**项目模块简介**

|  |  |  |
| --- | --- | --- |
| 名称 | 介绍 | 说明 |
| config-server | 远程配置管理服务 | spring cloud config配置中心，演示基本的config功能 |
| erureka-server | cloud微服务注册中心 | 基于REST的定位服务，以实现云端中间层服务发现和故障转移 |
| mall-portal | mall商城入口服务 | 用于演示 feign+ribbon+hystrix+zuul 等组件基本的使用和配置 |
| service-member | mall会员微服务(仅演示) | SpringBoot+MybatisPlus框架的业务模块微服务，可随意扩展重建，  附带feign+config+bus等组件的演示，  整合Redis+RabbitMQ中间件的基本使用 |
| service-order | mall订单微服务(仅演示) | SpringBoot+MybatisPlus框架的业务模块微服务，可随意扩展重建，  附带feign+config+bus等组件的演示，  整合Redis+RabbitMQ中间件的基本使用 |
| service-product | mall商品微服务(仅演示) | SpringBoot+MybatisPlus框架的业务模块微服务，可随意扩展重建，  附带feign+config+bus等组件的演示，  整合Redis+RabbitMQ中间件的基本使用 |

**Docker容器中间件部署**

Mysql

docker pull mysql:5.7

sudo docker run --name pwc-mysql -e MYSQL\_ROOT\_PASSWORD=root -p 3306:3306 -d mysql:5.7

Redis

docker pull redis:3.2

docker run -p 6379:6379 --name redis -v /etc/localtime:/etc/localtime:ro -v /mnt/docker/redis/data:/data -d redis:3.2

Rabbitmq

docker pull rabbitmq:management

docker run -p 5672:5672 -p 15672:15672 --name rabbitmq -v /etc/localtime:/etc/localtime:ro -d rabbitmq:management

Service启动后的测试地址

#erureka-server

<http://localhost:1001/>

#config-server

[http://](http://192.168.0.146:2001/service-member-dev.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-member-dev.yml](http://192.168.0.146:2001/service-member-dev.yml)

[http://](http://192.168.0.146:2001/service-member-test.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-member-test.yml](http://192.168.0.146:2001/service-member-test.yml)

[http://](http://192.168.0.146:2001/service-member-prd.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-member-prd.yml](http://192.168.0.146:2001/service-member-prd.yml)

[http://](http://192.168.0.146:2001/service-product-dev.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-product-dev.yml](http://192.168.0.146:2001/service-product-dev.yml)

[http://](http://192.168.0.146:2001/service-product-test.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-product-test.yml](http://192.168.0.146:2001/service-product-test.yml)

[http://](http://192.168.0.146:2001/service-product-prd.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-product-prd.yml](http://192.168.0.146:2001/service-product-prd.yml)

[http://](http://192.168.0.146:2001/service-order-dev.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-order-dev.yml](http://192.168.0.146:2001/service-order-dev.yml)

[http://](http://192.168.0.146:2001/service-order-test.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-order-test.yml](http://192.168.0.146:2001/service-order-test.yml)

[http://](http://192.168.0.146:2001/service-order-prd.yml)[localhost](http://192.168.0.146:3001/)[:2001/service-order-prd.yml](http://192.168.0.146:2001/service-order-prd.yml)

[http://](http://192.168.0.146:2001/actuator/bus-refresh)[localhost](http://192.168.0.146:3001/)[:2001/actuator/bus-refresh](http://192.168.0.146:2001/actuator/bus-refresh) POST

#service-member

[http://](http://192.168.5.20:8001/profile)[localhost](http://192.168.0.146:3001/)[:8001/profile](http://192.168.5.20:8001/profile)

[http://](http://192.168.5.20:8001/api/user?username=baicai)[localhost](http://192.168.0.146:3001/)[:8001/api/user?username=baicai](http://192.168.5.20:8001/api/user?username=baicai)

[http://](http://192.168.5.20:8001/api/user/cache?username=baicai)[localhost](http://192.168.0.146:3001/)[:8001/api/user/cache?username=baicai](http://192.168.5.20:8001/api/user/cache?username=baicai)

#service-product

[http://](http://192.168.5.20:8003/profile)[localhost](http://192.168.0.146:3001/)[:8003/profile](http://192.168.5.20:8003/profile)

[http://](http://192.168.5.20:8003/api/product?sn=SN123456)[localhost](http://192.168.0.146:3001/)[:8003/api/product?sn=SN123456](http://192.168.5.20:8003/api/product?sn=SN123456)

[http://](http://192.168.5.20:8003/api/product/cache?sn=SN123456)[localhost](http://192.168.0.146:3001/)[:8003/api/product/cache?sn=SN123456](http://192.168.5.20:8003/api/product/cache?sn=SN123456)

#service-order

[http://](http://192.168.5.20:8002/profile)[localhost](http://192.168.0.146:3001/)[:8002/profile](http://192.168.5.20:8002/profile)

[http://](http://192.168.5.20:8002/api/order?sn=Q123456)[localhost](http://192.168.0.146:3001/)[:8002/api/order?sn=Q123456](http://192.168.5.20:8002/api/order?sn=Q123456)

[http://](http://192.168.5.20:8002/api/order/cache?sn=Q123456)[localhost](http://192.168.0.146:3001/)[:8002/api/order/cache?sn=Q123456](http://192.168.5.20:8002/api/order/cache?sn=Q123456)

[http://localhost:8002/api/order/tx?sn=Q123456&productId=1&memberId=1](http://192.168.5.20:8002/api/order/tx?sn=Q123456&productId=1&memberId=1)

#mall-portal

[http://](http://192.168.5.20:9001/api/find/data?username=baicai&productSn=SN123456&orderSn=Q123456)[localhost](http://192.168.0.146:3001/)[:9001/api/find/data?username=baicai&productSn=SN123456&orderSn=Q123456](http://192.168.5.20:9001/api/find/data?username=baicai&productSn=SN123456&orderSn=Q123456)

[http://](http://192.168.5.20:9001/apigateway/member/api/user?username=baicai)[localhost](http://192.168.0.146:3001/)[:9001/apigateway/member/api/user?username=baicai](http://192.168.5.20:9001/apigateway/member/api/user?username=baicai)

[http://](http://192.168.5.20:9001/apigateway/product/api/product?sn=SN123456)[localhost](http://192.168.0.146:3001/)[:9001/apigateway/product/api/product?sn=SN123456](http://192.168.5.20:9001/apigateway/product/api/product?sn=SN123456)

[http://](http://192.168.5.20:9001/apigateway/order/api/order?sn=Q123456)[localhost](http://192.168.0.146:3001/)[:9001/apigateway/order/api/order?sn=Q123456](http://192.168.5.20:9001/apigateway/order/api/order?sn=Q123456)

Docker Compose编排项目

**注意：**

1、因为order，member和product服务启动需要读取config服务，所以需要将docker-compose拆分，让eureka和config先启动

2、因为容器没法直接访问宿主机的真实ip，只能访问宿主机给容器映射的ip(用ifconfig命令查出来的docker0网卡的ip)，所以容器要访问安装在宿主机上的那些基础服务必须用宿主机的映射ip

docker-compose.yml内容如下，

version: '2' #docker的文件格式版本

services:

eureka-server: #docker服务名

image: eureka-server #docker镜像

ports:

- "1001:1001"

config-server:

image: config-server

command:

- "--mysql.address=172.17.0.1"

ports:

- "2001:2001"

service-member:

image: service-member

command:

- "--mysql.address=172.17.0.1"

- "--redis.address=172.17.0.1"

- "--rabbitmq.address=172.17.0.1"

ports:

- "8001:8001"

service-order:

image: service-order

command:

- "--mysql.address=172.17.0.1"

- "--redis.address=172.17.0.1"

- "--rabbitmq.address=172.17.0.1"

ports:

- "8002:8002"

service-product:

image: service-product

command:

- "--mysql.address=172.17.0.1"

- "--redis.address=172.17.0.1"

- "--rabbitmq.address=172.17.0.1"

ports:

- "8003:8003"

mall-portal:

image: mall-portal

ports:

- "9001:9001"