springBoot集成Dubbo最新版本（spring-boot 2.0.3 + dubbo-spring-boot-starter 2.0.0）

francis\_zl 2018-09-17 19:39:28 13716 收藏 11

版权

前言：

自dubbo 更新出了springboot版本后，各项配置都简便了许多，集成起来也很迅速。

下面我们就来讲讲怎么用 spring-boot 2.0.3集成dubbo-spring-boot-starter 2.0.0(Git地址)

用zookeeper做注册中心来注册dubbo服务（zookeeper自行下载安装，这里不做介绍）

POM依赖：

首先创建一个maven多模块项目，在父项目加入springboot的各项依赖（这里只给出父依赖其他的省略，个人根据项目需要添加）

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.0.3.RELEASE</version>

</parent>

然后加入dubbo 和zookeeper依赖

<!--dubbo 依赖 -->

<dependency>

<groupId>com.alibaba.spring.boot</groupId>

<artifactId>dubbo-spring-boot-starter</artifactId>

<version>2.0.0</version>

</dependency>

<!-- zookeeper client依赖 -->

<dependency>

<groupId>com.101tec</groupId>

<artifactId>zkclient</artifactId>

<version>0.10</version>

</dependency>

提供者-provider

接着创建子工程项目dubbo-provider（提供者），创建application.yml，在里面加入dubbo 配置

server:

port: 8080

spring:

dubbo:

application:

name: provider

server: true

registry:

address: zookeeper://127.0.0.1:2181

protocol:

name: dubbo

port: 20880

创建springboot启动类，注意使用dubbo-spring-boot-starter版本的必须在启动类上加@EnableDubboConfiguration注解来启用dubbo

package com.dubbo.consumer;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import com.alibaba.dubbo.spring.boot.annotation.EnableDubboConfiguration;

@SpringBootApplication

@EnableDubboConfiguration

public class Application {

public static void main(String[] args) {

// 程序启动入口

// 启动嵌入式的 TomCat 并初始化 Spring 环境及其各 Spring 组件

SpringApplication.run(Application.class, args);

}

}

创建service

public interface ProviderService {

String sayHello();

}

创建实现类，这个时候实现类上面需要用到dubbo 的@Service 注解。

其他版本的dubbo 暴露服务接口的方式可以通过<dubbo:service/>或者spring.dubbo.scan，但是dubbo-spring-boot-starter版本的只需要在@Service注解里加上interfaceClass 属性来指定当前要暴露的服务。如下：

import org.springframework.stereotype.Component;

import com.alibaba.dubbo.config.annotation.Service;

import com.dubbo.provider.business.service.ProviderService;

@Service(interfaceClass = ProviderService.class)

@Component

public class ProviderServiceImpl implements ProviderService{

@Override

public String sayHello() {

return "hello!!!";

}

}

消费者-consumer

然后创建消费者项目dubbo-consumer(启动类同provider)：

yml内容如下：

server:

port: 8081

spring:

dubbo:

application: #应用配置，用于配置当前应用信息，不管该应用是提供者还是消费者。

name: consumer

registry: #注册中心配置，用于配置连接注册中心相关信息。

address: zookeeper://192.168.1.115:2181

protocol: #协议配置，用于配置提供服务的协议信息，协议由提供方指定，消费方被动接受（订阅）。

name: dubbo

port: 20880

consumer:

check: false

reference:

loadbalance: roundrobin #轮询机制

#loadbalance: random #随机机制

#loadbalance: leastactive #最少活跃调用数机制

其中reference:

        loadbalance: roundrobin #轮询机制

        #loadbalance: random #随机机制

        #loadbalance: leastactive #最少活跃调用数机制

是指的dubbo的负载均衡策略，有兴趣的可以去了解了，用多台机器或者改dubbo注册服务的端口号，一般在实际项目我们配置权重或负载均衡时不在代码中写死，动态使用默认配置，需要调节时通过dubbo admin管控台上进行配置。

然后创建启动类 跟provider的Application一样

在pom 文件里依赖下提供者项目、

<dependency>

<groupId>com.boot.dubbo</groupId>

<artifactId>dubbo-boot-provider</artifactId>

<version>0.0.1-SNAPSHOT</version>

</dependency>

接着创建一个controller

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.alibaba.dubbo.config.annotation.Reference;

import com.dubbo.provider.business.service.ProviderService;

@RestController

@RequestMapping("/hello/")

public class ConsumerController {

@Reference

private ProviderService providerService;

@GetMapping

public String sayHello() {

return providerService.sayHello();

}

}

我们可以看到这里的提供者Service 不再是用@Autowired去装载了，而是用dubbo的@Reference注解去连接提供者提供服务了。

当然有可能业务需求，需要我们指定提供者服务版本，我们可以在@Service加上属性@Service(version="1.0.0")，@Reference去链接的时候也同样指定@Reference（version="1.0.0"）。

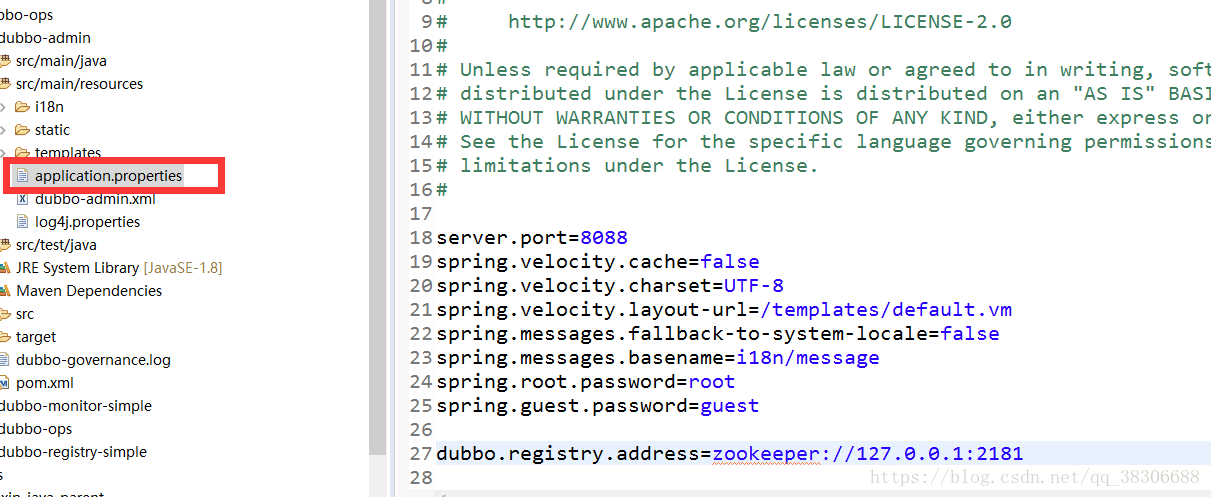
Dubbo-admin

一位博主保存的的老版springboot版的git 地址 https://github.com/light0x000/incubator-dubbo-ops

新版页面改成了vue 版

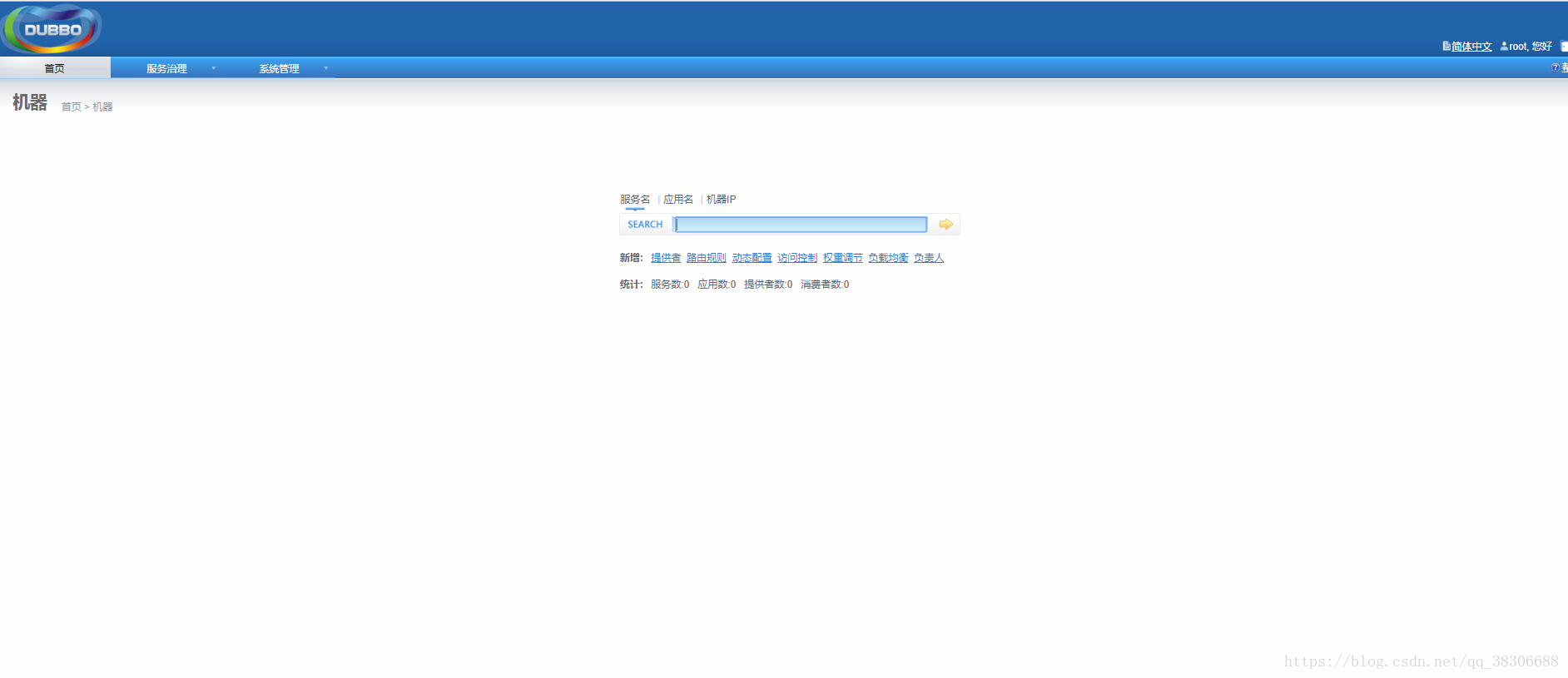
把dubbo-admin项目下载下来

根据自己要求修改其application.properties（port 或者zookeeper连接地址）



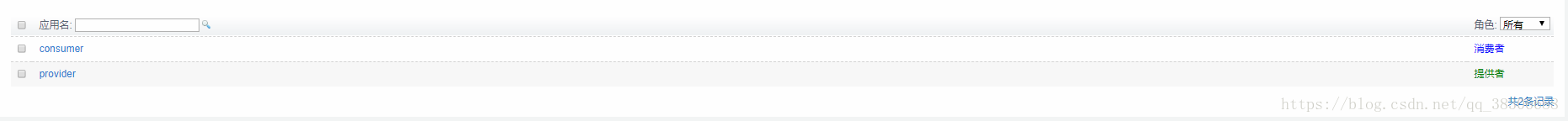
 最后打成jar 或者war包运行，

浏览器访问就可以看到管理台页面了，用root root 或者 guest guest登录，



接着依次启动提供者项目和消费者项目

 就可以在admin 上面看到注册的两个服务了



外传：

       我曾经遇到过一个诡异的问题，在一个正在开发的项目基础上加入dubbo ，模块比较多，导致我消费者直接依赖提供者后出现了一些问题，先启动提供者，这时候提供者注册服务已经开启了20880端口，当我再启动消费者的时候发现端口占用，仔细一检查发现消费者并没有订阅20880端口的服务而是在往里面注册， 很百思不得其解，可以是pom依赖的模块问题，导致两个项目结构网混乱，于是我把service 的interface抽出来单独做一个子模块项目，这样消费者和提供者只需要依赖这个子模块，从而避免了消费者和提供者项目的直接关联导致服务注册出现冲突。（遇到过类似问题的 可以尝试用次方法解决）

----我是francis， 谨以此记录自己精彩的程序人生。

————————————————

版权声明：本文为CSDN博主「francis\_zl」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/qq\_38306688/article/details/82707503