

百度APP

全民战疫情 百度在行动

新型肺炎患者同乘查询

微服务入门这一篇就够了

centychen [关注](#)

13 2019.08.09 17:28:58 字数 2,593 阅读 143,735

开篇

刚开始进入软件行业时还是单体应用的时代，前后端分离的概念都还没普及，开发的时候需要花大量的时间在“强大”的JSP上面，那时候SOA已经算是新技术了。现在，微服务已经大行其道，有哪个互联网产品不说自己是微服务架构呢？

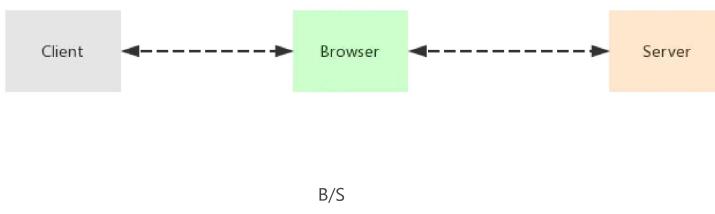
但是，对于微服务的理解每个人都不太一样，这篇文章主要是聊一聊我对微服务的理解以及如何搭建经典的微服务架构，目的是梳理一下自己的一些想法，如果存在不同看法的欢迎指正！

什么是微服务

首先，什么是微服务呢？

单体应用

相对的，要理解什么是微服务，那么可以先理解什么是单体应用，在没有提出微服务的概念的“远古”年代，一个软件应用，往往会将应用所有功能都开发和打包在一起，那时候的一个B/S应用架构往往是这样的：



但是，当用户访问量变大导致一台服务器无法支撑时怎么办呢？加服务器加负载均衡，架构就变成这样了：

推荐阅读

微服务、容器、云原生、
Kubernetes、SOA、PaaS平台、...
阅读 820

网关背景分类及常用框架
阅读 178

小白也能看懂的dubbo3应用级服务
发现详解
阅读 325

rabowl---umi-qiankun微服务框架企业实践(1)
阅读 220

feign和openfeign的区别
阅读 595

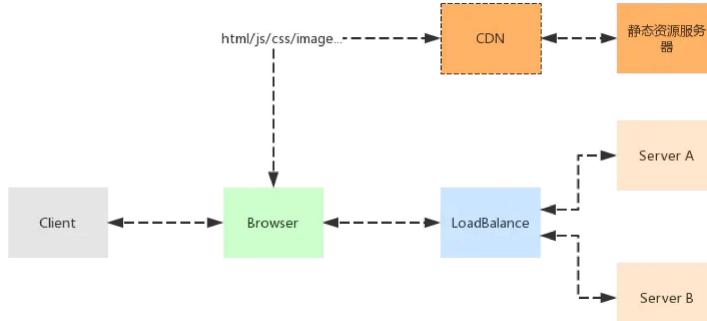
阅读 612



电脑摄像头



后面发现把静态文件独立出来，通过CDN等手段进行加速，可以提升应用的整体响应，单体应用的架构就变成：



B/S+前端分离

上面3中架构都还是单体应用，只是在部署方面进行了优化，所以避免不了单体应用的根本的缺点：

- 代码臃肿，应用启动时间长；（代码超过1G的项目都有！）
- 回归测试周期长，修复一个小小bug可能都需要对所有关键业务进行回归测试。
- 应用容错性差，某个小小功能的程序错误可能导致整个系统宕机；
- 伸缩困难，单体应用扩展性能时只能整个应用进行扩展，造成计算资源浪费。
- 开发协作困难，一个大型应用系统，可能几十个甚至上百个开发人员，大家都在维护一套代码的话，代码merge复杂度急剧增加。

微服务

我认为任何技术的演进都是有迹可循的，任何新技术的出现都是为了解决原有技术无法解决的需求，所以，微服务的出现就是因为原来单体应用架构已经无法满足当前互联网产品的技术需求。

在微服务架构之前还有一个概念：SOA (Service-Oriented Architecture) -面向服务的体系架构。我认为的SOA只是一个架构模型的方法论，并不是一个明确而严谨的架构标准，只是后面很多人将SOA与The Open Group的SOA参考模型等同了，认为严格按照TOG-SOA标准的才算真正的SOA架构。SOA就已经提出的面向服务的架构思想，所以微服务应该算是SOA的一种演进吧。

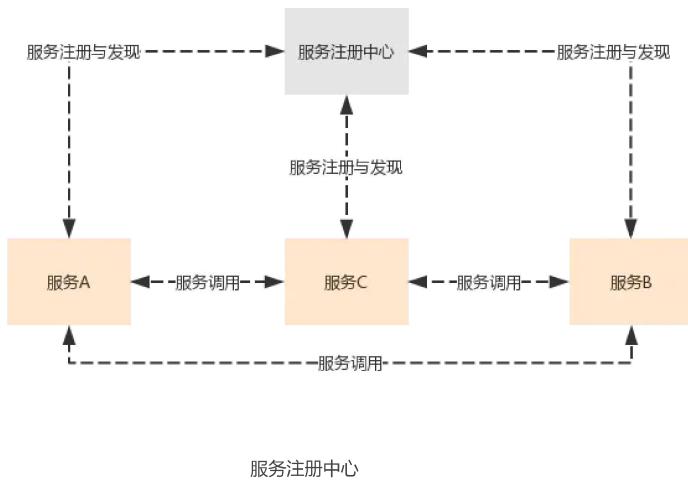
撇开架构先不说，什么样的服务才算微服务呢？

- 单一职责的。一个微服务应该都是单一职责的，这才是“微”的体现，一个微服务解决一个业务问题（注意是一个业务问题而不是一个接口）。
- 面向服务的。将自己的业务能力封装并对外提供服务，这是继承SOA的核心思想，一个微服务本身也可能使用到其它微服务的能力。

我觉得满足以上两点就可以认为典型的微服务。



应用微服务化之后，首先遇到的第一个问题就是服务发现问题，一个微服务如何发现其他微服务呢？最简单的方式就是每个微服务里面配置其他微服务的地址，但是当微服务数量众多的时候，这样做明显不现实。所以需要使用到微服务架构中的一个最重要的组件：**服务注册中心**，所有服务都注册到服务注册中心，同时也可以从服务注册中心获取当前可用的服务清单：



推荐阅读

微服务、容器、云原生、Kubernetes、SOA、PaaS平台、...
阅读 820

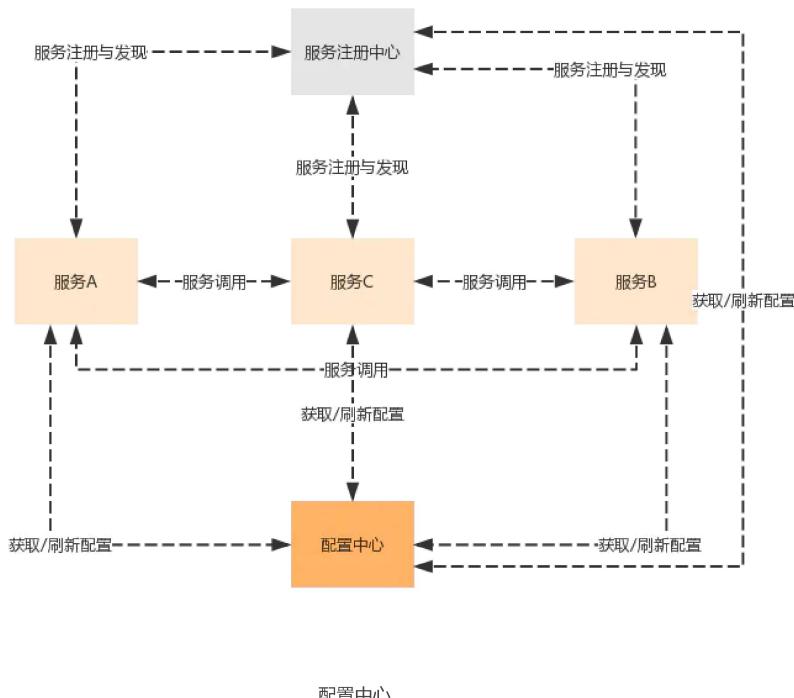
网关背景分类及常用框架
阅读 178

小白也能看懂的dubbo3应用级服务发现详解
阅读 325

rabowl---umi-qiankun微服务框架企业实践(1)
阅读 220

feign和openfeign的区别
阅读 595

解决服务发现问题后，接着需要解决微服务分布式部署带来的第二个问题：服务配置管理的问题。当服务数量超过一定程度之后，如果需要在每个服务里面分别维护每一个服务的配置文件，运维人员估计要哭了。那么，就需要用到微服务架构里面第二个重要的组件：**配置中心**，微服务架构就变成下面这样了：



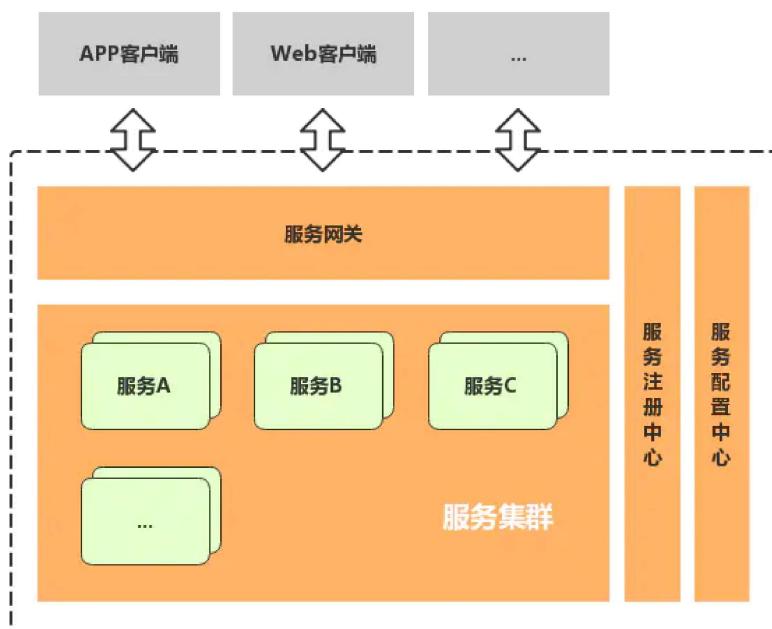
以上应用内部的服务治理，当客户端或外部应用调用服务的时候怎么处理呢？服务A可能有多

个节点

评论16

赞193





典型微服务架构

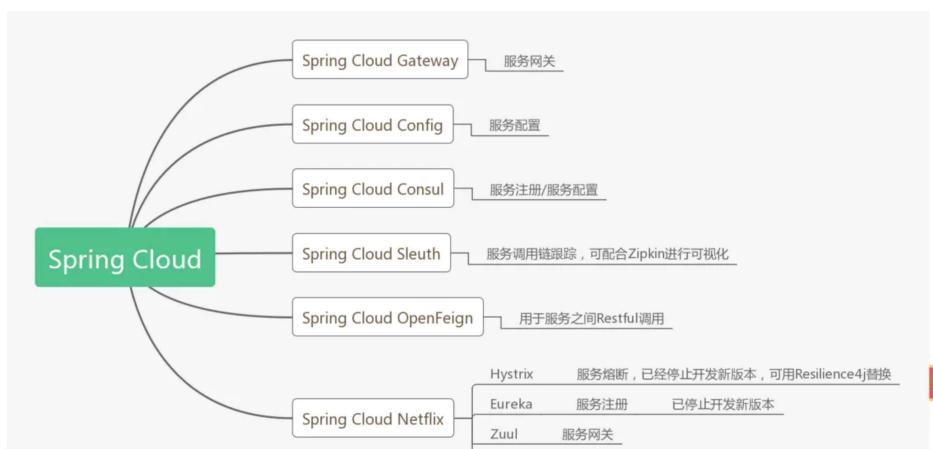
上面是一个典型的微服务架构，当然微服务的服务治理还涉及很多内容，比如：

- 通过熔断、限流等机制保证高可用；
- 微服务之间调用的负载均衡；
- 分布式事务（2PC、3PC、TCC、LCN等）；
- 服务调用链跟踪等等。

微服务框架

目前国内企业使用的微服务框架主要是Spring Cloud和Dubbo（或者DubboX），但是Dubbo那两年的停更严重打击了开发人员对它的信心，Spring Cloud已经逐渐成为主流，比较两个框架的优劣势的文章在网上有很多，这里就不重复了，选择什么框架还是按业务需求来吧，业务框架决定技术框架。

Spring Cloud全家桶提供了各种各样的组件，基本可以覆盖微服务的服务治理的方方面面，以下列出了Spring Cloud一些常用组件：



写下你的评论...

评论16

赞193

推荐阅读

微服务、容器、云原生、
Kubernetes、SOA、PaaS平台、...
阅读 820

网关背景分类及常用框架
阅读 178

小白也能看懂的dubbo3应用级服务
发现详解
阅读 325

rabowl---umi-qiankun微服务框架企
业实践(1)
阅读 220

feign和openfeign的区别
阅读 595

搭建典型微服务架构

本章节主要介绍如何基于Spring Cloud相关组件搭建一个典型的微服务架构。

首先，创建一个Maven父项目 `spring-cloud-examples`，用于管理项目依赖包版本。由于Spring Cloud组件很多，为保证不同组件之间的兼容性，一般通过 `spring-cloud-dependencies` 统一管理Spring Cloud组件版本，而非每个组件单独引入。

pom.xml配置如下：

```

1 <!-- 继承SpringBoot父项目，注意与SpringCloud版本的匹配 -->
2 <parent>
3   <groupId>org.springframework.boot</groupId>
4   <artifactId>spring-boot-starter-parent</artifactId>
5   <version>2.1.4.RELEASE</version>
6 </parent>
7
8 <properties>
9   <spring.boot.version>2.1.4.RELEASE</spring.boot.version>
10  <spring.cloud.version>Greenwich.SR1</spring.cloud.version>
11  <lombok.version>1.18.8</lombok.version>
12  <maven.compiler.plugin.version>3.8.1</maven.compiler.plugin.version>
13 </properties>
14
15
16 <dependencyManagement>
17   <dependencies>
18     <dependency>
19       <groupId>org.springframework.cloud</groupId>
20       <artifactId>spring-cloud-dependencies</artifactId>
21       <version>${spring.cloud.version}</version>
22       <type>pom</type>
23       <scope>import</scope>
24     </dependency>
25   </dependencies>
26 </dependencyManagement>
```

搭建服务配置中心

- 在 `spring-cloud-examples` 项目下创建一个子项目 `spring-cloud-example-config`，添加Spring Cloud Config Server端的相关依赖包：

```

1 <dependencies>
2   <dependency>
3     <groupId>org.springframework.cloud</groupId>
4     <artifactId>spring-cloud-config-server</artifactId>
5   </dependency>
6 </dependencies>
```

- 添加Spring Boot配置文件 `application.yml`，配置如下：

```

1 spring:
2   application:
3     name: spring-cloud-example-config
4   profiles:
5     active: native #启用本地配置文件
6   cloud:
7     config:
8       server:
9         native:
10          search_LOCATIONS: classpath:/configs/ #配置文件扫描目录
11
12 server:
13   port: 8000 #服务端口
```



```

1 @SpringBootApplication
2 @EnableConfigServer
3 public class Application {
4
5     public static void main(String[] args) {
6         SpringApplication.run(Application.class, args);
7     }
8 }
```

搭建服务注册中心

- 在 `spring-cloud-examples` 项目下创建一个子项目 `spring-cloud-example-registry`，在 `pom.xml` 中添加 Eureka Server 相关依赖包：

```

1 <dependencies>
2     <dependency>
3         <groupId>org.springframework.cloud</groupId>
4         <artifactId>spring-cloud-netflix-eureka-server</artifactId>
5     </dependency>
6
7     <dependency>
8         <groupId>org.springframework.cloud</groupId>
9         <artifactId>spring-cloud-starter-config</artifactId>
10    </dependency>
11 </dependencies>
```

- 在 `spring-cloud-example-config` 配置中心项目的 `src/main/resource/configs` 目录下添加一个服务配置文件 `spring-cloud-example-registry.yml`，配置如下：

```

1 spring:
2     application:
3         name: spring-cloud-example-registry
4
5 # Eureka相关配置
6 eureka:
7     client:
8         register-with-eureka: false #不注册服务
9         fetch-registry: false #不拉去服务清单
10        serviceUrl:
11            defaultZone: http://localhost:${server.port}/eureka/ #多个通过英文逗号分隔
12
13 server:
14     port: 8001
```

- 在 `spring-cloud-example-registry` 项目的 `src/main/resource/` 目录添加 `bootstrap.yml` 配置文件，配置如下：

```

1 spring:
2     cloud:
3         config:
4             name: spring-cloud-example-registry #配置文件名称，多个通过逗号分隔
5             uri: http://localhost:8000 #Config Server服务地址
```

- 启动类添加注解 `@EnableEurekaServer` 通过启用 Eureka Server 服务。

```

1 @SpringBootApplication
2 @EnableEurekaServer
3 public class Application {
4
5     public static void main(String[] args) {
```



搭建业务服务A

- 在 `spring-cloud-examples` 项目下创建一个业务服务A的子项目 `spring-cloud-example-biz-a`，在 `pom.xml` 中添加以下依赖包：

```

1 <dependencies>
2   <!-- Spring Boot Web Starter -->
3   <dependency>
4     <groupId>org.springframework.boot</groupId>
5     <artifactId>spring-boot-starter-web</artifactId>
6   </dependency>
7
8   <!-- feign -->
9   <dependency>
10    <groupId>org.springframework.cloud</groupId>
11    <artifactId>spring-cloud-starter-openfeign</artifactId>
12  </dependency>
13
14   <!-- Eureka Client Starter -->
15   <dependency>
16     <groupId>org.springframework.cloud</groupId>
17     <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
18   </dependency>
19
20   <!-- Config Client Starter -->
21   <dependency>
22     <groupId>org.springframework.cloud</groupId>
23     <artifactId>spring-cloud-starter-config</artifactId>
24   </dependency>
25 </dependencies>

```

- 在 `spring-cloud-example-config` 配置中心项目的 `src/main/resource/configs` 目录下添加一个服务

配置文件 `spring-cloud-example-biz-a.yml`，配置如下：

```

1 spring:
2   application:
3     name: spring-cloud-example-biz-a
4
5 server:
6   port: 8010
7
8 # Eureka相关配置
9 eureka:
10  client:
11    serviceUrl:
12      defaultZone: http://localhost:8001/eureka/
13  instance:
14    lease-renewal-interval-in-seconds: 10      # 心跳时间，即服务续约间隔时间（缺省为30s）
15    lease-expiration-duration-in-seconds: 60    # 告警时间，即服务续约到期时间（缺省为90s）
16    prefer-ip-address: true
17    instance-id: ${spring.application.name}:${spring.application.instance_id:${server.port}}

```

- 在 `spring-cloud-example-biz-a` 项目的 `src/main/resource/` 目录添加 `bootstrap.yml` 配置文件，配置如下：

```

1 spring:
2   cloud:
3     config:
4       name: spring-cloud-example-biz-a #配置文件名称，多个通过逗号分隔
5       uri: http://localhost:8000 #Config Server服务地址

```

- 添加一个示例接口，代码参考：



推荐阅读

微服务、容器、云原生、
Kubernetes、SOA、PaaS平台、...

阅读 820

网关背景分类及常用框架

阅读 178

小白也能看懂的dubbo3应用级服务
发现详解

阅读 325

rabowl---umi-qiankun微服务框架企
业实践(1)

阅读 220

feign和openfeign的区别

阅读 595

```

7     * @return
8     */
9     @GetMapping
10    public String sayHello() {
11        return "Hello,This is Biz-A Service.";
12    }
13 }
```

推荐阅读

微服务、容器、云原生、
Kubernetes、SOA、PaaS平台、...
阅读 820

网关背景分类及常用框架
阅读 178

小白也能看懂的dubbo3应用级服务
发现详解
阅读 325

rabowl---umi-qiankun微服务框架企
业实践(1)
阅读 220

feign和openfeign的区别
阅读 595

搭建业务服务B

参考上面业务服务A搭建另外一个业务服务B。

搭建服务网关

- 在 `spring-cloud-examples` 项目下创建一个业务服务A的子项目 `spring-cloud-example-gateway`，在 `pom.xml` 中添加以下依赖包：

```

1 <dependencies>
2     <!-- zuul -->
3     <dependency>
4         <groupId>org.springframework.cloud</groupId>
5         <artifactId>spring-cloud-starter-netflix-zuul</artifactId>
6     </dependency>
7
8     <!-- Eureka Client Starter -->
9     <dependency>
10        <groupId>org.springframework.cloud</groupId>
11        <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
12    </dependency>
13
14     <!-- Config Client Starter -->
15     <dependency>
16         <groupId>org.springframework.cloud</groupId>
17         <artifactId>spring-cloud-starter-config</artifactId>
18     </dependency>
19 </dependencies>
```

- 在 `spring-cloud-example-config` 配置中心项目的 `src/main/resource/configs` 目录下添加一个服务

配置文件 `spring-cloud-example-gateway.yml`，配置如下：

```

1 spring:
2     application:
3         name: spring-cloud-example-gateway
4
5 server:
6     port: 8002
7
8 # Eureka相关配置
9 eureka:
10    client:
11        serviceUrl:
12            defaultZone: http://localhost:8001/eureka/
13    instance:
14        lease-renewal-interval-in-seconds: 10      # 心跳时间，即服务续约间隔时间（缺省为30s）
15        lease-expiration-duration-in-seconds: 60   # 发呆时间，即服务续约到期时间（缺省为90s）
16        prefer-ip-address: true
17        instance-id: ${spring.application.name}:${spring.application.instance_id:${server.port}}
```

- 在 `spring-cloud-example-gateway` 项目的 `src/main/resource/` 目录添加 `bootstrap.yml` 配置文件，配置如下：

```
1 spring:
```

写下你的评论...

评论16

赞193



- 启动类添加注解 `@EnableZuulProxy` 通过启用网关代理服务。

```

1  @SpringBootApplication
2  @EnableZuulProxy
3  public class Application {
4
5      public static void main(String[] args) {
6          SpringApplication.run(Application.class, args);
7      }
8  }

```

启动示例

- 启动顺序

```

spring-cloud-example-config >> spring-cloud-example-eureka >> spring-cloud-example-biz-
a / spring-cloud-example-biz-b / spring-cloud-example-gateway

```

- 通过网关访问服务A接口

← → ⌂ ⓘ localhost:8002/spring-cloud-example-biz-a/hello

Hello,This is Biz-A Service.

服务A调用

- 通过网关访问服务B接口

← → ⌂ ⓘ localhost:8002/spring-cloud-example-biz-b/hello

Hello,This is Biz-B Service.

服务B调用

服务之间调用

- 在业务服务A中添加一个Feign Client Bean，参考代码如下：

```

1  @FeignClient(name = "spring-cloud-example-biz-b") # 指定服务名称
2  public interface RemoteService {
3
4      /**
5       * 调用服务B的hello方法
6       *
7       * @return
8       */
9      @GetMapping("/hello") #指定请求地址
10     String sayHello();

```



写下你的评论...

评论16

赞193

```

1  @RestController
2  @RequestMapping("/hello")
3  public class HelloController {
4
5      @Autowired
6      private RemoteService remoteService;
7
8      /**
9       * 示例方法
10      *
11      * @return
12      */
13     @GetMapping
14     public String sayHello() {
15         return "Hello,This is Biz-A Service.";
16     }
17
18     /**
19      * 示例方法: 调用服务B
20      *
21      * @return
22      */
23     @GetMapping(path = "/call2b")
24     public String sayHello2B() {
25         return remoteService.sayHello();
26     }
27 }
```

推荐阅读

微服务、容器、云原生、
Kubernetes、SOA、PaaS平台、...
阅读 820

网关背景分类及常用框架
阅读 178

小白也能看懂的dubbo3应用级服务
发现详解
阅读 325

rabowl---umi-qiankun微服务框架企
业实践(1)
阅读 220

feign和openfeign的区别
阅读 595

- 重启业务服务A，通过调用 `/hello/call2b` 接口：

localhost:8002/spring-cloud-example-biz-a/hello/call2b

Hello,This is Biz-B Service.

服务之间调用

示例代码

- 码云：[centychen/spring-cloud-examples](#)

下一代微服务

目前网上很多说是下一代微服务架构就是Service Mesh，Service Mesh主流框架有Linkerd和Istio，其中Istio有大厂加持所以呼声更高。Service Mesh我接触还不多，但是个人感觉并不一定能称为下一代微服务架构，可能认为是服务治理的另外一种解决方案更合适，是否能够取代当前的微服务架构还需要持续观察。



193人点赞 >



Spring Cloud

...

更多精彩内容，就在简书APP

"小礼物走一走，来简书关注我"

赞赏支持



抽奖



还没有人赞赏，支持一下

写下你的评论...

评论16

赞193

推荐阅读

微服务、容器、云原生、
Kubernetes、SOA、PaaS平台、...
阅读 820

网关背景分类及常用框架
阅读 178

小白也能看懂的dubbo3应用级服务
发现详解
阅读 325

rabowl---umi-qiankun微服务框架企
业实践(1)
阅读 220

feign和openfeign的区别
阅读 595

百度搜索疫情实时大数据

写下你的评论...

| 全部评论 16 只看作者

按时间倒序 按时间正序



努力成为架构师
20楼 10.12 15:19

强! ↗

赞 回复



莫不是花海
19楼 09.09 14:30

感谢分享

赞 回复

签名文件一键自动生成



blue_bird
18楼 08.30 09:07

very good

赞 回复



blue_bird
17楼 08.16 12:00

讲解的通俗易懂，而且还配了示例项目，好文

赞 回复



玛门
16楼 05.30 15:49

谢谢分享~

赞 回复



Jefitar
15楼 04.27 19:27

厉害↗



写下你的评论...

评论16

赞193

写的真好

拇指 赞 回复



轰鸣之雨

13楼 2020.12.23 09:15

感谢分享

拇指 赞 回复



行走的300块_48ca

12楼 2020.10.22 19:05

想要调用其他服务的接口，要在启动类上添加 @EnableFeignClients 注解，不然会提示扫描不到对应的bean

拇指 赞 回复



1962bd77a46a

5楼 2019.11.26 16:42

spring-cloud-examples / spring-cloud-example-config / pom.xml里加个
<version>2.1.1.RELEASE</version>

拇指 赞 回复

1

2

下一页

被以下专题收入，发现更多相似内容

微服务入门 软件测试 微服务 SpringC... docker...

分布式系统 java 展开更多 ▾



写下你的评论...

评论16

赞193