
# Module 25 Supplement: Spring RESTful Web Services

CS544: Enterprise Architecture

Spring Web Services:

# SPRING RESTFUL WEB SERVICES

# RESTful Web Services, JAX-RS & JSON

- REST and RESTful Web Services:
  - REST: REpresentational State Transfer – a software architectural style, defined around 2000 by Roy Fielding as part of his PhD dissertation
  - In a REST architecture, data and functionality are considered resources that can be accessed via URIs (i.e. links)

- RESTful Web Services – simple, lightweight, high-performant, scalable.
  - Resource Identification through URI
  - Resource manipulation using a fixed set of operations – get, put, post delete
  - E.g URI - http://www.webserver.com/resource/id/123/ - Resource with Id of 123

# RESTful Web Services, JAX-RS & JSON

- JAX-RS
  - JAVA API for RESTful Web Services
  - Reference Implementation – Project Jersey
    - https://jersey.java.net/
  - Provides support for annotations which simplifies WS implementation
    - e.g. @RequestMapping(“/resources”)
  - Well supported in Spring Framework

# RESTful Web Service with Spring

- Create a Resource Representation Class

```
package com.cs544.model;
public class Resource {
      private int id;
      private String content;
      public Resource (int id, String content) {
        this.id = id;
        this.content = content;
      }
      public int getId() { return id; }
      public void setId(int id) { … }
      public String getContent() { return content; }
      public void setContent(String content) { …}

}
```

# RESTful Web Service with Spring

- Create a Resource Controller

```
package com.cs544.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
…
@Controller // Note: Spring 4 supports @RESTController
 @RequestMapping(“/resource/id”)
public class ResourceController {
      @RequestMapping(value="{id}", method = RequestMethod.GET,
     produces="application/json")
     public @ResponseBody Resource getResourceById (@PathVariable int id) {

       //ResourceRepository.findResourceById(id);
       Resource res = new Resource(); res.setId(id); res.setContent(“something here”);
       return res;
     }
}
```

# Surfacing JSON with Spring REST

- Two options exist for doing this:

  Option 1

  If the following four conditions are met:

  i. Jackson library present in classpath
  ii. @Controller annotation on controller class
  iii. Spring config has mvc:annotation-driven enabled
  iv. Return type of Controller method is annotated with @ResponseBody

  This is the Default behavior

# Surfacing JSON with Spring REST

- Two options exist for doing this:

Option 2

- This entails overriding the Default behavior by explicitly setting appropriate configurations in Spring-Config.xml
- Includes mainly setting appropriate bean class for handling ContentNegotiation and specifying supported mediaTypes

# RESTful WS – Spring Config

CS544SpringREST/pom.xml | Author.java | AuthorsController.java | web.xml | spring-rest-dispatcher-servlet.xml

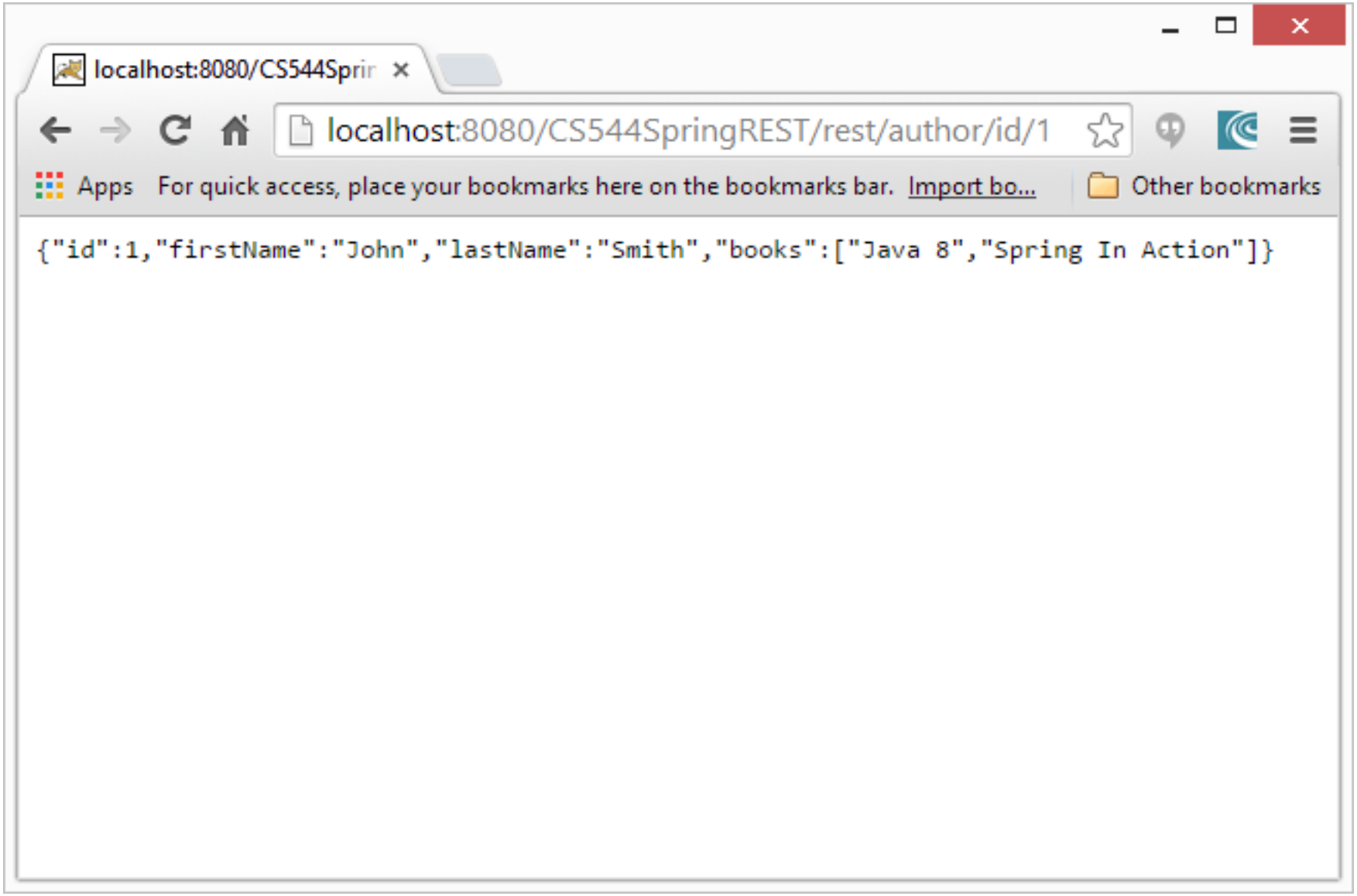
```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:mvc="http://www.springframework.org/schema/mvc" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context.xsd
        http://www.springframework.org/schema/mvc
        http://www.springframework.org/schema/mvc/spring-mvc.xsd">

    <context:component-scan base-package="com.cs544.rest.web.services.controller" />

    <!-- activates annotation driven binding -->
    <mvc:annotation-driven />

    <!-- Handle json and other output -->
    <bean class="org.springframework.web.servlet.view.ContentNegotiatingViewResolver">
      <property name="mediaTypes">
        <map>
          <entry key="json" value="application/json"/>
        </map>
      </property>
      <property name="viewResolvers">
        <list>
          <bean class="org.springframework.web.servlet.view.BeanNameViewResolver"/>
        </list>
      </property>
      <property name="defaultViews">
        <list>
          <bean class="org.springframework.web.servlet.view.json.MappingJackson2JsonView" />
        </list>
      </property>
    </bean>
```

# RESTful WS – Spring Config

CS544SpringREST/pom.xml | Author.java | AuthorsController.java | web.xml | spring-rest-dispatcher-servlet.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation=
3    <display-name>CS544SpringREST</display-name>
4
5      <servlet>
6          <servlet-name>spring-rest-dispatcher</servlet-name>
7          <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
8          <load-on-startup>1</load-on-startup>
9      </servlet>
10
11     <servlet-mapping>
12         <servlet-name>spring-rest-dispatcher</servlet-name>
13         <url-pattern>/rest/*</url-pattern>
14     </servlet-mapping>
15
16     <context-param>
17         <param-name>contextConfigLocation</param-name>
18         <param-value>/WEB-INF/spring-rest-dispatcher-servlet.xml</param-value>
19     </context-param>
20
21     <listener>
22         <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
23     </listener>
24
25   <welcome-file-list>
26     <welcome-file>index.html</welcome-file>
27     <welcome-file>index.htm</welcome-file>
28     <welcome-file>index.jsp</welcome-file>
29     <welcome-file>default.html</welcome-file>
30     <welcome-file>default.htm</welcome-file>
31     <welcome-file>default.jsp</welcome-file>
32   </welcome-file-list>
33 </web-app>
```

# SPRING RESTFUL WS – JSON OUTPUT



# Spring REST – Additional Exercise

- Read through the sample implementation code for producing Author object data in JSON format

- Modify and extend the solution to produce a list of Authors when the service is invoked as:
  - http://localhost:8080/CS544SpringREST/rest/authors