GOLAB

Easily build healthy and reliable web apps using metrics and tracing

Thorsten Essig
Carsten Dietrich

Who we are

Carsten Dietrich
E: carsten.dietrich@aoe.com
T: @carstencodes

- Software Developer at AOE
- ~2 years of Go experience
- Strong PHP Background

Thorsten Essig
E: thorsten.essig@aoe.com
T: @thorstenessig

- Software Developer at AOE
- ~3 years of Go experience
- Focus on E-Commerce and Developer Trainings

Agenda

- Challenges of modern web applications
- What is OpenCensus?
- Live Coding
    - The Flamingo Framework
    - Writing an application from scratch
- Q&A

Challenges

- Users expect applications to be always available and response within a glimpse of a second
- Increased complexity in (web) applications
- Service Level Agreements
- Microservice architecture

How to predict outages?

- Collect and interpret data (in real time)
- Monitoring is key
- There is more than just logging
- Health checks
- Metrics

A metric is a measure of software/business characteristics which are quantifiable or countable.

Metrics

- **Application Metrics** (e.g. Webshop)
  - Number of user registrations
  - Number of started / finished checkouts
- **Vendor Metrics** (e.g. Go Webframework)
  - Go Routine count
  - Allocated memory
- **Platform Metrics** (e.g. Kubernetes cluster)
  - Requests per minute
  - Storage usage

Types of metrics

- **Counter**

  A cumulative metric that only ever increases

  (E.g. requests served, tasks completed, errors occurred)
- **Gauge**

  A metric that can arbitrarily go up or down

  (E.g. temperature, memory usage)
- **Histogram**

  Binned measurement of a continuous variable

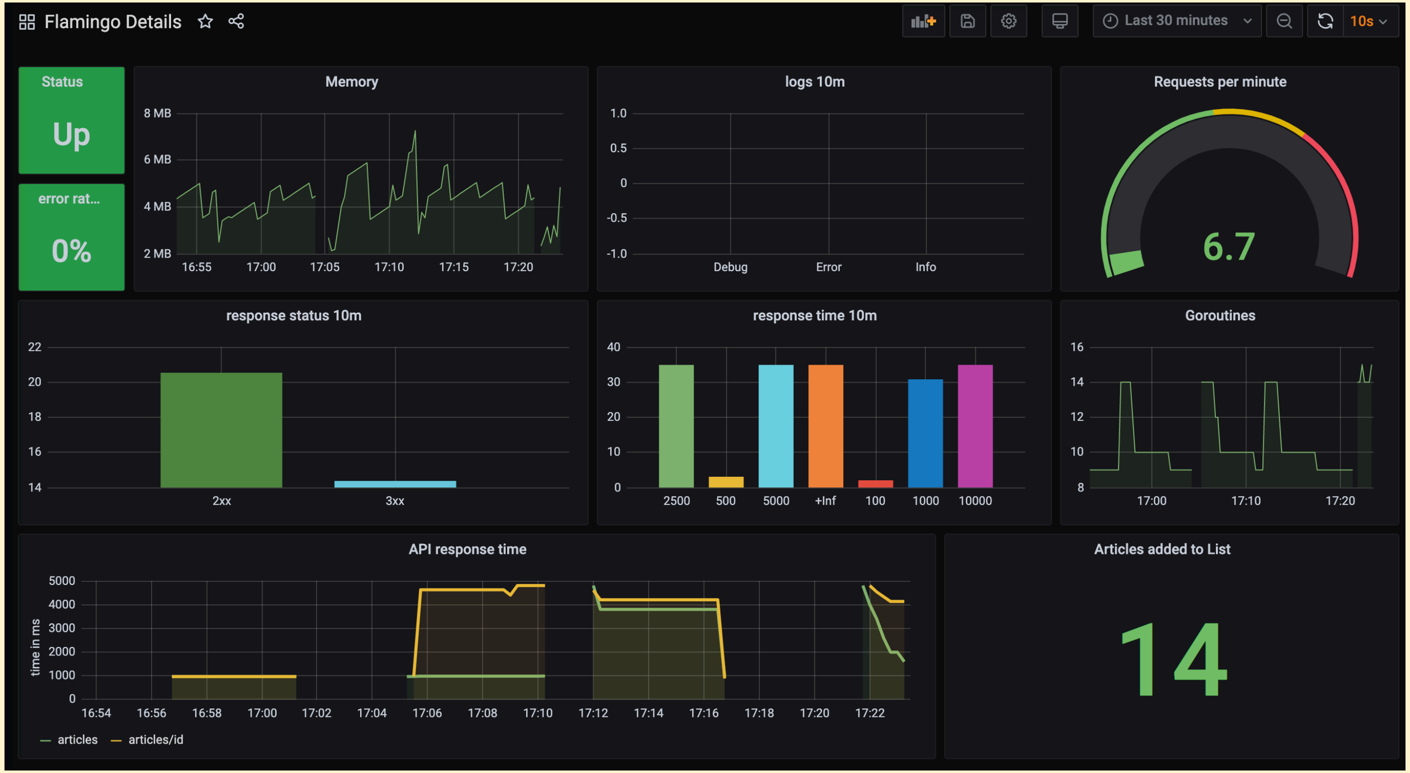
  (E.g. latency, request duration, age)

Metrics in the wild

```
# HELP flamingo_zap_logs Count of logs
# TYPE flamingo_zap_logs counter
flamingo_zap_logs{area="root",level="Debug"} 3
flamingo_zap_logs{area="root",level="Error"} 2
flamingo_zap_logs{area="root",level="Info"} 3
# HELP process_cpu_goroutines Number of goroutines that currently exist
# TYPE process_cpu_goroutines gauge
process_cpu_goroutines 15
# HELP process_heap_alloc Process heap allocation
# TYPE process_heap_alloc gauge
process_heap_alloc 3.049616e+06
# HELP process_heap_objects The number of objects allocated on the heap
# TYPE process_heap_objects gauge
process_heap_objects 14576
```

Metrics in Grafana

24/10/2020



How to Find Performance issues?

- Collect and interpret data (in real time)
- Use Metrics
- Find time consuming parts of your app
- Keep an overview over the global architecture
- Tracing

Tracing is understanding the path of a request as it is traversing through the parts/layers of your application or infrastructure

Tracing - Some Definitions

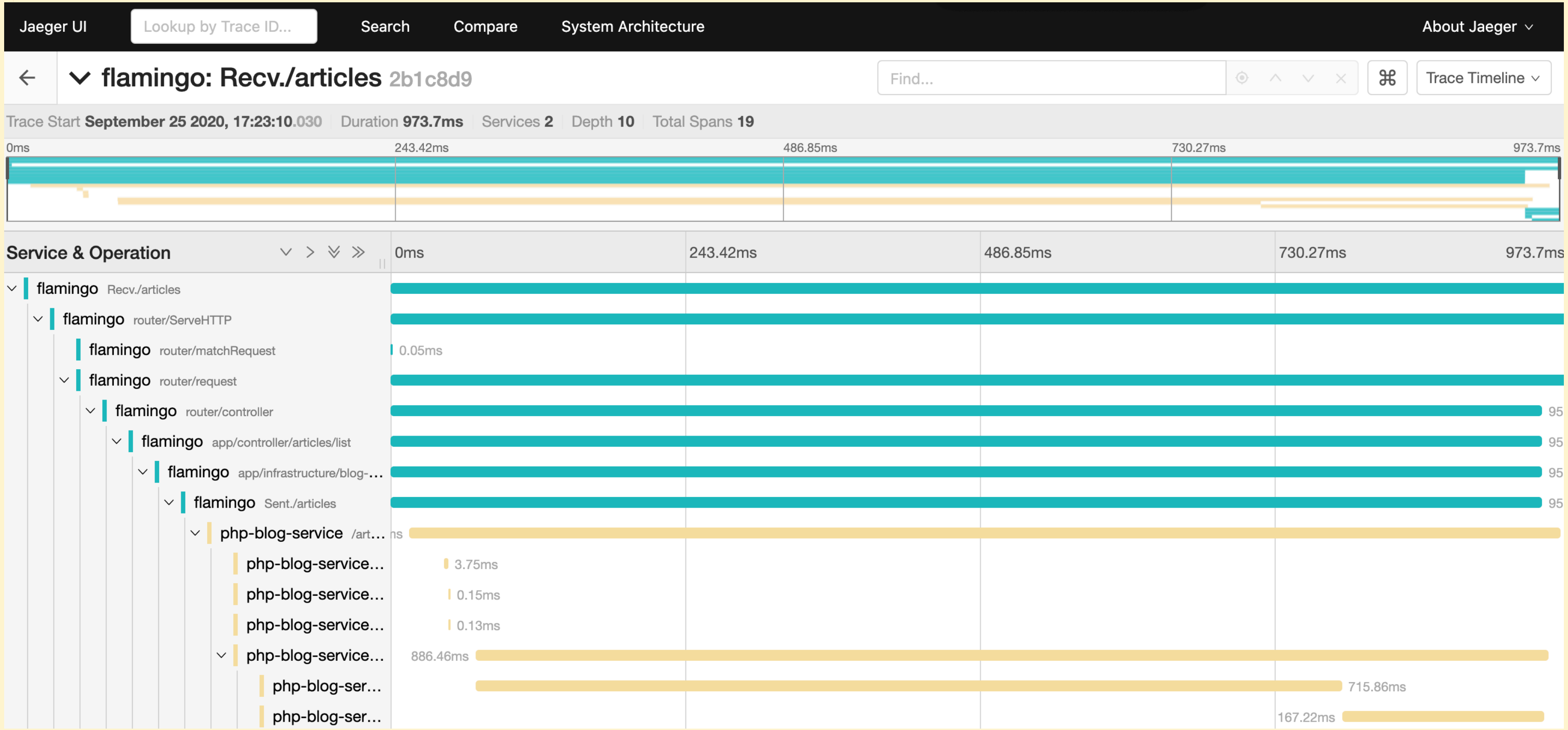
- **Trace**
  - The record of the complete request, recording the actual work by each part as a collection of Spans
- **Span**
  - A recording of a single operation
  - Child spans are possible to record more details.
  - Connected to a trace via the trace ID, and optionally to a parent span via a parent ID

Tracing under the hood

- A Trace can be viewed as tree with the triplet (TraceID, SpanID, ParentID) defining the exact position of a span
- Span Context = (TraceID, SpanID, ParentID)
- In Go, the Span Context is normally passed down via the `context.Context`
- For distributed tracing, it can be passed through protocol headers

Traces in Jaeger

24/10/2020



flamingo: Recv./articles 2b1c8d9

Trace Start September 25 2020, 17:23:10.030 | Duration 973.7ms | Services 2 | Depth 10 | Total Spans 19

0ms 243.42ms 486.85ms 730.27ms 973.7ms

```
Service & Operation                              0ms   243.42ms   486.85ms   730.27ms   973.7ms
flamingo Recv./articles
  flamingo router/ServeHTTP
    flamingo router/matchRequest                 0.05ms
    flamingo router/request
      flamingo router/controller                                                        95
        flamingo app/controller/articles/list                                           95
          flamingo app/infrastructure/blog-…                                            95
            flamingo Sent./articles                                                     95
              php-blog-service /art…             ns
                php-blog-service…                3.75ms
                php-blog-service…                0.15ms
                php-blog-service…                0.13ms
                php-blog-service…                886.46ms
                  php-blog-ser…                                                715.86ms
                  php-blog-ser…                                           167.22ms
```

GOLAB

What is OpenCensus?

- „OpenCensus is a set of libraries for various languages that allow you to collect application metrics and distributed traces, then transfer the data to a backend of your choice in real time. This data can be analyzed by developers and admins to understand the health of the application and debug problems.“ *
- OpenCensus originates from Google and is open source
- Will be merged into OpenTelemetry (currently in beta)

* from: https://opencensus.io/

Time for coding

A simple reading list

- We want a simple web application where the user can collect articles for a later reading session
- Sprint 1: Basic Setup
  - MVP: Web application that serves a basic page is set up

...

- Sprint N: Production readiness
  - Adding Health Check, Metrics, Tracing

Let’s use the Flamingo Framework

- Modular architecture
  - separation of bounded contexts
  - Interchangeable
  - Expandable
  - Scoping
- Module for Configuration
  - Configuration Merging and Loading
  - Default Configuration, Config Injection
  - Context and Area Support
- Module for Routing
  - Reverse Routing (productPage(foo) -> /en/product/foo)
  - Prefix Routing (/en/foo + /de/foo)
- Modules for Operational Readiness
  - Separate „internal“ port
  - Metrics, tracing, ping, health check
  - Logging
- Modules for Security Middleware
- Module for Authentication and Authorization (OpenID / oAuth2 / ...)
- Modules for Command & Command Registration

Flamingo

https://github.com/i-love-flamingo/flamingo

GOLAB

Thank you for listening!

- https://github.com/tessig/flamingo-readinglist

https://gitter.im/i-love-flamingo

GITTER

Carsten Dietrich
carsten.dietrich@aoe.com

@carstencodes

Thorsten Essig
thorsten.essig@aoe.com

@thorstenessig

Gophers by egonelbre

Powered by develer