servicecomb-kie Documentation

xiaoliang

Jan 16, 2020

User Documentations

1 Introductions **1**
- 1.1 What is ServiceComb kie . . . 1
- 1.2 Why use ServiceComb kie . . . 1
- 1.3 Concepts . . . 1
  - 1.3.1 Labels . . . 1
  - 1.3.2 key value . . . 1

2 Get started **3**
- 2.1 Quick start . . . 3
  - 2.1.1 With docker . . . 3
  - 2.1.2 Run locally with Docker compose . . . 4
  - 2.1.3 Run locally without Docker . . . 4

3 User guides **7**
- 3.1 Storage . . . 7
  - 3.1.1 Options . . . 7
  - 3.1.2 Example . . . 7
- 3.2 Registry . . . 8
  - 3.2.1 Options . . . 8
  - 3.2.2 Example . . . 8

4 Development guide **9**
- 4.1 Build . . . 9

CHAPTER 1

Introductions

1.1 What is ServiceComb kie

ServiceComb kie is a config server which manage configurations in a distributed system. It is also a micro service in ServiceComb ecosystem and developed by go-chassis we call it ServiceComb Native application.

1.2 Why use ServiceComb kie

Since it is a ServiceComb Native application, the user of ServiceComb is able to integrate with it easily, or even replace some implements of kie to develop a config server on-premise.

1.3 Concepts

1.3.1 Labels

key value must belong to a identical label, a labels is equivalent to a map, it is represent as a json object

```
{
"app": "some_app",
"service": "payment",
"environment": "production",
"version": "1.0.1"
}
```

1.3.2 key value

A key is usually related to some function in your program, let's say a new web UI should enable or not. the labels is just like a map to tell you where this program located in. the map says, it is located in production environment,the

version is 1.0.1

```
{
        "value":"true",
        "labels":{
              "service": "web-console",
              "environment": "production",
              "version": "1.0.1"
             }
}
```

CHAPTER 2

Get started

2.1 Quick start

2.1.1 With docker

Run mongodb server

write a script to create a user

```
cat <<EOM > db.js
db.createUser(
    {
        user: "root",
        pwd: "root",
        roles:[
            {
                role: "readWrite",
                db:   "kie"
            }
        ]
    }
);
EOM
```

```
sudo docker run --name mongo -d \
    -e "MONGO_INITDB_DATABASE=kie" \
    -e "MONGO_INITDB_ROOT_USERNAME=root" \
    -e "MONGO_INITDB_ROOT_PASSWORD=root" \
    -p 27017:27017 \
    -v ${PWD}/db.js:/docker-entrypoint-initdb.d/db.js:ro \
    mongo:3.4
```

```
export MONGO_IP=`sudo docker inspect --format '{{ .NetworkSettings.IPAddress }}'␣
↪mongo`
```

Run kie server

```
sudo docker run --name kie-server -d \
    -e "MONGODB_ADDR=${MONGO_IP}:27017" \
    -e "MONGODB_USER=root" \
    -e "MONGODB_PWD=root" \
    -p 30110:30110 \
    servicecomb/kie
```

Put a key

```
curl -X PUT \
  http://127.0.0.1:30110/v1/default/kie/kv/ingressRule.http \
  -H 'Content-Type: application/json' \
  -d '{
        "value":"some rule",
        "type": "yaml",
        "labels":{"app":"default"}
}'
```

response is

```
{
    "_id": "5d6f27c5a1b287c5074e4538",
    "label_id": "5d6f27c5a1b287c5074e4537",
    "key": "ingressRule.http",
    "value": "rule",
    "value_type": "text",
    "labels": {
        "app": "default"
    },
    "revision": 1
}
```

2.1.2 Run locally with Docker compose

```
git clone git@github.com:apache/servicecomb-kie.git
cd servicecomb-kie/deployments/docker
sudo docker-compose up
```

it will launch 3 components

- mongodb: 127.0.0.1:27017
- mongodb UI:http://127.0.0.1:8081
- servicecomb-kie: http://127.0.0.1:30110

2.1.3 Run locally without Docker

Download and run mongodb, see MongoDB Community Edition Installation Tutorials

Write a script to create a user

```
cat <<EOM > native_db.js
db.createUser(
    {
        user: "root",
        pwd: "root",
        roles:[
            {
                role: "readWrite",
                db:   "kie"
            }
        ]
    }
);
EOM
```

Exec native_db.js

```
mongo 127.0.0.1/kie native_db.js
```

Download the binary of kie, see https://apache.org/dyn/closer.cgi/servicecomb/servicecomb-kie/0.1.0/ Unzip and Run it. For Example:

```
tar -xzvf apache-servicecomb-kie--${platform}-amd64.tar.gz
cd apache-servicecomb-kie--${platform}-amd64
./kie --config conf/kie-conf.yaml
```

CHAPTER 3

User guides

3.1 Storage

you can use mongo db as kie server storage to save configuration

3.1.1 Options

uri

(required, string) db uri

timeout

(optional, string) db operation timeout

sslEnabled

(optional, bool) enable TLS communication to mongodb server

rootCAFile

(optional, bool) if sslEnabled is true, you must give a ca file

3.1.2 Example

```
db:
  uri: mongodb://kie:123@127.0.0.1:27017
  poolSize: 10
  timeout:  5m
  sslEnabled: true
  rootCAFile: /opt/kie/ca.crt
```

3.2 Registry

kie server is able to register to a registry, so that other client is able to discover it registry service. by default registry feature is disabled.

```
cse:
  service:
    registry:
      disabled: true
...
```

For example, kie is able to register to ServiceComb service center.

3.2.1 Options

the function is powered by go-chassis in chassis.yaml file refer to https://docs.go-chassis.com/user-guides/registry.html

3.2.2 Example

Register to ServiceComb service center

```
cse:
  service:
    registry:
      address: http://127.0.0.1:30100
...
```

CHAPTER 4

Development guide

4.1 Build

Download and run mongodb, see MongoDB Community Edition Installation Tutorials

Write a script to create a user

```
cat <<EOM > native_db.js
db.createUser(
    {
        user: "root",
        pwd: "root",
        roles:[
            {
                role: "readWrite",
                db:   "kie"
            }
        ]
    }
);
EOM
```

Exec native_db.js

```
mongo 127.0.0.1/kie native_db.js
```

Build serivcecomb-kie binary

```
cd ${servicecomb_root_path}
build/build_binary.sh
```

This will build 4 packages for 4 different platforms in ${servicecomb_root_path}/release/kie directory. Choose the right one and unzip and run it. For Example:

```
cd ${servicecomb_root_path}/release/kie
tar -xzvf apache-servicecomb-kie--${platform}-amd64.tar.gz
cd apache-servicecomb-kie--${platform}-amd64
./kie --config conf/kie-conf.yaml
```