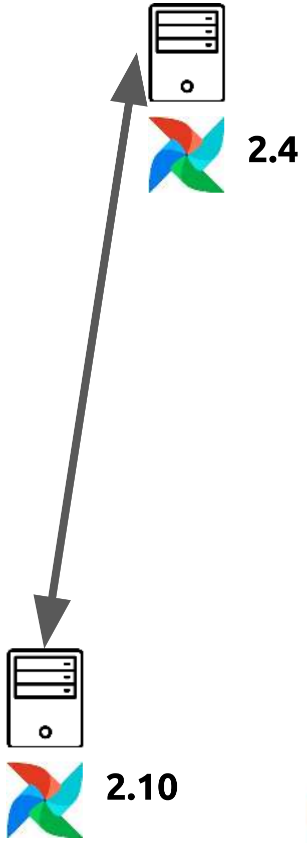
=

…

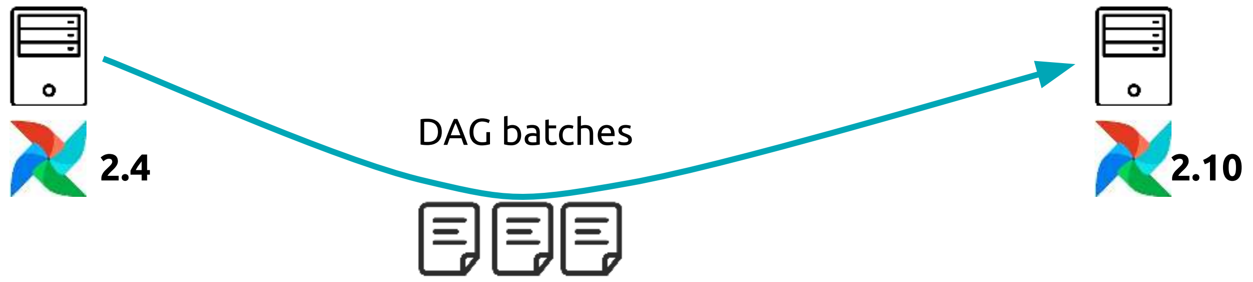
Synchronizing Airflow Assets

Our use case

- Airflow Migration 2.4 → 2.10 (latest version)
- Complete dependency overhaul (Python & Spark)
- Big dag library (~2000 DAGs)
- From old instance to new
- Upgrading code as we migrate



Our use case



- **Migrating DAGs by batches based on**
  - Complexity
  - Dependencies
  - Missing features
  - Urgency

- **Specific needs:**
  - Cross instance DAG Trigger
  - Cross instance Task sensor

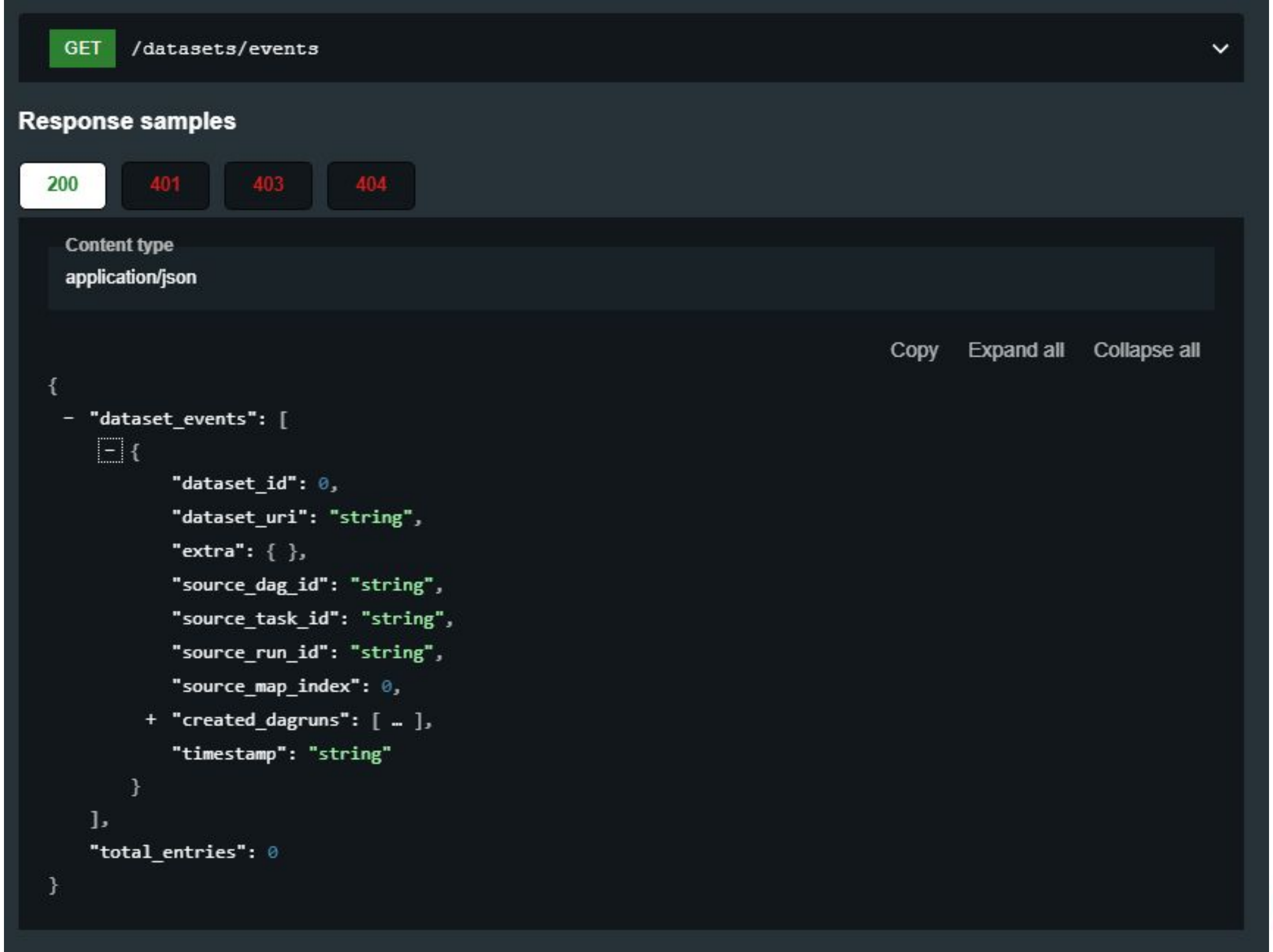
- **What about Airflow Assets?** ⇒ **Synchronization !**

Push or Pull ?

Push

How do we start?

By getting the Asset Events!



GET /datasets/events

Response samples

200 401 403 404

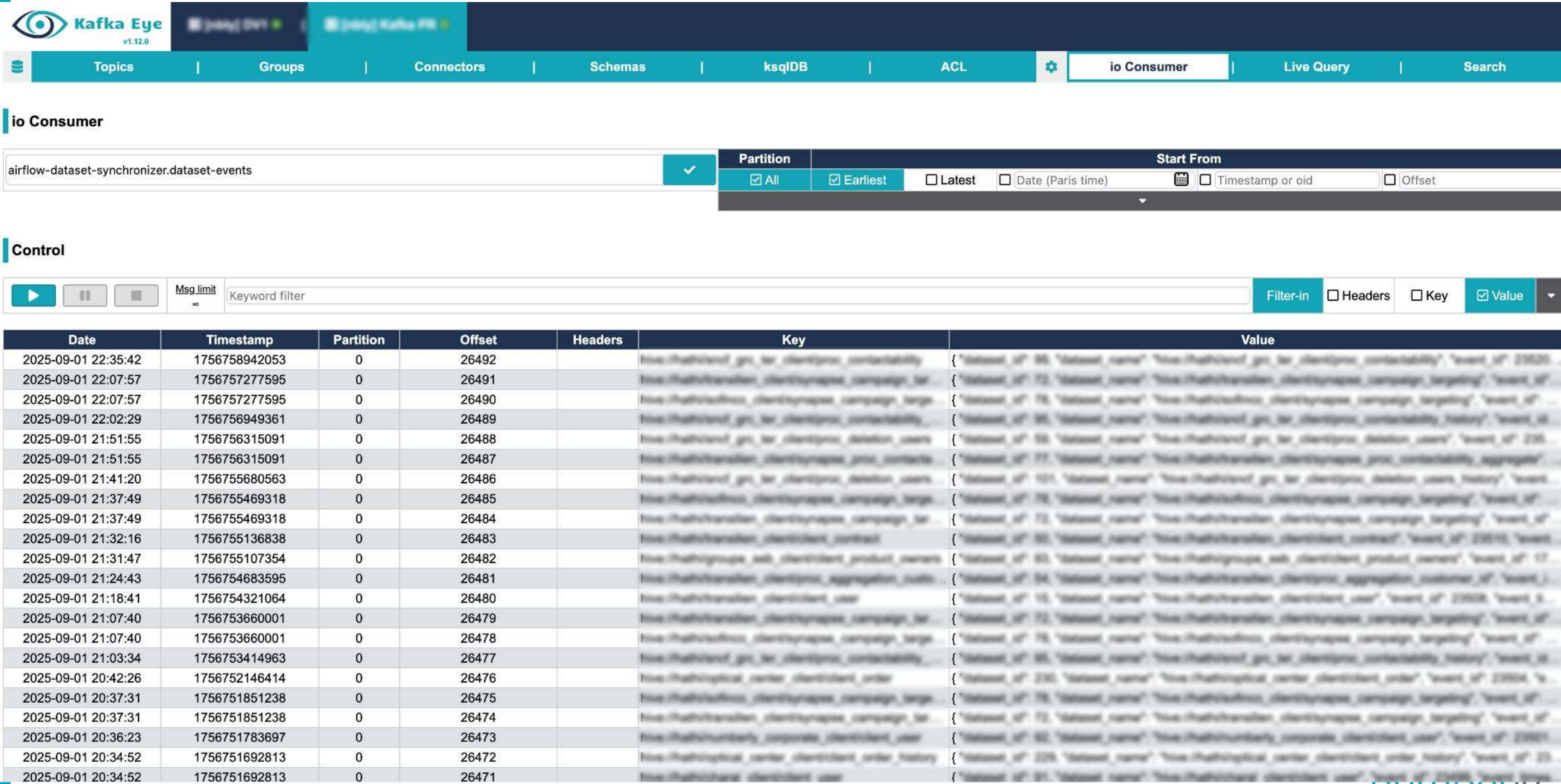
Content type
application/json

Copy Expand all Collapse all

```
{
  - "dataset_events": [
      - {
            "dataset_id": 0,
            "dataset_uri": "string",
            "extra": { },
            "source_dag_id": "string",
            "source_task_id": "string",
            "source_run_id": "string",
            "source_map_index": 0,
          + "created_dagruns": [ … ],
            "timestamp": "string"
        }
    ],
    "total_entries": 0
}
```

Design choices

- **Using Airflow API to get Asset Events**
    - Filter by -timestamp & store the “offset” in Airflow Variables


Kafka Eye
v1.12.0
Topics | Groups | Connectors | Schemas | ksqlDB | ACL
io Consumer | Live Query | Search

io Consumer

airflow-dataset-synchronizer.dataset-events

Partition	Start From				
☑ All	☑ Earliest	☐ Latest	☐ Date (Paris time)	☐ Timestamp or oid	☐ Offset

Control

Msg limit ∞ | Keyword filter | Filter-in | ☐ Headers | ☐ Key | ☑ Value

Date	Timestamp	Partition	Offset	Headers	Key	Value
2025-09-01 22:35:42	1756758942053	0	26492		[illegible]	[illegible]
2025-09-01 22:07:57	1756757277595	0	26491		[illegible]	[illegible]
2025-09-01 22:07:57	1756757277595	0	26490		[illegible]	[illegible]
2025-09-01 22:02:29	1756756949361	0	26489		[illegible]	[illegible]
2025-09-01 21:51:55	1756756315091	0	26488		[illegible]	[illegible]
2025-09-01 21:51:55	1756756315091	0	26487		[illegible]	[illegible]
2025-09-01 21:41:20	1756755680563	0	26486		[illegible]	[illegible]
2025-09-01 21:37:49	1756755469318	0	26485		[illegible]	[illegible]
2025-09-01 21:37:49	1756755469318	0	26484		[illegible]	[illegible]
2025-09-01 21:32:16	1756755136838	0	26483		[illegible]	[illegible]
2025-09-01 21:31:47	1756755107354	0	26482		[illegible]	[illegible]
2025-09-01 21:24:43	1756754683595	0	26481		[illegible]	[illegible]
2025-09-01 21:18:41	1756754321064	0	26480		[illegible]	[illegible]
2025-09-01 21:07:40	1756753660001	0	26479		[illegible]	[illegible]
2025-09-01 21:07:40	1756753660001	0	26478		[illegible]	[illegible]
2025-09-01 21:03:34	1756753414963	0	26477		[illegible]	[illegible]
2025-09-01 20:42:26	1756752146414	0	26476		[illegible]	[illegible]
2025-09-01 20:37:31	1756751851238	0	26475		[illegible]	[illegible]
2025-09-01 20:37:31	1756751851238	0	26474		[illegible]	[illegible]
2025-09-01 20:36:23	1756751783697	0	26473		[illegible]	[illegible]
2025-09-01 20:34:52	1756751692813	0	26472		[illegible]	[illegible]
2025-09-01 20:34:52	1756751692813	0	26471		[illegible]	[illegible]

Design choices

- **Using Airflow API to get Asset Events**
  - Filter by -timestamp & store the “offset” in Airflow Variables
- **Push them to a Kafka topic ⇒ Single source of truth for all Asset Events**

Almost Done !

Some 403 issues

- **No POST endpoint for Dataset Events -** Solved in 2.9
  - Dirty DB editor DAG
- **Dataset Events POST doesn't create NEW Datasets -** "Solved" in 3.0 with Materialize EP
  - Make my own materialize DAG

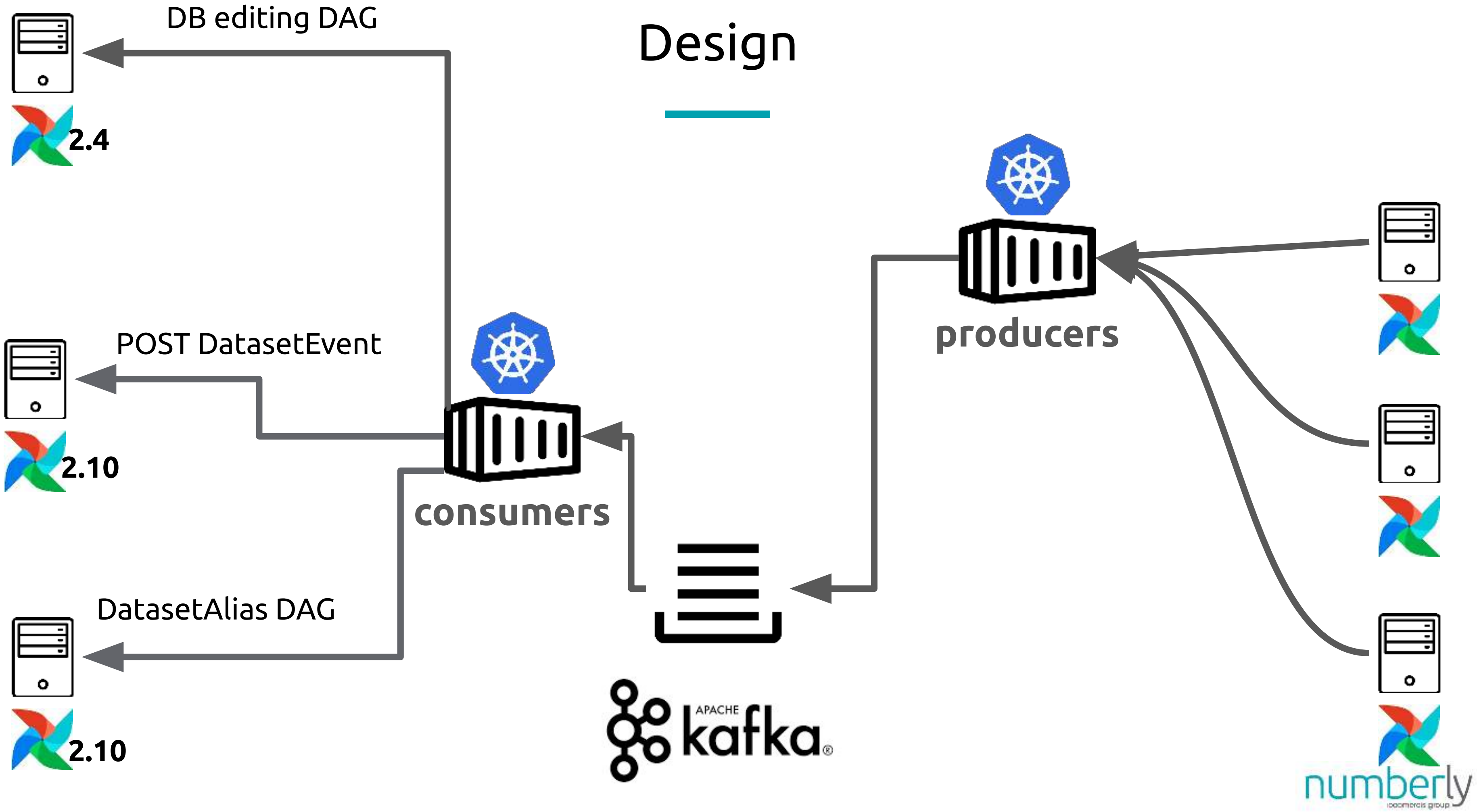
=====> Conflict between parsed objects (Dataset) and created objects (Dataset Events)

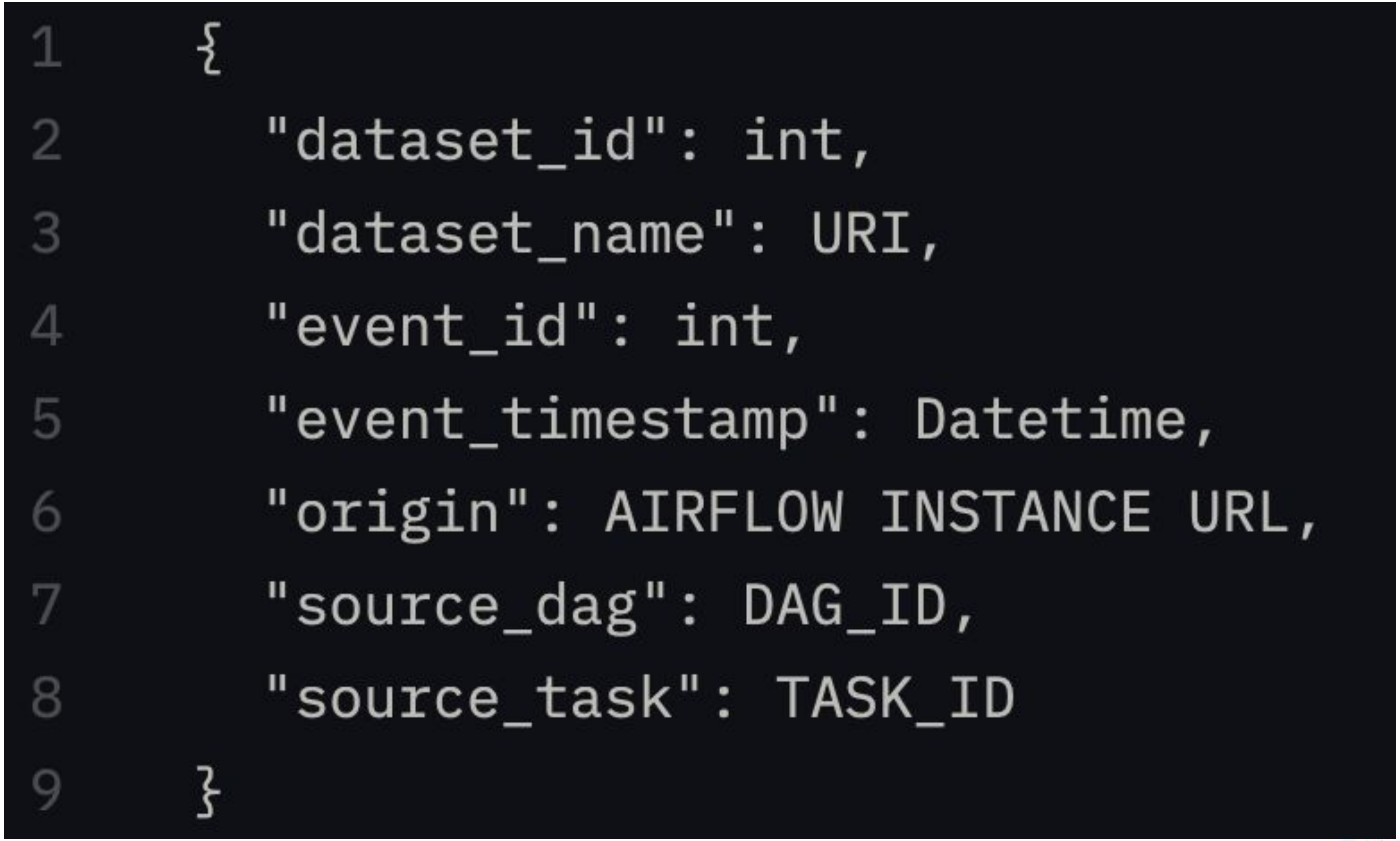
- **Datasets cannot be dynamically created -** Solved in 2.10 with DatasetAlias
  - Go to latest version (at the time) 2.10

Design choices

- **Using Airflow API to get Asset Events**
  - Filter by -timestamp & store the “offset” in Airflow Variables
- **Kafka topic ⇒ Single source of truth for all Asset Events**
- **Push back to Airflow through 3 different methods**

Design


DB editing DAG
2.4
POST DatasetEvent
2.10
DatasetAlias DAG
2.10
consumers
producers
APACHE kafka
numberly



```
{
    "dataset_id": int,
    "dataset_name": URI,
    "event_id": int,
    "event_timestamp": Datetime,
    "origin": AIRFLOW INSTANCE URL,
    "source_dag": DAG_ID,
    "source_task": TASK_ID
}
```

It works !

Or Pull ?

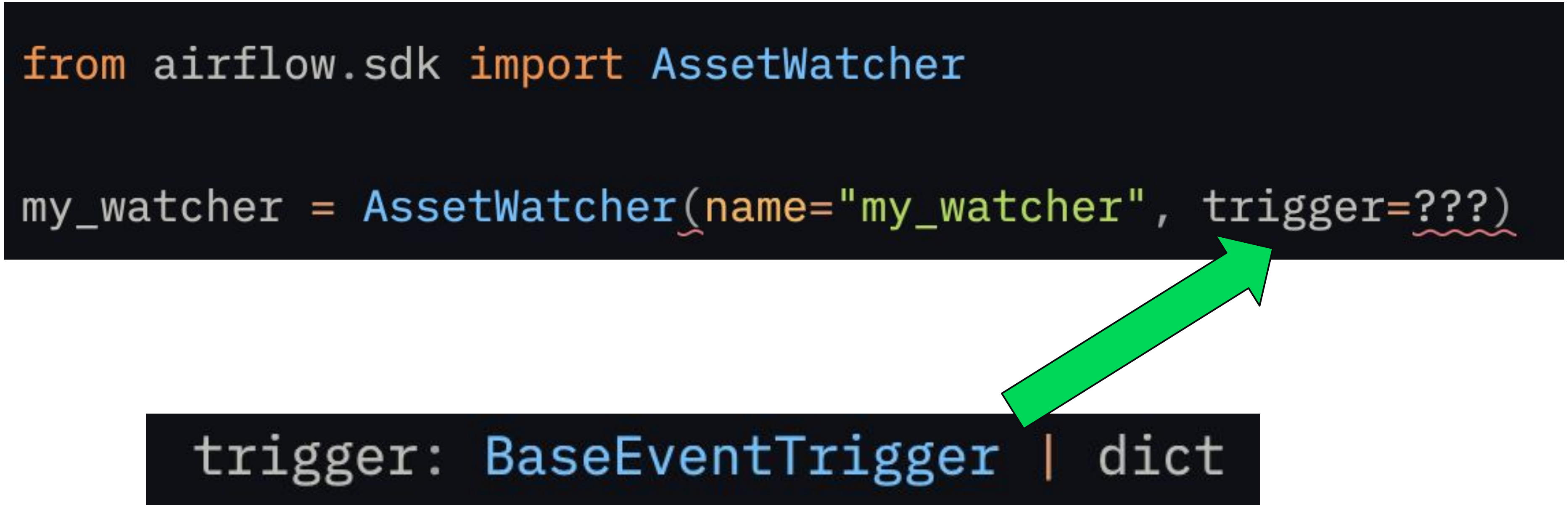
Pull Based Scheduling

- **Airflow PULL from external instances directly**
- **Use Event based scheduling**
  - **<u>Ignore</u>** the fact that we need another migration
- **Poll Asset Events continuously to stay updated**

Asset Watcher ⇒ Asset Event ⇒ DAG execution

Let’s code together

Using an Asset Watcher



```python
from airflow.sdk import AssetWatcher

my_watcher = AssetWatcher(name="my_watcher", trigger=???)
```

```python
trigger: BaseEventTrigger | dict
```

Base Event Trigger

CLASS:

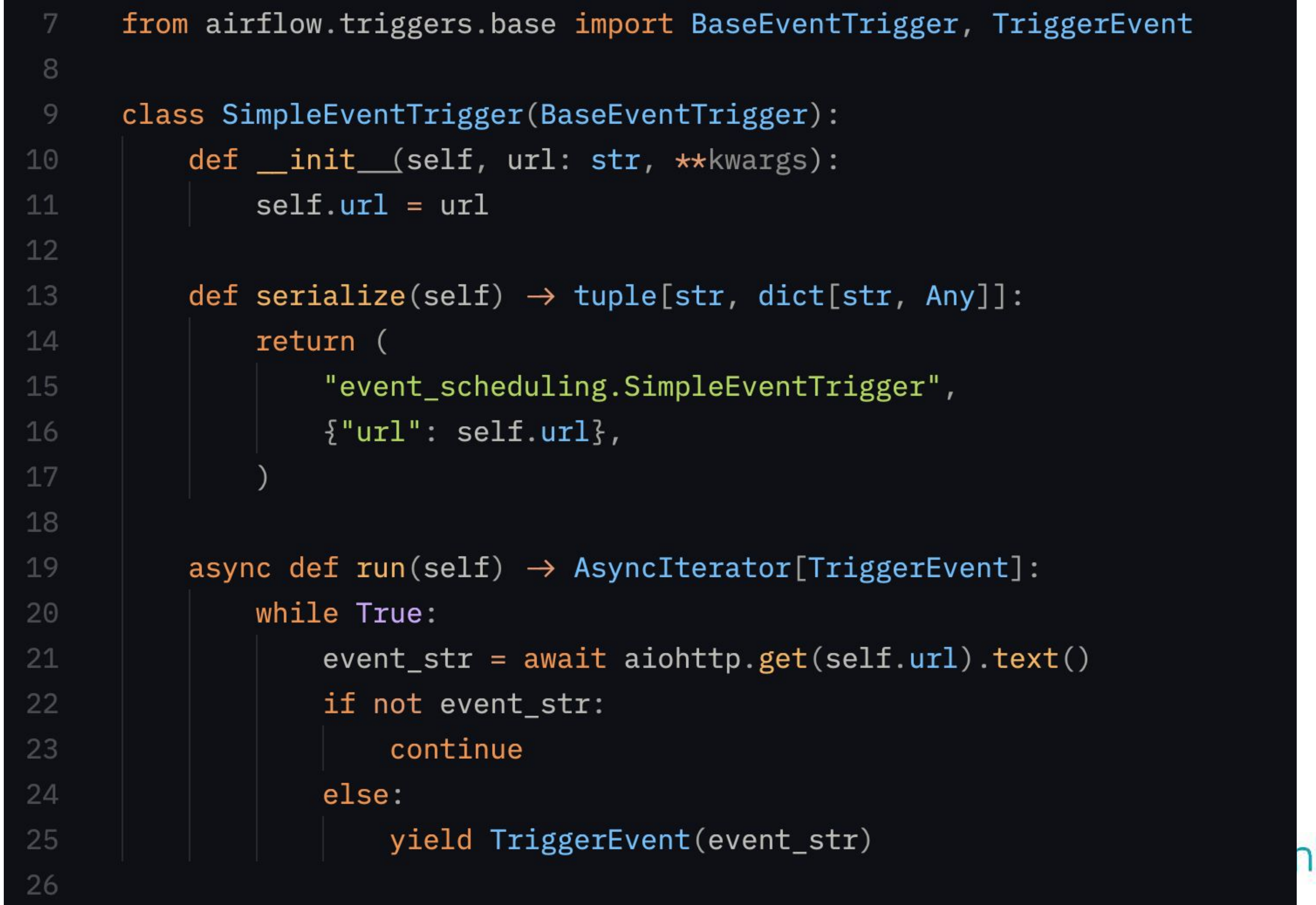
- **Inherits from BaseEventTrigger**

OBLIGATORY METHODS:

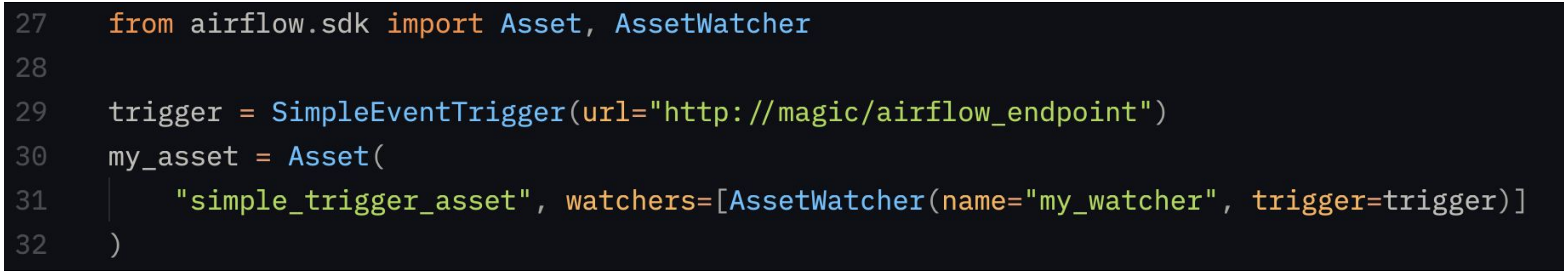
- **serialize**
  - Class path & arguments required for initialization must be serializable
- **run**
  - The actual trigger operation
- **__init__**

OPTIONAL METHODS:

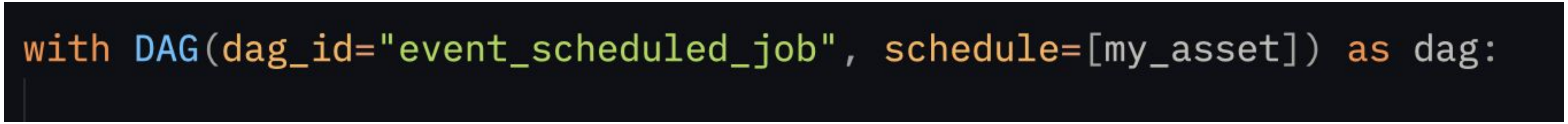
- **cleanup**
- **_set_context**



```python
from airflow.triggers.base import BaseEventTrigger, TriggerEvent

class SimpleEventTrigger(BaseEventTrigger):
    def __init__(self, url: str, **kwargs):
        self.url = url

    def serialize(self) -> tuple[str, dict[str, Any]]:
        return (
            "event_scheduling.SimpleEventTrigger",
            {"url": self.url},
        )

    async def run(self) -> AsyncIterator[TriggerEvent]:
        while True:
            event_str = await aiohttp.get(self.url).text()
            if not event_str:
                continue
            else:
                yield TriggerEvent(event_str)

```

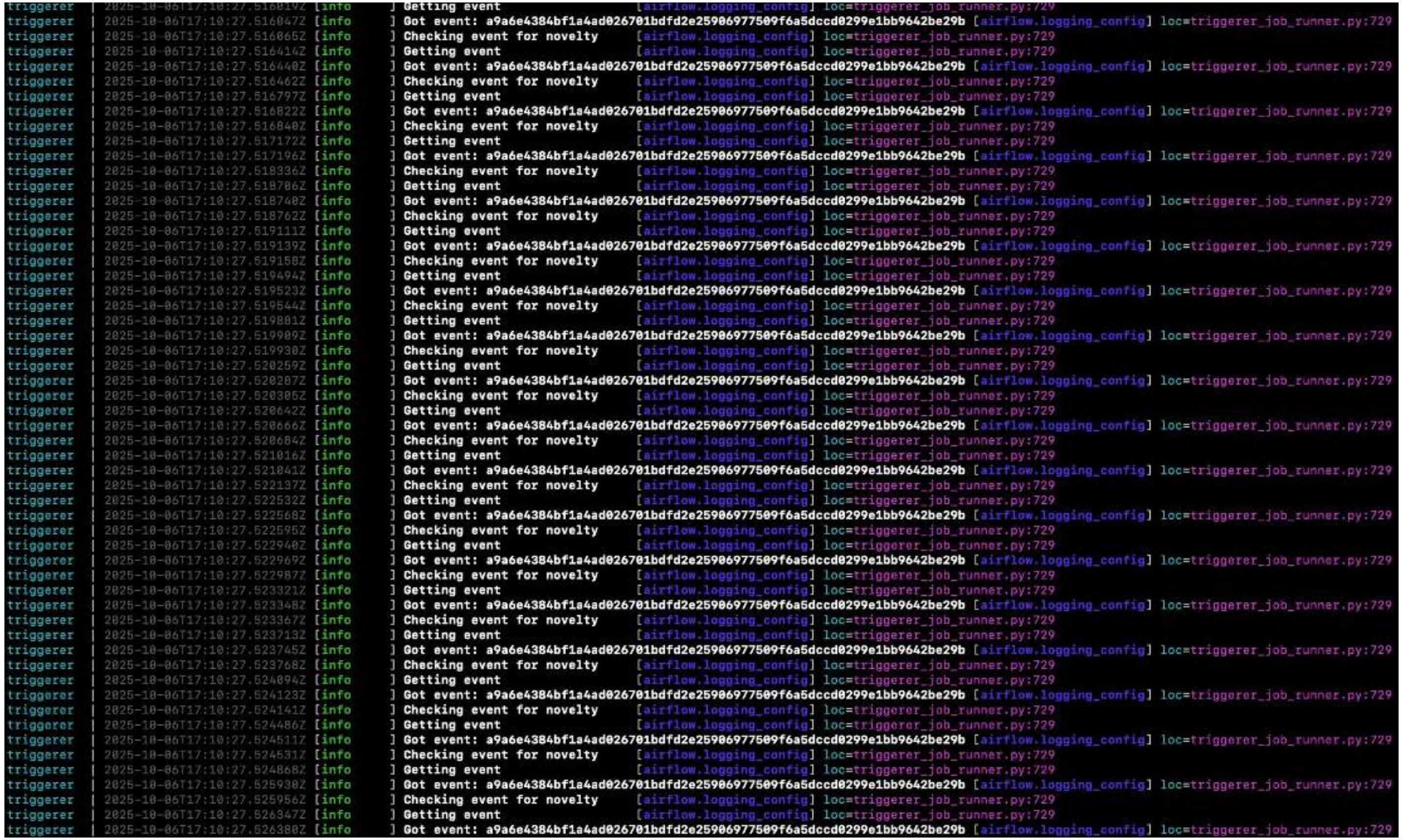


```python
from airflow.sdk import Asset, AssetWatcher

trigger = SimpleEventTrigger(url="http://magic/airflow_endpoint")
my_asset = Asset(
    "simple_trigger_asset", watchers=[AssetWatcher(name="my_watcher", trigger=trigger)]
)
```



```python
with DAG(dag_id="event_scheduled_job", schedule=[my_asset]) as dag:
```

Triggerer Process



```
triggerer  | 2025-10-06T17:10:27.516019Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.516047Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.516065Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.516414Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.516440Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.516462Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.516797Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.516822Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.516840Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.517172Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.517196Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.518336Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.518706Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.518740Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.518762Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519111Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519139Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519158Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519494Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519523Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519544Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519881Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519909Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.519930Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.520259Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.520287Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.520305Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.520642Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.520666Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.520684Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.521016Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.521041Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.522137Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.522532Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.522568Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.522595Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.522940Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.522969Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.522987Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.523321Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.523348Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.523367Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.523713Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.523745Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.523768Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.524094Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.524123Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.524141Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.524486Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.524511Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.524531Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.524868Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.525930Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.525956Z [info     ] Checking event for novelty     [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.526347Z [info     ] Getting event                  [airflow.logging_config] loc=triggerer_job_runner.py:729
triggerer  | 2025-10-06T17:10:27.526380Z [info     ] Got event: a9a6e4384bf1a4ad026701bdfd2e25906977509f6a5dccd0299e1bb9642be29b [airflow.logging_config] loc=triggerer_job_runner.py:729
```

Existing Event Triggers

Event Triggers:

- **KAFKA:** KafkaMessageQueueTrigger
- **GENERIC MQ:** MessageQueueTrigger

Base Triggers:

- **REDIS:** AwaitMessageTrigger
- **GOOGLE:** GCSBlobTrigger
- **AWS:** EC2StateSensorTrigger, S3KeyTrigger, …

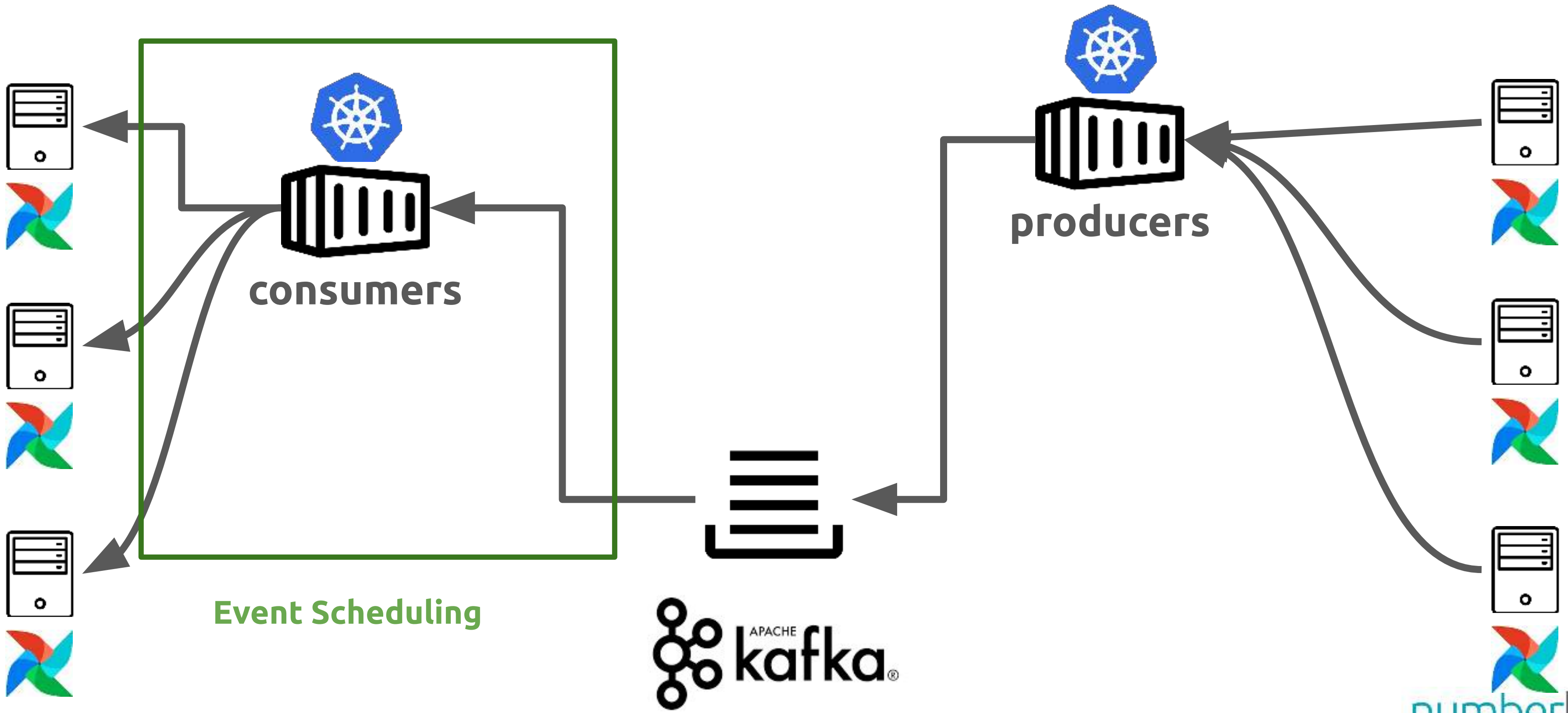
Pull based scheduling

Caveats:

- **“run” method is king**
  - Error handling
  - Pagination
  - Authentication
  - No blocking
  - Not heavy
- **Know where your Triggerers are running**
  - Do you have one ?
  - Is it sharing scheduler resources ?
- **Use cases outside message queues are “exploratory” if possible**

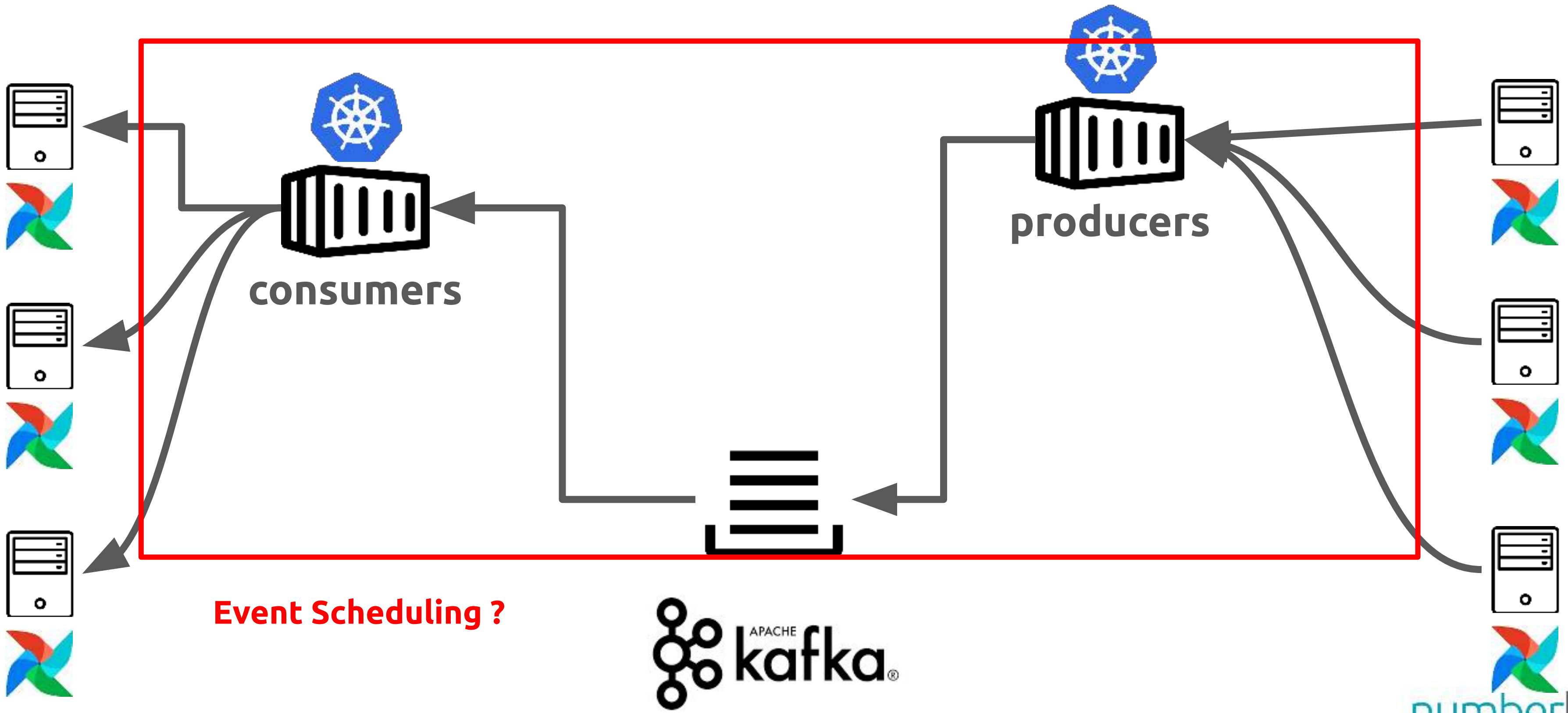
So… can we pull ?

Using Pull ?



numberly

Using Pull ?



numberly

Cost of pulling

With the queue ?

- **Constant triggerer polling**
  - Less accurate than push
  - Or noisy
- **Still need to push to queue**
  - Keep dedicated service?
  - Use custom “high frequency” DAGs ?

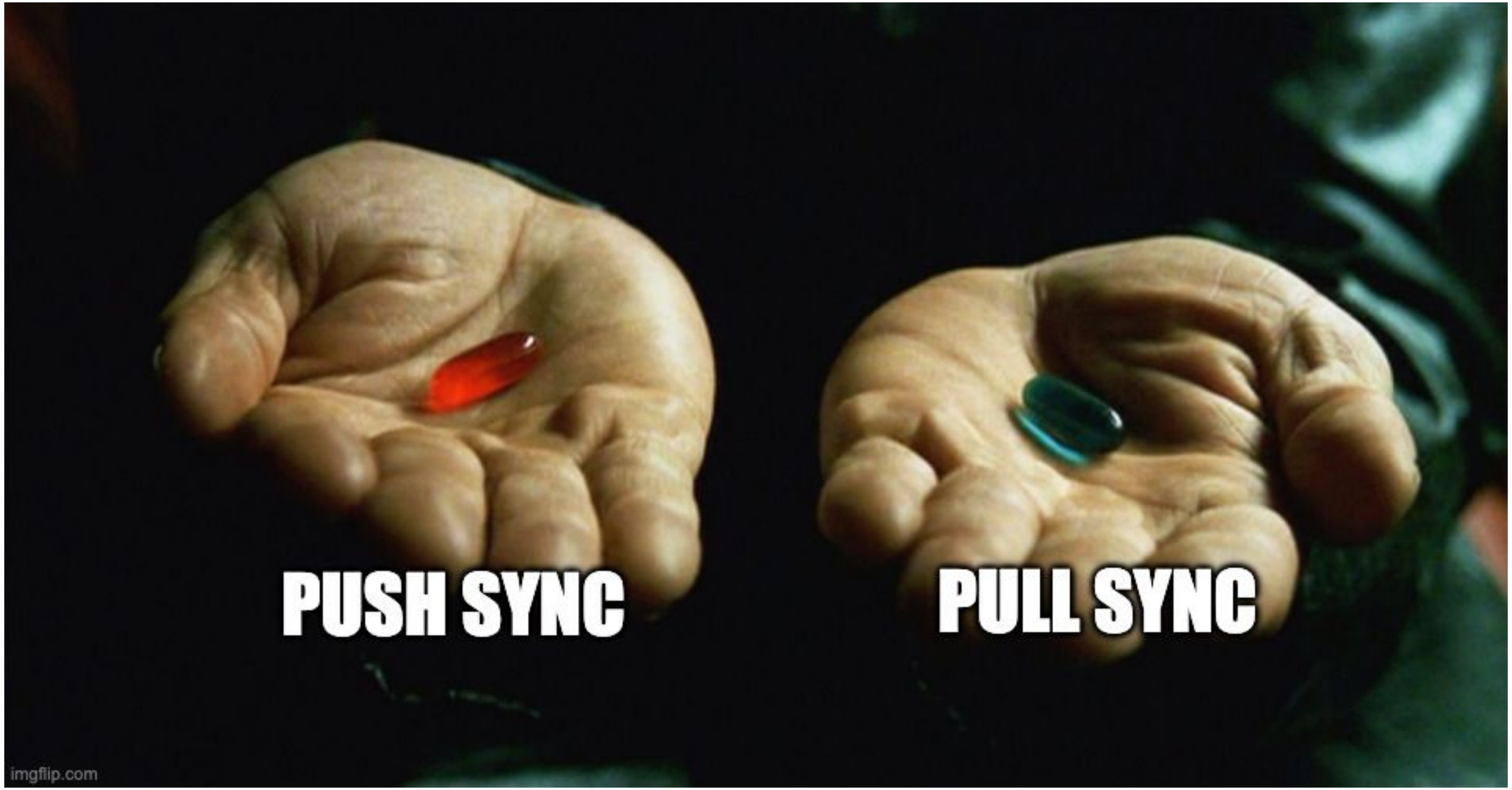
And without the queue?

- **API request + Airflow Variable**
- **O(n^2) network calls**
- **Even More Airflow Activity**

numberly

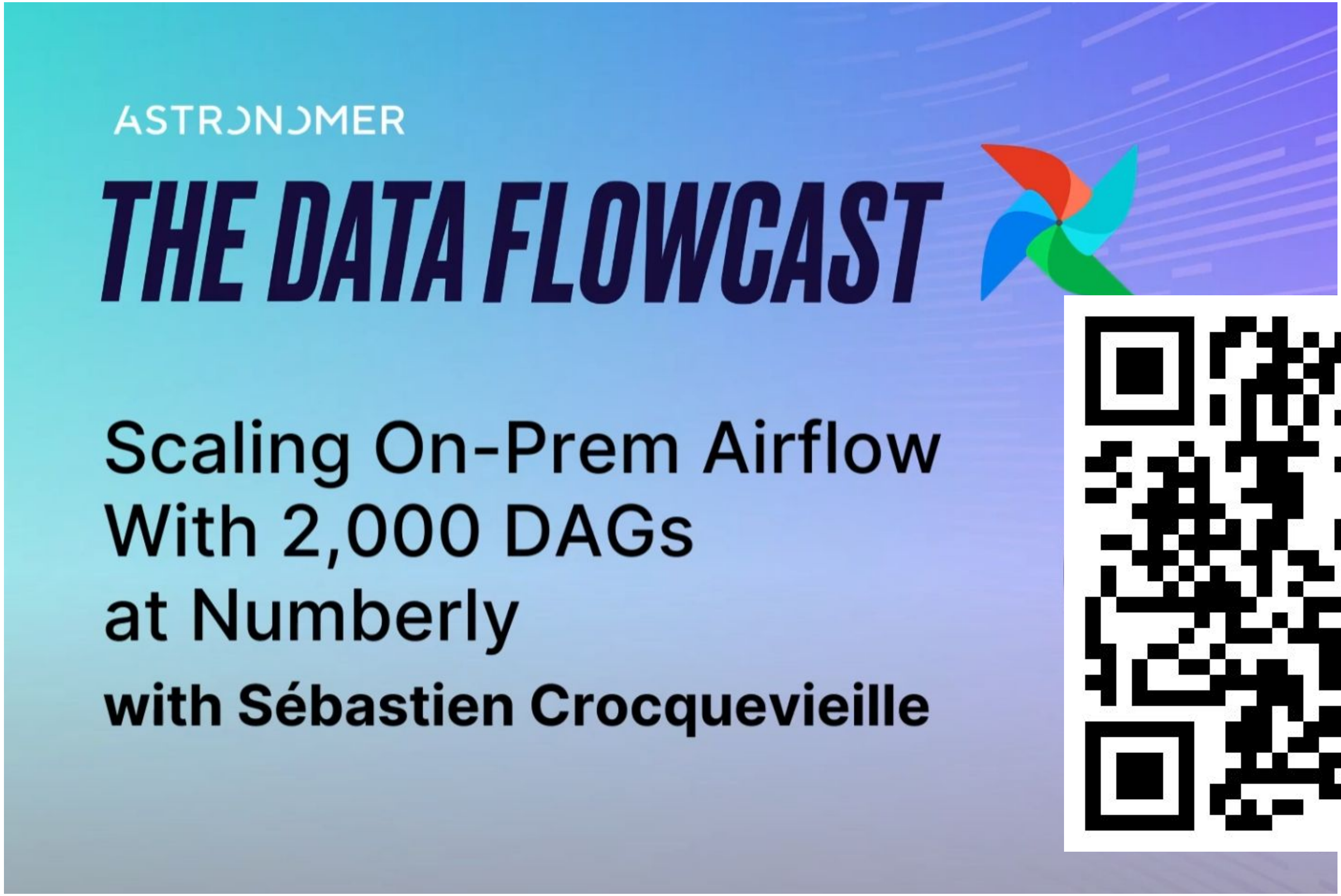
So the choice is

I’m still on Airflow 2 !



Question time !

Bonus



ASTRONOMER

THE DATA FLOWCAST

Scaling On-Prem Airflow With 2,000 DAGs at Numberly

with Sébastien Crocquevieille