



```

15. ResponseEntity<BetResource> createBet(@RequestBody Bet body) {
16.     Bet bet = betService.createBet(body.getMarketId(),
17.         body.getSelectionId(), body.getPrice(), body.getStake(),
18.         body.getType());
19.     BetResource resource = betResourceAssembler.toResource(bet);
20.     return new ResponseEntity<BetResource>(resource,
        HttpStatus.CREATED);
21. }
22.
23. @RequestMapping(method = RequestMethod.PUT, value =("/{betId}")
24. ResponseEntity<BetResource> updateBet(@PathVariable Long betId,
25. @RequestBody Bet body) throws BetNotFoundException,
        BetNotUnmatchedException {
26.     Bet bet = betService.updateBet(betId, body);
27.     BetResource resource = betResourceAssembler.toResource(bet);
28.     return new ResponseEntity<BetResource>(resource, HttpStatus.OK);
29. }
30.
31. @RequestMapping(method = RequestMethod.GET, value =("/{betId}")
32. ResponseEntity<BetResource> getBet(@PathVariable Long betId) throws
        BetNotFoundException {
33.     Bet bet = betService.getBet(betId);
34.     BetResource resource = betResourceAssembler.toResource(bet);
35.     if (bet.getStatus() == BetStatus.UNMATCHED) {
36.         resource.add(linkTo(BetController.class).slash(bet.getId()).withRel
37.     }
38.     return new ResponseEntity<BetResource>(resource, HttpStatus.OK);
39. }
40.
41. @RequestMapping(method = RequestMethod.GET)
42. ResponseEntity<List<BetResource>> getBets() {
43.     List<Bet> betList = betService.getAllBets();
44.     List<BetResource> resourceList =
        betResourceAssembler.toResources(betList);
45.     return new ResponseEntity<List<BetResource>>(resourceList,
        HttpStatus.OK);
46. }
47.
48. @RequestMapping(method = RequestMethod.DELETE, value =("/{betId}")
49. ResponseEntity<BetResource> cancelBet(@PathVariable Long betId) {
50.     Bet bet = betService.cancelBet(betId);
51.     BetResource resource = betResourceAssembler.toResource(bet);
52.     return new ResponseEntity<BetResource>(resource, HttpStatus.OK);
53. }
54.
55. @ExceptionHandler
56. ResponseEntity handleExceptions(Exception ex) {
57.     ResponseEntity responseEntity = null;
58.     if (ex instanceof BetNotFoundException) {
59.         responseEntity = new ResponseEntity(HttpStatus.NOT_FOUND);
60.     } else if (ex instanceof BetNotUnmatchedException) {
61.         responseEntity = new ResponseEntity(HttpStatus.CONFLICT);
62.     } else {
63.         responseEntity = new
            ResponseEntity(HttpStatus.INTERNAL_SERVER_ERROR);
64.     }
65.     return responseEntity;
66. }
67.
68. }

```

All the operations will create a *BetResource* for returning to the client. This is done by calling *toResource* on the *BetResourceAssembler* class.



### Spotlight Features



An Exhaustive List of  
Developer Topic Newsletters



Oracle Confirms New Java 9  
Features



DevNexus Conference:  
Gearing up for DEVolution



My Most Frequent Code  
Review Comment

```

01. public class BetResourceAssembler extends
    ResourceAssemblerSupport<Bet, BetResource> {
02.
03. public BetResourceAssembler() {
04.     super(BetController.class, BetResource.class);
05. }
06.
07. public BetResource toResource(Bet bet) {
08.     BetResource resource = instantiateResource(bet);
09.     resource.bet = bet;
10.     resource.add(linkTo(BetController.class).slash(bet.getId()).withSelfLink());
11.     return resource;
12. }
13.
14. }

```

This class extends *ResourceAssemblerSupport* which requires the implementation of a *toResource* method as it implements the *ResourceAssembler* interface. This is where the mapping between *Bet* and *BetResource* is done. In this case, *BetResource* is just a wrapper for *Bet* so it is simply a case of setting the *bet* attribute. The *instantiateResource* method will return a *BetResource* without any links so links can be added at this point if required. In this example a link to self is added. An alternative approach would be to use *createResourceWithId* which will return a *BetResource* with the self link.

```

1. public class BetResource extends ResourceSupport {
2.
3.     public Bet bet;
4.
5. }

```

Also in this example, links are added to the *BetResource* within the *BetController* class to ensure the application of the HATEOAS constraint. If the REST service receives a GET request then a check is made on the status of the *Bet*. If the *Bet* is *UNMATCHED*, then a link to cancel the *Bet* can be added to the *BetResource*. This is done in similar fashion to the self link but with the relationship attribute name of *cancel*. An alternative approach to this is to build a link to a method as opposed to constructing a URI.

```

1. resource.add(linkTo(methodOn(BetController.class).cancelBet(bet.getId())).withRel("cancel"));
2.

```

The *methodOn* would create a proxy of the *BetController* class and as a result the return type of the *cancelBet* method would have to be capable of proxying. Therefore in this example the return type of *cancelBet* method would be *HttpEntity<Bet>* and not *ResponseEntity<Bet>*. If the latter, then the likely exception from the server would be:

```

[org.springframework.http.ResponseEntity com.city81.hateoas.rest.BetResource]
com.city81.hateoas.controller.BetController.getBet(java.lang.Long) throws
com.city81.hateoas.BetNotFoundException: org.springframework.aop.framework.AopConfigException:
Could not generate CGLIB subclass of class [class org.springframework.http.ResponseEntity]: common
causes of this problem include using a final class or a non-visible class; nested exception is
java.lang.IllegalArgumentException: Superclass has no null constructors but no arguments were given
Back to the GET request, and the returned JSON for requesting a Bet resource which has a status
of UNMATCHED is shown below:

```

```

{
  "links": [
    { "rel": "self", "href": "http://localhost:8080/hateoas-1-SNAPSHOT/bets/0" },
    { "rel": "cancel", "href": "http://localhost:8080/hateoas-1-SNAPSHOT/bets/0" } ],
  "bet": { "id": 0, "marketId": 1, "selectionId": 22, "price": 4.0, "stake": 2.0, "type": "BACK", "status": "UNMATCHED" }
}

```

The client can therefore use the self link for retrieving and updating the *Bet*, and also the cancel link to effectively delete it.

This post describes just some of the functionality of the Spring-HATEOAS project which is evolving all

the time. For an up to date and more detailed explanation, visit the [GitHub](#) pages.

Published at DZone with permission of [Geraint Jones](#), author and DZone MVB. ([source](#))

(Note: Opinions expressed in this article and its replies are the opinions of their respective authors and not those of DZone, Inc.)

Tags: [Java](#) [REST](#) [Frameworks](#)

The *Enterprise Integration Zone* is brought to you in partnership with [WSO2](#). Learn more about [WSO2's API Management](#).

## AROUND THE DZONE NETWORK

### ARCHITECTS

Top Posts of 2013: Big Data Beyond MapReduce: Goog...

### JAVALOBBY

Top Posts of 2013: The Principles of Java Applicat...

### ARCHITECTS

5 Things a Java Developer Should Consider This Yea...

### JAVALOBBY

Top Posts of 2013: There Are Only 2 Roles of Code

### JAVALOBBY

Singleton Design Pattern – An Introspection w/ B...

### SERVER

Best Best Practices Ever

## YOU MIGHT ALSO LIKE

[Code Golf: Fibonacci's Sequence Part Deux](#)

[The Codeless Code: Case 1 - The Small Stuff](#)

[What Are the Leading Trends in Cloud Computing?](#)

[Being Agile Is about the Journey, Not the Destination](#)

[Some Thoughts on Self-Organization in Agile Teams](#)

[REST is Not About APIs \(Part 1\)](#)

[Geek Reading February 26, 2015](#)

[Can You Mandate Your Agile Transformation?](#)

[Bash Script to Convert Subversion to Git](#)

[Video: MobileFirst for Bluemix \(MBaaS\)](#)

[Using MongoDB with Hadoop & Spark: Part 1 - Introduction & Setup](#)

[Raspberry PI Automation, Docker, and Other Inanities](#)

[Anxiety Causes Selfish Behavior](#)

[The Crafty Consultant's Guide to... Devops](#)

[How to Use SQL PIVOT To Compare Two Tables in Your Database](#)

## POPULAR ON JAVALOBBY

- [Spring Batch - Hello World](#)
- [Is Hibernate the best choice?](#)
- [How to Create Visual Applications in Java?](#)
- [9 Programming Languages To Watch In 2011](#)
- [Introduction to Oracle's ADF Faces Rich Client Framework](#)
- [Interview: John De Goes Introduces a Newly Free Source Code Editor](#)
- [Lucene's FuzzyQuery is 100 times faster in 4.0](#)
- [Time Slider: OpenSolaris 2008.11 Killer Feature](#)

## LATEST ARTICLES

- [The Road To Awesome - Welcome JBoss Champions Program](#)
- [Arduino Launches the Zero Pro](#)
- [The Best of DZone: Feb. 25 - Mar. 4](#)
- [Determining File Types in Java](#)
- [Why Graph Databases are Perfect for the Internet of Things](#)
- [Dell BYOD Upgrades Latest in Efforts to Simplify BYOD](#)
- [7 Things Engineers Can Do On Their Commute](#)
- [The Are No Silver Bullets: Which Error Handling Style to Pick For a Given Configuration of Constraints?](#)

## SPOTLIGHT RESOURCES



**Essential Couchbase APIs: Open Source NoSQL Data Access from Java, Ruby, and .NET**



**Camel Essential Components**

DZone's 170th Refcard is an essential reference to Camel, an open-source, lightweight, integration library. This Refcard is authored by...



**Practical DNS: Managing Domains for Safety, Reliability, and Speed**

 Search

### DZone

[Refcardz](#)  
[Tech Library](#)  
[Snippets](#)  
[About DZone](#)  
[Tools & Buttons](#)

[Book Reviews](#)  
[IT Questions](#)  
[My Profile](#)  
[Advertise](#)  
[Send Feedback](#)

### Topics

[HTML5](#)  
[Cloud](#)  
[.NET](#)  
[PHP](#)  
[Performance](#)  
[Agile](#)

[Windows Phone](#)  
[Mobile](#)  
[Java](#)  
[Eclipse](#)  
[Big Data](#)  
[DevOps](#)

### Follow Us

[Google +](#)  
[Facebook](#)  
[LinkedIn](#)  
[Twitter](#)

**"Starting from scratch" is seductive but disease ridden**  
 -Pithy Advice for Programmers

