

Industrial strength Java/JEE Career Companion to open more doors

[Home](#)
[Java FAQs](#)
[600+ Java Q&As](#)
[Career](#)
[Tutorials](#)
[Member](#)
[Why?](#)
[Can u Debug?](#)
[Java 8 ready?](#)
[Top X](#)
[Productivity Tools](#)
[Judging Experience?](#)

[Home](#) > [member-paid](#) > 07: HATEOAS RESTFul Service with Spring tutorial

07: HATEOAS RESTFul Service with Spring tutorial

Posted on July 2, 2015 by [Arulkumaran Kumaraswamipillai](#)

0
Like
Share

Tweet

0
G+1
Share

Step 1: Create a Maven project structure

```
1 mvn archetype:generate -DgroupId=com.mytutorial -
```

Step 2: pom.xml to get right jars

```
1 <project xmlns="http://maven.apache.org/POM/4.0.0"
2   xsi:schemaLocation="http://maven.apache.org/
3     <modelVersion>4.0.0</modelVersion>
4
5     <groupId>com.mytutorial</groupId>
6     <artifactId>simpleSpringRestWeb</artifactId>
7     <version>1.0-SNAPSHOT</version>
8     <packaging>war</packaging>
9
10    <name>simpleSpringRestWeb</name>
11    <url>http://maven.apache.org</url>
12
13    <properties>
```

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

[open all](#) | [close all](#)

[Ice Breaker Interview](#)

[Core Java Interview C](#)

[JEE Interview Q&A \(3](#)

[JEE Overview \(2\)](#)

[Web basics \(8\)](#)

[WebService \(11\)](#)

[01: ♥♦ 40+ Java](#)

[02: ♦ 6 Java RE](#)

[03: ♥ JAX-RS hc](#)

[04: 5 JAXB inter](#)

[05: RESTFul We](#)

[06: RESTful Wel](#)

[07: HATEOAS R](#)

[08: REST constr](#)

[09: 11 SOAP We](#)

[10: SOAP Web](#)

[11: ♥ JAX-WS hc](#)

[JPA \(2\)](#)

[JTA \(1\)](#)

[JDBC \(4\)](#)

[JMS \(5\)](#)

```

14     <project.build.sourceEncoding>UTF-8</project>
15 </properties>
16
17 <dependencies>
18     <dependency>
19         <groupId>junit</groupId>
20         <artifactId>junit</artifactId>
21         <version>3.8.1</version>
22         <scope>test</scope>
23     </dependency>
24     <dependency>
25         <groupId>com.sun.xml.bind</groupId>
26         <artifactId>jaxb-impl</artifactId>
27         <scope>provided</scope>
28         <version>2.1.3</version>
29     </dependency>
30     <dependency>
31         <groupId>xml-apis</groupId>
32         <artifactId>xml-apis</artifactId>
33         <scope>provided</scope>
34         <version>1.0.b2</version>
35     </dependency>
36     <dependency>
37         <groupId>javax.servlet</groupId>
38         <artifactId>javax.servlet-api</artifactId>
39         <scope>provided</scope>
40         <version>3.0.1</version>
41     </dependency>
42     <dependency>
43         <groupId>org.logback-extensions</groupId>
44         <artifactId>logback-ext-spring</artifactId>
45         <version>0.1.1</version>
46     </dependency>
47     <dependency>
48         <groupId>org.springframework.hateoas</groupId>
49         <artifactId>spring-hateoas</artifactId>
50         <version>0.17.0.RELEASE</version>
51     </dependency>
52     <dependency>
53         <groupId>com.fasterxml.jackson.datatype</groupId>
54         <artifactId>jackson-datatype-joda</artifactId>
55         <version>2.3.2</version>
56     </dependency>
57 </dependencies>
58 </project>

```












Note that “spring-hateoas” is included. The “com.fasterxml.jackson.datatype” is required for the JSON processing.

Step 3: Hello class with “ResourceSupport”

```














1 package com.mytutorial;
2
3 import org.springframework.hateoas.ResourceSupport;
4
5 import com.fasterxml.jackson.annotation.JsonCreator;
6 import com.fasterxml.jackson.annotation.JsonProperty;
7

```

-  [JMX \(3\)](#)
-  [JNDI and LDAP \(1\)](#)
-  [Pressed for time? Java](#)
-  [SQL, XML, UML, JSC](#)
-  [Hadoop & BigData Int](#)
-  [Java Architecture Inte](#)
-  [Scala Interview Q&As](#)
-  [Spring, Hibernate, & I](#)
-  [Testing & Profiling/Sa](#)
-  [Other Interview Q&A I](#)
-  [Free Java Interview](#)

16 Technical Key Areas

[open all](#) | [close all](#)

-  [Best Practice \(6\)](#)
-  [Coding \(26\)](#)
-  [Concurrency \(6\)](#)
-  [Design Concepts \(7\)](#)
-  [Design Patterns \(11\)](#)
-  [Exception Handling \(3\)](#)
-  [Java Debugging \(21\)](#)
-  [Judging Experience I](#)
-  [Low Latency \(7\)](#)
-  [Memory Management](#)
-  [Performance \(13\)](#)
-  [QoS \(8\)](#)
-  [Scalability \(4\)](#)
-  [SDLC \(6\)](#)
-  [Security \(13\)](#)
-  [Transaction Managen](#)

80+ step by step Java Tutorials

[open all](#) | [close all](#)

-  [Setting up Tutorial \(6\)](#)

```

8 public class Hello extends ResourceSupport{
9
10     private final String content;
11
12     @JsonCreator
13     public Hello(@JsonProperty("content") String
14         this.content = content;
15     }
16
17     public String getContent() {
18         return content;
19     }
20 }

```

The “jackson-datatype-joda” artifact brings in the dependency jackson libraries.

@JsonCreator – signal on how Jackson can create an instance of this POJO

@JsonProperty – clearly marks what field Jackson should put this constructor argument into.

Step 4: HelloControllerImpl with REST endpoint

```

1 package com.mytutorial;
2
3 //linkTo & methodOn static methods are from Cont
4 import static org.springframework.hateoas.mvc.Co
5 import static org.springframework.hateoas.mvc.Co
6
7 import org.springframework.http.HttpEntity;
8 import org.springframework.http.HttpStatus;
9 import org.springframework.http.ResponseEntity;
10 import org.springframework.stereotype.Controller
11 import org.springframework.web.bind.annotation.R
12 import org.springframework.web.bind.annotation.R
13 import org.springframework.web.bind.annotation.R
14
15
16 @Controller
17 public class HelloControllerImpl {
18
19     private static final String FORMAT = "Hello,
20
21     @RequestMapping("/greeting")
22     @ResponseBody
23     public HttpEntity<Hello> sayHello(@RequestPa
24         value = "name",
25         required = false,
26         defaultValue = "World") String name) {
27
28         Hello hello = new Hello(String.format(F0
29         hello.add(linkTo(methodOn(HelloControlle
30         return new ResponseEntity<Hello>(hello,
31     }
32 }

```

- [Tutorial - Diagnosis \(2](#)
- [Akka Tutorial \(9\)](#)
- [Core Java Tutorials \(2](#)
- [Hadoop & Spark Tuto](#)
- [JEE Tutorials \(19\)](#)
- [Scala Tutorials \(1\)](#)
- [Spring & Hibernate Ti](#)
- [Tools Tutorials \(19\)](#)
- [Other Tutorials \(45\)](#)

100+ Java pre-interview coding tests

[open all](#) | [close all](#)

- [Can you write code? \(](#)
- [♦ Complete the given](#)
- [Converting from A to I](#)
- [Designing your classe](#)
- [Java Data Structures](#)
- [Passing the unit tests](#)
- [What is wrong with th](#)
- [Writing Code Home A](#)
- [Written Test Core Jav](#)
- [Written Test JEE \(1\)](#)

How good are your?

[open all](#) | [close all](#)

- [Career Making Know-](#)
- [Job Hunting & Resum](#)

Take note of the “**linkTo**” and “**methodOn**” static methods from the “org.springframework.hateoas.mvc.ControllerLinkBuilder” class.

Step 5: Spring @JavaConfig class

```
1 package com.mytutorial;
2
3 import org.springframework.context.annotation.Co
4 import org.springframework.context.annotation.Co
5 import org.springframework.web.servlet.config.an
6 import org.springframework.web.servlet.config.an
7
8 @Configuration
9 @EnableWebMvc
10 @ComponentScan("com.mytutorial")
11 public class SimpleConfig extends WebMvcConfigur
12
13 }
```

Step 6: web.xml file

```
1 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSc
2   xsi:schemaLocation="http://java.sun.com/xml/
3   version="2.5">
4
5   <display-name>Simple Spring Web Service Appl
6
7   <servlet>
8     <servlet-name>simpleSpringWeb</servlet-n
9     <servlet-class>org.springframework.web.s
10    <init-param>
11      <param-name>contextClass</param-name
12      <param-value>org.springframework.web
13    </init-param>
14    <init-param>
15      <param-name>contextConfigLocation</p
16      <param-value>com.mytutorial.SimpleCo
17    </init-param>
18    <load-on-startup>1</load-on-startup>
19  </servlet>
20
21  <servlet-mapping>
22    <servlet-name>simpleSpringWeb</servlet-n
23    <url-pattern>/*</url-pattern>
24  </servlet-mapping>
25
26 </web-app>
27
```

If you don't add “contextClass” as
“org.springframework.web.context.support.AnnotationConfig

WebApplicationContext" it will be looking for the Spring context XML file in **WEB-INF** folder.

Step 7: jboss-web.xml if deploying to JBoss server

```
1 <!DOCTYPE jboss-web PUBLIC "-//JBoss//DTD Web App
2 "http://www.jboss.org/j2ee/dtd/jboss-web_5_0.
3 <jboss-web>
4   <context-root>/simpleSpring</context-root>
5 </jboss-web>
```

Step 8: Invoke the URL

After building & deploying the application hit the url

```
1 http://localhost:8080/simpleSpring/greeting/
```

You will get the JSON data as shown below

```
1 {"content":"Hello, World!",
2  "links":[{"rel":"self","href":"http://localhost
3  }]
```

if you use the url

```
1 http://localhost:8080/simpleSpring/greeting?name=
```

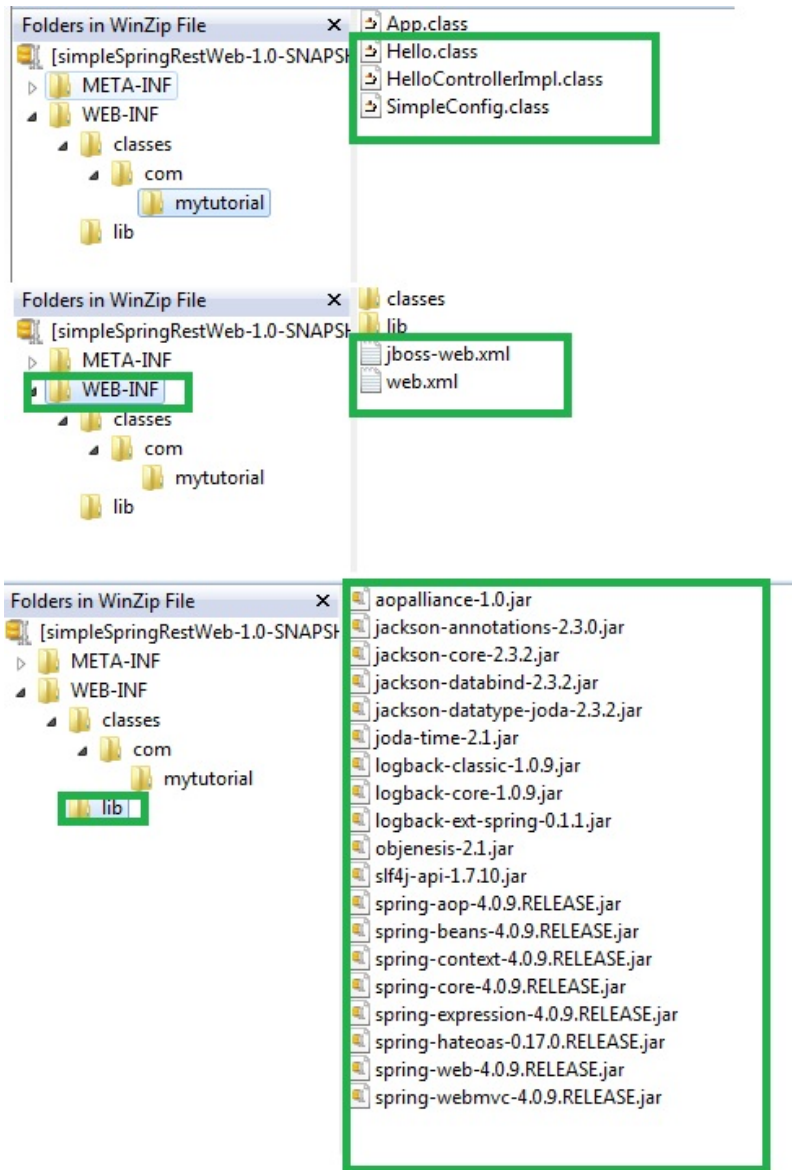
You will get the response of

```
1 {"content":"Hello,
2  John!", "links":[{"rel":"self","href":"http://loc
3  }]
4
```

Project Structure



<http://www.java-success.com/hateoas-restful-service-with-spring-tutorial/>



HATEOAS Spring Web WAR

Popular Posts

♦ [11 Spring boot interview questions & answers](#)

825 views

♦ [Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers](#)

766 views

[18 Java scenarios based interview Questions and Answers](#)

400 views

001A: ♦ 7+ Java integration styles & patterns

interview questions & answers

388 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

295 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

285 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

279 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

239 views

001B: ♦ Java architecture & design concepts interview questions & answers

201 views

Bio

Latest Posts



Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site. **945+** paid members. [join my LinkedIn Group](#). [Reviews](#)



About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job



interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via [Amazon.com](https://www.amazon.com) in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site. **945+** paid members. [join my LinkedIn Group](#). [Reviews](#)

◀ ♥ Hidden Unix, Git & SVN config files

Setting up Cygwin with Git Tutorial ▶

Posted in member-paid, WebService

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ [Java generics in no time](#) ☀ [Top 6 tips to transforming your thinking from OOP to FP](#) ☀ [How does a HashMap internally work? What is a hashing function?](#)
☀ [10+ Java String class interview Q&As](#) ☀ [Java auto un/boxing benefits & caveats](#) ☀ [Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect](#)

Non-Technical Know Hows

☀ [6 Aspects that can motivate you to fast-track your career & go places](#) ☀ [Are you reinventing yourself as a Java developer?](#) ☀ [8 tips to safeguard your Java career against offshoring](#) ☀ [My top 5 career mistakes](#)

Prepare to succeed

☀ [Turn readers of your Java CV go from "Blah blah" to "Wow"? ☀ \[How to prepare for Java job interviews?\]\(#\) ☀ \[16 Technical Key Areas\]\(#\) ☀ \[How to choose from multiple Java job offers?\]\(#\)](#)

Select Category ▼

© Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.