Airflow Orchestration Core Concepts

Hayde Martinez
@haydemtzl

Airflow History

General Overview

Core Concepts

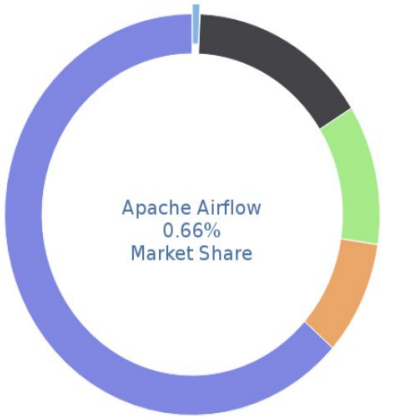
More resources

Airflow History

How was Airflow born and raised? :D

866

Companies using Apache Airflow



powered by enlyft.com

How it begin

In 2015, Airbnb experienced a **problem**. They were growing like crazy and had a massive amount of data that was only getting larger.

Now

- As of December 2020, Airflow has over 1,400 contributors, 11,230 commits, and 19,800 stars on Github.

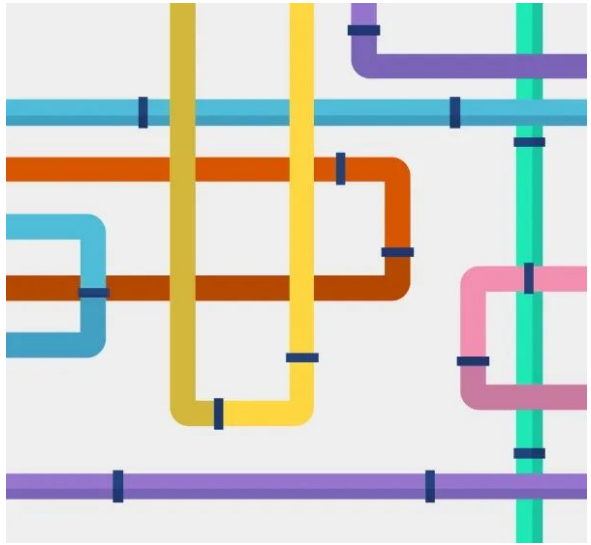
- Airflow is used by thousands of Data Engineering teams around the world and continues to be adopted as the community grows stronger.

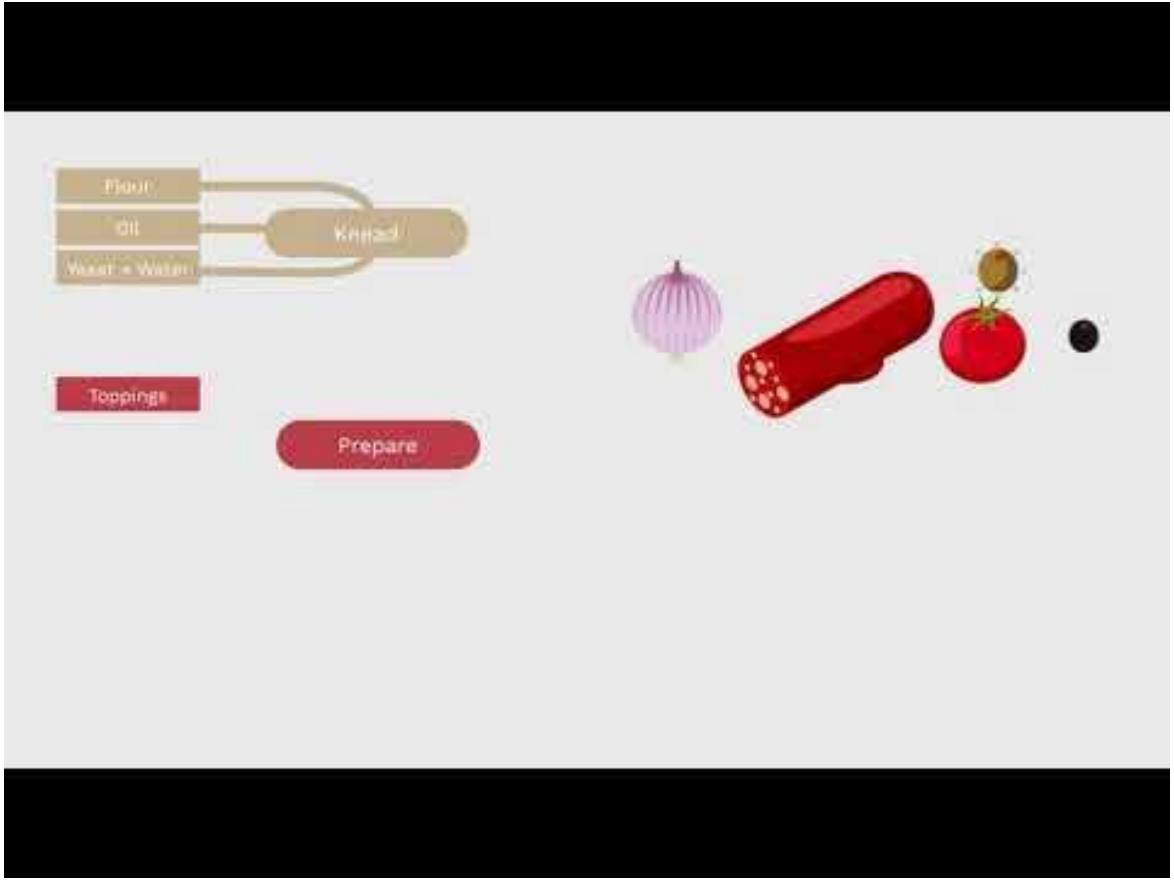
Airflow Overview

Ok, so, What's Airflow? :D

What 's Airflow?

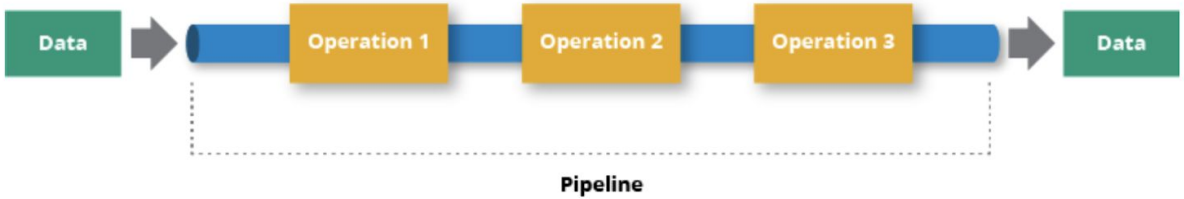
Apache Airflow is a **platform** for programmatically **authoring**, **scheduling**, and **monitoring** workflows. It is completely open source and is especially useful in architecting and orchestrating complex data **pipelines**.



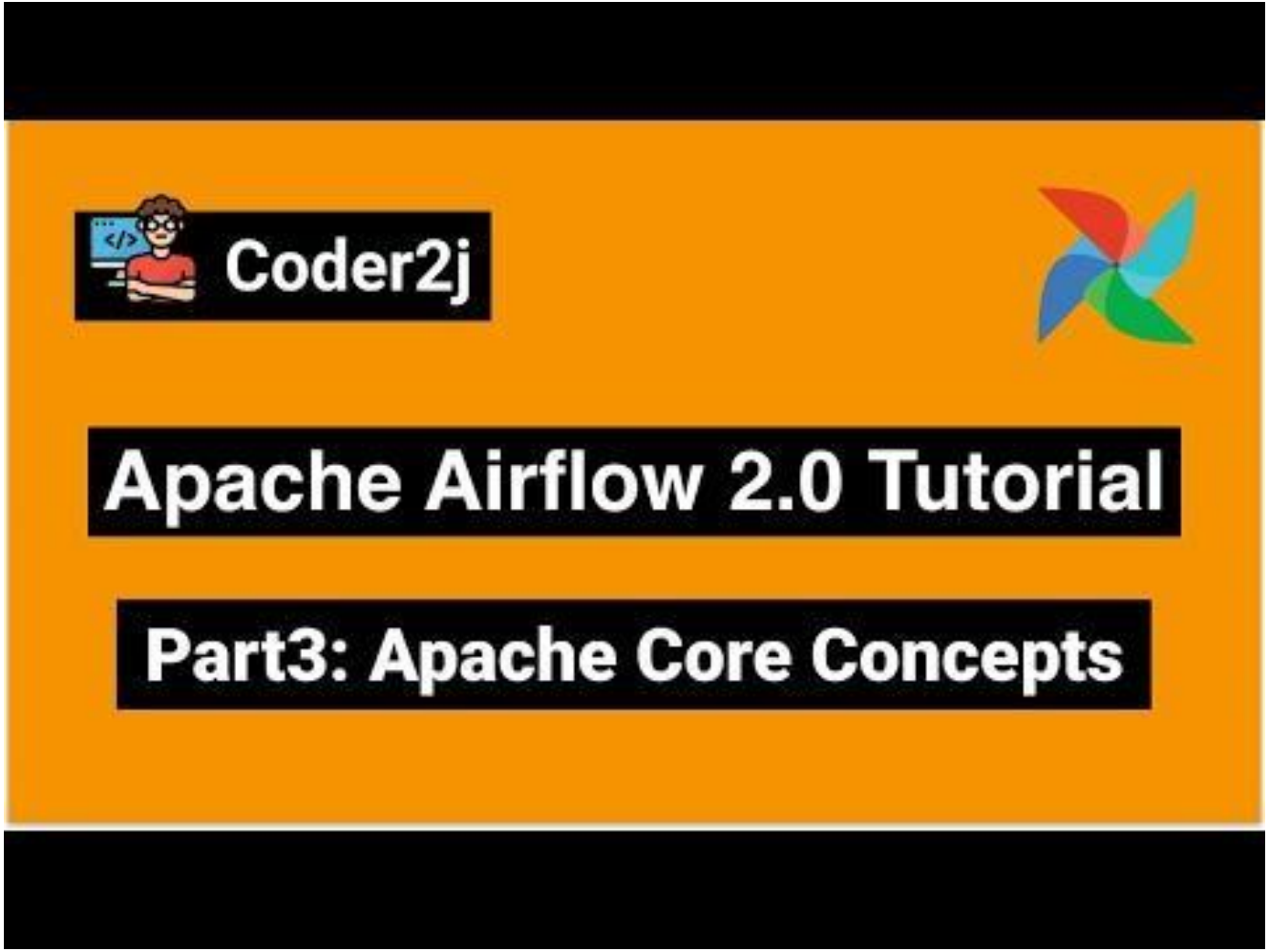

Flour
Oil
Yeast + Water
Kneaded
Toppings
Prepare

Remembering the concept: Data Pipeline

A **data pipeline** is a series of data processing steps. If the data is not currently loaded into the data platform, then it is ingested at the beginning of the pipeline.



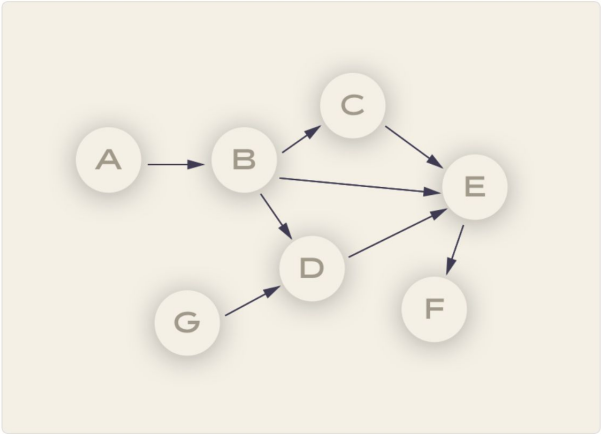
Core Concepts


Coder2j
Apache Airflow 2.0 Tutorial
Part3: Apache Core Concepts

DAG's Mathematical Background

Directed Graph: A directed graph is any graph where the vertices and edges have some sort of order or direction associated with them.

Directed Acyclic Graph: Finally, a directed acyclic graph is a directed graph without any cycles. A cycle is just a series of vertices that connect back to each other in a closed chain.



Node A could be the code for pulling data out of an API.

Node B could be the code for anonymizing the data and dropping any IP address.

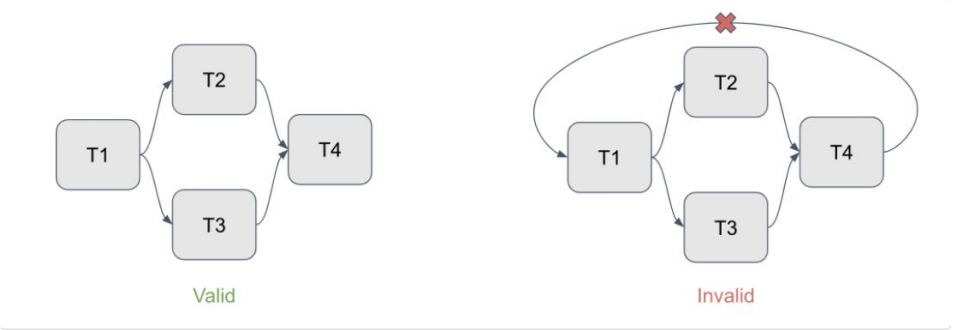
Node D could be the code for checking that no duplicate record ids exist.

Node E could be putting that data into a database.

Node F could be running a SQL query on the new tables to update a dashboard.

DAG's in Airflow

A Directed Acyclic Graph, or **DAG**, is a data pipeline defined in Python code. Each **DAG** represents a collection of tasks you want to run and is organized to show relationships between tasks in Airflow's UI.



Recap

DAGs are a natural fit for batch architecture - they allow you to model natural dependencies that come up in data processing without and force you to architect your workflow with a sense of "completion."

- **Directed** - If multiple tasks exist, each must have at least one defined upstream (previous) or downstream (subsequent) tasks, although they could easily have both.
- **Acyclic** - No task can create data that goes on to reference itself. This could cause an infinite loop that would be, um, it'd be bad. Don't do that.
- **Graph** - All tasks are laid out in a clear structure with discrete processes occurring at set points and clear relationships made to other tasks.

Tasks in Airflow

A Task is the **basic unit of execution** in Airflow. Tasks are arranged into **DAGs**, and then have upstream and downstream dependencies set between them into order to express the order they should run in.

```
first_task >> second_task >> [third_task, fourth_task]
```

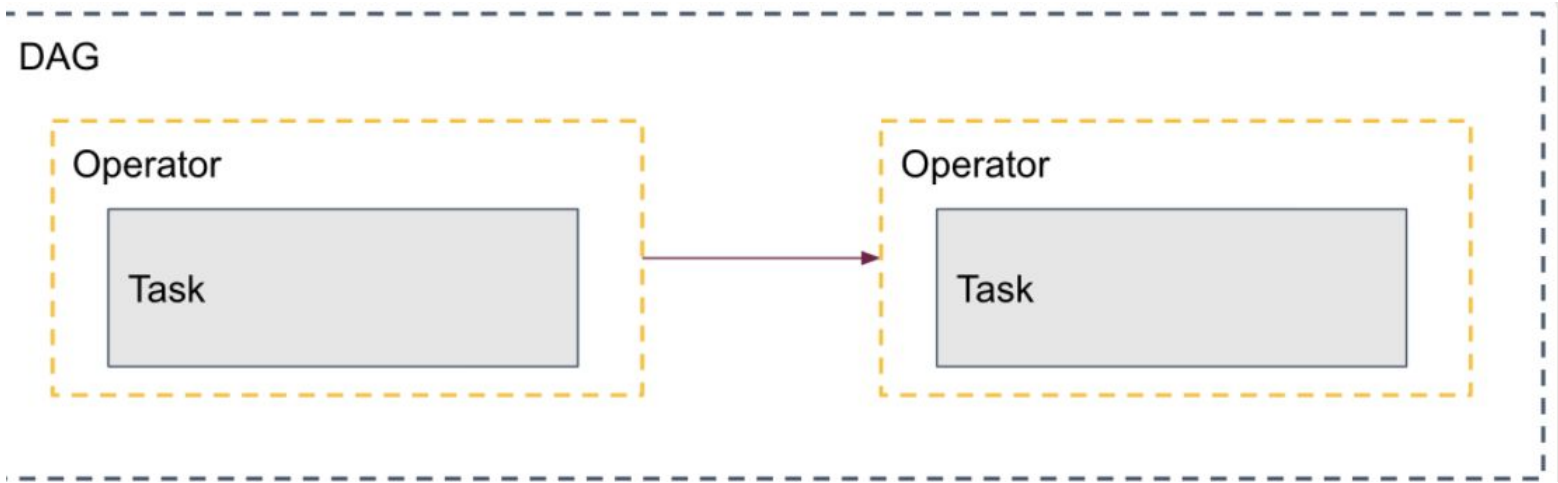
Operators in Airflow

Operators are the **building blocks of Airflow**, and determine the actual work that gets done. They can be thought of as a wrapper around a single task, or node of a DAG, that defines how that task will be run.

There are three main categories of operators:
* **Action Operators** execute a function, like the PythonOperator or BashOperator
* **Transfer Operators** move data from a source to a destination, like the S3ToRedshiftOperator
* **Sensor Operators** wait for something to happen, like the ExternalTaskSensor

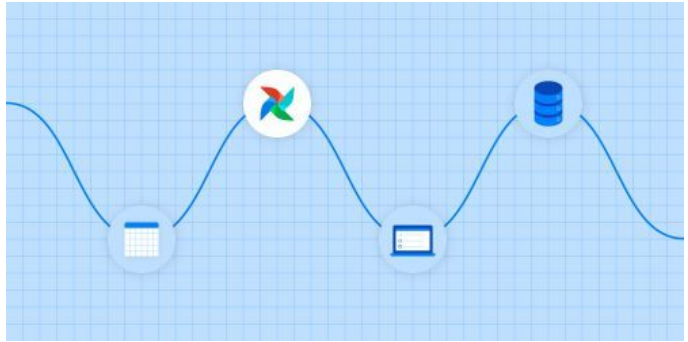
operator

```
file = open("myfile", "r")
print(f.read())
```


DAG
Operator
Task
Operator
Task

Hooks in Airflow

Hooks are **Airflow's way of interfacing with third-party systems.** They allow you to connect to external APIs and databases like Hive, S3, GCS, MySQL, Postgres, etc.



More Resources

I know you are all excited and want to learn and practice more!!!

Hands on resources

Airflow best practices:
https://github.com/jghoman/awesome-apache-airflow#best-practices-lessons-learned-and-cool-use-cases

Try airflow with no mess:
https://github.com/astronomer/astro-cli

Airflow oficial docs:
https://airflow.apache.org/docs/apache-airflow/stable/

Interesting further Reading

1. https://medium.com/hashmapinc/orchestration-and-dag-design-in-apache-airflow-two-approaches-35edd3eaf7c0
2. https://livebook.manning.com/book/data-pipelines-with-apache-airflow/chapter-5/v-5/1