INTRO

OPENTRACING

ALWAYS START WITH WHY

- debugging and monitoring distributed software architectures

- debug and optimize code (performance & latency)

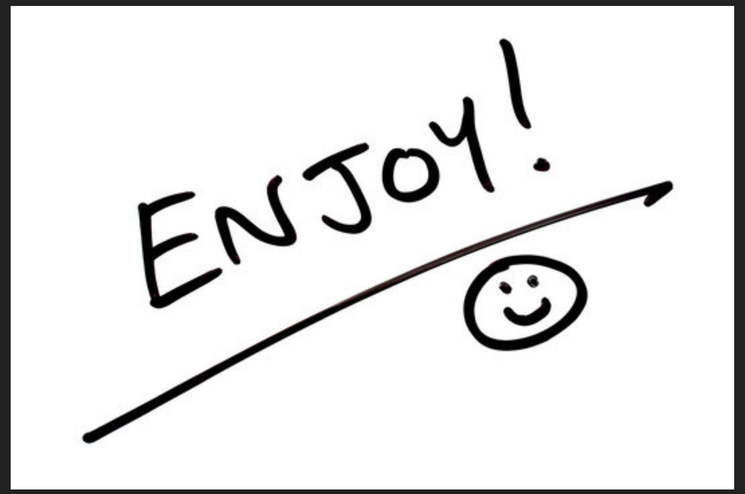
- root cause analysis

- service dependency analysis

- distributed context propagation

PLAN

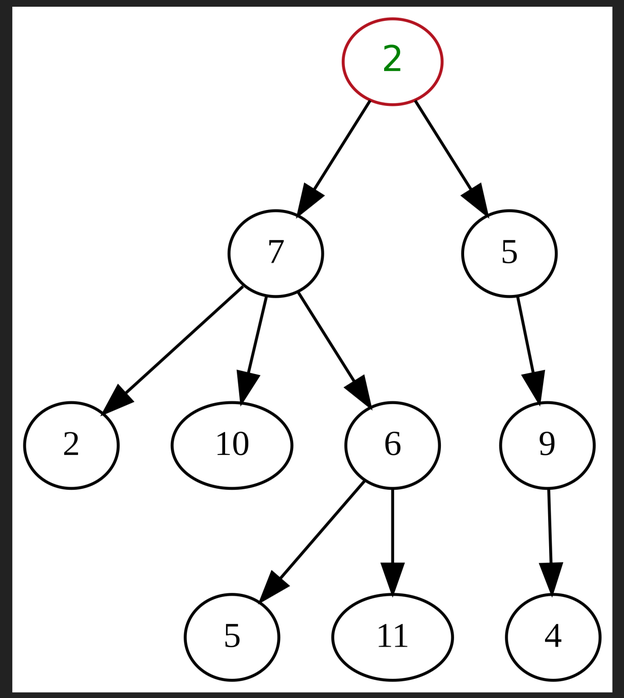
- meet the subject (theory)
- show demo (practice)
- set up server for tracing
- with interested teams apply & test

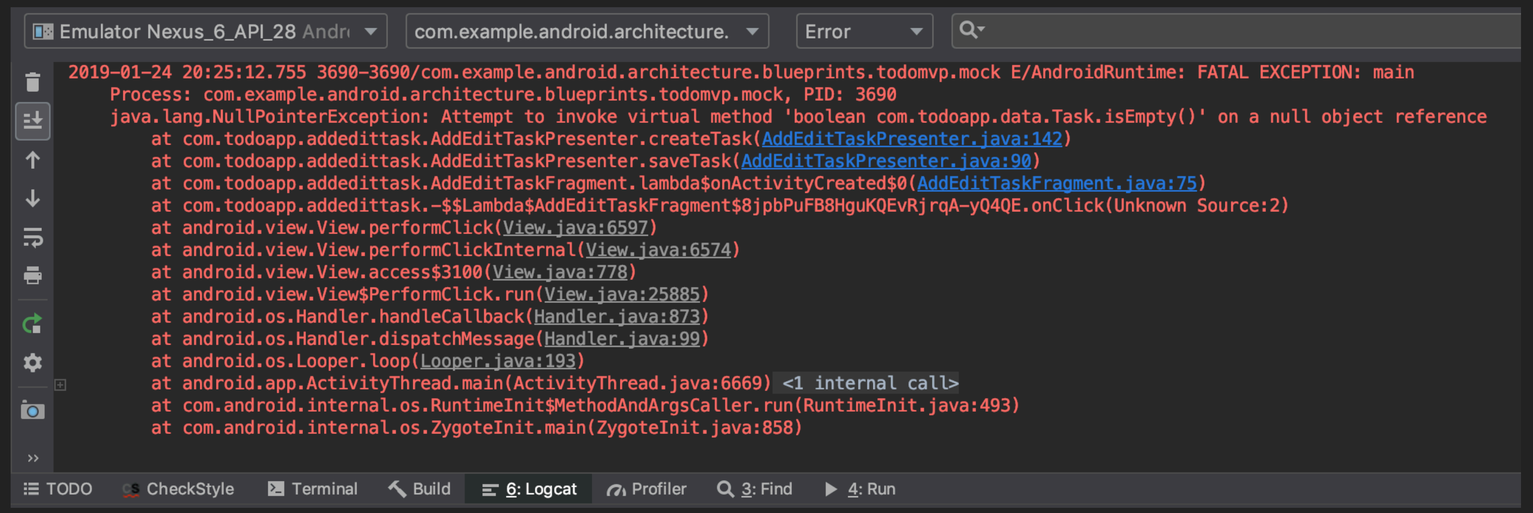


SPECS

- what is “trace” ?

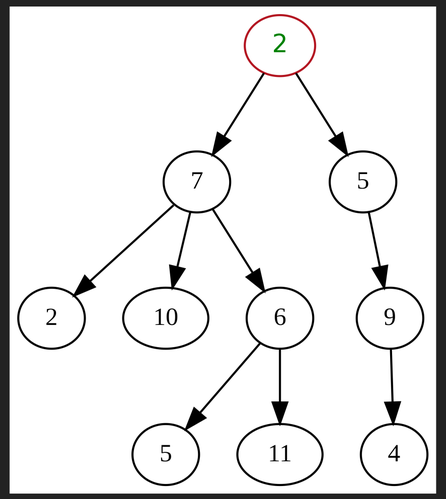
Trace can be thought of as a directed acyclic graph (DAG) of **Spans.**

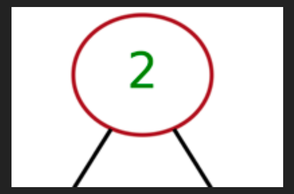




SPECS

- meet the “span”





SPAN - encapsulates following states:

- name (operation/function name)
- start time
- finish time
- set of [k => v] tags
- set of [k => v] logs
- SpanContext
  - implementation dependent state
  - baggage items
- Refs (child of, batch…)

SPECS

- meet the “tracer”

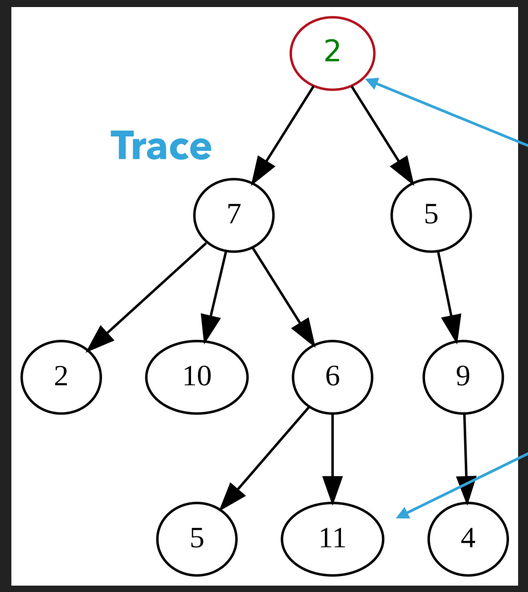
Tracer - defined by it’s abilities (methods)

- **creates** span
- **injects span into carrier**
- **extracts span from carrier**

SPECS

TraceR (also known as GlobalTracer)

- **creates** span
- **injects span into carrier**
- **extracts span from carrier**

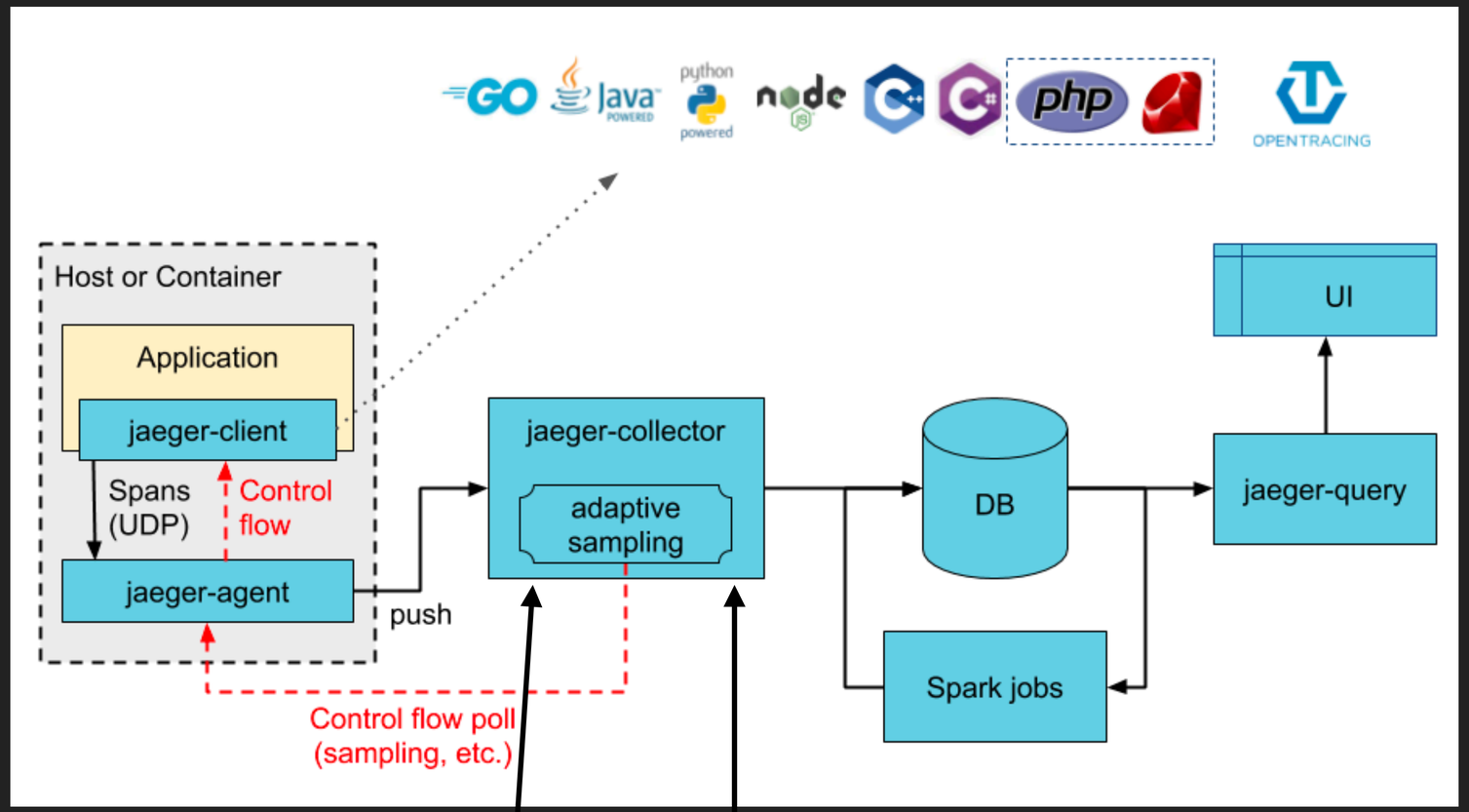


Span

- name (operation/function name)
- start time
- finish time
- set of [k => v] tags
- set of [k => v] logs
- SpanContext
  - implementation dependent state
  - baggage items
- Refs (child of, batch…)

DEMO TIME !

JAEGER-ARCHITECTURE



application: any (our) application

jaeger-client: library to work with jaeger-api

jaeger-agent: daemon process that sends traces to…

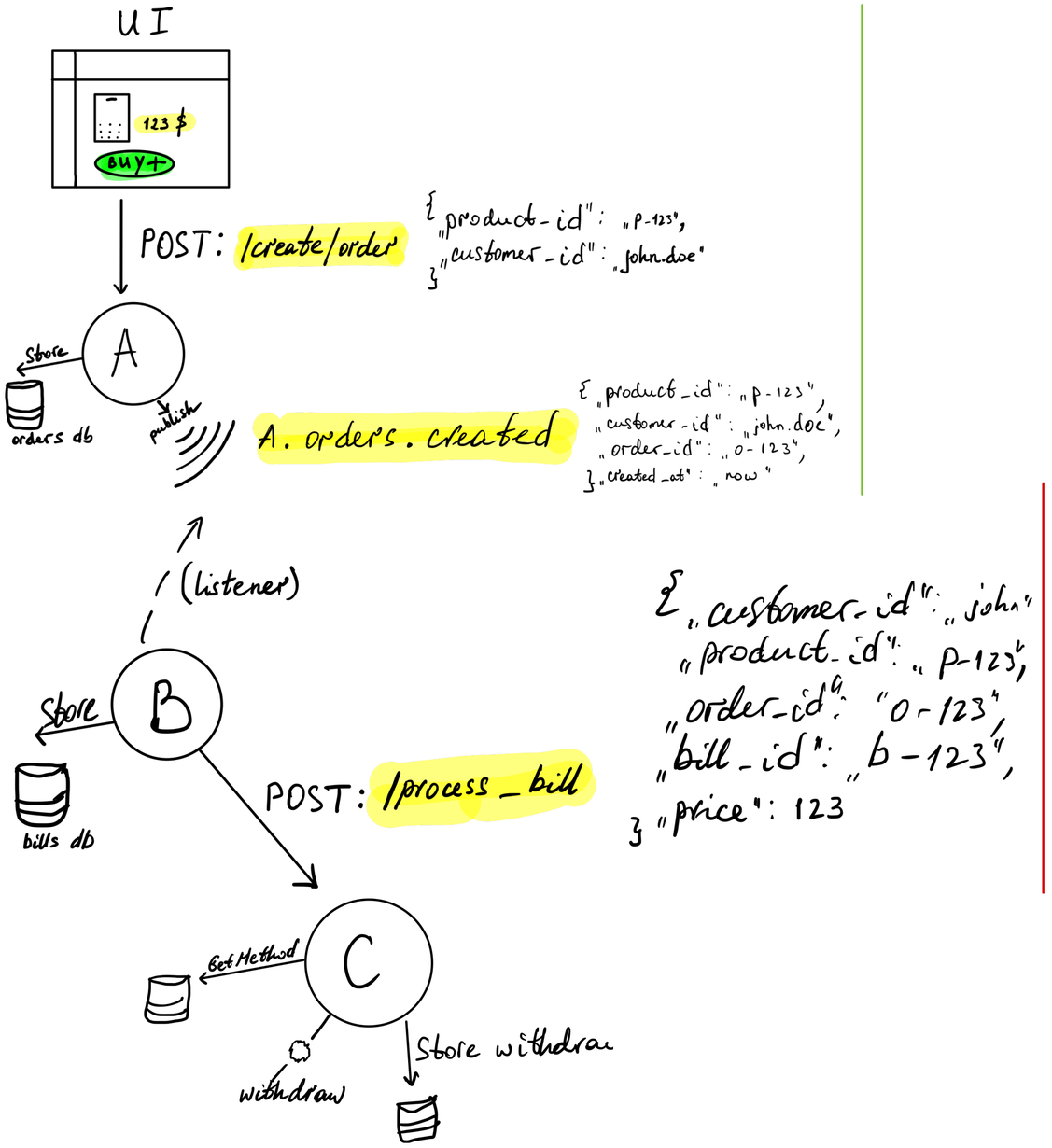
jaeger-collector: receives and stores spans

jaeger-query: serv. for fetching and viewing traces in **UI**

2) CONTAINER WITH ITS AGENT

3) CONTAINER WITH ITS AGENT

SOA EXAMPLE


UI
123 $
buy
POST: /create/order
{ "product_id": "p-123",
"customer_id": "john.doe" }
A
Store
orders db
publish
A. orders. created
{ "product_id": "p-123",
"customer_id": "john.doe",
"order_id": "o-123",
"created_at": "now" }
(listener)
B
Store
bills db
POST: /process_bill
{ "customer_id": "john"
"product_id": "p-123",
"order_id": "o-123",
"bill_id": "b-123",
"price": 123 }
C
Get Method
withdraw
Store withdraw

QUESTIONS ?

LINKS

- https://opentracing.io/specification
- https://www.jaegertracing.io
- https://medium.com/jaegertracing/embracing-context-propagation-7100b9b6029a
- https://github.com/yurishkuro/opentracing-tutorial
- demo: https://github.com/komron-m/opentracing_jaeger