# http://blog.csdn.net/hejingyuan6/article/details/47403299

# [【Dubbo实战】 Dubbo+Zookeeper+Spring整合应用篇-Dubbo基于Zookeeper实现分布式服务（二）](http://blog.csdn.net/hejingyuan6/article/details/47403299)

# Dubbo与Zookeeper、Spring整合使用

Dubbo采用全**[spring](http://lib.csdn.net/base/javaee" \o "Java EE知识库" \t "http://blog.csdn.net/hejingyuan6/article/details/_blank)**配置方式，透明化接入应用，对应用没有任何API侵入，只需用Spring加载Dubbo的配置即可，Dubbo基于Spring的Schema扩展进行加载。

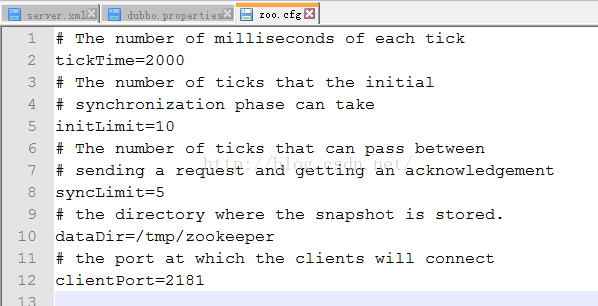
## 一：单机模式安装zookeeper

1,下载zookeeper注册中心，下载地址：[http://www.apache.org/dyn/closer.cgi/zookeeper/](http://www.apache.org/dyn/closer.cgi/zookeeper/" \t "http://blog.csdn.net/hejingyuan6/article/details/_blank) 下载后解压即可，进入E:\zookeeper-3.3.6\zookeeper-3.3.6\bin，

双击zkServer.cmd启动注册中心服务。

zkServer.sh【**[Linux](http://lib.csdn.net/base/linux" \o "Linux知识库" \t "http://blog.csdn.net/hejingyuan6/article/details/_blank)**】或zkServer.cmd【Windows】

2,在你执行启动脚本之前，还有几个基本的配置项需要配置一下，Zookeeper的配置文件在 conf 目录下，这个目录下有 zoo\_sample.cfg 和 log4j.properties，你需要做的就是将zoo\_sample.cfg 改名为 zoo.cfg，因为 Zookeeper在启动时会找这个文件作为默认配置文件。下面详细介绍一下，这个配置文件中各个配置项的意义。



   •tickTime：这个时间是作为Zookeeper 服务器之间或客户端与服务器之间维持心跳的时间间隔，也就是每个 tickTime 时间就会发送一个心跳。

   •dataDir：顾名思义就是 Zookeeper保存数据的目录，默认情况下，Zookeeper 将写数据的日志文件也保存在这个目录里。

   •dataLogDir：顾名思义就是Zookeeper 保存日志文件的目录

   •clientPort：这个端口就是客户端连接Zookeeper 服务器的端口，Zookeeper 会监听这个端口，接受客户端的访问请求

配置好后，zookeeper会监听本机的2181端口。

当这些配置项配置好后，你现在就可以启动 Zookeeper 了，启动后要检查 Zookeeper 是否已经在服务，可以通过 netstat – ano 命令查看是否有你配置的 clientPort 端口号在监听服务。

## 二：服务提供者

定义服务接口:(该接口需单独打包，在服务提供方和消费方共享)

**[java]** [view plain](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "view plain) [copy](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "copy) [print](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "print)[?](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "?)

1. **package** com.unj.dubbotest.provider;
3. **import** java.util.List;
5. **public** **interface** DemoService {
7. String sayHello(String name);
9. **public** List getUsers();
11. }

在服务提供方实现接口：(对服务消费方隐藏实现)

**[java]** [view plain](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "view plain) [copy](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "copy) [print](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "print)[?](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "?)

1. **package** com.unj.dubbotest.provider.impl;
3. **import** java.util.ArrayList;
4. **import** java.util.List;
6. **import** com.unj.dubbotest.provider.DemoService;
8. **public** **class** DemoServiceImpl **implements** DemoService {
10. **public** String sayHello(String name) {
11. **return** "Hello " + name;
12. }
14. **public** List getUsers() {
15. List list = **new** ArrayList();
16. User u1 = **new** User();
17. u1.setName("hejingyuan");
18. u1.setAge(20);
19. u1.setSex("f");
21. User u2 = **new** User();
22. u2.setName("xvshu");
23. u2.setAge(21);
24. u2.setSex("m");

27. list.add(u1);
28. list.add(u2);
30. **return** list;
31. }
32. }

用Spring配置声明暴露服务：

**[html]** [view plain](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "view plain) [copy](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "copy) [print](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "print)[?](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "?)

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<beans** xmlns="http://www.springframework.org/schema/beans"
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"
4. xsi:schemaLocation="http://www.springframework.org/schema/beans
5. http://www.springframework.org/schema/beans/spring-beans.xsd
6. http://code.alibabatech.com/schema/dubbo
7. http://code.alibabatech.com/schema/dubbo/dubbo.xsd
8. "**>**
10. <!-- 具体的实现bean -->
11. **<bean** id="demoService" class="com.unj.dubbotest.provider.impl.DemoServiceImpl" **/>**
13. <!-- 提供方应用信息，用于计算依赖关系 -->
14. **<dubbo:application** name="xs\_provider" **/>**
16. <!-- 使用multicast广播注册中心暴露服务地址 -->
17. <!--<dubbo:registry address="multicast://224.5.6.7:1234" /> -->
19. <!-- 使用zookeeper注册中心暴露服务地址 --即zookeeper的所在服务器ip地址和端口号 -->
20. **<dubbo:registry** address="zookeeper://192.168.24.213:2181" **/>**
22. <!-- 用dubbo协议在20880端口暴露服务 -->
23. **<dubbo:protocol** name="dubbo" port="20880" **/>**
25. <!-- 声明需要暴露的服务接口 -->
26. **<dubbo:service** interface="com.unj.dubbotest.provider.DemoService"
27. ref="demoService" **/>**
29. **</beans>**

加载Spring配置，启动服务（或者将项目建为web项目，然后在web.xml中配置好spring的启动，然后扔到tomcat中即可提供服务）：

**[java]** [view plain](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "view plain) [copy](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "copy) [print](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "print)[?](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "?)

1. **package** com.unj.dubbotest.provider.impl;
3. **import** org.springframework.context.support.ClassPathXmlApplicationContext;
5. **public** **class** Provider {
7. **public** **static** **void** main(String[] args) **throws** Exception {
8. ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext(
9. **new** String[] { "applicationContext.xml" });
10. context.start();
11. System.in.read(); // 为保证服务一直开着，利用输入流的阻塞来模拟
12. }
13. }

## 三：服务消费者

通过Spring配置引用远程服务：

**[html]** [view plain](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "view plain) [copy](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "copy) [print](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "print)[?](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "?)

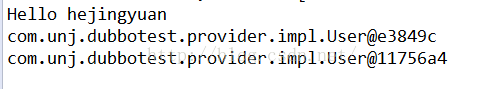
1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<beans** xmlns="http://www.springframework.org/schema/beans"
3. xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:dubbo="http://code.alibabatech.com/schema/dubbo"
4. xsi:schemaLocation="http://www.springframework.org/schema/beans
5. http://www.springframework.org/schema/beans/spring-beans.xsd
6. http://code.alibabatech.com/schema/dubbo
7. http://code.alibabatech.com/schema/dubbo/dubbo.xsd
8. "**>**
10. <!-- 消费方应用名，用于计算依赖关系，不是匹配条件，不要与提供方一样 -->
11. **<dubbo:application** name="hjy\_consumer" **/>**
13. <!-- 使用zookeeper注册中心暴露服务地址 -->
14. <!-- <dubbo:registry address="multicast://224.5.6.7:1234" /> -->
15. **<dubbo:registry** address="zookeeper://192.168.24.213:2181" **/>**
17. <!-- 生成远程服务代理，可以像使用本地bean一样使用demoService -->
18. **<dubbo:reference** id="demoService"
19. interface="com.unj.dubbotest.provider.DemoService" **/>**
21. **</beans>**

调用服务**[测试](http://lib.csdn.net/base/softwaretest" \o "软件测试知识库" \t "http://blog.csdn.net/hejingyuan6/article/details/_blank)**：

**[java]** [view plain](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "view plain) [copy](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "copy) [print](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "print)[?](http://blog.csdn.net/hejingyuan6/article/details/47403299" \o "?)

1. **package** com.alibaba.dubbo.demo.pp;
3. **import** java.util.List;
5. **import** org.springframework.context.support.ClassPathXmlApplicationContext;
7. **import** com.unj.dubbotest.provider.DemoService;
9. **public** **class** Consumer {
11. **public** **static** **void** main(String[] args) **throws** Exception {
12. ClassPathXmlApplicationContext context = **new** ClassPathXmlApplicationContext(
13. **new** String[] { "applicationContext.xml" });
14. context.start();
16. DemoService demoService = (DemoService) context.getBean("demoService");
17. String hello = demoService.sayHello("hejingyuan");
18. System.out.println(hello);
20. List list = demoService.getUsers();
21. **if** (list != **null** && list.size() > 0) {
22. **for** (**int** i = 0; i < list.size(); i++) {
23. System.out.println(list.get(i));
24. }
25. }
26. System.in.read();
27. }
29. }

测试结果：



# 附：Dubbo的管理页面

需要下载：dubbo-admin-2.5.3的war包

下载地址：[http://download.csdn.net/detail/u013286716/7041185](http://download.csdn.net/detail/u013286716/7041185" \t "http://blog.csdn.net/hejingyuan6/article/details/_blank)

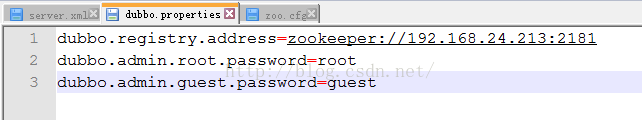
操作如下：

1，替换掉tomcat/webapps下自带的ROOT文件夹内容（即替换tomcat的启动主页），将下载的war包解压到webapps/ROOT中，直接替换即可

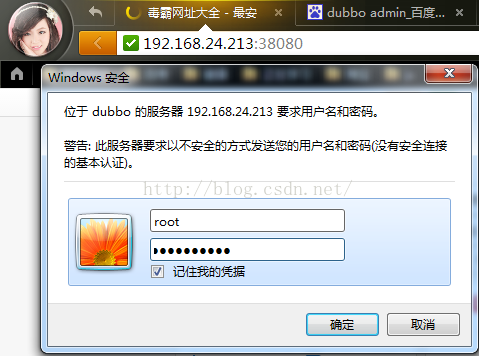
   注意：jdk不要使用1.8，本次实验使用的为1.6

2，启动tomcat，访问ip:8080即可或者如果是本地的话使用localhost:8080

输入用户名密码，在E:\apache-tomcat-7.0.6-dubbo\webapps\ROOT\WEB-INF下的dubbo.properties文件中即可查看到，如：



3，访问 [http://192.168.24.213:38080/](http://192.168.24.213:38080/" \t "http://blog.csdn.net/hejingyuan6/article/details/_blank)



4，启动我们的服务提供者和消费者即可查看到





# 整体说明：

    Zookeeper（注册中心）部署到213的机子上，服务提供者和服务消费者均在215的机子上运行，当然我们也可以分别将服务提供者和服务消费者部署到不同的两台机子上。

优点：

    服务提供者和服务消费者只需要知道注册中心即可，它们之间打交道需通过注册中心这个第三方，只要是注册中心中已经注册的服务，我们均可以使用，实现了服务提供者和服务消费者间的解耦。