Dubbo-demo

创建 dubbo-provider maven web项目

## pom 依赖

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>com.lyj.dubbo</groupId>

<artifactId>dubbo-provider</artifactId>

<version>1.0.0-SNAPSHOT</version>

<packaging>war</packaging>

<dependencies>

<!-- dubbo采用spring配置方式，所以需要导入spring容器依赖 -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>4.1.3.RELEASE</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-log4j12</artifactId>

<version>1.6.4</version>

</dependency>

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>dubbo</artifactId>

<version>2.5.3</version>

<exclusions>

<exclusion>

<!-- 排除传递spring依赖 -->

<artifactId>spring</artifactId>

<groupId>org.springframework</groupId>

</exclusion>

</exclusions>

</dependency>

<!-- zookeeper -->

<dependency>

<groupId>org.apache.zookeeper</groupId>

<artifactId>zookeeper</artifactId>

<version>3.3.3</version>

</dependency>

<dependency>

<groupId>com.github.sgroschupf</groupId>

<artifactId>zkclient</artifactId>

<version>0.1</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.tomcat.maven</groupId>

<artifactId>tomcat7-maven-plugin</artifactId>

<version>2.2</version>

<configuration>

<port>8187</port>

<path>/</path>

</configuration>

</plugin>

</plugins>

</build>

</project>

## Bean

**package** com.lyj.dubbo.bean;

// 使用dubbo要求传输的对象必须实现序列化接口

**public** **class** User **implements** java.io.Serializable {

/\*\*

\* serialVersionUID

\*/

**private** **static** **final** **long** ***serialVersionUID*** = -5330406141916172644L;

**private** Long id;

**private** String username;

**private** String password;

**private** Integer age;

**private** Integer gender;

**public** Integer getGender() {

**return** gender;

}

**public** **void** setGender(Integer gender) {

**this**.gender = gender;

}

**public** Long getId() {

**return** id;

}

**public** **void** setId(Long id) {

**this**.id = id;

}

**public** String getUsername() {

**return** username;

}

**public** **void** setUsername(String username) {

**this**.username = username;

}

**public** String getPassword() {

**return** password;

}

**public** **void** setPassword(String password) {

**this**.password = password;

}

**public** Integer getAge() {

**return** age;

}

**public** **void** setAge(Integer age) {

**this**.age = age;

}

}

## 创建服务接口

**public** **interface** UserService {

/\*\*

\* 查询所有用户

\* **@return**

\*/

**public** List<User> queryAll();

}

## 创建实现类

package com.lyj.dubbo.service.impl;

import java.util.ArrayList;

import java.util.List;

import com.lyj.dubbo.bean.User;

import com.lyj.dubbo.service.UserService;

public class UserServiceImpl implements UserService {

/\*\*

\* 模拟数据库查询

\*/

public List<User> queryAll() {

List<User> list = new ArrayList<User>();

for (int i = 0; i< 10; i++) {

User user = new User();

user.setId(Long.valueOf(i + 1));

user.setUsername("uname\_" + i);

user.setPassword("000000");

user.setAge(10 + i);

user.setGender(i%2);

list.add(user);

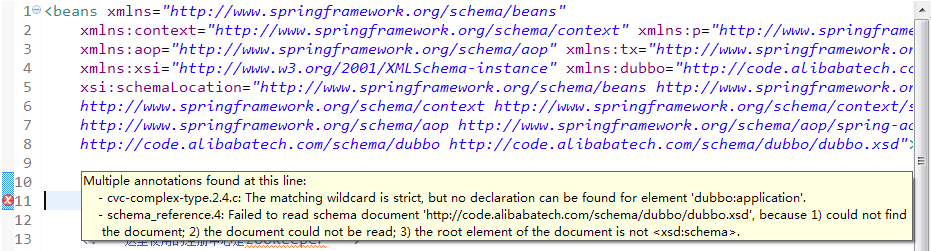
}

return list;

}

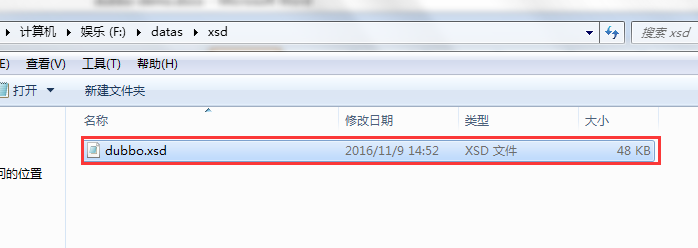
}

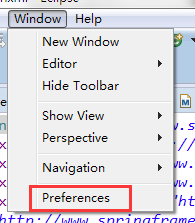
## 编写 Dubbo 配置文件

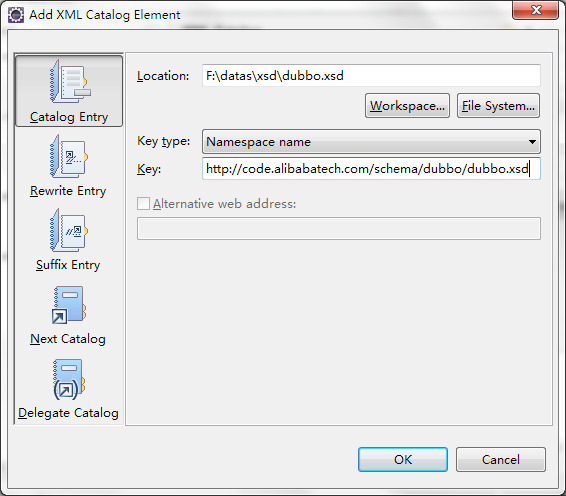


引入约束文件 xsd

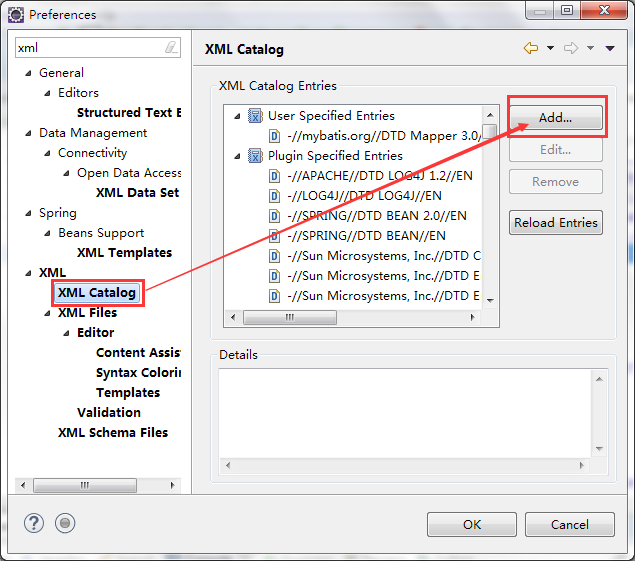
<http://code.alibabatech.com/schema/dubbo/dubbo.xsd>

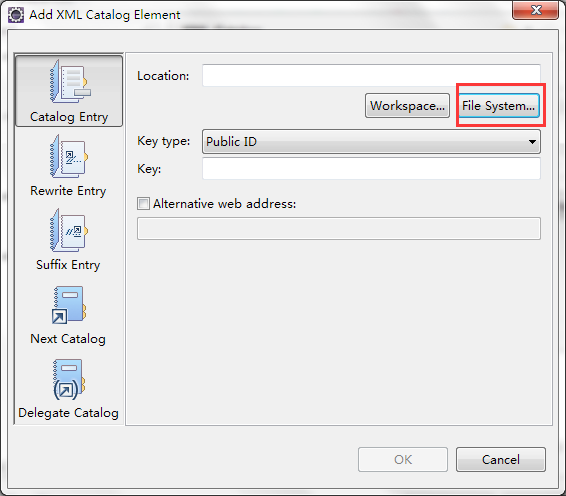






重新启动当前 Editor





选择刚刚的 dubbo.xsd 文件

## 配置 web.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xmlns=*"http://java.sun.com/xml/ns/javaee"*

xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"*

id=*"WebApp\_ID"* version=*"2.5"*>

<display-name>dubbo-provider</display-name>

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:dubbo/dubbo-\*.xml</param-value>

</context-param>

<!--Spring的ApplicationContext 载入 -->

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

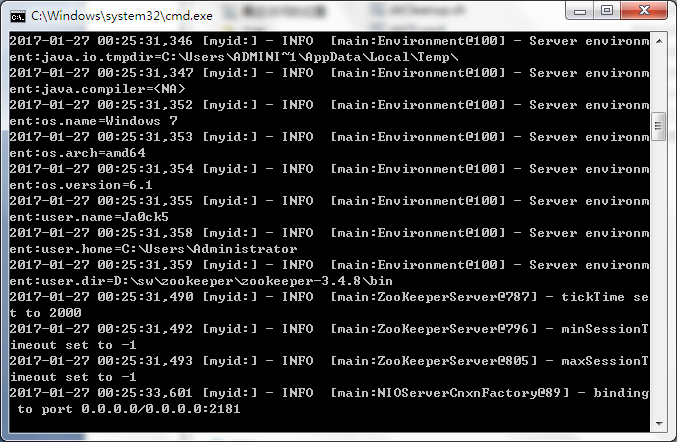
<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

</welcome-file-list>

</web-app>

## 启动 zookeeper



## 启动项目

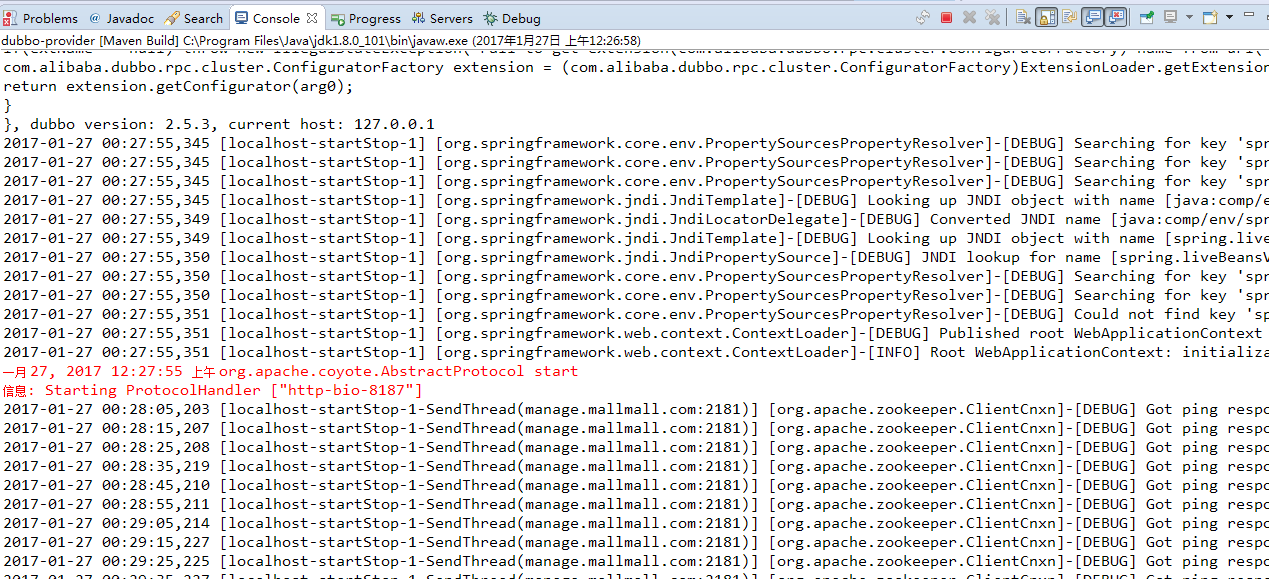
2017-01-27 00:27:53,976 [localhost-startStop-1] [com.alibaba.dubbo.config.AbstractConfig]-[INFO] [DUBBO] Export dubbo service com.lyj.dubbo.service.UserService to local registry, dubbo version: 2.5.3, current host: 127.0.0.1

2017-01-27 00:27:53,977 [localhost-startStop-1] [com.alibaba.dubbo.config.AbstractConfig]-[INFO] [DUBBO] Export dubbo service com.lyj.dubbo.service.UserService to url dubbo://192.168.64.1:20880/com.lyj.dubbo.service.UserService?anyhost=true&application=dubbo-provider-server&dubbo=2.5.3&interface=com.lyj.dubbo.service.UserService&methods=queryAll&pid=4708&side=provider&timestamp=1485448073650, dubbo version: 2.5.3, current host: 127.0.0.1

2017-01-27 00:27:53,977 [localhost-startStop-1] [com.alibaba.dubbo.config.AbstractConfig]-[INFO] [DUBBO] Register dubbo service com.lyj.dubbo.service.UserService url dubbo://192.168.64.1:20880/com.lyj.dubbo.service.UserService?anyhost=true&application=dubbo-provider-server&dubbo=2.5.3&interface=com.lyj.dubbo.service.UserService&methods=queryAll&pid=4708&side=provider&timestamp=1485448073650 to registry registry://127.0.0.1:2181/com.alibaba.dubbo.registry.RegistryService?application=dubbo-provider-server&client=zkclient&dubbo=2.5.3&pid=4708&registry=zookeeper&timestamp=1485448073622, dubbo version: 2.5.3, current host: 127.0.0.1

2017-01-27 00:27:54,011 [localhost-startStop-1] [com.alibaba.dubbo.common.extension.ExtensionLoader]-[DEBUG] [DUBBO] package com.alibaba.dubbo.registry;

## 启动成功



## 消费者

创建消费者 dubbo-consumer

pom

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>com.lyj.dubbo</groupId>

<artifactId>dubbo-consumer</artifactId>

<version>1.0.0-SNAPSHOT</version>

<dependencies>

<!-- 单元测试 -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.10</version>

<scope>test</scope>

</dependency>

<!-- dubbo采用spring配置方式，所以需要导入spring容器依赖 -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>4.1.3.RELEASE</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-log4j12</artifactId>

<version>1.6.4</version>

</dependency>

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>dubbo</artifactId>

<version>2.5.3</version>

<exclusions>

<exclusion>

<!-- 排除传递spring依赖 -->

<artifactId>spring</artifactId>

<groupId>org.springframework</groupId>

</exclusion>

</exclusions>

</dependency>

<dependency>

<groupId>org.apache.zookeeper</groupId>

<artifactId>zookeeper</artifactId>

<version>3.3.3</version>

</dependency>

<dependency>

<groupId>com.github.sgroschupf</groupId>

<artifactId>zkclient</artifactId>

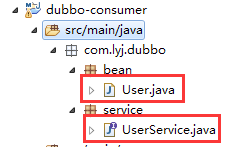
<version>0.1</version>

</dependency>

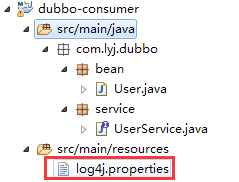
</dependencies>

</project>

## Consumer (Copy)使用Provider 的User 对象和 UserService 接口



引入日志



## 编写dubbo配置文件 dubbo-consumer.xml

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:context=*"http://www.springframework.org/schema/context"* xmlns:p=*"http://www.springframework.org/schema/p"*

xmlns:aop=*"http://www.springframework.org/schema/aop"* xmlns:tx=*"http://www.springframework.org/schema/tx"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:dubbo=*"http://code.alibabatech.com/schema/dubbo"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd*

*http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd*

*http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.0.xsd http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.0.xsd*

*http://code.alibabatech.com/schema/dubbo http://code.alibabatech.com/schema/dubbo/dubbo.xsd"*>

<!-- 提供方应用信息，用于计算依赖关系 -->

<dubbo:application name=*"dubbo-a-consumer"*/>

<!-- 这里使用的注册中心是zookeeper -->

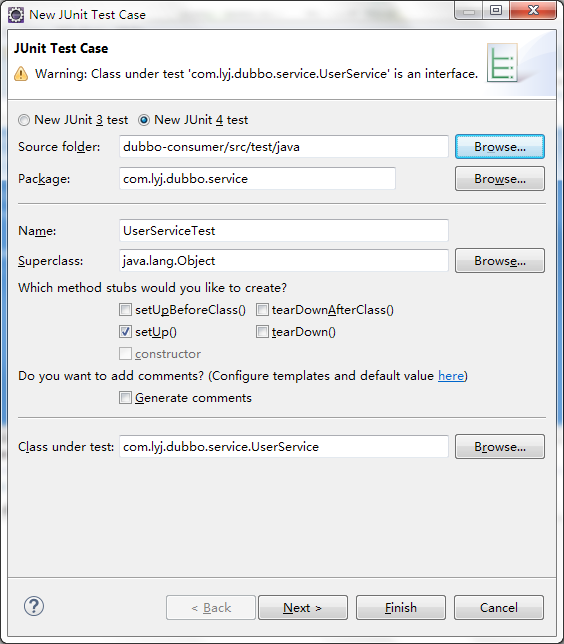
<dubbo:registry address=*"zookeeper://127.0.0.1:2181"* client=*"zkclient"*/>

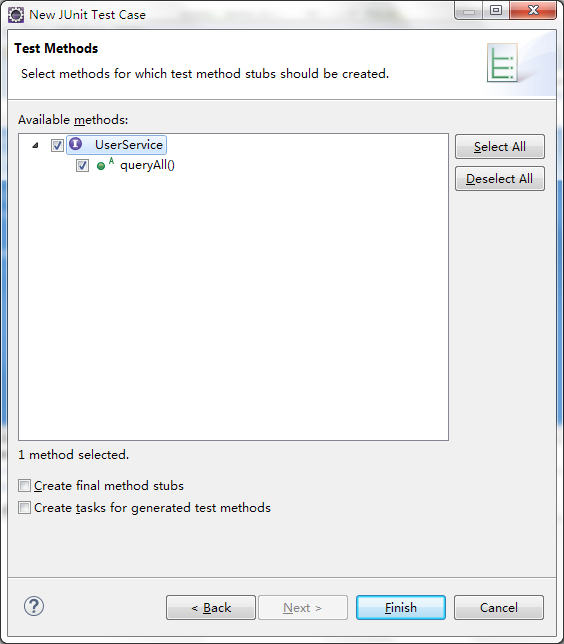
<!-- 从注册中心中查找服务 -->

<dubbo:reference id=*"userService"* interface=*"com.lyj.dubbo.service.UserService"*/>

</beans>

## 测试





测试代码：

**package** com.lyj.dubbo.service;

**import** java.util.List;

**import** org.junit.Before;

**import** org.junit.Test;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**import** com.lyj.dubbo.bean.User;

**public** **class** UserServiceTest {

**private** UserService userService;

@Before

**public** **void** setUp() **throws** Exception {

ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext("classpath:dubbo/dubbo-\*.xml");

**this**.userService = applicationContext.getBean(UserService.**class**);

}

@Test

**public** **void** testQueryAll() {

List<User> list = **this**.userService.queryAll();

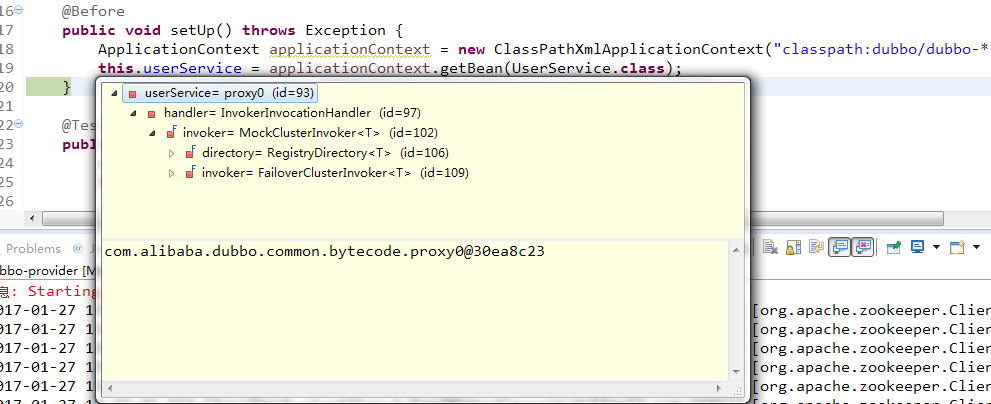
**for** (User user : list) {

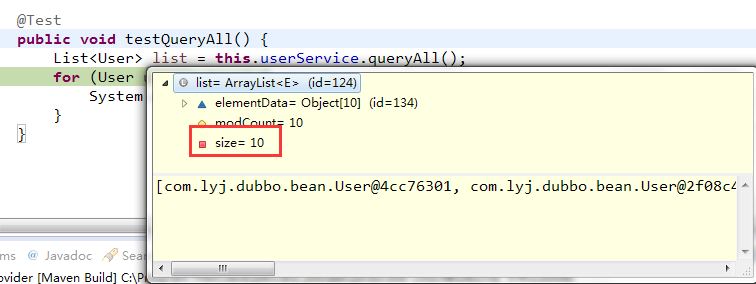
System.***out***.println(user);

}

}

}





远程调用成功。