<http://www.cnblogs.com/yjmyzz/p/spring-mvc-rest-service-with-xml-and-json-format.html>

之前写过一篇 [利用JAX-RS快速开发RESTful 服务](http://www.cnblogs.com/yjmyzz/p/javaee-jax-rs-tutorial.html) 今天来看下spring-mvc框架如何实现类似的功能:

**一、**pom.xml

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

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

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/maven-v4\_0\_0.xsd">

4 <modelVersion>4.0.0</modelVersion>

5 <groupId>infosky</groupId>

6 <artifactId>Spring-MVC-REST</artifactId>

7 <packaging>war</packaging>

8 <version>1.0</version>

9 <properties>

10 <java-version>1.6</java-version>

11 <spring.version>3.2.8.RELEASE</spring.version>

12 <org.aspectj-version>1.7.3</org.aspectj-version>

13 </properties>

14 <dependencies>

15

16 <!-- Spring -->

17 <dependency>

18 <groupId>org.springframework</groupId>

19 <artifactId>spring-core</artifactId>

20 <version>${spring.version}</version>

21 </dependency>

22

23 <dependency>

24 <groupId>org.springframework</groupId>

25 <artifactId>spring-expression</artifactId>

26 <version>${spring.version}</version>

27 </dependency>

28

29 <dependency>

30 <groupId>org.springframework</groupId>

31 <artifactId>spring-beans</artifactId>

32 <version>${spring.version}</version>

33 </dependency>

34

35 <dependency>

36 <groupId>org.springframework</groupId>

37 <artifactId>spring-context</artifactId>

38 <version>${spring.version}</version>

39 </dependency>

40 <dependency>

41 <groupId>org.springframework</groupId>

42 <artifactId>spring-context-support</artifactId>

43 <version>${spring.version}</version>

44 </dependency>

45 <dependency>

46 <groupId>org.springframework</groupId>

47 <artifactId>spring-web</artifactId>

48 <version>${spring.version}</version>

49 </dependency>

50

51 <dependency>

52 <groupId>org.springframework</groupId>

53 <artifactId>spring-webmvc</artifactId>

54 <version>${spring.version}</version>

55 </dependency>

56

57 <dependency>

58 <groupId>org.springframework</groupId>

59 <artifactId>spring-oxm</artifactId>

60 <version>${spring.version}</version>

61 </dependency>

62

63 <!-- json -->

64 <dependency>

65 <groupId>org.codehaus.jackson</groupId>

66 <artifactId>jackson-mapper-asl</artifactId>

67 <version>1.9.3</version>

68 </dependency>

69

70 <dependency>

71 <groupId>org.codehaus.jackson</groupId>

72 <artifactId>jackson-jaxrs</artifactId>

73 <version>1.9.9-redhat-2</version>

74 </dependency>

75

76 <!-- Logging -->

77 <dependency>

78 <groupId>log4j</groupId>

79 <artifactId>log4j</artifactId>

80 <version>1.2.17</version>

81 </dependency>

82

83 <!-- Servlet -->

84 <dependency>

85 <groupId>javax.servlet</groupId>

86 <artifactId>servlet-api</artifactId>

87 <version>2.5</version>

88 <scope>provided</scope>

89 </dependency>

90

91

92 </dependencies>

93

94 <profiles>

95 <profile>

96 <!-- 本地环境 -->

97 <id>local</id>

98 <properties>

99

100 </properties>

101 </profile>

102 <profile>

103 <!-- 开发环境 -->

104 <id>dev</id>

105 <properties>

106

107 </properties>

108 <!-- 默认激活本环境 -->

109 <activation>

110 <activeByDefault>true</activeByDefault>

111 </activation>

112 </profile>

113 <profile>

114 <!-- 测试环境 -->

115 <id>test</id>

116 <properties>

117

118 </properties>

119 </profile>

120 <profile>

121 <!-- 预发布环境 -->

122 <id>pre</id>

123 <properties>

124

125 </properties>

126 </profile>

127 <profile>

128 <!-- 生产环境 -->

129 <id>prod</id>

130 <properties>

131

132 </properties>

133 </profile>

134 </profiles>

135

136 <build>

137 <resources>

138 <resource>

139 <directory>src/main/resources</directory>

140 <filtering>true</filtering>

141 </resource>

142 </resources>

143 <plugins>

144 <plugin>

145 <groupId>org.apache.maven.plugins</groupId>

146 <artifactId>maven-compiler-plugin</artifactId>

147 <version>2.5.1</version>

148 <configuration>

149 <source>1.6</source>

150 <target>1.6</target>

151 <encoding>utf-8</encoding>

152 </configuration>

153 </plugin>

154 </plugins>

155 </build>

156 <organization>

157 <name>infosky</name>

158 <url>www.infosky.com.cn</url>

159 </organization>

160 </project>

[复制代码](javascript:void(0);)

**二、**web.xml

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

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

2 <web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"

3 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

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

5 <context-param>

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

7 <param-value>classpath:root-context.xml</param-value>

8 </context-param>

9 <filter>

10 <filter-name>CharacterEncodingFilter</filter-name>

11 <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>

12 <init-param>

13 <param-name>encoding</param-name>

14 <param-value>utf-8</param-value>

15 </init-param>

16 </filter>

17 <filter-mapping>

18 <filter-name>CharacterEncodingFilter</filter-name>

19 <url-pattern>/\*</url-pattern>

20 </filter-mapping>

21 <listener>

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

23 </listener>

24 <servlet>

25 <servlet-name>appServlet</servlet-name>

26 <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

27 <init-param>

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

29 <param-value>classpath:servlet-context.xml</param-value>

30 </init-param>

31 <load-on-startup>0</load-on-startup>

32 </servlet>

33 <servlet-mapping>

34 <servlet-name>appServlet</servlet-name>

35 <url-pattern>/</url-pattern>

36 </servlet-mapping>

37

38 </web-app>

[复制代码](javascript:void(0);)

**三、**servlet-context.xml (关键配置)

[复制代码](javascript:void(0);)

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:mvc="http://www.springframework.org/schema/mvc"

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

5 xsi:schemaLocation="http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc.xsd

6 http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd

7 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd">

8

9 <mvc:annotation-driven />

10 <context:component-scan base-package="com.cnblogs.yjmyzz" />

11 <mvc:resources mapping="/resources/\*\*" location="/resources/" />

12

13

14 <bean

15 class="org.springframework.web.servlet.view.ContentNegotiatingViewResolver">

16 <property name="order" value="1" />

17 <property name="favorParameter" value="false" />

18 <property name="ignoreAcceptHeader" value="true" />

19 <property name="defaultContentType" value="text/html" />

20 <property name="mediaTypes">

21 <map>

22 <entry key="json" value="application/json" />

23 <entry key="xml" value="application/xml" />

24 </map>

25 </property>

26 <property name="viewResolvers">

27 <list>

28 <bean class="org.springframework.web.servlet.view.BeanNameViewResolver" />

29 <bean

30 class="org.springframework.web.servlet.view.InternalResourceViewResolver">

31 <property name="prefix" value="/WEB-INF/views/" />

32 <property name="suffix" value=".jsp" />

33 </bean>

34 </list>

35 </property>

36 <property name="defaultViews">

37 <list>

38 <bean id="jsonView"

39 class="org.springframework.web.servlet.view.json.MappingJacksonJsonView" />

40 <bean id="xmlView"

41 class="org.springframework.web.servlet.view.xml.MarshallingView">

42 <constructor-arg>

43 <bean class="org.springframework.oxm.jaxb.Jaxb2Marshaller">

44 <property name="classesToBeBound">

45 <list>

46 <value>com.cnblogs.yjmyzz.dto.UserInfo</value>

47 <value>com.cnblogs.yjmyzz.dto.ListBean</value>

48 </list>

49 </property>

50 </bean>

51 </constructor-arg>

52 </bean>

53 </list>

54 </property>

55 </bean>

56

57 </beans>

[复制代码](javascript:void(0);)

注：46，47行对应 Rest Service中涉及的对象类名，根据需要自行添加。

**四、**DTO对象

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

1 package com.cnblogs.yjmyzz.dto;

2

3 import java.io.Serializable;

4 import java.math.BigDecimal;

5 import java.util.Date;

6

7 import javax.xml.bind.annotation.XmlElement;

8 import javax.xml.bind.annotation.XmlRootElement;

9

10 import com.cnblogs.yjmyzz.utils.DateUtil;

11 import com.cnblogs.yjmyzz.utils.ListUtil;

12

13 @XmlRootElement(name = "user")

14 public class UserInfo implements Serializable {

15

16 private static final long serialVersionUID = -5461373449802431627L;

17 private String userName;

18 private BigDecimal salary;

19 private Date birthday;

20 private boolean isVip;

21 private int id;

22 private ListBean hobbies;

23

24 @XmlElement

25 public String getUserName() {

26 return userName;

27 }

28

29 public void setUserName(String userName) {

30 this.userName = userName;

31 }

32

33 @XmlElement

34 public BigDecimal getSalary() {

35 return salary;

36 }

37

38 public void setSalary(BigDecimal salary) {

39 this.salary = salary;

40 }

41

42 @XmlElement

43 public Date getBirthday() {

44 return birthday;

45 }

46

47 public void setBirthday(Date birthday) {

48 this.birthday = birthday;

49 }

50

51 @XmlElement

52 public boolean isVip() {

53 return isVip;

54 }

55

56 public void setVip(boolean isVip) {

57 this.isVip = isVip;

58 }

59

60 @XmlElement

61 public int getId() {

62 return id;

63 }

64

65 public void setId(int id) {

66 this.id = id;

67 }

68

69 @XmlElement

70 public ListBean getHobbies() {

71 return hobbies;

72 }

73

74 public void setHobbies(ListBean hobbies) {

75 this.hobbies = hobbies;

76 }

77

78 public String toString() {

79 return "id:" + this.id + ",userName:" + this.userName + ",isVip="

80 + this.isVip + ",birthday="

81 + DateUtil.formatDate(this.birthday) + ",hobbies:"

82 + ListUtil.getString(this.hobbies.getList());

83 }

84

85 }

[复制代码](javascript:void(0);)

为了实现List的XML序列化，还需要一个辅助类ListBean

[复制代码](javascript:void(0);)

1 package com.cnblogs.yjmyzz.dto;

2

3 import java.util.List;

4

5 import javax.xml.bind.annotation.XmlElement;

6 import javax.xml.bind.annotation.XmlElements;

7 import javax.xml.bind.annotation.XmlRootElement;

8

9 @XmlRootElement(name = "list")

10 public class ListBean {

11

12 public ListBean() {

13 }

14

15 public ListBean(List<?> list) {

16 this.list = list;

17 }

18

19 private List<?> list;

20

21 @XmlElements({ @XmlElement(name = "user", type = UserInfo.class),

22 @XmlElement(name = "hobby", type = String.class)

23 // 如果还有其它类型，继续添加

24 })

25 public List<?> getList() {

26 return list;

27 }

28

29 public void setList(List<?> list) {

30 this.list = list;

31 }

32 }

[复制代码](javascript:void(0);)

注：25行getList前的注解，根据实际需要自行修改

**五、**Controller

[复制代码](javascript:void(0);)

1 package com.cnblogs.yjmyzz.web.controller;

2

3 import javax.servlet.http.HttpServletRequest;

4 import javax.servlet.http.HttpServletResponse;

5

6 import org.springframework.beans.factory.annotation.Autowired;

7 import org.springframework.stereotype.Controller;

8 import org.springframework.web.bind.annotation.PathVariable;

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

10 import org.springframework.web.bind.annotation.RequestMethod;

11

12 import com.cnblogs.yjmyzz.dto.ListBean;

13 import com.cnblogs.yjmyzz.dto.UserInfo;

14 import com.cnblogs.yjmyzz.service.UserService;

15

16 @Controller

17 @RequestMapping(value = "/rest", method = RequestMethod.GET)

18 public class RestController {

19

20 @Autowired

21 UserService userService;

22

23 @RequestMapping(value = "/user/{id}", method = RequestMethod.GET)

24 public UserInfo show(@PathVariable int id, HttpServletRequest request,

25 HttpServletResponse response) throws Exception {

26 return userService.getUserInfo(id);

27

28 }

29

30 @RequestMapping(value = "/user/list", method = RequestMethod.GET)

31 public ListBean getAll() throws Exception {

32 return userService.getAllUsers();

33

34 }

35

36 }

[复制代码](javascript:void(0);)

其中UserService的代码为：

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

1 package com.cnblogs.yjmyzz.service;

2

3 import java.math.BigDecimal;

4 import java.util.ArrayList;

5 import java.util.List;

6

7 import org.springframework.beans.factory.annotation.Autowired;

8 import org.springframework.stereotype.Service;

9

10 import com.cnblogs.yjmyzz.dto.ListBean;

11 import com.cnblogs.yjmyzz.dto.UserInfo;

12 import com.cnblogs.yjmyzz.utils.DateUtil;

13

14 @Service("userService")

15 public class UserService {

16

17 @Autowired

18 UserInfo defaultUserInfo;

19

20 List<UserInfo> userInfos = null;

21

22 public UserService() {

23 init();

24 }

25

26 private void init() {

27 userInfos = new ArrayList<UserInfo>();

28 UserInfo user1 = new UserInfo();

29 user1.setBirthday(DateUtil.getDate(1985, 1, 1));

30 user1.setId(1);

31 user1.setUserName("A");

32 user1.setVip(true);

33 user1.setSalary(new BigDecimal(5000.00d));

34 List<String> hobbyNames = new ArrayList<String>();

35 hobbyNames.add("music");

36 hobbyNames.add("movie");

37 user1.setHobbies(new ListBean(hobbyNames));

38 userInfos.add(user1);

39

40 UserInfo user2 = new UserInfo();

41 user2.setBirthday(DateUtil.getDate(1988, 10, 2));

42 user2.setId(2);

43 user2.setUserName("B");

44 user2.setVip(false);

45 user2.setSalary(new BigDecimal(8000.60d));

46 userInfos.add(user2);

47

48 }

49

50 public UserInfo getUserInfo(int userId) {

51 for (UserInfo u : userInfos) {

52 if (u.getId() == userId) {

53 return u;

54 }

55 }

56 return defaultUserInfo;

57

58 }

59

60 public ListBean getAllUsers() {

61 return new ListBean(userInfos);

62 }

63 }

[复制代码](javascript:void(0);)

其中DefaultUserInfo是通过配置由Spring注入的默认用户对象

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

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"

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

5

6 <bean id="dateFormat" class="java.text.SimpleDateFormat">

7 <constructor-arg value="yyyy-MM-dd" />

8 </bean>

9

10 <bean name="defaultUserBean" class="com.cnblogs.yjmyzz.dto.UserInfo">

11 <property name="userName" value="匿名"></property>

12 <property name="salary" value="1000.00"></property>

13 <property name="vip" value="false"></property>

14 <property name="id" value="-1"></property>

15 <property name="birthday">

16 <bean factory-bean="dateFormat" factory-method="parse">

17 <constructor-arg value="1985-06-01" />

18 </bean>

19 </property>

20 </bean>

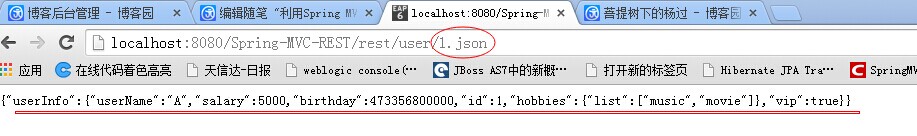
21

22 </beans>

[复制代码](javascript:void(0);)

**效果：**

http://localhost:8080/Spring-MVC-REST/rest/user/1.**json** (返回json格式)



http://localhost:8080/Spring-MVC-REST/rest/user/1.**xml** (返回xml格式)



上面都是GET的示例，最后来一个POST的demo:

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

1 @RequestMapping(value = "/user/create", method = RequestMethod.POST)

2 @ResponseBody

3 public UserInfo create(@RequestBody(required=true) UserInfo user, HttpServletRequest request,

4 HttpServletResponse response) throws Exception {

5 logger.debug("日志记录成功：" + user);

6 user.setId(9999);//只是示例下，模拟db生成了主键id

7 return user;

8 }

[复制代码](javascript:void(0);)

源代码下载：[Spring-MVC-REST.zip](http://files.cnblogs.com/yjmyzz/Spring-MVC-REST.zip)

最后附上一个链接，关于@RequestParam、@RequestBody、@RequestHeader、@CookieValue 这堆注解的用途，可参看这里

http://blog.csdn.net/kobejayandy/article/details/12690161