VV1) <https://thepracticaldeveloper.com/2017/07/31/guide-spring-boot-controller-tests/> hitech

1)<https://medium.com/@gustavo.ponce.ch/spring-boot-restful-junit-mockito-hamcrest-eclemma-5add7f725d4e> god of gods full

3)<http://www.springboottutorial.com/unit-testing-for-spring-boot-rest-services> god controller test

2v) <https://springframework.guru/unit-testing-spring-mvc-spring-boot-1-4-part-1/>

2) <https://springframework.guru/mockito-mock-vs-spy-in-spring-boot-tests/> service

2)<https://dzone.com/articles/spring-boot-rest-api-unit-testing-with-junit>

4) <https://krishnasblog.com/2013/02/22/junit-testing-of-spring-mvc-application-introduction/>

v1) <https://howtodoinjava.com/best-practices/how-you-should-unit-test-dao-layer/> god simple dao

2) http://javasampleapproach.com/testing/unit-test-for-spring-mvc-controller-with-spring-boot

<https://dzone.com/articles/a-guide-to-mocking-with-mockito>

https://dzone.com/articles/spring-boot-unit-testing-and-mocking-with-mockito

https://docs.spring.io/spring-security/site/docs/current/reference/html/test-mockmvc.html

{<https://www.petrikainulainen.net/spring-mvc-test-tutorial/> god best

<https://github.com/pkainulainen/spring-mvc-test-examples> }

<http://www.springboottutorial.com/spring-boot-unit-testing-and-mocking-with-mockito-and-junit> bvn

<https://stackoverflow.com/questions/38159132/mockito-testing-service-layer> try it once

https://www.javacodegeeks.com/2012/10/junit-testing-spring-service-and-dao.html

<https://docs.spring.io/spring/docs/3.0.x/spring-framework-reference/html/testing.html#integration-testing> bvn

http://www.devglan.com/spring-mvc/writing-junit-tests-in-spring4-mvc

<https://github.com/matsev/spring-testing> bvnbvn

http://websystique.com/springmvc/spring-4-mvc-and-hibernate4-integration-testing-example-using-annotations/

<http://www.springboottutorial.com/unit-testing-for-spring-boot-rest-services> bvnbvn

https://www.petrikainulainen.net/writing-tests-for-data-access-code/

{ <http://jlearning.in/unit-testing-example-using-junit-mockito-spring-framework/>

<http://jlearning.in/category/junit-mockito/> }

https://julienprog.wordpress.com/2016/10/23/mock-hibernatetemplate-with-mockito/

<https://stackoverflow.com/questions/12467966/should-i-write-unit-tests-for-crud-operations-when-i-have-already-integration-te>

<https://dzone.com/articles/unit-and-integration-tests-in-spring-boot>

<http://blog.saa-solutions.com/lession-06-create-crud-spring-mvc-using-hibernate-unit-test-junit/>

https://github.com/in28minutes/spring-boot-master-class/tree/master/05.Spring-Boot-Advanced

---

https://www.testingexcellence.com/4-different-ways-iterate-map-java/

https://teamtreehouse.com/library/unit-testing-a-spring-application

Test CRUD of RestTemplate

<https://www.javacodegeeks.com/2014/02/migrating-spring-mvc-restful-web-services-to-spring-4.html> rest test bvn

<http://www.devglan.com/spring-mvc/writing-junit-tests-in-spring4-mvc> bvn

<https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-testing.html>

<https://medium.com/@konstantinnalum/input-validation-and-unit-testing-of-a-rest-controller-in-spring-boot-304b82ca7526> mock bvn

<https://www.concretepage.com/spring-boot/spring-boot-rest-jpa-hibernate-mysql-example> bvn god code n test both

<https://www.programcreek.com/java-api-examples/index.php?class=org.springframework.web.client.RestTemplate&method=getForEntity>

**Spring Testing Tools**

* [@RunWith](http://junit.sourceforge.net/javadoc/org/junit/runner/RunWith.html) – This is a JUnit annotation allowing one to run a test with a different runner than the one provided by JUnit.
* [SpringJUnit4ClassRunner](http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/test/context/junit4/SpringJUnit4ClassRunner.html) – This is a JUnit test runner for Spring applications. Typically, test classes are annoted with @RunWith(SpringJUnit4ClassRunner.class).
* [@ContextConfiguration](http://static.springsource.org/spring/docs/3.0.x/javadoc-api/org/springframework/test/context/ContextConfiguration.html) – This annotation can be used to specify how to load an applicationContext in Spring test class. This can be configured via XML files or Java configuration objects.

https://www.javacodegeeks.com/2012/10/junit-testing-spring-service-and-dao.html

9. Client Code with RestTemplate to Test CRUD Application

**RestClientUtil.java**

package com.concretepage.client;

import java.net.URI;

import org.springframework.http.HttpEntity;

import org.springframework.http.HttpHeaders;

import org.springframework.http.HttpMethod;

import org.springframework.http.MediaType;

import org.springframework.http.ResponseEntity;

import org.springframework.web.client.RestTemplate;

import com.concretepage.entity.Article;

public class RestClientUtil {

public void getArticleByIdDemo() {

HttpHeaders headers = new HttpHeaders();

headers.setContentType(MediaType.APPLICATION\_JSON);

RestTemplate restTemplate = new RestTemplate();

String url = "http://localhost:8080/spring-app/user/article/{id}";

HttpEntity<String> requestEntity = new HttpEntity<String>(headers);

ResponseEntity<Article> responseEntity = restTemplate.exchange(url, HttpMethod.GET, requestEntity, Article.class, 1);

Article article = responseEntity.getBody();

System.out.println("Id:"+article.getArticleId()+", Title:"+article.getTitle()

+", Category:"+article.getCategory());

}

public void getAllArticlesDemo() {

HttpHeaders headers = new HttpHeaders();

headers.setContentType(MediaType.APPLICATION\_JSON);

RestTemplate restTemplate = new RestTemplate();

String url = "http://localhost:8080/spring-app/user/articles";

HttpEntity<String> requestEntity = new HttpEntity<String>(headers);

ResponseEntity<Article[]> responseEntity = restTemplate.exchange(url, HttpMethod.GET, requestEntity, Article[].class);

Article[] articles = responseEntity.getBody();

for(Article article : articles) {

System.out.println("Id:"+article.getArticleId()+", Title:"+article.getTitle()

+", Category: "+article.getCategory());

}

}

public void addArticleDemo() {

HttpHeaders headers = new HttpHeaders();

headers.setContentType(MediaType.APPLICATION\_JSON);

RestTemplate restTemplate = new RestTemplate();

String url = "http://localhost:8080/spring-app/user/article";

Article objArticle = new Article();

objArticle.setTitle("Spring REST Security using Hibernate");

objArticle.setCategory("Spring");

HttpEntity<Article> requestEntity = new HttpEntity<Article>(objArticle, headers);

URI uri = restTemplate.postForLocation(url, requestEntity);

System.out.println(uri.getPath());

}

public void updateArticleDemo() {

HttpHeaders headers = new HttpHeaders();

headers.setContentType(MediaType.APPLICATION\_JSON);

RestTemplate restTemplate = new RestTemplate();

String url = "http://localhost:8080/spring-app/user/article";

Article objArticle = new Article();

objArticle.setArticleId(1);

objArticle.setTitle("Update:Java Concurrency");

objArticle.setCategory("Java");

HttpEntity<Article> requestEntity = new HttpEntity<Article>(objArticle, headers);

restTemplate.put(url, requestEntity);

}

public void deleteArticleDemo() {

HttpHeaders headers = new HttpHeaders();

headers.setContentType(MediaType.APPLICATION\_JSON);

RestTemplate restTemplate = new RestTemplate();

String url = "http://localhost:8080/spring-app/user/article/{id}";

HttpEntity<Article> requestEntity = new HttpEntity<Article>(headers);

restTemplate.exchange(url, HttpMethod.DELETE, requestEntity, Void.class, 4);

}

public static void main(String args[]) {

RestClientUtil util = new RestClientUtil();

//util.getArticleByIdDemo();

util.addArticleDemo();

//util.updateArticleDemo();

//util.deleteArticleDemo();

util.getAllArticlesDemo();

}

}

<https://memorynotfound.com/unit-test-spring-mvc-rest-service-junit-mockito/>

--

REST n Us testing :

<https://piotrminkowski.wordpress.com/tag/testing/> god bvn