# Airflow Docker Compose

데엔스쿨

- **사용 목적**
  - Airflow 시스템은 worker, database, scheduler 등 여러가지 서비스들로 구성되어 있음
  - 각 서비스들을 컨테이너 환경에서 구성할 수 있도록 해주는 파일 → Docker Compose

- **파일 작성**
  - 들여쓰기로 계층 속성을 정의하는 YAML 포맷을 사용
  - XML, JSON 형식과 비슷하지만 더 간결하고 사람이 이해하기 쉬운 작성 방식

- **실행**
  - ‘docker-compose.yaml’ 파일이 있는 위치에서 ‘docker compose up -d’ 실행

# Docker Compose 파일 분석

데엔스쿨

- x-airflow-common
  - Airflow 시스템에서의 공통 설정들을 정의하는 구간
  - &airflow-common으로 정의한 뒤, 각 서비스에서 <<: *airflow-common 형태로 재사용
  - 이미지 명, 버전, 환경 변수, 볼륨 마운트, 의존성, 설치 패키지 등을 설정
  - airflow.cfg 에 설정된 내용을 오버라이딩

- services
  - Airflow 시스템을 구성하는 컨테이너화 된 서비스들을 정의하는 구간

- volumes
  - 메타DB로 설정된 데이터베이스와 연결될 Docker Volume을 정의하는 구간

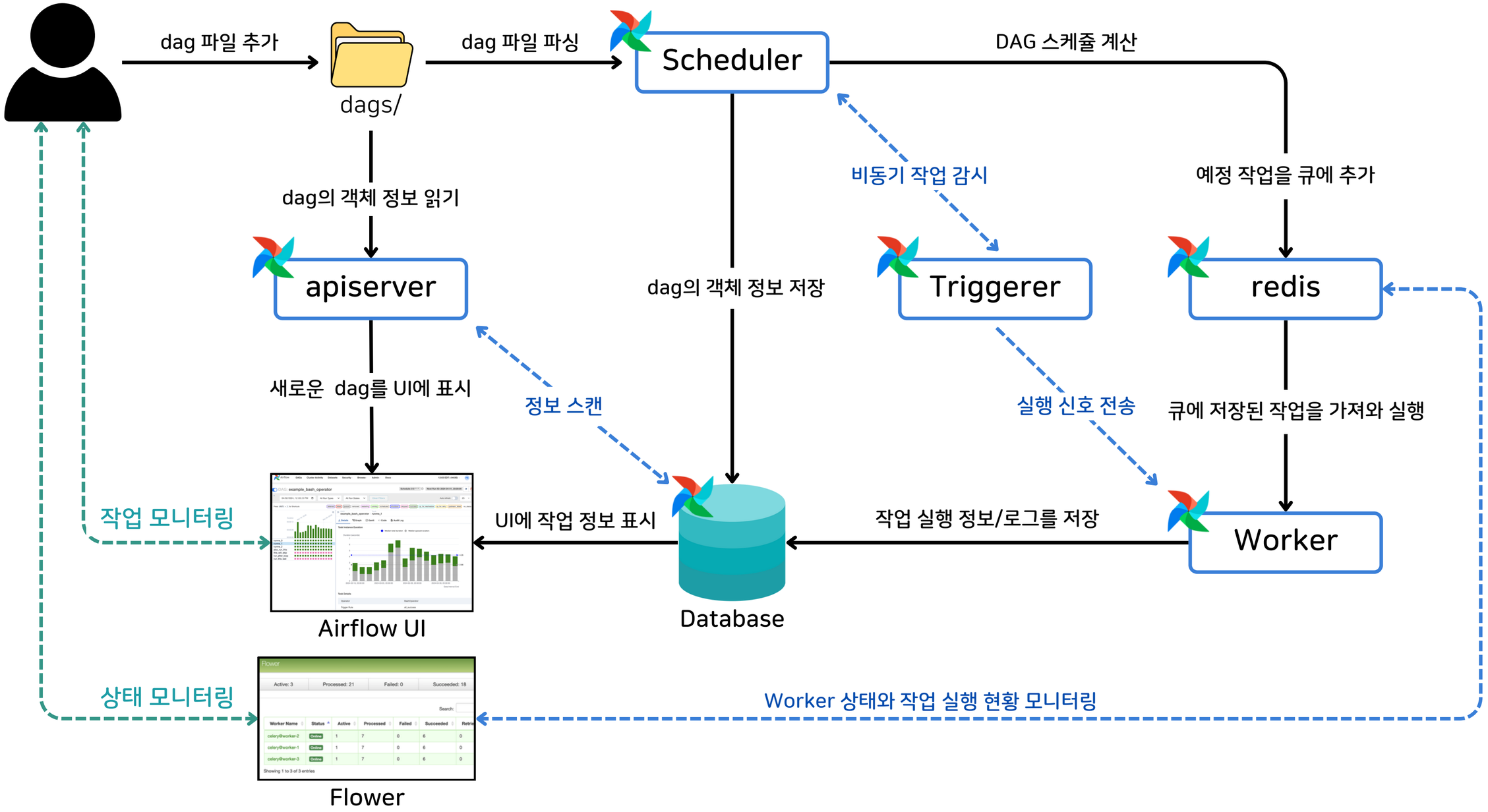
# Airflow 컨테이너의 구성

데엔스쿨

Airflow 2.10.5 기준

| 서비스 | 역할 | 필수 여부 |
|---|---|---|
| postgres | Airflow의 메타데이터, 작업 상태, 변수 등을 저장하고 관리 | ✅ |
| redis | CeleryExecutor를 사용시 작업 대기열(queue)을 관리하는 브로커 역할 | ✅ |
| airflow-webserver | Airflow의 웹 인터페이스(UI) 제공 | ✅ |
| airflow-scheduler | DAG를 주기적으로 스캔하여 정해진 시간에 Worker에 작업을 할당 | ✅ |
| airflow-worker | DAG에 명시된 실제 Task를 실행하고 결과를 처리 | ✅ |
| airflow-triggerer | 트리거 기반 비동기 작업 처리를 담당 | ✅ |
| airflow-init | Airflow 초기 설정 및 데이터베이스 초기화 수행 | ✅ |
| airflow-cli | Airflow 명령어를 실행하기 위한 임시 컨테이너 | ❌ |
| flower | Celery 워커의 상태를 모니터링하는 웹 UI 서비스 | ❌ |

# Airflow 2.x 버전 시스템 아키텍처



# Docker Compose를 통한 설치

데엔스쿨

- Airflow 공식 문서 → Running Airflow in Docker

1. Airflow 폴더 생성 및 이동
2. docker-compose.yaml 파일 다운로드
3. AIRFLOW__CORE__LOAD_EXAMPLES: 'false' 로 설정
4. WSL / powershell 터미널 실행
5. mkdir -p ./dags ./logs ./plugins ./config
6. echo -e "AIRFLOW_UID=$(id -u)" > .env
7. '.env' 파일 내용 변경 → AIRFLOW_UID=50000
8. docker compose up airflow-init
9. docker compose up -d