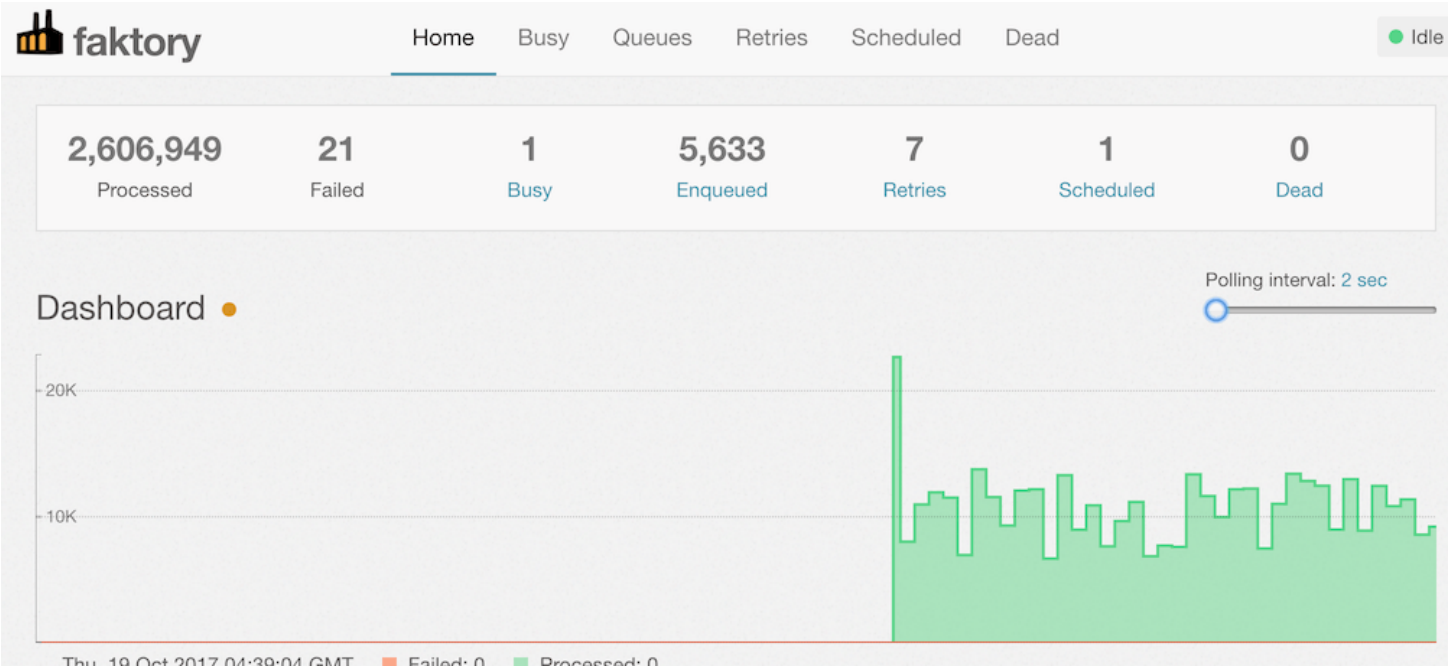
Faktory 0.9.0 - Hello, Redis!

2018-10-16

Faktory is my new background job system which brings Sidekiq-like background jobs to all languages. Want Sidekiq in PHP? Python? JavaScript? You got it!



Faktory 0.9 has just been released with a major architectural overhaul. I've replaced the previous storage engine, RocksDB, with Redis. This change had hugely important, very good downstream effects!

Want to try Faktory? The wiki has everything you need!

Why Redis?

So why did I replace RocksDB? RocksDB has two advantages over Redis:

- it is very fast
- it is embeddable (which means it links into your process and provides an API your code calls directly).

The problem is that **everything else** is a disadvantage:

- It has dozens of barely documented options
- It's written in C++, meaning a much more complex build process to embed it into a Go process
- It has virtually no usage outside of large corporate silos (e.g. Facebook, Yahoo, Pinterest) where internal developers can learn and maintain it.
- Its API is key/value only, very low-level, with no higher-level structures supported.

Redis, on the other hand:

- has lots of great documentation.
- used everywhere, by everyone.
- has lots of commercial support and offerings.
- has a broad, stable, high-level API
- is not much slower, only 10-20%. It'll still handle 1000s of jobs/sec.
- and lastly, I have years of experience supporting it in production!

Now instead of a directory full of obscure files, we have a single `faktory.rdb` file with all data. Now we can use `redis-cli` and RDB tools to introspect the datastore, monitor commands in real-time, use existing Redis monitoring integrations to monitor Faktory's storage. **When we build on the shoulders of giants, the view is pretty good.**

The only issue I had to solve is how to "embed" Redis? I can't link it into the process directly so instead I updated Faktory to start Redis as a child process and maintain a named socket to it. By keeping it local, using a named socket, we ensure our usage of Redis is both high performance and secure (since Redis does not listen on the network).

What's Upcoming?

With 0.9 out, I'm very close to releasing Faktory Pro, my first commercially supported version of Faktory! The initial feature list:

1. Cron jobs - enqueue jobs on a periodic schedule
2. Expiring jobs - throw away jobs which have passed a given expiration date
3. Redis Gateway - expose Redis on a TCP port so you can attach a replica to it for real-time replication

This feature list is very likely to grow over time.

See the Faktory wiki for how to install and lots of documentation. Join us in the chatroom or open an issue if you have questions.