Observability Lab

Version 1.0

Table of Contents

3 Hour workshop

Agenda

- Intro
- Lab01 - Logging
- Tracing Terminology
- Lab02 - Tracing
- Lab03 - Metrics
- Lab04 - Observability Bugs

$whoami

Jeff Beck

Software at SmartThings

How this Workshop is Setup

Labs are in a state where they will compile but not all are 100% correct. The answers are in the corresponding modules.

System Overview

[diagramoverdev] | *_images/diagramoverdev.svg*

Infra

All the shared infrastructure for observability is in this directory you can run it with docker-compose.

Requests

A handful of simple requests that can exercise the system easily are in this directory. Each service has a file.

Services

Each service is named for its role then a dash with it’s framework. You only need to run one of each role.

Task List

There is a high level task list in each lab directory, that has a rough order on the things to explore. It also has general pointers of where to get started.

Wrap Up

We will do a wrap up discussion after each lab talking about more complex real world applications of the topics.

What is Observability

The property of systems that allows operators to clearly understand the state of the system.

Show and Tell

Lab 01 - Logging

Commands

From infra dir:

```
docker-compose up
```

From each project directory

```
./gradlew run
```

Lab 01 - Tasks

1. Get Logs to One Place
2. Dynamic Log Filtering
3. Log Formatting
4. Correlation IDs

GOAL All Logs Available in Kibania

Lab 01 - Wrap Up

- Correlation IDs are lightweight, good for small retrofit
- Dynamic Logging is for cost savings

- Formatting Matters

Tracing Terminology

What is Distributed Tracing

Distributed tracing systems collect end-to-end latency graphs (traces) in near real-time.

- Zipkin
- Jaeger
- Dapper

Terminology Lesson

- **Span** - An operation that took place.
- **Event** - Something that occurs in a span.
- **Tag** - Key value pair on a span.

Terminology Lesson

- **Trace** - End-to-end latency graph, made up of spans.
- **Tracer** - Library that records spans and passes context
- **Instrumentation** - Use of a tracer to record tasks.
- **Sample %** - How often to record a trace.

Lab 02 — Tracing

Same apps just add tracing.

Lab 02 - Tasks

1. Zipkin Support For Services
2. DataStore Tracing
3. Debug Issues
4. Debug Slow Transactions

Lab 02 — Wrap Up

- Customization is Key
- Service Mesh

- When to use annotations?
- When to use tags?

Lab 03 — Metrics

Lab 03 - Tasks

1. Expose Metrics for Prometheus
2. Scrape All the Metrics
3. Custom Metrics

Lab 03 — Discussion

- What metrics do you collect today?
- How do metrics lie to you?
- How do your metrics tie to users?

Lab 04 — Observability Bugs

Lab 04 — Tasks

1. Odd Behaviors

Lab 04 — Error Stories

- Traces with > 10k spans
- Error rates thrown off by service reporting the wrong name.
- Lost traces
- Broken Traces
- Fixed correlation IDs

Observability Discussion

- Data is step 1
- Actionable data is step 2
- Pair all the tools for maximum effect.

Questions

We are Hiring

http://bit.ly/SmartThingsJobs