Serving OGC API - Processes with pygeoapi and prefect

A high-level overview

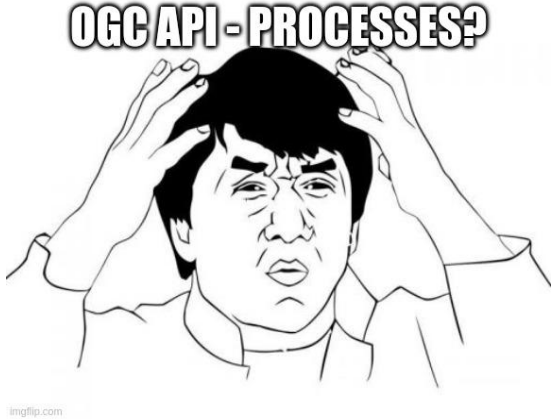
Ricardo Garcia Silva
ricardo.garcia.silva@gmail.com

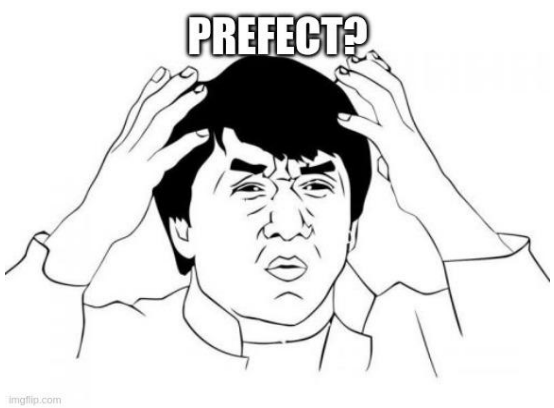
Introducing pygeoapi-prefect

A custom job manager for pygeoapi that integrates with Prefect and allows robust execution of OGC API - Processes requests


CUSTOM JOB MANAGER?
imgflip.com

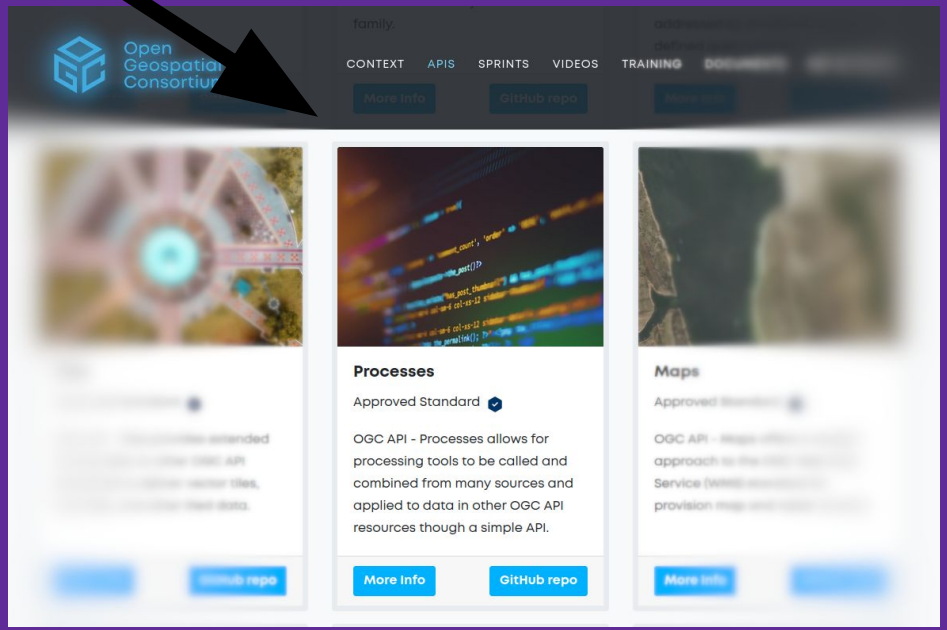

PYGEOAPI?
imgflip.com

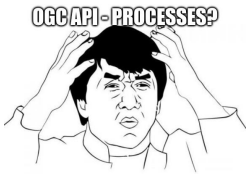

OGC API - PROCESSES?
imgflip.com


PREFECT?
imgflip.com

OGC API - Processes

A geospatial standard that specifies how servers can expose computational tasks via a web API

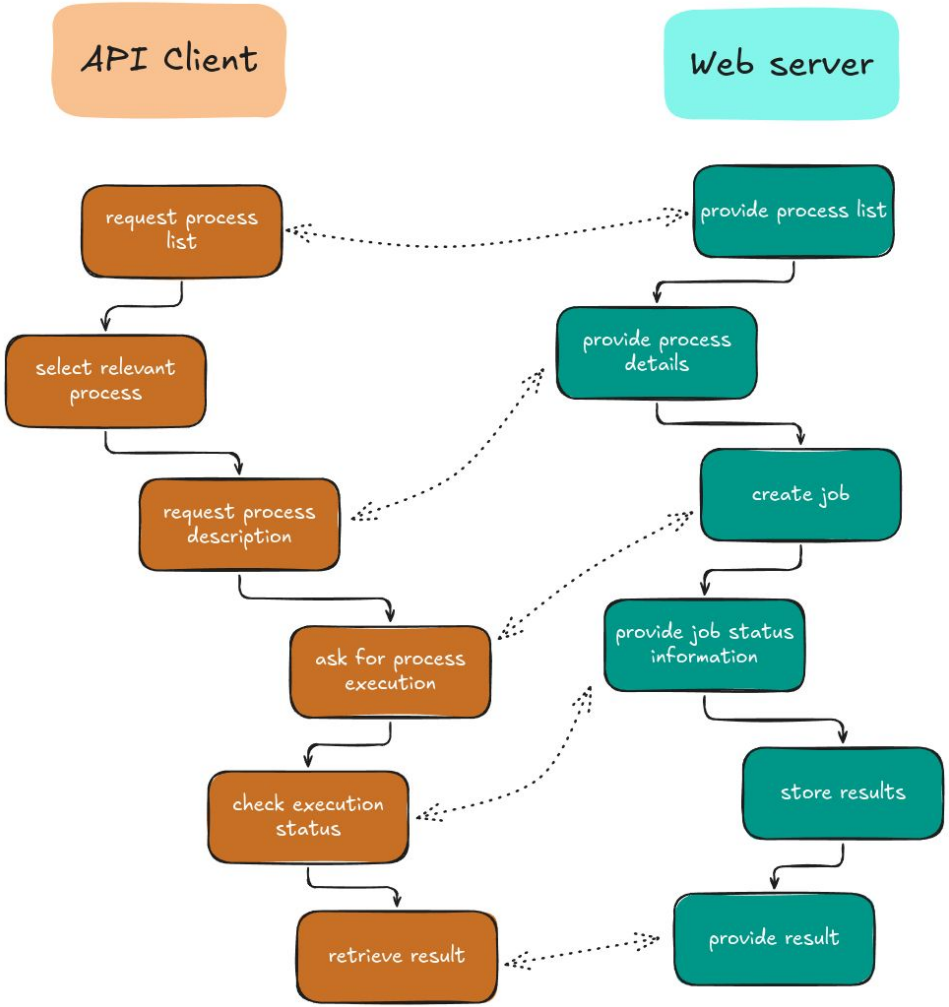




https://ogcapi.ogc.org/processes/

OGC API - Processes: Overview

- Servers expose **processes**
- A process describes a computation that the server is able to run
- Processes may take **inputs** and generate **outputs**
- A client requests **execution** of a process, providing relevant inputs
- Execution can be **synchronous** or **asynchronous**
- When execution is requested, the server eventually spawns a **job**
- Upon successful completion, job outputs are made available to the client


API Client
Web server
request process list
provide process list
select relevant process
provide process details
request process description
create job
ask for process execution
provide job status information
check execution status
store results
retrieve result
provide result

OGC API - Processes: example process description

curl -s https://demo.pygeoapi.io/stable/processes/hello-world


```
{
  "id": "hello-world",
  "jobControlOptions": ["sync-execute"],
  "inputs": {
    "name": {
      "schema": {"type": "string"},
      "minOccurs": 1,
      "maxOccurs": 1,
    }
  },
  "outputs": {
    "echo": {
      "schema": {"type": "object", "contentMediaType": "application/json"}
    }
  }
}
```

OGC API - Processes: execution request

Execution request

```
curl -s -X POST \
    https://demo.pygeoapi.io/stable/processes/hello-world/execution \
    -H "Content-Type: application/json" \
    -d '{"inputs": {"name": "ric"}}'
```

Response

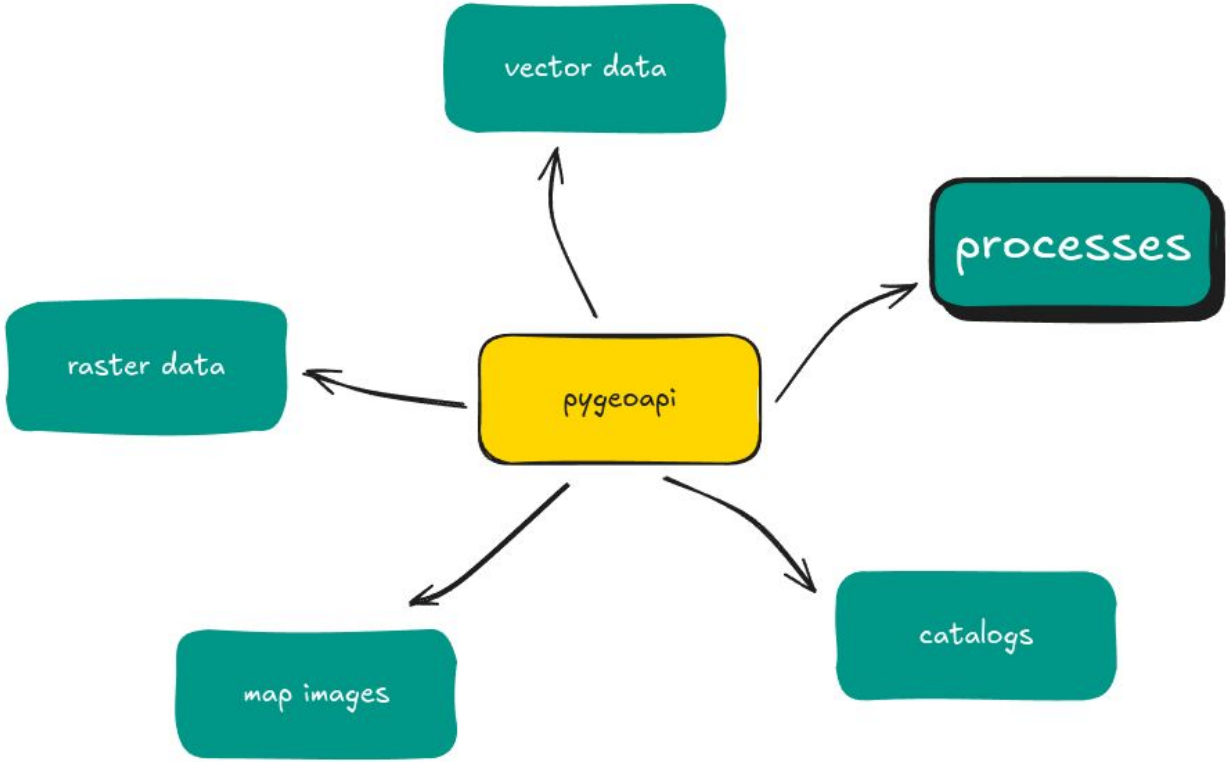
```
{"id":"echo","value":"Hello ric!"}
```

pygeoapi

Python-based web application that implements a number of OGC API standards

- Open source
- MIT license
- OGC Reference implementation
- Easily extendable through plugins
- Deployed in many places in scientific and governmental sectors

https://pygeoapi.io


vector data
processes
raster data
pygeoapi
map images
catalogs

pygeoapi - OAProc support

Process requests handled via a job manager, which can be provided by plugins

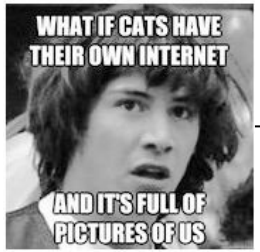
multiple builtin implementations

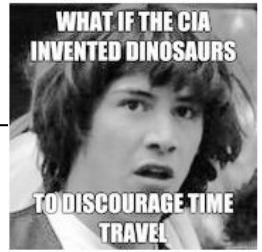
- TinyDB (default)
- PostgreSQL
- mongoDB

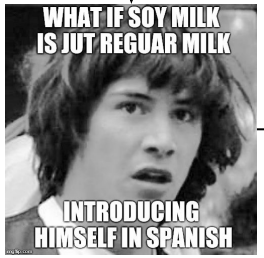
pygeoapi - OAProc support

The builtin implementations, while serviceable, offer limited support for robust operations

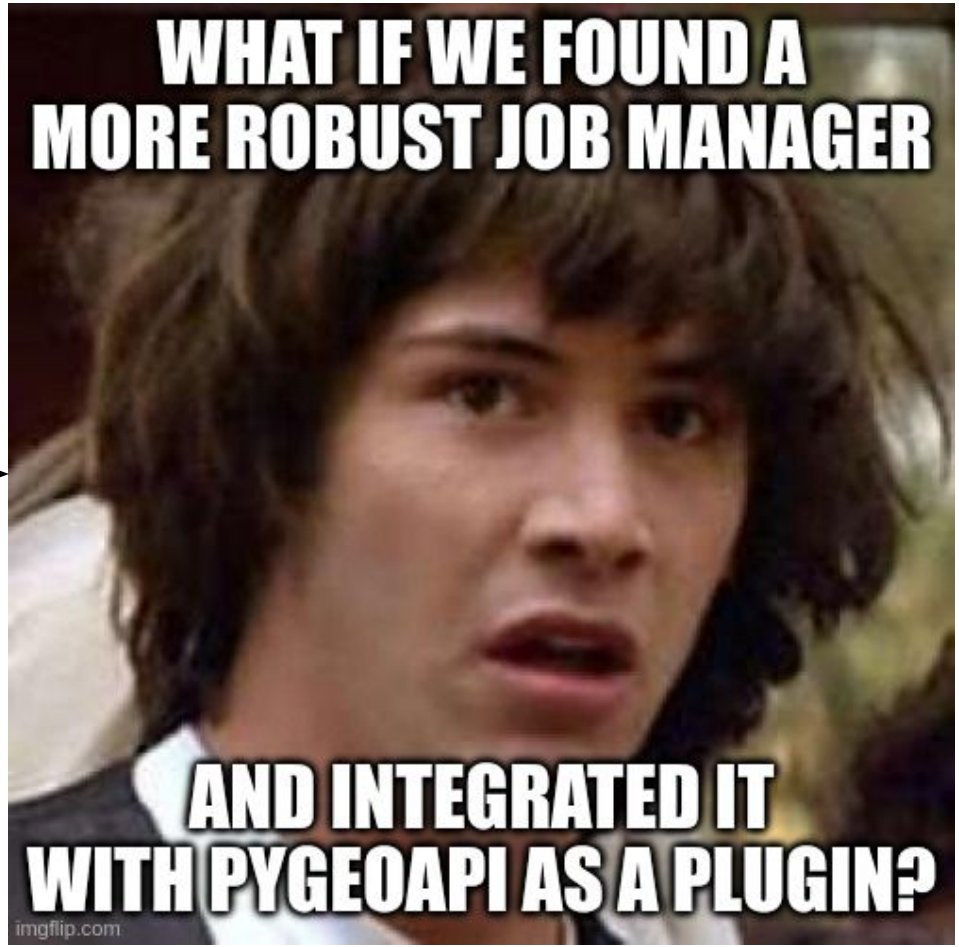
- No way to control job concurrency
- Not possible to horizontally scale compute nor result storage
- Job execution is always run on the same environment as pygeoapi server
- Basic job monitoring capabilities
- No error/retry capabilities


WHAT IF CATS HAVE THEIR OWN INTERNET
AND IT'S FULL OF PICTURES OF US


WHAT IF THE CIA INVENTED DINOSAURS
TO DISCOURAGE TIME TRAVEL


WHAT IF SOY MILK IS JUT REGUAR MILK
INTRODUCING HIMSELF IN SPANISH

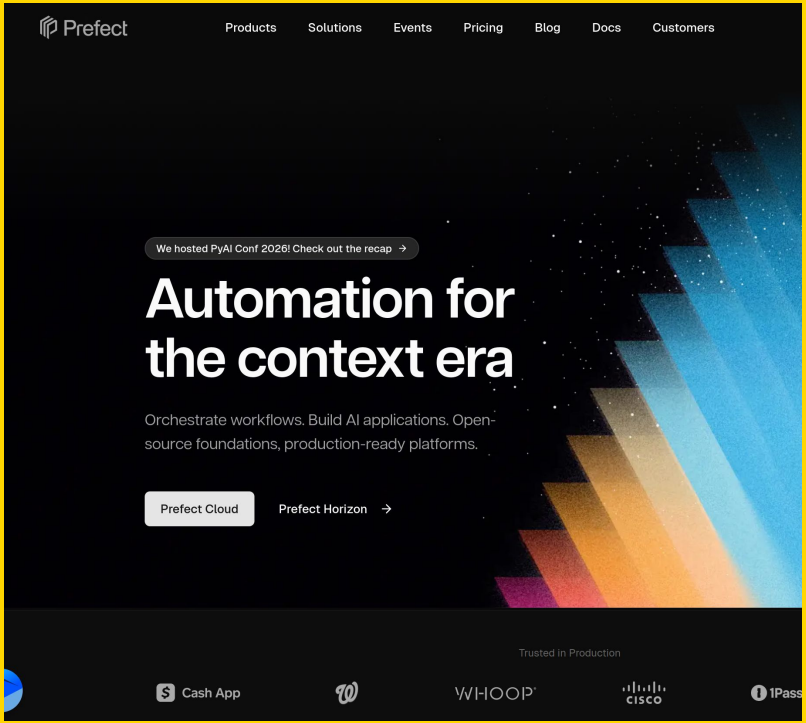

WHAT IF GOD WAS ONE OF US
JUST A STRANGER ON A BUS


WHAT IF WE FOUND A MORE ROBUST JOB MANAGER
AND INTEGRATED IT WITH PYGEOAPI AS A PLUGIN?
imgflip.com

Prefect

A robust orchestration engine for handling data pipelines at scale

- Open source
- Separates control plane from data plane
- Can be managed or self-hosted
- Multiple ways to set-up
- Built for scaling



https://www.prefect.io/

Prefect: overview

- **Flows** are Python functions that describe data processing algorithms
- Flows can be **deployed** to **work pools**
- There are different types of work pools, like multi-processing, docker, k8s, etc.
- Flows can be **scheduled** for execution and also executed **on-demand**
- Whenever a flow is to be executed, a **flow run** is generated
- The **scheduler** takes care of submitting flow runs to their respective work pool
- **Workers** consume job run execution requests for their respective work pool
- flow runs are recorded in the monitoring DB and can be inspected

Prefect: relevant features

- Robust asynchronous execution model
- Handles result storage transparently, with support for multiple backends (local filesystem, cloud, etc.)
- Allows horizontal scaling of workers
- Provides multiple ways to control resource usage and concurrency
- Allows dealing with errors and retries
- Provides a rich UI for monitoring both real-time and historical operations

Prefect is a ~~perfect~~
nice fit for managing
pygeoapi OAProc
execution requests

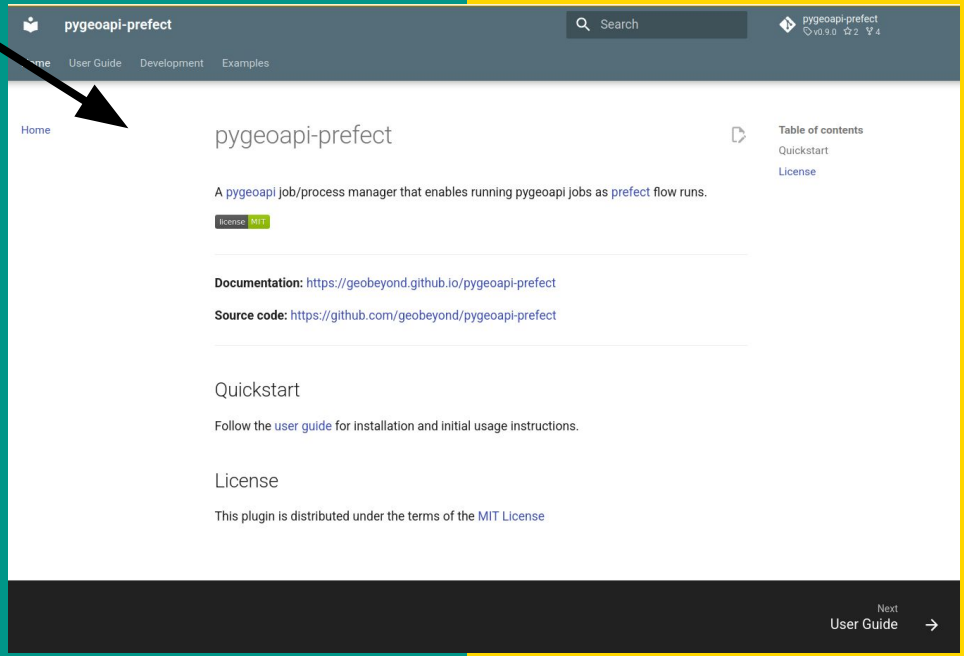
Introducing pygeoapi-prefect

https://geobeyond.github.io/pygeoapi-prefect/

A custom job manager for pygeoapi that integrates with Prefect and allows robust execution of OGC API - Processes requests

pygeoapi-prefect

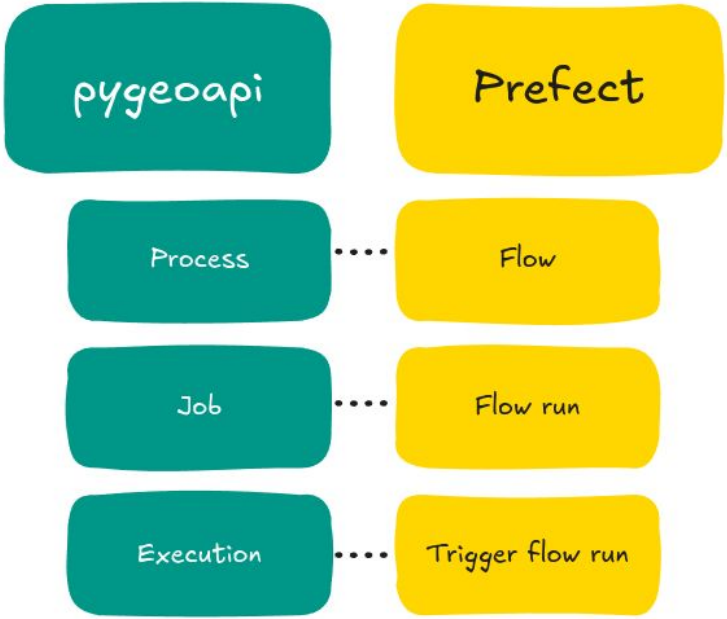
- A plugin for pygeoapi
- Provides a job manager for OGC API - Processes
- Integrates Prefect and pygeoapi together
- Open source
- MIT License



https://geobeyond.github.io/pygeoapi-prefect/

pygeoapi-prefect: concept mapping

- Process description and process/flow mapping is specified via configuration
- Job/Flow run mapping is done via common identifier
- Flow run status and information retrieved from Prefect DB dynamically
- Flow run results retrieved from Prefect when needed



pygeoapi-prefect

This work is funded by Geobeyond

- Italian-based geospatial company
- Core contributor to and service provider for pygeoapi
- Large experience deploying both pygeoapi and prefect in the wild

https://www.geobeyond.it/

Geobeyond

Making Geospatial Happen

pygeoapi-prefect: key features

Enables async execution by default (sync is also available)

Works with normal pygeoapi processors - Can be used as a drop-in replacement for the builtin job manager(s)

Works with deployed Prefect flows - Unlocks most Prefect features:

- Run jobs inside ephemeral docker containers
- Store job results in S3 or other dynamic storage
- Scale number of workers
- Limit concurrency
- Use Prefect UI to monitor execution and history

pygeoapi-prefect: TODO

Add more examples to the user guide

Complete OAProc - Part 1 support

- ***Dismiss*** feature - Cancel a running job and/or remove results
- ***Callback*** feature - Send results to a client-specified URL when they are ready

Plan support for future OAProc extensions:

- Workflows
- CRUD processes
- Quotations and billing

Serving OGC API - Processes with pygeoapi and prefect

A high-level overview

Ricardo Garcia Silva
ricardo.garcia.silva@gmail.com