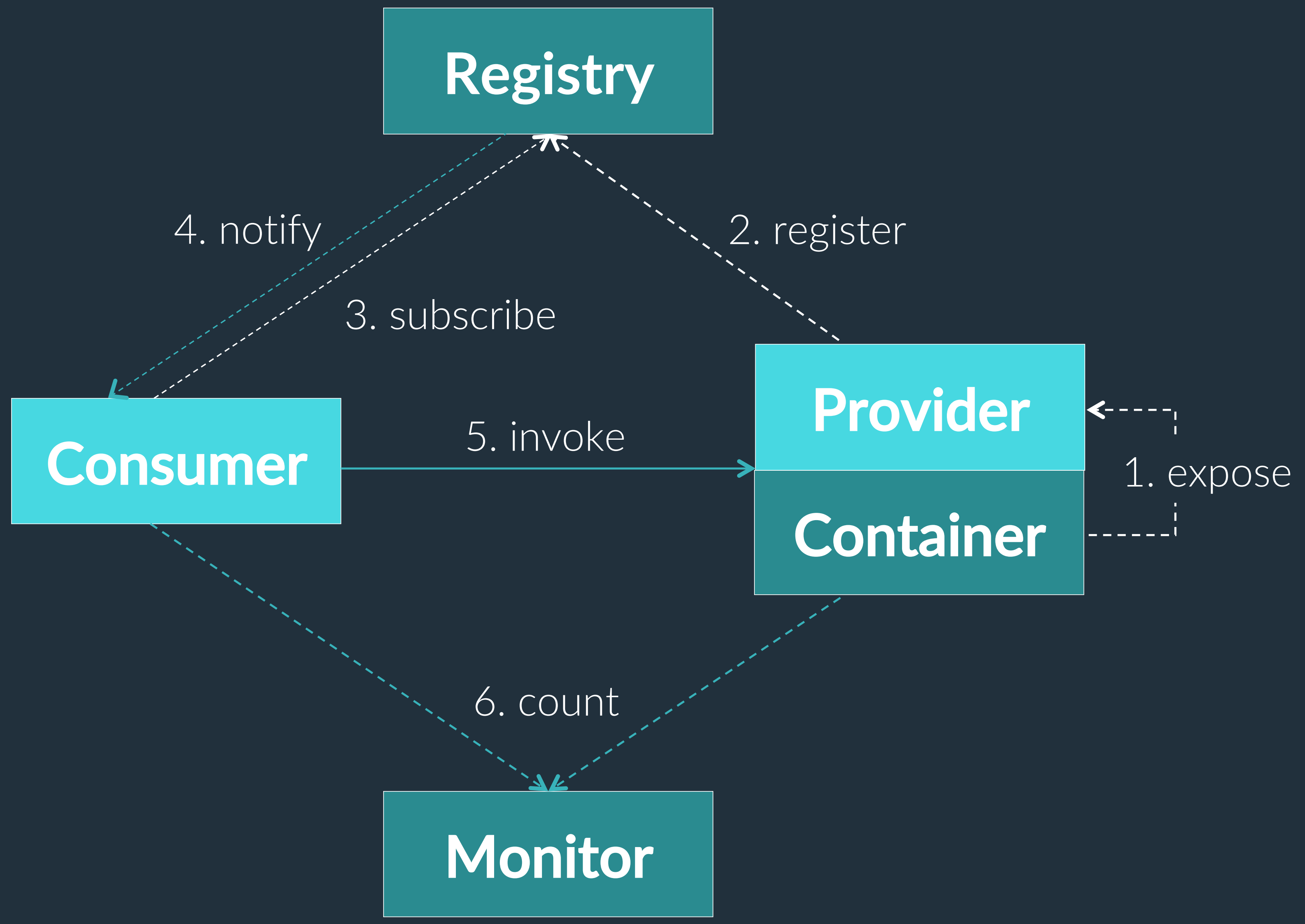
WHAT DUBBO IS & HOW IT WORKS

APACHECON NA 2018

dubbo.apache.org

WHAT'S DUBBO

Dubbo Basic Concepts



01 Expose

Service provider exposes service on the specified port

02 Register

Service provider registers service info to the service registry

03 Subscribe

Service consumer subscribes to registry center

04 Notify

Registry center notifies consumer the provider's addresses

05 Invoke

Service consumer picks one from received addresses, then fires invocation

06 Count

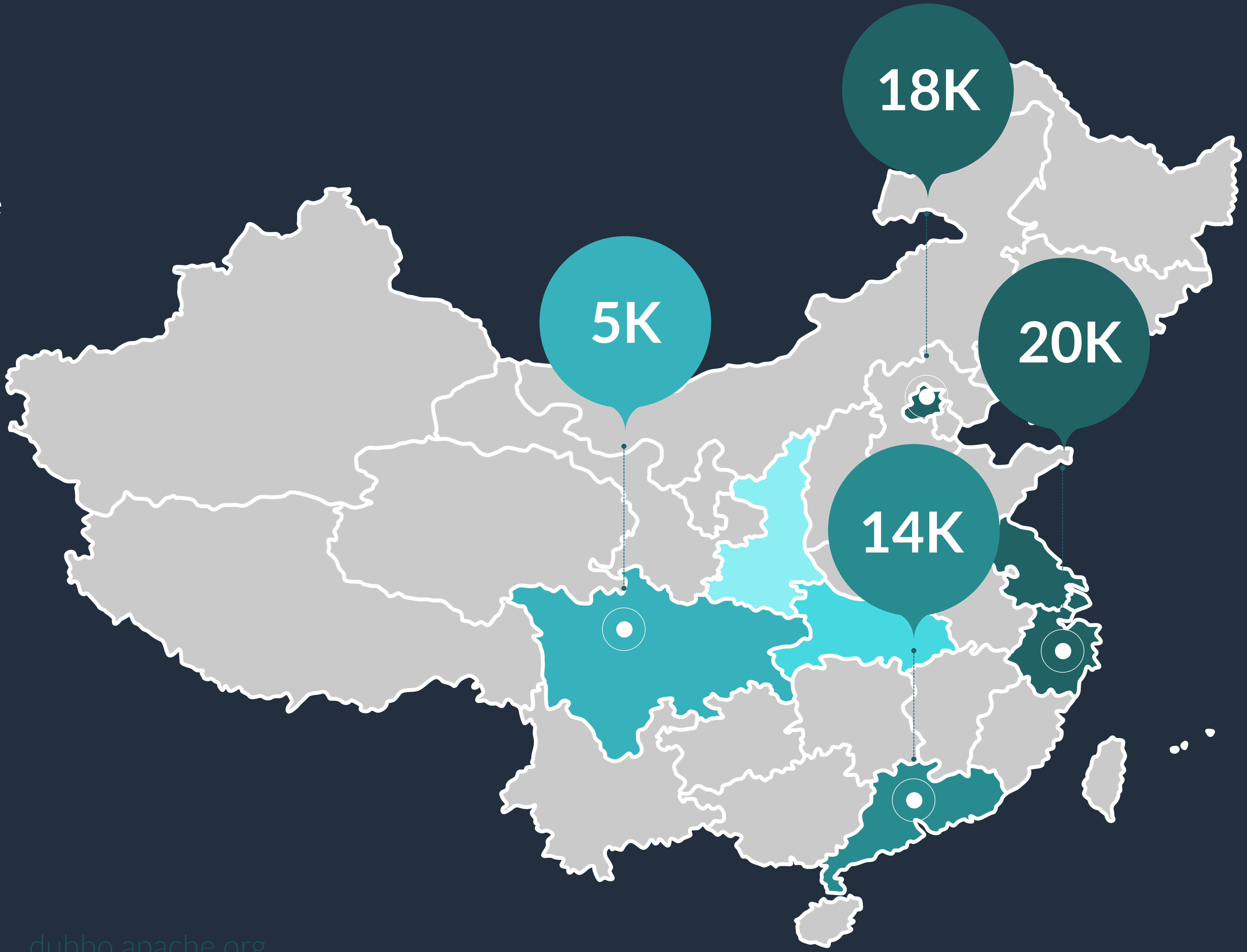
Both service provider and consumer reports stats to monitor center

Dubbo User Community

Two Month User Visits

87% Users in China. Most of them concentrate in Beijing, the Yangtze River Delta region, and the Pearl River Delta region.

YANGTZE RIVER DELTA	24.5%
BEIJING CITY	22.2%
PEARL RIVER DELTA	17.1%
SICUAN PROVINCE	6%
OTHERS	30.2%



Dubbo Opensource Status

21,679

STARS

120% star increment since last year, and rank 10th in Java projects on GitHub

14,945

FORKS

Code repo is forked 15K times by GitHub users

128

CONTRIBUTORS

2,542 commits from 128 contributors

16

COMMITERS

16 committers in Apache from different companies

DEVELOP DUBBO APP

Dubbo Quick Start

01 DUBBO SPRING INITIALIZR

Visit start.dubbo.io to generate project scaffold in lighting speed

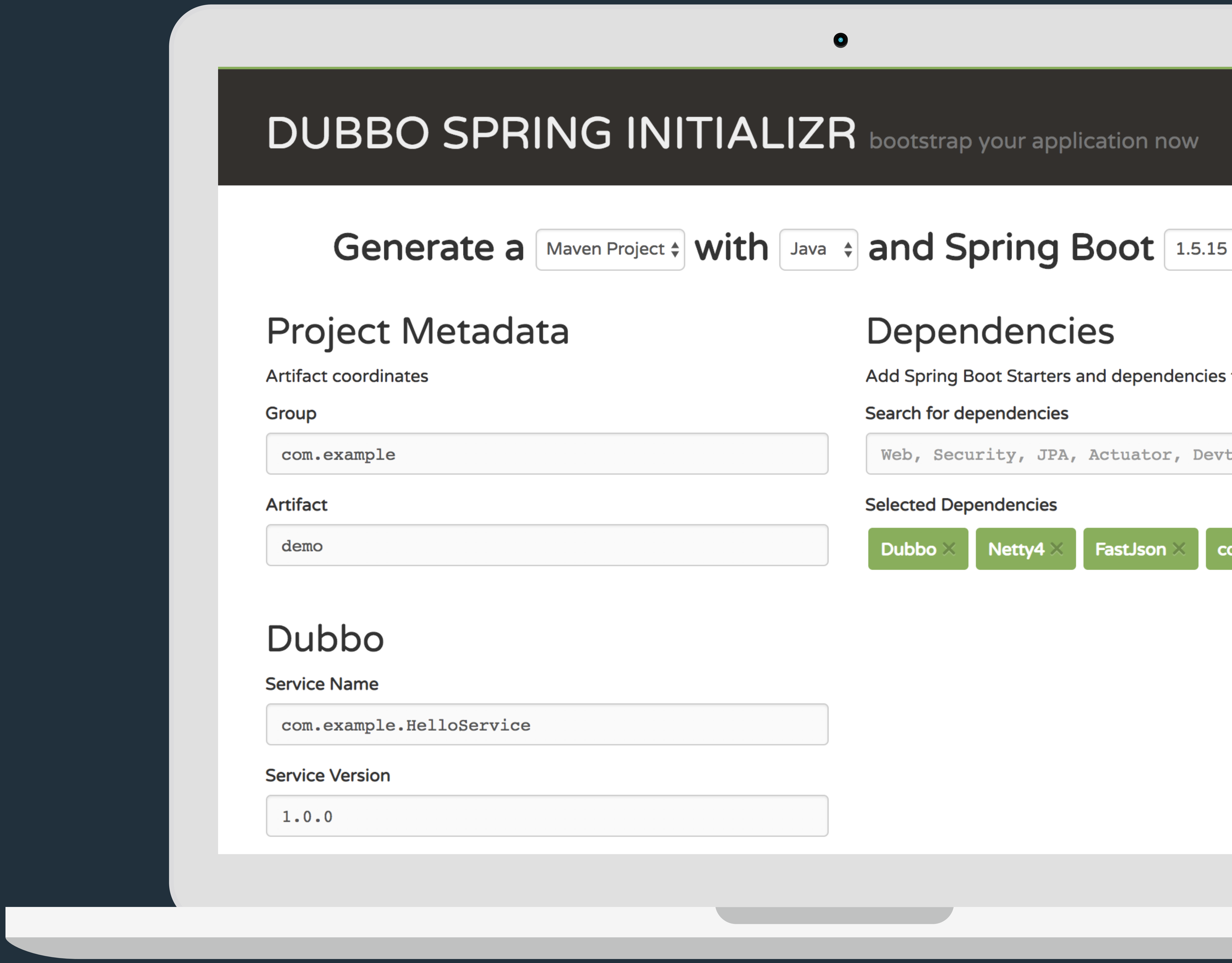
02 DUBBO IDE PLUGIN

Preferences(Settings) > Plugins > Browse repositories... > Search “Dubbo” to install in Intellij Idea

03 MORE

Sample code: github.com/dubbo/dubbo-samples

Quick start: dubbo.io/en-us/docs/user/quick-start.html



DUBBO SPRING INITIALIZR bootstrap your application now

Generate a Maven Project with Java and Spring Boot 1.5.15

Project Metadata

Artifact coordinates

Group: com.example

Artifact: demo

Dubbo

Service Name: com.example.HelloService

Service Version: 1.0.0

Dependencies

Add Spring Boot Starters and dependencies

Search for dependencies: Web, Security, JPA, Actuator, Devt

Selected Dependencies: Dubbo ✕ Netty4 ✕ FastJson ✕ co

start.dubbo.io

BASED ON SPRING INITIALIZR PROJECT

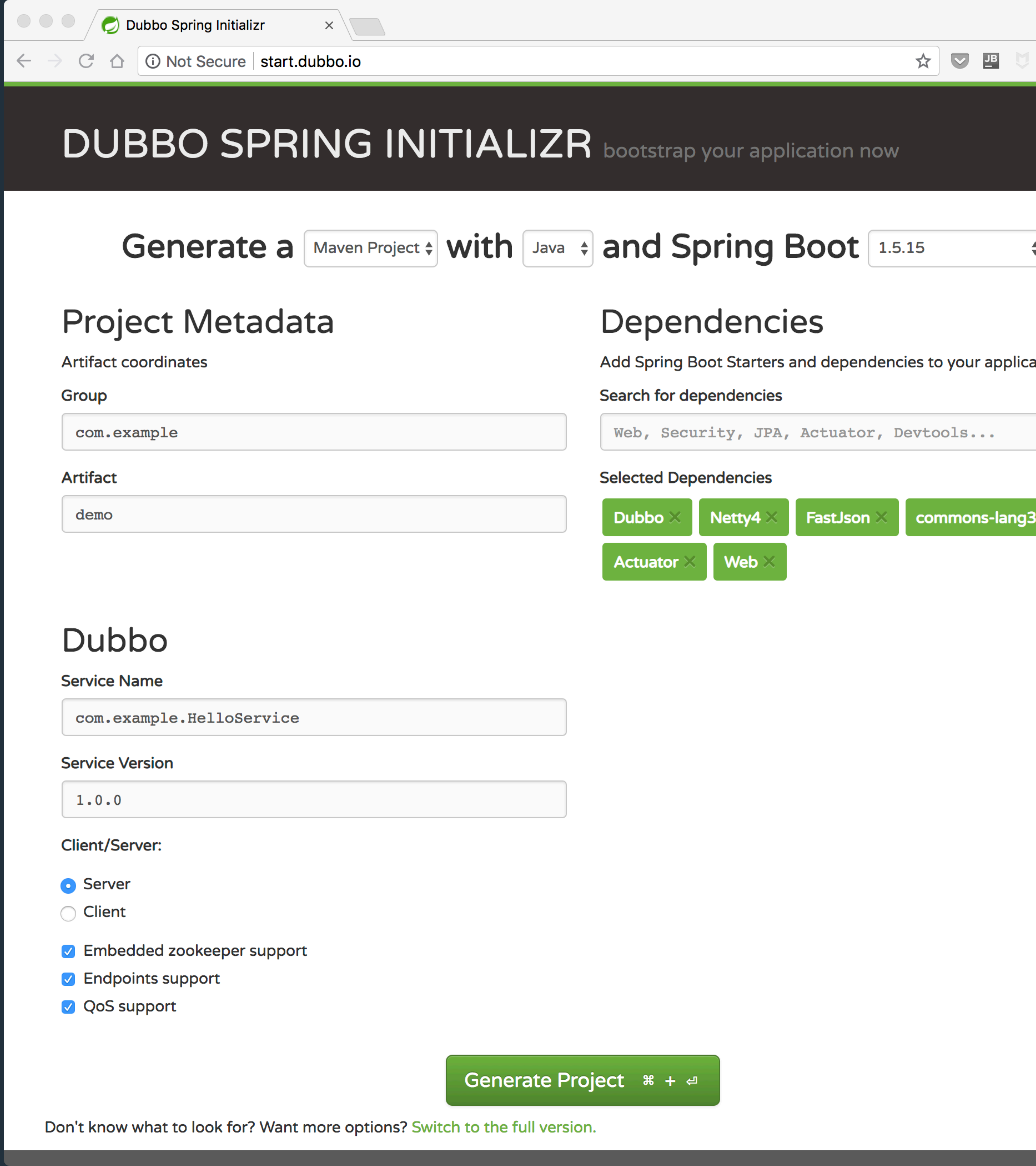
A spring initializr fork with dubbo addon. Dubbo code is generated, ready to run after downloaded

SERVER/CLIENT

Can generate projects for both dubbo server and dubbo client

MORE OPTIONS

Embedded zookeeper, Spring boot endpoints, Dubbo QoS support



DUBBO SPRING INITIALIZR bootstrap your application now

Generate a Maven Project with Java and Spring Boot 1.5.15

Project Metadata

Artifact coordinates

Group: com.example

Artifact: demo

Dependencies

Add Spring Boot Starters and dependencies to your applica

Search for dependencies: Web, Security, JPA, Actuator, Devtools...

Selected Dependencies: Dubbo ×, Netty4 ×, FastJson ×, commons-lang3, Actuator ×, Web ×

Dubbo

Service Name: com.example.HelloService

Service Version: 1.0.0

Client/Server:

● Server
○ Client

☑ Embedded zookeeper support
☑ Endpoints support
☑ QoS support

Generate Project ⌘ + ⏎

Don't know what to look for? Want more options? Switch to the full version.

Server Side Code

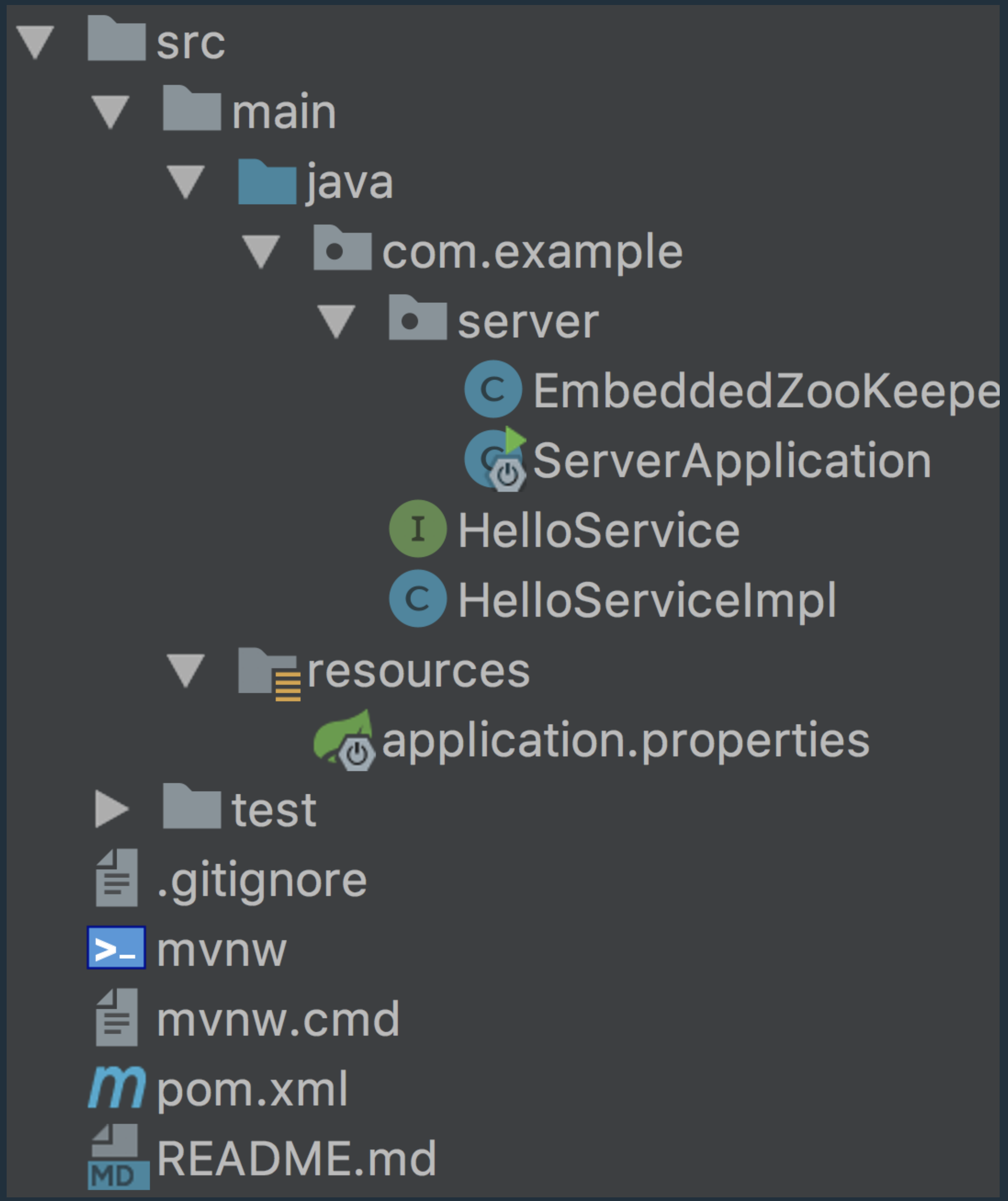
HELLOSERVICEIMPL.JAVA

```java
@Service(version = "1.0.0")
public class HelloServiceImpl implements HelloService {
    @Override
    public String sayHello(String name) {
        return "Hello, " + name + ", " + new Date();
    }
}
```

APPLICATION.PROPERTIES

```
dubbo.application.name = dubbo-demo-server
dubbo.scan.basePackages  = com.example
dubbo.registry.address = \
  zookeeper://localhost:2181?client=curator
management.security.enabled = false
endpoints.dubbo.enabled = true
```

PROJECT LAYOUT



```
▼ src
  ▼ main
    ▼ java
      ▼ com.example
        ▼ server
            EmbeddedZooKeepe
            ServerApplication
          HelloService
          HelloServiceImpl
    ▼ resources
        application.properties
  ▶ test
.gitignore
mvnw
mvnw.cmd
pom.xml
README.md
```

Client Side Code

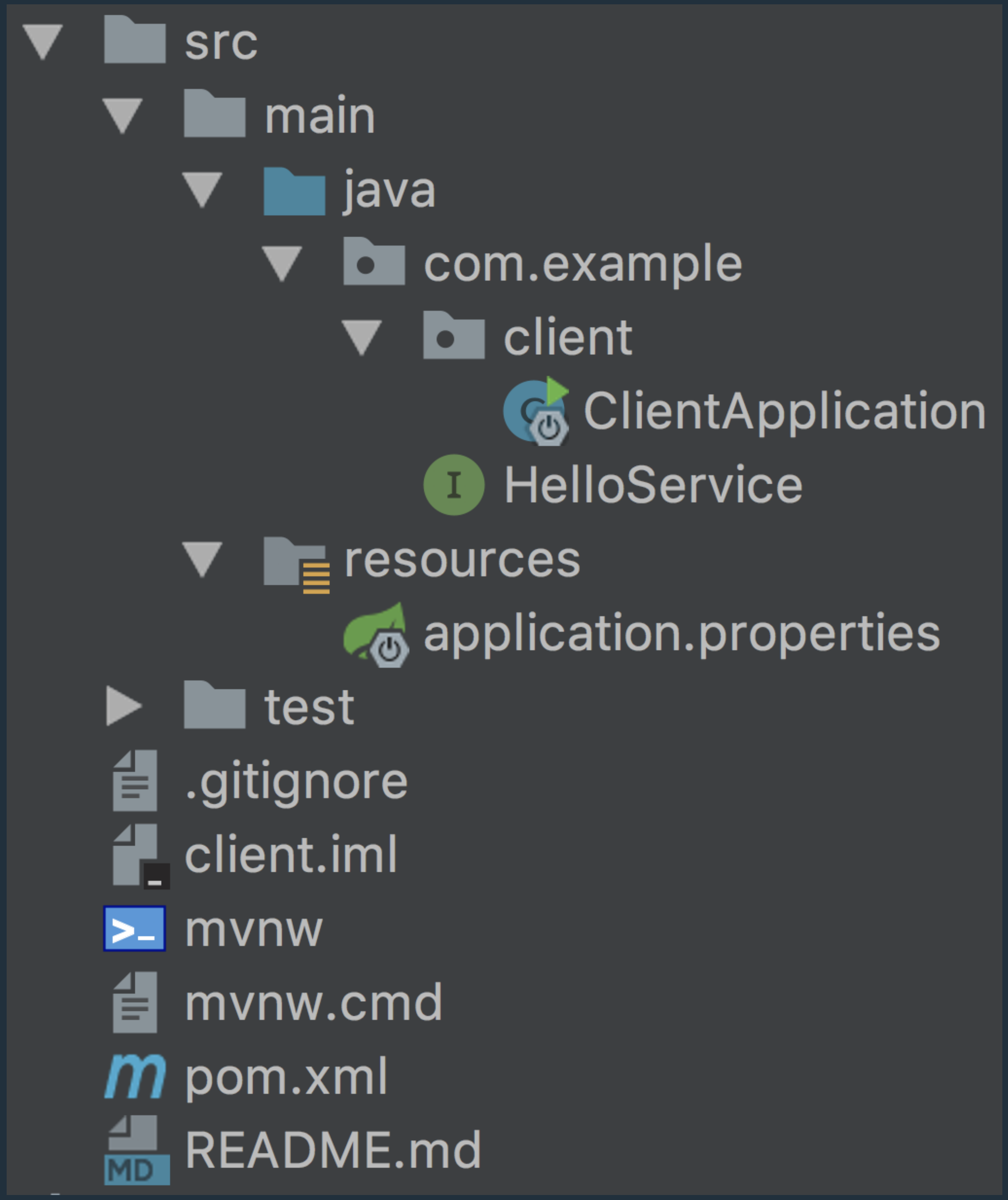
CLIENTAPPLICATION.JAVA

```java
@SpringBootApplication
public class ClientApplication {
    @Reference(version = "1.0.0")
    private HelloService demoService;

    public static void main(String[] args) {
        SpringApplication.run(ClientApplication.class);
    }

    @PostConstruct
    public void init() {
        String sayHello = demoService.sayHello("world");
        System.err.println(sayHello);
    }
}
```

PROJECT LAYOUT



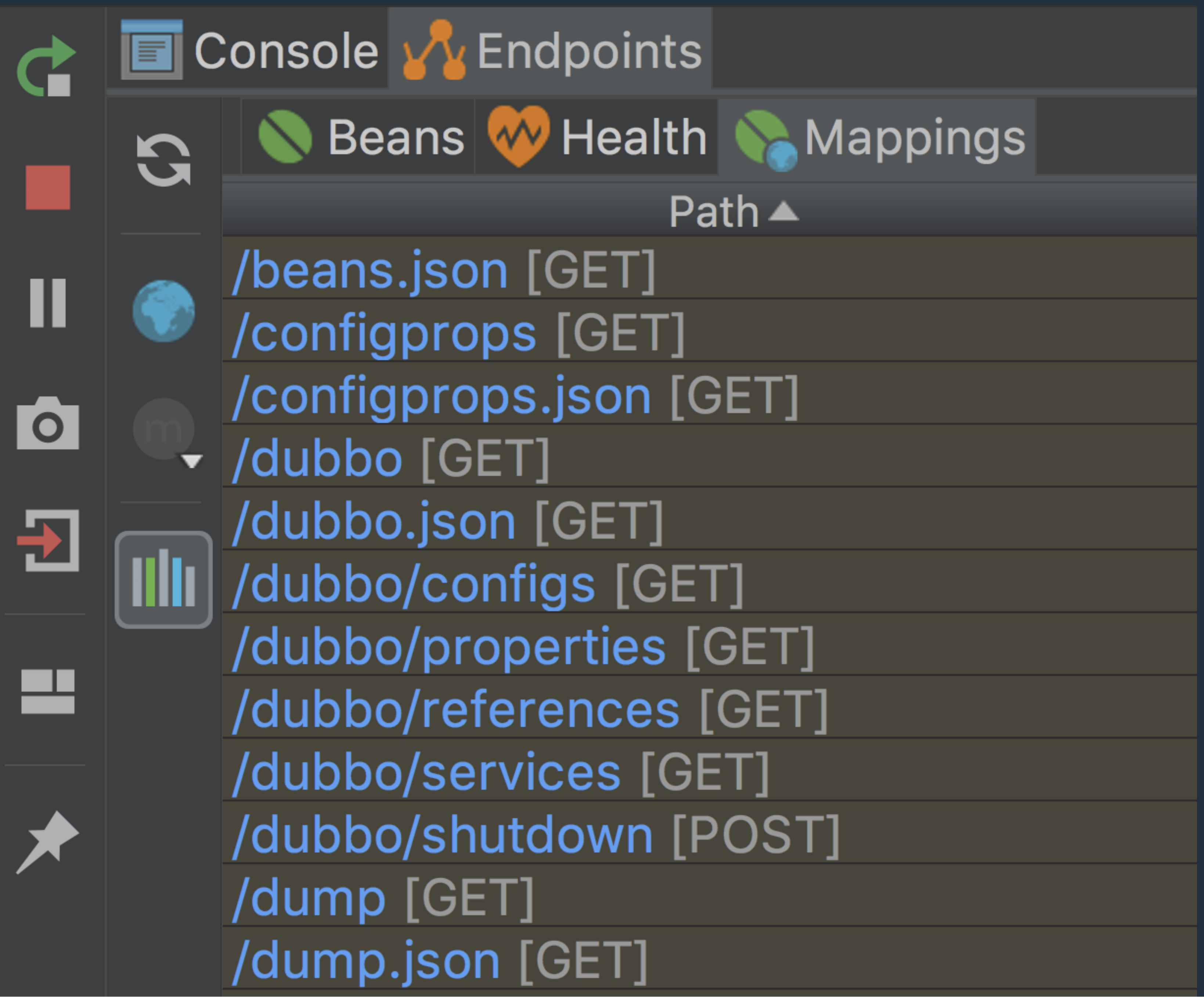
```
▼ src
    ▼ main
        ▼ java
            ▼ com.example
                ▼ client
                    ClientApplication
                  HelloService
        ▼ resources
              application.properties
    ▶ test
  .gitignore
  client.iml
  mvnw
  mvnw.cmd
  pom.xml
  README.md
```

MONITOR DUBBO

Dubbo Endpoints

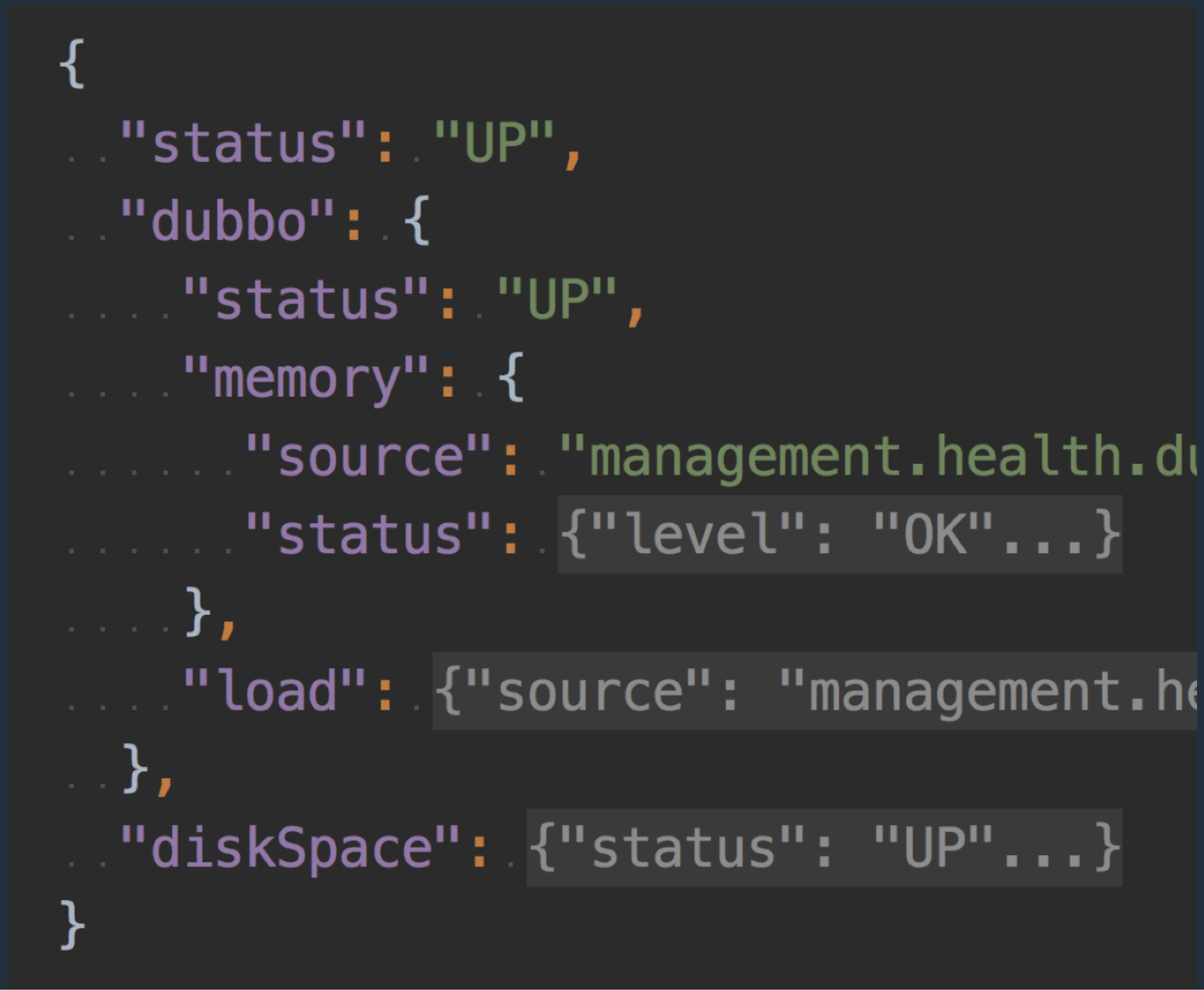
Dubbo provides the following spring boot endpoints: /dubbo to get all dubbo endpoints, /dubbo/services to get provided services, /dubbo/references to get consumed services, /dubbo/configs to get dubbo relevant configurations

dubbo endpoints



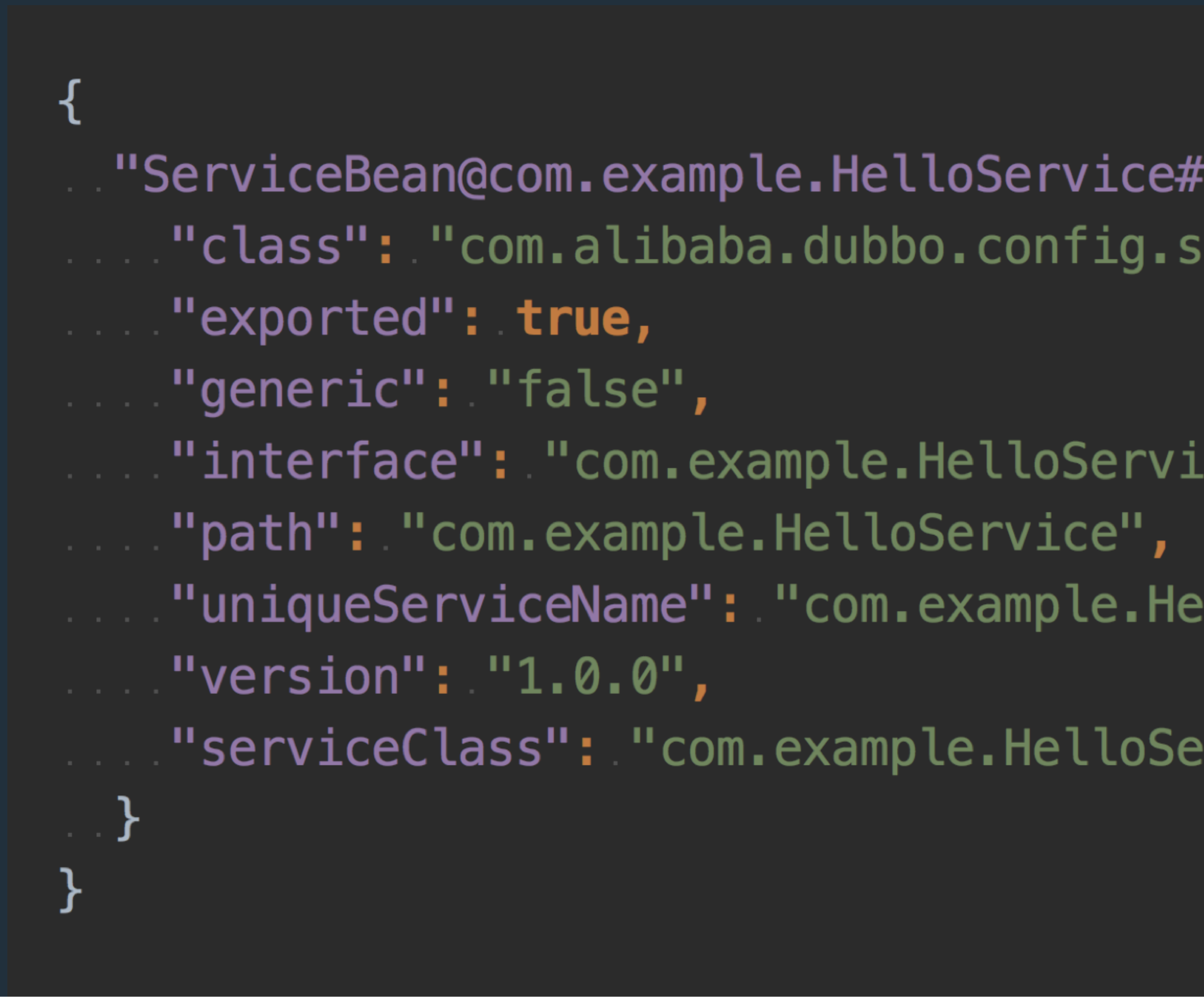
Path ▲
/beans.json [GET]
/configprops [GET]
/configprops.json [GET]
/dubbo [GET]
/dubbo.json [GET]
/dubbo/configs [GET]
/dubbo/properties [GET]
/dubbo/references [GET]
/dubbo/services [GET]
/dubbo/shutdown [POST]
/dump [GET]
/dump.json [GET]

/health



```
{
  "status": "UP",
  "dubbo": {
    "status": "UP",
    "memory": {
      "source": "management.health.du
      "status": {"level": "OK"...}
    },
    "load": {"source": "management.he
  },
  "diskSpace": {"status": "UP"...}
}
```

/dubbo/services



```
{
  "ServiceBean@com.example.HelloService#
    "class": "com.alibaba.dubbo.config.s
    "exported": true,
    "generic": "false",
    "interface": "com.example.HelloServi
    "path": "com.example.HelloService",
    "uniqueServiceName": "com.example.He
    "version": "1.0.0",
    "serviceClass": "com.example.HelloSe
  }
}
```

```
$ telnet localhost 22222
Trying ::1...
Connected to localhost.
Escape character is '^]’.

DUBBO

dubbo>ls
As Provider side:
+-------------------------------+---+
|     Provider Service Name     |PUB|
+-------------------------------+---+
|com.example.HelloService:1.0.0| Y |
+-------------------------------+---+
As Consumer side:
+---------------------+---+
|Consumer Service Name|NUM|
+---------------------+---+

dubbo>offline com.example.HelloService:1.0.0
OK
dubbo>ls
As Provider side:
+-------------------------------+---+
|     Provider Service Name     |PUB|
+-------------------------------+---+
|com.example.HelloService:1.0.0| N |
+-------------------------------+---+
As Consumer side:
+---------------------+---+
|Consumer Service Name|NUM|
+---------------------+---+

dubbo>online com.example.HelloService:1.0.0
```

Dubbo QoS Service

WHY

A convenient way for developer who’s prefer to command line to observe, manager services in development stage. Unlike endpoints, telnet interface doesn’t rely on Spring Boot since it’s a built-in functionality. It works with both telnet and http.

LIST

List all services the current node provides and consumes

ONLINE/OFFLINE

Manually online or offline the specified service from the registry center

EXTEND DUBBO

Dubbo Extension Loader

1. INJECT BY NAME

Dubbo's extension has name, default extension can be defined on @SPI – inject by type

2. AOP

A extension has copy constructor (wrapper), can use decorator pattern to achieve AOP

3. ON-CONDITION

@Adaptive on SPI's method. A suitable extension can be selected based on method parameter

4. FILTER / ORDER

@Activate on SPI's impl. Return a list of filtered and ordered extensions

SPI INTERFACE

```
@SPI(RandomLoadBalance.NAME)
public interface LoadBalance {
  @Adaptive("loadbalance")
  <T> Invoker<T> select(List<Invoker<T>> invokers,
    URL url, Invocation invocation)
      throws RpcException;
}
```

SPI CONFIGURATION

```
random=c.a.d.r.c.l.RandomLoadBalance
roundrobin=c.a.d.r.c.l.RoundRobinLoadBalance
leastactive=c.a.d.r.c.l.LeastActiveLoadBalance
consistenthash=c.a.d.r.c.l.ConsistentHashLoadBalance
demo=org.apache.dubbo.DemoLoadBalance
```

API TO FETCH IMPL

```
ExtensionLoader<LoadBalance> loader = ExtensionLoader.
  getExtensionLoader(LoadBalance.class);
LoadBalance lb = loader.getExtension("demo");
```

Extend a Filter

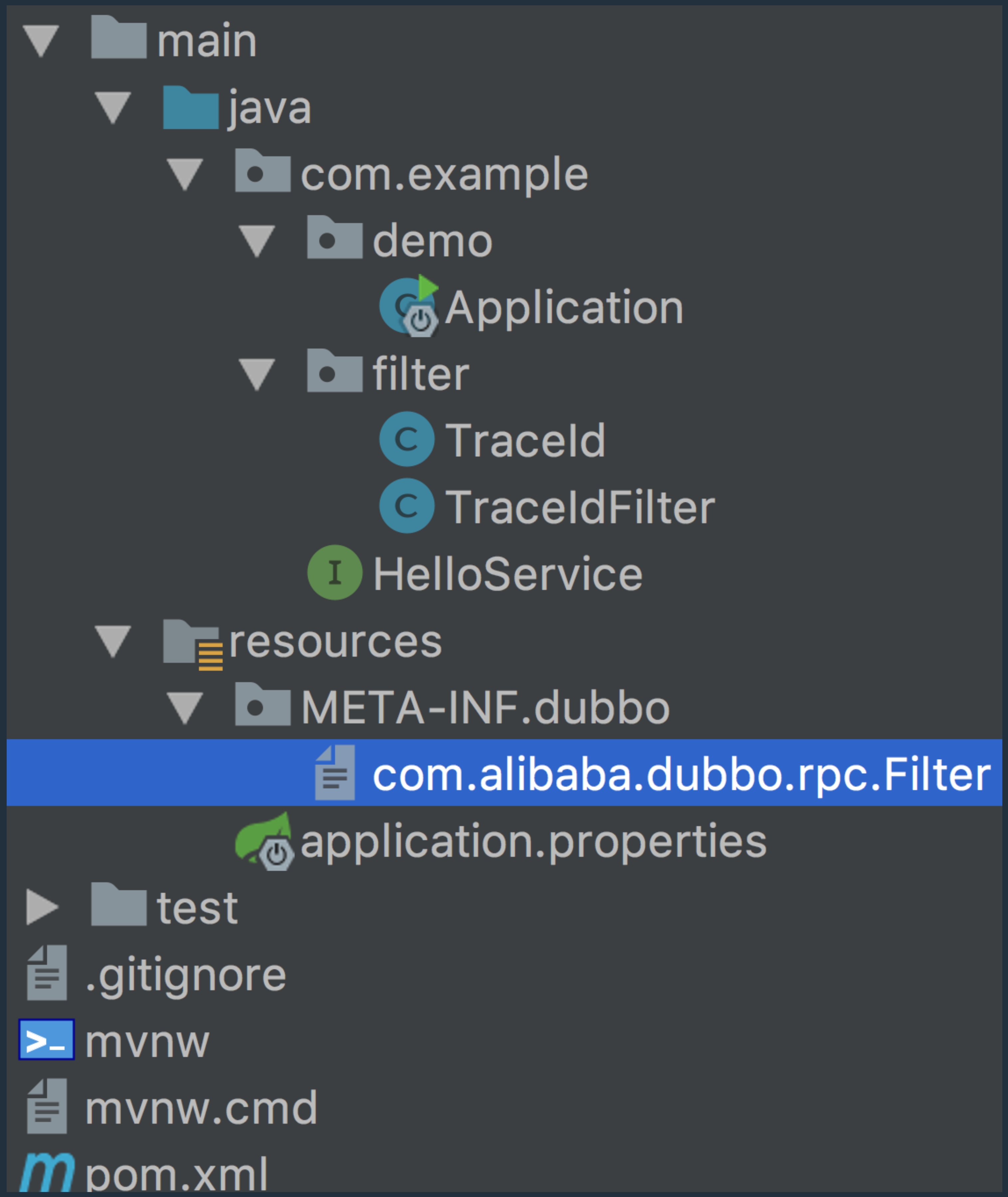
TRACEIDFILTER.JAVA

```
@Activate(group = {Constants.CONSUMER})
public class TraceIdFilter implements Filter {
    @Override
    public Result invoke(Invoker<?> invoker,
        Invocation invocation) throws RpcException {
        RpcContext.getContext().setAttachment("traceId",
            TraceId.get());
        return invoker.invoke(invocation);
    }
}
```

META-INF/DUBBO/COM.ALIBABA.DUBBO.RPC.FILTER

```
traceIdfilter=com.example.filter.TraceIdFilter
```

PROJECT LAYOUT



```
▼ main
    ▼ java
        ▼ com.example
            ▼ demo
                Application
            ▼ filter
                TraceId
                TraceIdFilter
            HelloService
    ▼ resources
        ▼ META-INF.dubbo
            com.alibaba.dubbo.rpc.Filter
        application.properties
▶ test
.gitignore
mvnw
mvnw.cmd
pom.xml
```

DEMO

https://github.com/dubbo/dubbo-spring-boot-demo

@ApacheDubbo

HOME

dubbo.apache.org

GITHUB

github.com/apache/incubator-dubbo

github.com/dubbo

MAILING LIST

dev@dubbo.apache.org

IM

gitter.im/alibaba/dubbo