

键盘敲出未来。。。IT民工，苦逼码农。。。RSS订阅

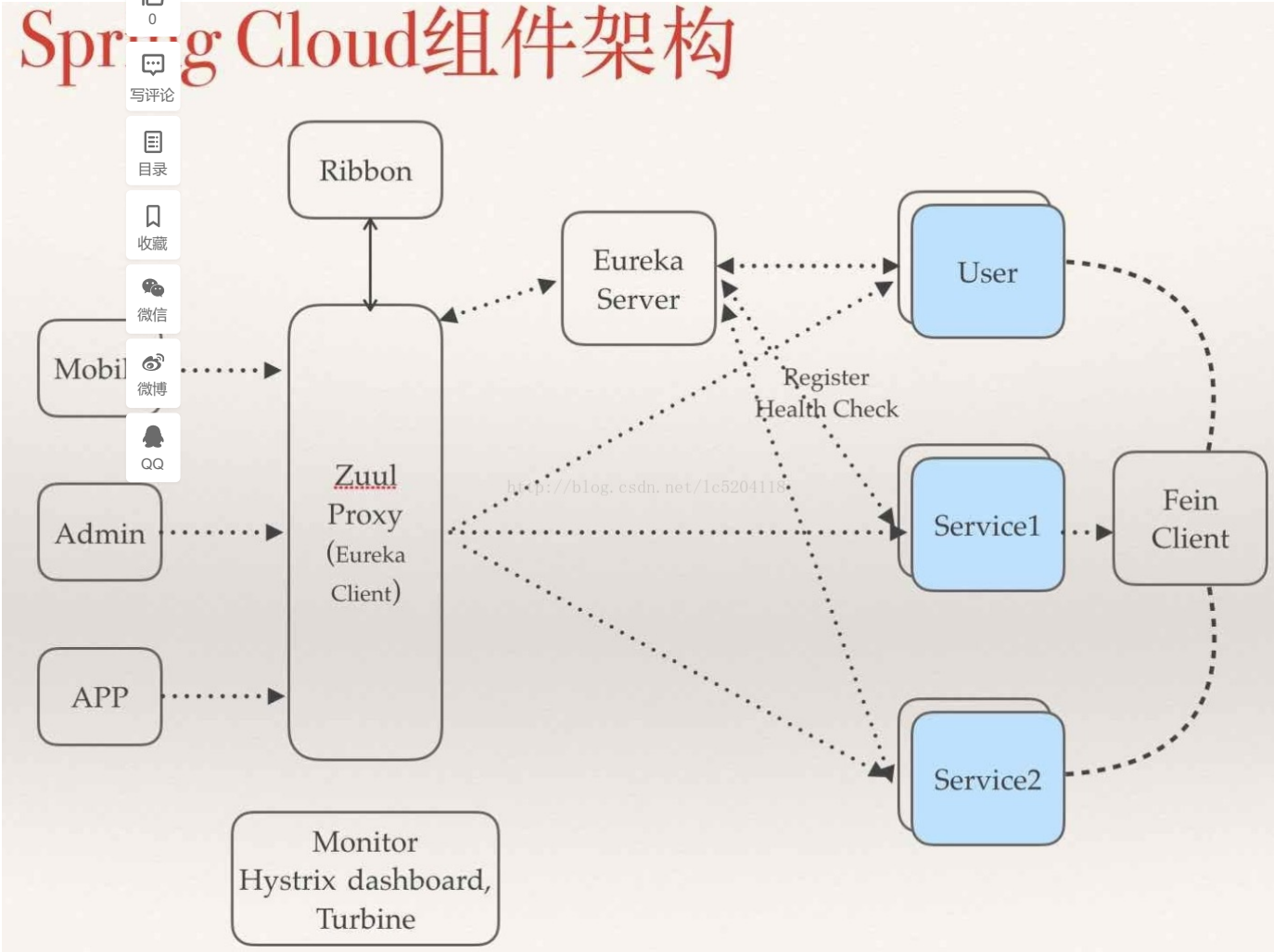
原

【微服务架构】springcloud微服务架构搭建

2017年11月08日 19:47:45

阅读数：4663

要会用，首先要了解。图懒得画，借鉴网上大牛的图吧，springcloud组建架构如图：



微服务架构的应用场景：

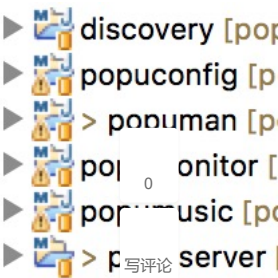
- 1、系统拆分，多个子系统
- 2、每个子系统可部署多个应用，应用之间负载均衡实现
- 3、需要一个服务注册中心，所有的服务都在注册中心注册，负载均衡也是通过在注册中心注册的服务来使用一定策略来实现。
- 4、所有的客户端都通过同一个网关地址访问后台的服务，通过路由配置，网关来判断一个URL请求由哪个服务处理。请求转发到服务上的时候也使用负载均衡。
- 5、服务之间有时候也需要相互访问。例如有一个用户模块，其他服务在处理一些业务的时候，要获取用户服务的用户数据。
- 6、需要一个断路器，及时处理服务调用时的超时和错误，防止由于其中一个服务的问题而导致整体系统的瘫痪。
- 7、还需要一个监控功能，监控每个服务调用花费的时间等。

Spring Cloud的优势

- 1 香港服务器，CN2中港专线
- 2 jQueryMiniUI，加速Web开发~90...

- 有spring Boot 这个独立干将可以省很多事，大大小小的活spring boot都搞的挺不错。
- 作为一个微服务治理的大家伙，考虑的很全面，几乎服务治理的方方面面都考虑到了，方便开发开箱即用。
- Spring Cloud 活跃度很高，教程很丰富，遇到问题很容易找到解决方案
- 轻轻松松几行代码就完成了熔断、均衡负责、服务中心的各种平台功能

废话少说，看代码：
项目架构：



一、discovery 注册发现

①、pom.xml

```
1 <?xml version="1.0"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"
4     >
5     <modelVersion>4.0.0</modelVersion>
6     <parent>
7         <groupId>com.gt</groupId>
8         <artifactId>popuserver</artifactId>
9         <version>0.0.1-SNAPSHOT</version>
10    </parent>
11
12
13    <artifactId>discovery</artifactId>
14    <name>discovery</name>
15    <url>http://www.popumusic</url>
16
17    <properties>
18        <start-class>com.wisely.discovery.DiscoveryApplication</start-class>
19    </properties>
20
21    <dependencies>
22        <dependency>
23            <groupId>org.springframework.cloud</groupId>
24            <artifactId>spring-cloud-starter-eureka-server</artifactId>
25        </dependency>
26    </dependencies>
27
28    <build>
29        <plugins>
30            <plugin>
31                <groupId>org.springframework.boot</groupId>
32                <artifactId>spring-boot-maven-plugin</artifactId>
33            </plugin>
34        </plugins>
35    </build>
36
37
38 </project>
39
```

②、DiscoveryApplication

```

2 | 3 | import org.springframework.boot.SpringApplication;
4 | import org.springframework.boot.autoconfigure.SpringBootApplication;
5 | import org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;
6 |
7 | @SpringBootApplication
8 | @EnableEurekaServer //1
9 | public class DiscoveryApplication {
10 |
11 |     public static void main(String[] args) {
12 |         SpringApplication.run(DiscoveryApplication.class, args);
13 |     }
14 |
15 | }

```

0

③、application.

写评论

```

1 | server:
2 |   port:
3 |   目录
4 | endpoint
5 |   shutdown
6 |   enabled: true
7 |   send: false
8 | eureka: 微信
9 |   instance:
10 |     prefer-ip-address: true #启用IP方式
11 |     ip-address: 127.0.0.1
12 |   client:
13 |     register-with-eureka: false #指向其他注册中心地址
14 |     fetch-registry: false
15 |     service-url:
16 |       defaultZone: http://127.0.0.1:8762/eureka/

```

二、monitor监控服务

①、pom.xml

```

1 | <?xml version="1.0"?>
2 | <project
3 |     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"
4 |     xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
5 |     <modelVersion>4.0.0</modelVersion>
6 |     <parent>
7 |         <groupId>com.gt</groupId>
8 |         <artifactId>popuserver</artifactId>
9 |         <version>0.0.1-SNAPSHOT</version>
10 |     </parent>
11 |     <groupId>com.poputar</groupId>
12 |     <artifactId>popumonitor</artifactId>
13 |     <version>0.0.1-SNAPSHOT</version>
14 |     <name>popumonitor</name>
15 |     <url>http://maven.apache.org</url>
16 |     <properties>
17 |         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
18 |     </properties>
19 |     <dependencies>
20 |         <dependency>
21 |             <groupId>org.springframework.cloud</groupId>
22 |             <artifactId>spring-cloud-starter</artifactId>
23 |             <version>1.1.7.RELEASE</version>
24 |         </dependency>
25 |         <dependency>
26 |             <groupId>org.springframework.cloud</groupId>
27 |             <artifactId>spring-cloud-starter-hystrix-dashboard</artifactId>
28 |             <version>1.2.2.RELEASE</version>
29 |         </dependency>

```

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

```
<dependency>31 | <groupId>org.springframework.cloud</groupId>
<artifactId>spring-cloud-starter-turbine</artifactId>
<version>1.1.7.RELEASE</version>
</dependency>
<dependency>
<groupId>junit</groupId>
<artifactId>junit</artifactId>
<version>3.8.1</version>
<scope>test</scope>
</dependency>
</dependencies>
<build>
<plugins>
<plugin>
<groupId>com.spotify</groupId>
<artifactId>docker-maven-plugin</artifactId>
<configuration>
<imageName>${project.name}:${project.version}</imageName>
<dockerDirectory>${project.basedir}/src/main/docker</dockerDirectory>
<skipDockerBuild>false</skipDockerBuild>
<resources>
<resource>
<directory>${project.build.directory}</directory>
<include>${project.build.finalName}.jar</include>
</resource>
</resources>
</configuration>
</plugin>
</plugins>
</build>
</project>
```

0

写评论

目录

收藏

微信

微博

QQ

②、PopumonitorApplication

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

```
package com.poputar;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.netflix.eureka.EnableEurekaClient;
import org.springframework.cloud.netflix.hystrix.dashboard.EnableHystrixDashboard;
import org.springframework.cloud.netflix.turbine.EnableTurbine;

/**
 * 监控服务
 */
@SpringBootApplication
@EnableEurekaClient
@EnableHystrixDashboard
@EnableTurbine
public class PopumonitorApplication
{
    public static void main( String[] args )
    {
        SpringApplication.run(PopumonitorApplication.class, args);
    }
}
```

③、application.yml

1

2

3

```
server:
  port: 8989
```

④、bootstrap.yml

```
1 spring:
2   application:
3     name: monitor
4
5 eureka:
6   instance:
7     nonSecurePort: ${server.port:8989}
8   client:
9     serviceUrl:
10      defaultZone: http://${eureka.host:localhost}:${eureka.port:8761}/eureka/
```

0

三、配置服务

①、pom.xml

```
1 <?xml version="1.0"?>
2 <project
3   xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"
5   >
6   <modelVersion>4.0.0</modelVersion>
7   <groupId>com.gt</groupId>
8   <artifactId>popuserver</artifactId>
9   <version>0.0.1-SNAPSHOT</version>
10  <parent>
11    <groupId>com.poputar</groupId>
12    <artifactId>popuconfig</artifactId>
13    <version>0.0.1-SNAPSHOT</version>
14    <name>popuconfig</name>
15    <url>http://maven.apache.org</url>
16    <properties>
17      <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
18    </properties>
19    <dependencies>
20      <dependency>
21        <groupId>org.springframework.cloud</groupId>
22        <artifactId>spring-cloud-starter</artifactId>
23        <version>1.1.7.RELEASE</version>
24      </dependency>
25      <dependency>
26        <groupId>org.springframework.cloud</groupId>
27        <artifactId>spring-cloud-config-server</artifactId>
28        <version>1.2.2.RELEASE</version>
29      </dependency>
30      <dependency>
31        <groupId>org.springframework.cloud</groupId>
32        <artifactId>spring-cloud-starter-eureka</artifactId>
33        <version>1.1.7.RELEASE</version>
34      </dependency>
35      <dependency>
36        <groupId>junit</groupId>
37        <artifactId>junit</artifactId>
38        <version>3.8.1</version>
39        <scope>test</scope>
40      </dependency>
41    </dependencies>
42    <build>
43      <plugins>
44        <plugin>
45          <groupId>com.spotify</groupId>
46          <artifactId>docker-maven-plugin</artifactId>
47          <configuration>
48            <imageName>${project.name}:${project.version}</imageName>
49            <dockerDirectory>${project.basedir}/src/main/docker</dockerDirectory>
50            <skipDockerBuild>false</skipDockerBuild>
```

```

51 |                                     <resources>
52 |                                     <resource>
53 |                                     <directory>${project.build.directory}</directory>
54 |                                     <include>${project.build.finalName}.jar</include>
55 |                                     </resource>
56 |                                     </resources>
57 |                                 </configuration>
58 |                             </plugin>
59 |                         </plugins>
60 |                     </build>
61 | </project>

```

②、Popuconf 0 ication

```

1 package popuconfig;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5 import org.springframework.cloud.config.server.EnableConfigServer;
6 import org.springframework.cloud.netflix.eureka.EnableEurekaClient;
7
8 /**
9  * 配置
10  * 微信
11  */
12 @SpringBootApplication
13 @EnableConfigServer
14 @EnableEurekaClient
15 public class PopuconfigApplication
16 {
17     public static void main( String[] args )
18     {
19         SpringApplication.run(PopuconfigApplication.class, args);
20     }
21 }

```

③、application.yml 配置文件放本地，读者可以自己研究下放在git服务上

```

1 spring:
2   cloud:
3     config:
4       server:
5         native:
6           search-locations: classpath:/config
7
8 server:
9   port: 8762

```

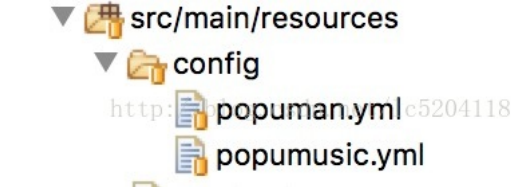
④、bootstrap.yml

```

1 spring:
2   application:
3     name: config #1
4   profiles:
5     active: native #2
6
7 eureka:
8   instance:
9     non-secure-port: ${server.port:8762} #3
10   metadata-map:
11     instanceId: ${spring.application.name}
12   client:
13     service-url:

```

⑤、src/main/resources/config下放应用所需的配置文件，命名方式跟appname相同，切记此处的命名是有规范的



四、用户服务

①、pom.xml引入外部jar包时，我已做注释，如下：

写评论

```
1 <?xml version="1.0"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"
4     >
5     <modelVersion>4.0.0</modelVersion>
6     <groupId>com.gt</groupId>
7     <artifactId>popuser</artifactId>
8     <version>0.0.1-SNAPSHOT</version>
9     <name>popuser</name>
10    <url>http://www.popumusic.com</url>
11    <dependencies>
12        <dependency>
13            <groupId>org.springframework.boot</groupId>
14            <artifactId>spring-boot-starter-data-redis</artifactId>
15        </dependency>
16        <dependency>
17            <groupId>org.springframework.cloud</groupId>
18            <artifactId>spring-cloud-starter-eureka</artifactId>
19        </dependency>
20        <dependency>
21            <groupId>org.springframework.boot</groupId>
22            <artifactId>spring-boot-starter-data-jpa</artifactId>
23        </dependency>
24        <dependency>
25            <groupId>mysql</groupId>
26            <artifactId>mysql-connector-java</artifactId>
27        </dependency>
28        <dependency>
29            <groupId>org.springframework.cloud</groupId>
30            <artifactId>spring-cloud-starter-config</artifactId>
31        </dependency>
32        <dependency>
33            <groupId>aliyun-java-sdk-dysmsapi</groupId>
34            <artifactId>aliyun-java-sdk-dysmsapi</artifactId>
35            <version>1.0.0</version>
36            <scope>system</scope>
37            <systemPath>${project.basedir}/lib/aliyun-java-sdk-dysmsapi-1.0.0.jar</systemPath>
38        </dependency>
39        <dependency>
40            <groupId>aliyun-java-sdk-core</groupId>
41            <artifactId>aliyun-java-sdk-core</artifactId>
42            <version>3.3.1</version>
43            <scope>system</scope>
44            <systemPath>${project.basedir}/lib/aliyun-java-sdk-core-3.3.1.jar</systemPath>
45        </dependency>
46    </dependencies>
47    </project>
```

```
53 |         </dependencies>
54 |     </build>
55 |     <build>
56 |         <plugins>
57 |             <plugin>
58 |                 <groupId>org.springframework.boot</groupId>
59 |                 <artifactId>spring-boot-maven-plugin</artifactId>
60 |             </plugin>
61 |         </plugins>
62 |
63 |         <!-- 将外部包打入jar docker部署时需要打开，否则报错，找不到jar -->
64 |         <!-- <resources>
65 |             <resource>
66 |                 <directory>lib</directory>
67 |                 <targetPath>BOOT-INF/lib/</targetPath>
68 |                 <includes>
69 |                     <include>/**/*.jar</include>
70 |                 </includes>
71 |             </resource>
72 |             <resource>
73 |                 <directory>src/main/resources</directory>
74 |                 <targetPath>BOOT-INF/classes/</targetPath>
75 |             </resource>
76 |         </resources> -->
77 |     </maven>
78 | </project>
79 |
```

0

写评论

目录

收藏

微信

微博

②、PopumanApplication增加了国际化配置

```
1 package com.gt;
2
3 import javax.validation.Validator;
4
5 import org.springframework.boot.SpringApplication;
6 import org.springframework.boot.autoconfigure.SpringBootApplication;
7 import org.springframework.cloud.netflix.eureka.EnableEurekaClient;
8 import org.springframework.context.annotation.Bean;
9 import org.springframework.context.support.ResourceBundleMessageSource;
10 import org.springframework.validation.beanvalidation.LocalValidatorFactoryBean;
11
12 @SpringBootApplication
13 @EnableEurekaClient
14 public class PopumanApplication {
15     public static void main(String[] args) {
16         SpringApplication.run(PopumanApplication.class, args);
17     }
18
19     public ResourceBundleMessageSource getMessageSource() throws Exception {
20         ResourceBundleMessageSource rbms = new ResourceBundleMessageSource();
21         rbms.setDefaultEncoding("UTF-8");
22         rbms.setBasenames("i18n/ValidationMessages");
23         return rbms;
24     }
25
26     @Bean
27     public Validator getValidator() throws Exception {
28         LocalValidatorFactoryBean validator = new LocalValidatorFactoryBean();
29         validator.setValidationMessageSource(getMessageSource());
30         return validator;
31     }
32 }
```

③、LocaleConfig 拦截器

```
1 package com.gt;
2
```



```

3 import java.util.Locale; 4
5 import org.springframework.boot.autoconfigure.EnableAutoConfiguration;
6 import org.springframework.context.annotation.Bean;
7 import org.springframework.context.annotation.ComponentScan;
8 import org.springframework.context.annotation.Configuration;
9 import org.springframework.web.servlet.LocaleResolver;
10 import org.springframework.web.servlet.config.annotation.InterceptorRegistry;
11 import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;
12 import org.springframework.web.servlet.i18n.LocaleChangeInterceptor;
13 import org.springframework.web.servlet.i18n.SessionLocaleResolver;
14
15 @Configuration
16 @EnableAutoConfiguration
17 @ComponentScan
18 public class LocaleConfig extends WebMvcConfigurerAdapter {
19     // 默认语言
20     // 目录
21     public LocaleResolver localeResolver() {
22         SessionLocaleResolver slr = new SessionLocaleResolver();
23         slr.setDefaultLocale(Locale.US);
24     }
25
26     @Bean
27     public LocaleChangeInterceptor localeChangeInterceptor() {
28         LocaleChangeInterceptor lci = new LocaleChangeInterceptor();
29         lci.setParamName("lang");
30     }
31
32     @Override
33     public void addInterceptors(InterceptorRegistry registry) {
34         registry.addInterceptor(localeChangeInterceptor());
35     }
36 }

```

④、MessageManager 读取国际化文件内容

```

1 package com.gt;
2
3 import java.util.Locale;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.context.MessageSource;
7 import org.springframework.context.i18n.LocaleContextHolder;
8 import org.springframework.stereotype.Component;
9
10 @Component
11 public class MessageManager {
12
13     private static MessageSource messageSource;
14
15     public static String getMsg(String key) {
16         Locale locale = LocaleContextHolder.getLocale();
17         return messageSource.getMessage(key, null, locale);
18     }
19
20     public static String getMsg(String key, String... arg) {
21         Locale locale = LocaleContextHolder.getLocale();
22         Object[] args = new Object[arg.length];
23         for (int i = 0; i < arg.length; i++) {
24             args[i] = arg[i];
25         }
26         return messageSource.getMessage(key, args, locale);
27     }
28 }

```

```

27     } 28
29     @Autowired(required = true)
30     public void setMessageSource(MessageSource messageSource) {
31         MessageManager.messageSource = messageSource;
32     }
33 }

```

⑤、application.yml

```

1 debug: true
2 server:
3   port: 0

```

写评论

⑥、bootstrap.yml docker环境下部署时需要指定ip，否则找不到配置服务，读取不了配置中心的相关配置

```

1 spring:
2   appli    1:
3   nam      收藏 human
4   cloud.
5   con
6   e 微信 l: true
7   d*covery: #配置服务发现，获取配置信息 配置文件命名要按照springcloud config配置文件命名规则命名
8   led: true
9   微博 lce-id: config
10 eureka:
11   insta
12   app      QQ popuman
13   client:
14     service-url:
15       defaultZone: http://${eureka.host:localhost}:${eureka.port:8761}/eureka/
16
17 #docker环境下需指定ip才能访问
18 #eureka:
19 #   instance:
20 #     appname: popuman
21 #     prefer-ip-address: true #启用IP方式
22 #     ip-address: 192.168.*.*
23 #   client:
24 #     service-url:
25 #       defaultZone: http://192.168.*.*:8761/eureka/

```

⑦、logback.xml 日志分级输出到文件，debug，error级别日志输出到各自的日志文件

```

1 <configuration>
2   <!-- %m输出的信息,%p日志级别,%t线程名,%d日期,%c类的全名,,, -->
3   <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
4     <encoder>
5       <pattern>%d %p (%file:%line\)- %m%n</pattern>
6       <charset>UTF-8</charset>
7     </encoder>
8   </appender>
9   <appender name="popuman"
10     class="ch.qos.logback.core.rolling.RollingFileAppender">
11     <File>/Users/david/Documents/Poputar/logs/popuman.log</File>
12     <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
13       <fileNamePattern>/Users/david/Documents/Poputar/logs/popuman.%d.%i</fileNamePattern>
14       <timeBasedFileNamingAndTriggeringPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP">
15         <!-- or whenever the file size reaches 64 MB -->
16         <maxFileSize>64 MB</maxFileSize>
17       </timeBasedFileNamingAndTriggeringPolicy>
18     </rollingPolicy>
19     <encoder>
20       <pattern>

```

```

21 |         %d %p (%file:%line\)- %m%n
22 |     </pattern>
23 |     <charset>UTF-8</charset> <!-- 此处设置字符集 -->
24 | </encoder>
25 |     <filter class="ch.qos.logback.classic.filter.LevelFilter"> <!-- 过滤错误日志 -->
26 |         <level>ERROR</level>
27 |         <onMatch>DENY</onMatch>
28 |         <onMismatch>ACCEPT</onMismatch>
29 |     </filter>
30 | </appender>
31 | <appender name="popuman_err"
32 |     class="ch.qos.logback.core.rolling.RollingFileAppender">
33 |     <File>/Users/david/Documents/Poputar/logs/popuman_err.log</File>
34 |     <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
35 |         <fileNamePattern>/Users/david/Documents/Poputar/logs/popuman_err.%d.%i</fileNamePattern>
36 |         <timeBasedFileNamingAndTriggeringPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP">
37 |             <!-- or whenever the file size reaches 64 MB -->
38 |             <maxFileSize>64 MB</maxFileSize>
39 |         </timeBasedFileNamingAndTriggeringPolicy>
40 |     </rollingPolicy>
41 |     <encoder>
42 |         <pattern>
43 |             %d %p (%file:%line\)- %m%n
44 |         </pattern>
45 |     </encoder>
46 |     <filter class="ch.qos.logback.classic.filter.LevelFilter"> <!-- 只打印错误日志 -->
47 |         <level>ERROR</level>
48 |         <onMatch>ACCEPT</onMatch>
49 |         <onMismatch>DENY</onMismatch>
50 |     </filter>
51 | </appender>
52 | </configuration>
53 | <root>
54 |     <appender-ref ref="STDOUT" />
55 | </root>
56 | <!-- 输出日志 -->
57 | <logger name="com.gt" level="DEBUG">
58 |     <appender-ref ref="popuman" />
59 |     <appender-ref ref="popuman_err" />
60 | </logger>
61 | </configuration>

```

五、popumusic项目代码就不贴了，同popuman项目类似。

部署到docker时，切记端口映射好，否则调不通。

六、应用之间服务的调用是通过springcloud的FeignClient调用，这种调用方式同样也是基于http协议，好处是不用我们再去封装httpClient手写post, get请求，通过调用方法的方式就可以调用其他服务接口

本架构采用springboot推荐的JPA方式来处理数据层，缓存采用redis，如果您要问，redis挂掉怎么办？那就要考虑redis的分布式，主从等，这里不做赘述。

springcloud是近两年新兴的微服务技术，目前我也是在学习，如有觉得我写的有不对的地方，还请批评指正，共同交流，学习一门新技术是枯燥的，难免走很多弯路，但是当你突破难关时，那样的轻松是何等畅快！在这里也感谢CSDN上大牛的技术文章分享，有分享才会有进步。希望本文对学习springcloud的同学有所帮助。

推荐几篇springcloud总结比较全的博客，也是本文项目搭建过程借鉴的技术文章

方志鹏大牛博客地址：<http://blog.csdn.net/forezp/article/category/6830968/1>

司青博客：<http://blog.csdn.net/neosmith/article/details/52449921>

<http://blog.csdn.net/f1576813783/article/details/76805195>

方志鹏<http://blog.csdn.net/forezp/article/category/68>

0

写评论

目录

收藏

文章标签：

springcloud

java

springboot

架构

开发框架

个人分类：

springcloud

微信

springcloud

java

相关热词：

微服务架构

微服务与

it微服务

微服务微服务发现

微服务调用微服务

上一篇springcloud异常： java.lang.NoClassDefFoundError: antlr/Recogniti...

QQ

2018年人工智能工程师平均年薪是多少？

机器学习|深度学习|图像处理|自然语言处理|无人驾驶，这些技术都会吗？看看真正的人工智能师都会那些关键技术？年薪比你高多少！

想对作者说点什么？

我来说两句

Spring Cloud 快速入门esclipse快速搭建微服务框架 （一）-注册与发现

1845

Spring-Cloud项目基本搭建 （一）随着近几年微服务架构理念的流行，越来越多微服务架构也进入人们的视野，目前大部分公司用的...

SpringCloud微服务架构搭建 （三）： 服务调用

2918

spring-cloud调用服务有两种方式，一种是Ribbon+RestTemplate, 另外一种Feign。 Ribbon是一个基于HTTP和TCP客户端的负载...

SpringCloud微服务架构之服务的调用 - CSDN博客

7-26

微服务架构中,业务都会被拆分成一个独立的服务,服务与服务的通讯是基于http restful的。Spring cloud有两种服务调用方式,一种是ribb...

一、基于SpringCloud的微服务架构 - CSDN博客

7-27

一、基于Springcloud的微服务架构 1、微服务架构 1.1 什么是微服务? 简单的说,微服务是系统架构上的一种...

云计算架构图

百度广告



基于 Spring Cloud 完整的微服务架构实战

2664

本项目是一个基于 Spring Boot、Spring Cloud、Spring Oauth2 和 Spring Cloud Netflix 等框架构建的微服务项目。@作者：Sheldo...

微服务架构与实践+SpringCloud微服务实战

7-24

包含了微服务实践架构与springcloud实战的两本好书,希望能够帮助大家学习关于微服务与springcloud相关知识

对于程序员来说，英语到底多重要

不背单词和语法，一个公式学好英语



SpringCloud微服务架构图

转自：黑马程序员springCloud微服务架构培训视频

1121

从 Spring Cloud 开始，聊聊微服务架构的实践之路

这是今天读的一篇文章，感觉挺好，可以对微服务有一个宏观的认识。 from:http://www.jianshu.com/p/45f35e05c350?ref=myread ...

3800

微服务架构与SpringCloud微服务实战

2018年03月26日 9.92MB 下载



学习spring cloud 第一课（微服务架构概述）

现在微服务这个概念越来越火了，公司最近也想使用微服务的技术，因此我就把我学习的东西记录下来，以备以后查询。既然要学习...

3569

深入理解Spring Cloud与微服务构建

2018年05月29日 9.75MB 下载



呼叫中心系统

专业的呼叫中心系统



利用SpringCloud搭建一个最简单的微服务框架

利用SpringCloud搭建一个最简单的微服务框架

5万

【微服务】之三：从零开始，轻松搞定SpringCloud微服务-配置中心

官方解释 Spring Cloud provides tools for developers to quickly build some of the common patterns in d...

623

SpringCloud微服务架构搭建（一）：注册与发现

首先spring-cloud相关的简介可以去百度搜索，这里就不多说了，这里分享一个翻译spring cloud官网的中文网站spring cloud中文网 ...

1072

java spring微服务springcloud实战项目

2018年06月09日 64B 下载



Spring Cloud构建微服务架构（五）服务网关

通过之前几篇Spring Cloud中几个核心组件的介绍，我们已经可以构建一个简略的（不够完善）微服务架构了。比如下图所示： alt ...

6995

50万码农评论：英语对于程序员有多重要！

不背单词和语法，老司机教你一个数学公式秒懂天下英语



一统江湖微服务架构之SpringCloud

2018年01月29日 1.34MB 下载



SpringBoot运行原理

SpringBoot提供了基于条件来配置Bean的能力，SpringBoot关于自动配置的源码在spring-boot-autoconfigure-1.4.1.RELEASE.jar内...

1932

使用Spring Cloud Netflix技术栈实施微服务架构

2.8万

前言系统一旦走向分布式，其复杂程度成倍增长，传统单体应用只考虑业务逻辑的开发方式已经不再适用。正因其复杂性，目前只有...

分布式-微服务-集群的区别

3556

1.分布式 将一个大的系统划分为多个业务模块，业务模块分别部署到不同的机器上，各个业务模块之间通过接口进行数据交互。区...

SpringBoot系列—Spring Cloud快速入门

1224

为了解决单块式架构可用性低，可伸缩性差，集中发布的生命周期以及违反单一功能原则，微服务（Microservice）应运而生了，将...

云计算架构

云计算的体系结构

写评论



Spring Cloud微服务分布式云架构 - spring cloud集成项目

738

Spring Cloud集成项目有很多，下面我们列举一下和Spring Cloud相关的优秀项目...

收藏

Spring Cloud微服务分布式云架构 - Spring Cloud简介

385

Spring Cloud是一个微服务架构的有序集合。利用Spring Boot的开发模式简化了分布式系统基础设施的开发，如服务发现、注册、配置...

微信

SpringCloud微服务架构搭建（四）：断路器

832

在Spring Cloud中，我们使用Hystrix 来实现断路器的功能。Hystrix是Netflix开源的微服务框架套件之一，该框架目标在于通过控制那些访...

QQ

何为分布式、微服务和集群！

394

一、分布式 小马正在经营一个在线购物网站，名叫TT猫，有商品管理、订单管理、用户管理、支付管理、购物车等模块，每个模块...

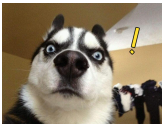
Spring基础：快速入门spring cloud（1）：Spring Cloud介绍

1.6万

分布式系统, 微服务, Java, 当这三个词放到一起的时候, 很多人自然而然地就会想起Spring Cloud. Spring Cloud是Spring总多的Proje...

程序猿不会英语怎么行？英语文档都看不懂！

不背单词和语法，一个公式教你读懂天下英文→



Spring Cloud构建微服务架构（二）服务消费者

4345

在上一篇《Spring Cloud构建微服务架构（一）服务注册与发现》中，我们已经成功创建了“服务注册中心”，实现并注册了一个“服务...

《Spring Cloud构建微服务架构》系列博

4128

http://blog.didispace.com/categories/Spring-Cloud/ 《Spring Cloud构建微服务架构》系列博文示例 chapter1-1-1: Sprin...

基于Spring Boot和Spring Cloud实现微服务架构学习

2.3万

看了几周Spring相关框架的书籍和官方demo，是时候开始总结下这中间的学习感悟。首先，最想说的是，当你要学习一套最新的技...

使用Springcloud及Docker实现微服务架构

714

SpringCloudSpring Cloud是在Spring Boot的基础上构建的，用于简化分布式系统构建的工具集，为开发人员提供快速建立分布式系...

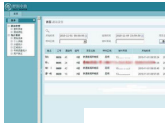
Java微服务框架一览

3475

原文：Java Microservices: Code Examples, Tutorials, and More 作者：Angela Stringfellow 翻译：雁惊寒 译者注...

呼叫中心系统

专业的呼叫中心系统



微服务架构(一): 什么是微服务

解析微服务架构系列文章将分几篇描述微服务的定义、特点、应用场景、企业集成架构的演进以及微服务转型思路和技术决策考虑等...

5.2万

一个简单的微服务框架 (RPC)

一个简单的微服务框架 (RPC) 参考书籍: 微服务分布式构架开发实战 龚鹏 RPC: Remote Procedure Call —远程过程调用。简单...

575

微服务框架对比...

功能点/服务框架 /SpringCloud Motan gRPC Thrift Dubbo/DubboX 功能定位 完整的微服务框架 RPC框架, 但整合了ZK或Con...

53

微服务架构与框架

目录 ...

3009

微服务框架--一台框架

微服务框架 微服务 : 一种架构风格, 将单体应用划分成一组小的服务, 服务之间相互协作, 实现业务功能; 每个服务运行在独...

701

程序猿不会英语怎么办? 英语文档都看不懂!

不背单词和语法 公式教你读懂天下英文→



QQ

微服务架构的核心要点和实现原理

摘要: 本文中, 我们将进一步理解微服务架构的核心要点和实现原理, 为读者的实践提供微服务的设计模式, 以期让微服务在读者正...

4882

为什么说搞定微服务架构, 先搞定RPC框架?

原文地址: http://mp.weixin.qq.com/s/jUgmW3ofIsTwyX-6kNZCfw 今天开始聊一些微服务的实践, 第一块, RPC框架的原理及实践...

3285

spring io 在线生成微服务项目

http://start.spring.io/

155

介绍一个Spring Cloud分布式微服务架构图

分布式、微服务、云架构 JAVA语言开发、跨平台、高性能、高可用、安全、服务化、模块化、组件化、驱动式开发模式 从现在开始...

541

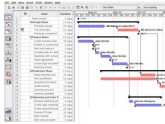
微服务架构实施原理

1 题记基于微服务架构和Docker容器技术的PaaS云平台建设目标是给我们的开发人员提供一套服务快速开发、部署、运维管理、持...

162

项目管理软件

project项目管理工具软件



利用Java上手微服务架构

几乎每个人都在关注微服务架构, 我们也不例外。作为一个与时俱进的程序员, 我一直在努力了解这一架构, 希望寻找一种通过Spr...

2562

一篇文章带你快速理解微服务架构, 由浅入深带你走进微服务架构的核心

什么是微服务关注作者的微信公众号: "Java架构师学习"一个只分享Java架构干货的公众号首先微服务并没有一个官方的定义, 想要...

4185

没有更多推荐了, 返回首页

0

写评论

目录

收藏

微信

微博

QQ

个人资料



南山无情

关注

原创20

粉丝7

喜欢0

评论1

等级：

博客

3

访问：3万+

积分：540

排名：10万+

反制无人机



最新文章

springboot异常： java.lang.NoClassDefFou
ndError: antlr/RecognitionException解决

tomcat启动报SEVERE: Unable to process
Jar entry [javassist/util/proxy/SerializedPro
xy.class] from

Description Resource Path Location Type
Cannot change version of project facet Dy
namic Web Module to

Nginx动静分离经典案例配置

jquery如何获取一个select里面的所有option
的值

个人分类

tomcat3篇

java-eclipse3篇

java5篇

springboot2篇

springcloud1篇

归档

2017年11月1篇

2017年10月1篇

2016年5月1篇

2016年4月1篇

2016年3月2篇

展开

热门文章

tomcat启动报SEVERE: Unable to process
Jar entry [javassist/util/proxy/SerializedPro

阅读量：9954

【微服务架构】springcloud微服务架构搭建

阅读量：4625

0

写评论

目录

收藏

微信

微博

QQ

- Description Resource Path Location Type

Cannot change version of project facet Dy

阅读量：4046
- qq好友列表窗口 java实现

阅读量：1494
- java项目名那里出现了红色感叹号?怎么去除?

阅读量：1385

最新评论

struts+spring+myb...
linfengisme： nice

自考和成考的区别



联系我们




请扫描二维码联系客服

 webmaster@csdn.net

 400-660-0108

 QQ客服  客服论坛

关于 招聘 广告服务 网站地图
©2018 CSDN版权所有 京ICP证09002463号
 百度提供支持

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心