gusty

Chris Cardillo

4/17/23

Table of contents

Preface

Orchestration, or the routine scheduling and exection of dependent tasks, is a core component of modern data work. Orchestration continues to reach more and more data workers - it was originally a focus for data engineers, but it now permeates the work of data analysts, analytics engineers, data scientists, and machine learning engineers. The easier it is for any class of data worker to orchestrate their code, the easier it is for any member of an organization to derive value from the outputs of that code.

Flavors of Orchestration Code

Orchestration with Python is a vast and opinionated landscape, but there are three clear flavors of orchestration to have emerged over time:

1. **Object-oriented** orchestration, where tasks are objects and dependencies between tasks are handled with methods. Airflow's classic style is a good example of object-oriented orchestration.
2. **Decorative** orchestration, where tasks are functions and decorators are used to configure the tasks. Airflow's taskflow API and Dagster's entire API are good examples of decorative orchestation.
3. **"File as a Task" (FaaT)** orchestration, where tasks are files. Tools like Mage, dbt, and Orchest exemplify FaaT orchestration.

What is gusty?

gusty is a FaaT framework for Airflow, the absolute standard for orchestrators today. While other orchestrators natively support FaaT, Airflow is a Top-Level Apache Project with sustained development, a gigantic ecosystem of provider packages, and is offered as a hosted service by major public clouds and other Airflow-focused companies. If you are reading this, you're probably already familiar with - or using - Airflow.

gusty exists to make FaaT orchestration fun and easy using Airflow, allowing for FaaT DAGs to be incorporated in existing Airflow projects without any need to change existing work or Airflow code.

1 gusty Basics

Check out the gusty README for now.