Serverless under thatched roof

Introduction to Fn Project

26/09/2019

szymon.winiarz@sonalake.com
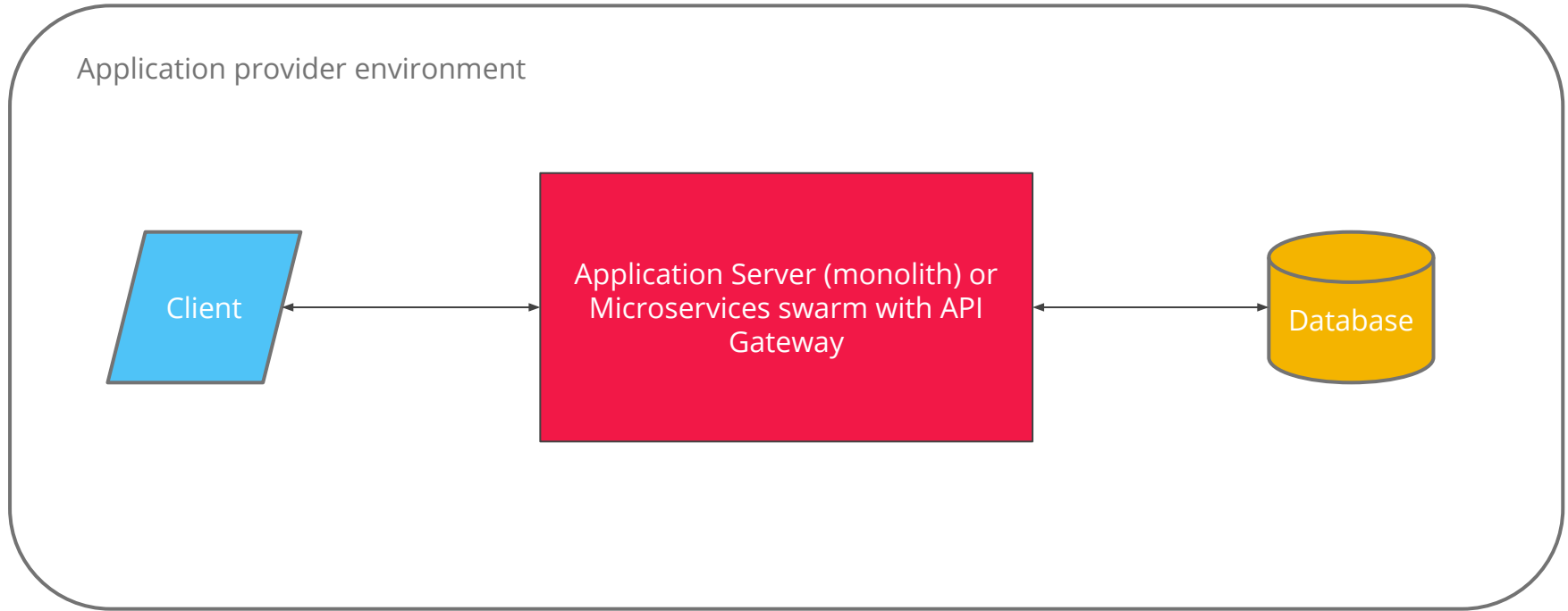
github.com/SzymonWiniarz

Serverless

Serverless

What is Serverless architecture?

- Buzzword, first time used in 2012, top of the top in 2016

- Is all about eliminating software and/or hardware

- Generally very wide concept

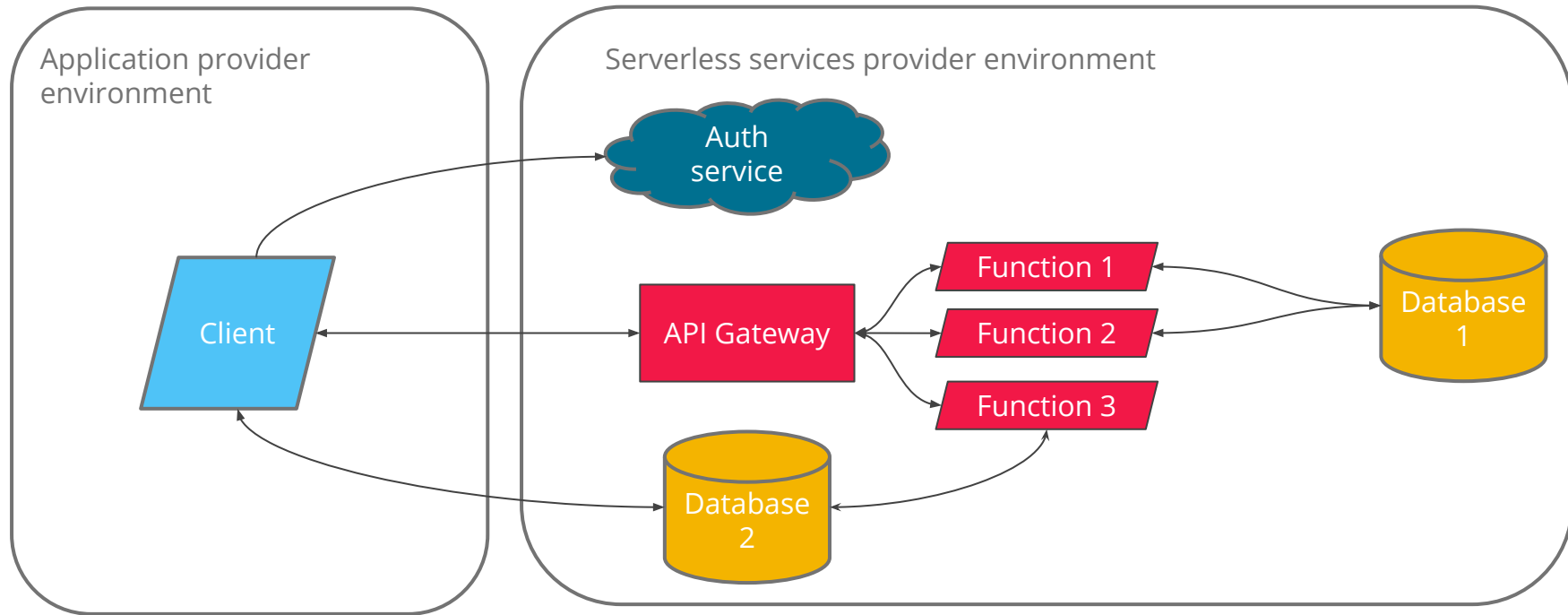- We distinguish between BaaS and FaaS

Serverless

Traditional webapp architecture

Application provider environment

Client → Application Server (monolith) or Microservices swarm with API Gateway → Database

SONALAKE

Serverless

Serverless architecture

Application provider environment

Serverless services provider environment

Auth service

Client

API Gateway

Function 1

Function 2

Function 3

Database 1

Database 2

SONALAKE

Serverless

Most popular Serverless solutions

AWS Lambda

Microsoft Azure

Google Cloud Functions

SONALAKE

Serverless

Yes! One of them is:

Fn Project

SONALAKE

Fn Project

> " *The Fn project is an open-source container-native serverless platform that you can run anywhere -- **any cloud or on-premise**. It's easy to use, supports every programming language, and is extensible and performant.*

— https://fnproject.io

SONALAKE

Serverless platform that requires a **server**?!

Fn Project

Characteristics of Fn Project

- Serverless platform (FaaS) that can be installed on premise or in cloud

- Written in Go

- Open source (Apache 2.0 license)

- Has FDKs (Functions Developer Kits) for several languages: Go, Java, Node.js, Python, Ruby

- Container-based (Docker)

- Provides CLI and Web UI for managing and monitoring

SONALAKE

Fn Project

Requirements

- Docker 17.10.0-ce or later

- Docker Hub account or any other Docker Registry

- One needs to be logged in into Docker Hub (`docker login`)

- FN Project CLI*

*Is not particularly required, but really helpful while working with Fn Project. In my examples I'm also using CLI. This is why I mentioned it here.

Fn Project

Creating and running functions

- Fn server startup: `fn start`

- Creating an empty function: `fn init --runtime java myfun`

  - With HTTP trigger: `--trigger http`

- Deploying a function on Fn server: `fn deploy --app myapp`

  - Locally, without pushing to Docker Hub: `--local`

  - Without bumping the version: `--no-bump`

- Invoking a function: `fn invoke myapp myfun`

  - If a function has HTTP trigger: `curl http://localhost:8080/t/myapp/myfun`

- Rebuilding a function: `fn build`

```
Deploying simple-function to app: myapp
Bumped to version 0.0.2
Building image fndemouser/simple-function:0.0.2
FN_REGISTRY:  fndemouser
Current Context:  default
Sending build context to Docker daemon  14.34kB
Step 1/11 : FROM fnproject/fn-java-fdk-build:1.0.100 as build-stage
 ---> 43818d2b84e5
Step 2/11 : WORKDIR /function
 ---> Using cache
 ---> 5f495be1aa14
Step 3/11 : ENV MAVEN_OPTS -Dhttp.proxyHost= -Dhttp.proxyPort= -Dhttps.proxyHost= -Dhttps.proxyPort= -Dhttp.nonProxyHosts= -Dmaven.r
epo.local=/usr/share/maven/ref/repository
 ---> Using cache
 ---> 1414d1949712
Step 4/11 : ADD pom.xml /function/pom.xml
 ---> Using cache
 ---> a6f8f66649ac
Step 5/11 : RUN ["mvn", "package", "dependency:copy-dependencies", "-DincludeScope=runtime", "-DskipTests=true", "-Dmdep.prependGrou
pId=true", "-DoutputDirectory=target", "--fail-never"]
 ---> Using cache
 ---> ad1a1081a675
Step 6/11 : ADD src /function/src
 ---> Using cache
 ---> 9745cef92d58
Step 7/11 : RUN ["mvn", "package"]
 ---> Using cache
 ---> 6543abc50498
Step 8/11 : FROM fnproject/fn-java-fdk:1.0.100
 ---> b572060cd8b5
Step 9/11 : WORKDIR /function
 ---> Using cache
 ---> 5eb939eff15d
Step 10/11 : COPY --from=build-stage /function/target/*.jar /function/app/
```

Fn Project

Docker HUB

Fn Project is, by default, integrated with Docker Hub, so functions that we write can be uploaded to that platform. This is happening automatically every time we install a new version of our function on Fn server (`fn deploy`).
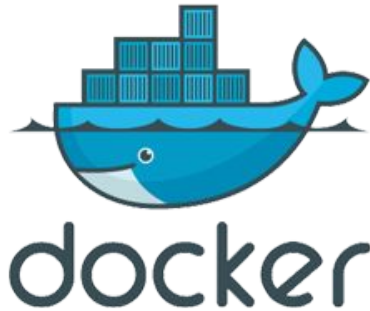
```
pId=true", "-DoutputDirectory=target", "--fail-never"]
 ---> Using cache
 ---> ad1a1081a675
Step 6/11 : ADD src /function/src
 ---> Using cache
 ---> 9745cef92d58
Step 7/11 : RUN ["mvn", "package"]
 ---> Using cache
 ---> 6543abc50498
Step 8/11 : FROM fnproject/fn-java-fdk:1.0.100
 ---> b572060cd8b5
Step 9/11 : WORKDIR /function
 ---> Using cache
 ---> 5eb939eff15d
Step 10/11 : COPY --from=build-stage /function/target/*.jar /function/app/
 ---> Using cache
 ---> c8c52e923130
Step 11/11 : CMD ["com.example.fn.HelloFunction::handleRequest"]
 ---> Using cache
 ---> 815484427303
Successfully built 815484427303
Successfully tagged szwiniarz/simple-function:0.0.3

Parts:  [szwiniarz simple-function:0.0.3]
Pushing szwiniarz/simple-function:0.0.3 to docker registry...The push refers to repository [docker.io/szwiniarz/simple-function]
4cf5d894eb9d: Preparing
dcbab89609db: Preparing
3cef6f91b606: Preparing
97fa3afd23f2: Preparing
1fbfb6acf90c: Preparing
e54bd3566d9e: Waiting
eb25e0278d41: Waiting
2bf534399aca: Waiting
1c95c77433e8: Waiting
```
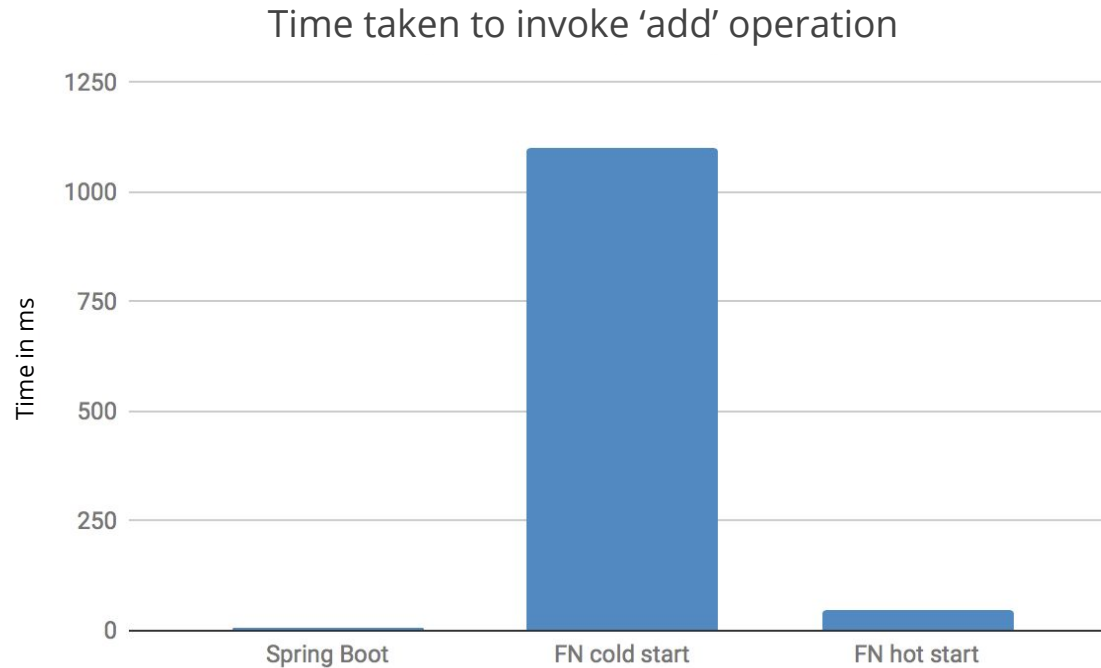
Fn Project

Grouping functions into applications

- Creating an application: file `app.yaml` with content: `name: myapp`

- Deploying the whole application at once: `fn deploy --all`

  - `--create-app` : creates the application on Fn server if it doesn't exist yet

- Listing of all the available applications: `fn list apps`

- Listing of all the functions of an app: `fn list functions myapp`

- Listing of all the triggers of an app: `fn list triggers myapp`

```
admins-MacBook-Pro-V29ANH:fn-calculator-app szymon.winiarz$ curl localhost:8080/t/calculator-app/add -d '{"a": 10, "b": 5}'
{"result":15.0}
admins-MacBook-Pro-V29ANH:fn-calculator-app szymon.winiarz$ curl localhost:8080/t/calculator-app/subtract -d '{"a": 10, "b": 5}'
{"result":5.0}
admins-MacBook-Pro-V29ANH:fn-calculator-app szymon.winiarz$ curl localhost:8080/t/calculator-app/multiply -d '{"a": 10, "b": 5}'
{"result":50.0}
admins-MacBook-Pro-V29ANH:fn-calculator-app szymon.winiarz$ curl localhost:8080/t/calculator-app/di -d '{"a": 10, "b": 5}'
```

Fn Project

Performance comparison between Fn project and traditional Servlet app

Time taken to invoke 'add' operation

Fn Project

Performance comparison between Fn project and traditional Servlet app

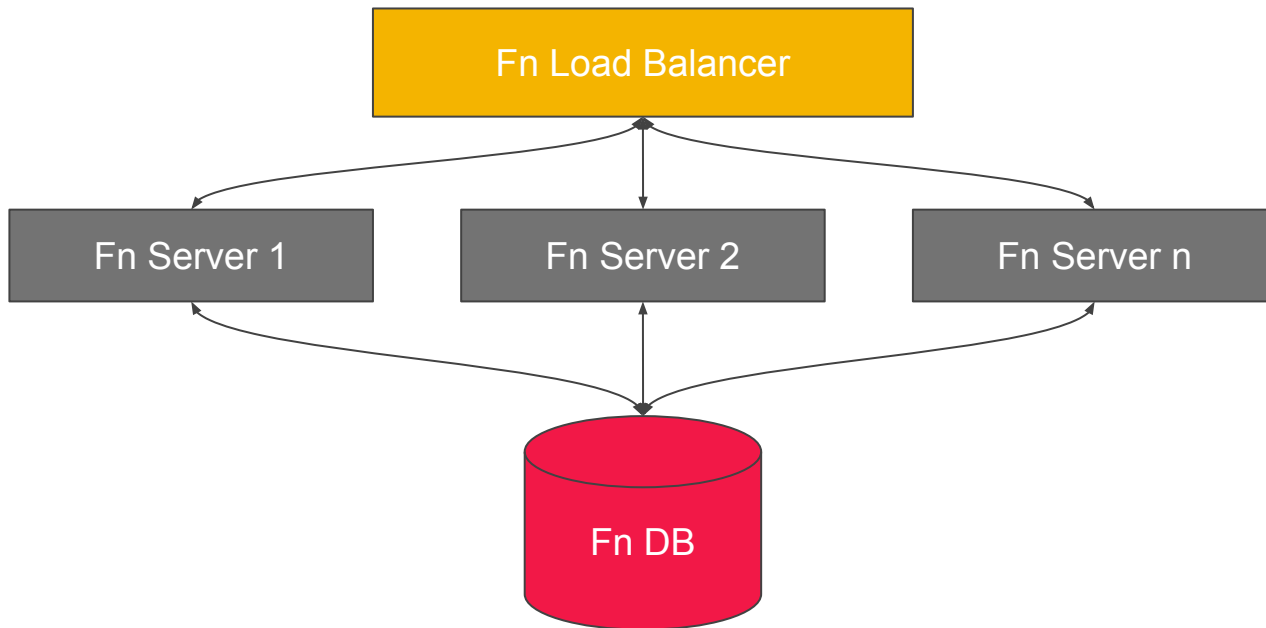Time taken to calculate the nth number of the Fibonacci sequence
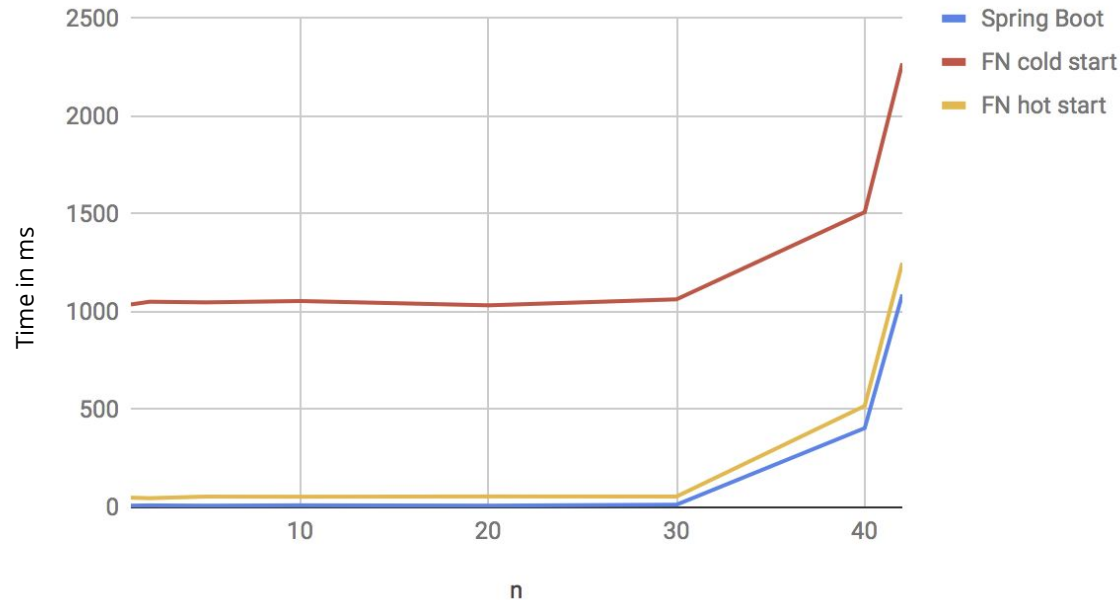
SONALAKE

Fn Project - advanced topics

Fn Project - advanced topics

Fn Project in cluster

SONALAKE

Fn Project - advanced topics
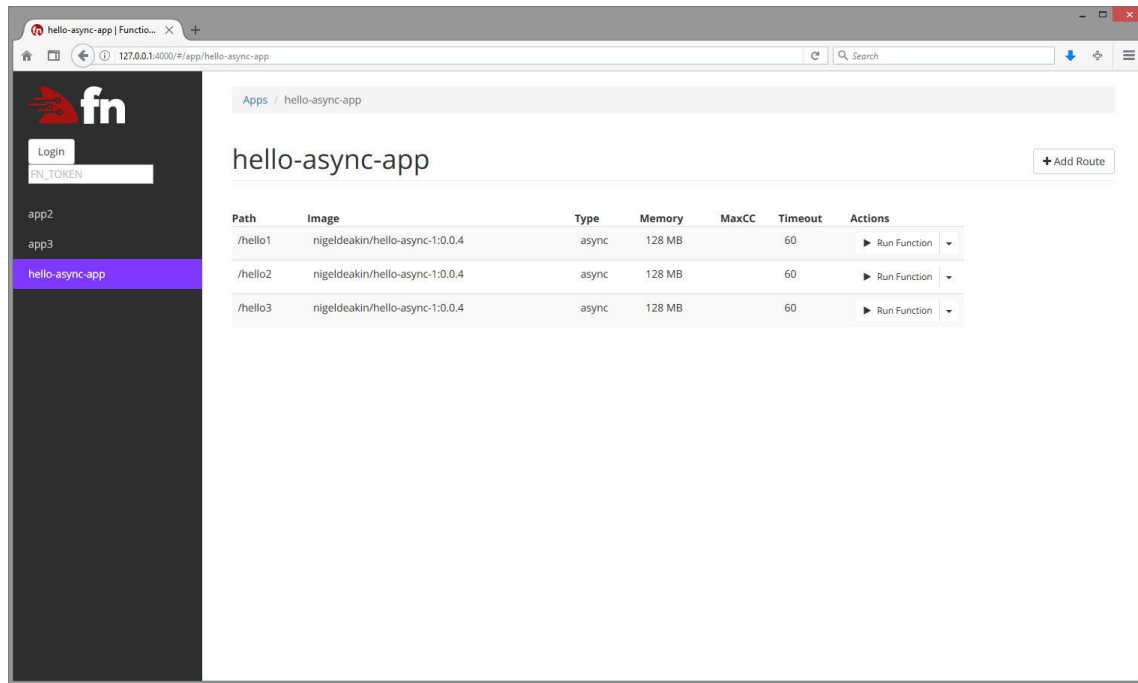
Fn Project in cluster

TL;DR:

1. Start a shared DB (e.g. MySQL; Docker image: `mysql:5.7.22`)
2. Start an API Server (Docker image: `fnproject/fnserver`, `FN_NODE_TYPE=api`)
3. Start runners (Docker image: `fnproject/fnserver`, `FN_NODE_TYPE="pure-runner"`)
4. Start a Load Balancer (Docker image: `fnproject/fnserver`, `FN_NODE_TYPE=lb`)

Full instruction here:
https://github.com/fnproject/docs/blob/master/fn/operate/runner_pools.md

Fn Project - advanced topics

Web UI

Fn Project - advanced topics

Web UI

How to run:

- `fn start` - starts Fn server that will be monitored/managed
- `docker run --rm -it --link fnserver:api -p 4000:4000 -e "FN_API_URL=http://api:8080" fnproject/ui` - starts UI server

Fn Project - advanced topics

Native Java functions - Fn Project + GraalVM

TL;DR

- GraalVM used for compiling Java to native code
- Native Java functions perform better, when it comes to speed and memory consumption, than functions written in Go
- Size of executables comparable to Go functions thanks to using `scratch` as a base Docker image

More can be found here:

https://medium.com/criciumadev/serverless-native-java-functions-using-graalvm-and-fn-project-c9b10a4a4859

Summary

Summary

Fn Project:

- Serverless platform (FaaS) that can be installed on premise
- Supports functions written in any programming language
- Container-based (Docker)
- Provides CLI that speeds up and facilitates the work
- Performance comparable to similar Spring Boot application
- Scalable (supports load balancing)
- Provides Web UI for easier monitoring and functions management
- Poor default exception handling
- Still feels a bit immature

Sources and other interesting materials

About Serverless - https://martinfowler.com/articles/serverless.html

Article firstly mentioning Serverless from 2012 -
https://readwrite.com/2012/10/15/why-the-future-of-software-and-apps-is-serverless/

Fn Project home page - https://fnproject.io

Web UI for Fn Project - https://github.com/fnproject/ui

Private registry instead of Docker Hub -
https://github.com/fnproject/docs/blob/master/fn/operate/private_registries.md

Load Balancing in Fn -
https://github.com/fnproject/docs/blob/master/fn/operate/runner_pools.md

Hot/cold start in Fn Project-
https://medium.com/fnproject/fn-hot-docker-functions-e02d15033392

Article about GraalVM + Fn -
https://medium.com/criciumadev/serverless-native-java-functions-using-graalvm-and-fn-project-c9b10a4a4859

Thank you!

Questions?