Qu'est-ce qu'Apache Airflow ?

Apache Airflow est une plateforme open-source utilisée pour la création, la planification et la surveillance des **workflows** ou **pipelines** de données. Il permet d'orchestrer des tâches en définissant des dépendances entre elles.

Principales caractéristiques

— Définition de workflows sous forme de code Python (*DAG*).

— Planification automatique des tâches.

— Visualisation graphique des dépendances.

— Reprise automatique en cas d'échec.

Concepts clés

DAG (Directed Acyclic Graph)

Un DAG représente un workflow où les tâches sont exécutées dans un ordre défini, sans boucle.

Operator

Un **Operator** est une tâche unique. Par exemple :

— `PythonOperator` : exécute une fonction Python.

— `BashOperator` : exécute une commande Bash.

Task Instance

Une **Task Instance** est une exécution unique d'une tâche dans un DAG.

Scheduler et Executor

Le **Scheduler** planifie les tâches, et l'**Executor** les exécute.

Exemple simple de DAG

```python
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from datetime import datetime

def hello_world():
    print("Hello, Airflow!")
```



```
7
8  with DAG('example_dag', start_date=datetime(2025,11,7),
       schedule_interval='@daily') as dag:
9      task1 = PythonOperator(
10         task_id='hello_task',
11         python_callable=hello_world
12     )
```

Explication :

— Création d'un DAG nommé $example_dag$.

- Une tâche $hello_task qui exécute la fonction$ hello_world.

- Le DAG est planifié pour s'exécuter tous les jours.

Exemple de dépendances entre tâches

```
1  def task1_func():
2      print("T che 1")
3
4  def task2_func():
5      print("T che 2")
6
7  with DAG('dependency_dag', start_date=datetime(2025,11,7),
       schedule_interval='@daily') as dag:
8      task1 = PythonOperator(task_id='task1', python_callable=task1_func)
9      task2 = PythonOperator(task_id='task2', python_callable=task2_func)
10
11     task1 >> task2  # task2 sera ex cut e apr s task1
```

Challenge pratique facile

Objectif : Créer un DAG avec 3 tâches :

1. start_task : affiche "Début du workflow".
2. process_task : affiche "Traitement des données".
3. end_task : affiche "Fin du workflow".

Instructions :

— Définir les dépendances : start_task » process_task » end_task.

— Planifier le DAG pour une exécution quotidienne.

Ressources supplémentaires

— Documentation officielle : https://airflow.apache.org/docs/

— Tutoriel simple : `https://airflow.apache.org/docs/apache-airflow/stable/tutorial.html`

