**1. What does REST stand for? (answer)**

REST stands for REpresentational State Transfer, which uses HTTP protocol to send data from client to server e.g. a book in the server can be delivered to the client using JSON or XML.

**2. What is a resource? (answer)**

A resource is how data is represented in REST architecture. By exposing entities as the resource it allows a client to read, write, modify, and create resources using HTTP methods like [GET](http://javarevisited.blogspot.sg/2012/03/get-post-method-in-http-and-https.html), [POST](http://www.java67.com/2014/08/difference-between-post-and-get-request.html), [PUT](http://www.java67.com/2016/09/when-to-use-put-or-post-in-restful-web-services.html), DELETE, etc.

**3. What are safe REST operations? (answer)**

REST API uses HTTP methods to perform operations. Some of the HTTP operations which don't modify the resource at the server are known as safe operations e.g. GET and HEAD. On the other hand, [PUT](http://javarevisited.blogspot.sg/2016/10/difference-between-put-and-post-in-restful-web-service.html), POST, and DELETE are unsafe because they modify the resource on the server.

**4. What are idempotent operations? Why is idempotency important? (**[**answer**](http://javarevisited.blogspot.sg/2016/05/what-are-idempotent-and-safe-methods-of-HTTP-and-REST.html)**)**

There are some HTTP methods e.g. GET which produce same response no matter how many times you use them e.g. sending multiple GET request to the same URI will result in same response without any side-effect hence it is known as idempotent.  
  
On the other hand, the POST is not idempotent because if you send multiple POST requests, it will result in multiple resource creation on the server, but again, PUT is idempotent if you are using it to update the resource.  
  
Even, multiple PUT requests to update a resource on a server will give the same end result. You can further take the [**HTTP Fundamentals**](http://pluralsight.pxf.io/c/1193463/424552/7490?u=https%3A%2F%2Fwww.pluralsight.com%2Fcourses%2Fxhttp-fund) course by Pluralsight to learn more about idempotent methods of the HTTP protocol and HTTP in general.

**5. Is REST scalable and/or interoperable? (answer)**

Yes, REST is Scalable and interoperable. It doesn't mandate a specific choice of technology either at the client or server end. You can use Java, C++, Python or JavaScript to create RESTful Web Services and Consume them at the client end. I suggest you read a good book on REST API e.g. [RESTful Web Services](http://javarevisited.blogspot.sg/2017/02/top-5-books-to-learn-rest-and-restful-web-services-in-java.html) to learn more about REST.

**6. What are the advantages of the RestTemplate? (**[**answer**](http://javarevisited.blogspot.sg/2017/02/how-to-consume-json-from-restful-web-services-Spring-RESTTemplate-Example.html)**)**

The RestTemplate class is an implementation of the Template method pattern in the Spring framework. Similar to other popular template classes like JdbcTemplate or JmsTempalte, it also simplifies the interaction with RESTful Web Services on the client-side. You can use it to consume a RESTful Web Servicer very easily as shown in this example.

**7. Which HTTP methods does REST use? (**[**answer**](http://javarevisited.blogspot.sg/2016/04/what-is-purpose-of-http-request-types-in-RESTful-web-service.html#axzz56WGunSwy)**)**

REST can use any HTTP methods but the most popular ones are GET for retrieving a resource, POST for creating a resource, PUt for updating the resource and DELETE for removing a resource from the server.

**8. What is an HttpMessageConverter in Spring REST? (answer)**

An HttpMessageConverter is a [Strategy interface](http://www.java67.com/2014/12/strategy-pattern-in-java-with-example.html) that specifies a converter that can convert from and to HTTP requests and responses. Spring REST uses this interface to convert HTTP response to various formats e.g. JSON or XML.  
  
Each HttpMessageConverter implementation has one or several MIME Types associated with it. Spring uses the "Accept" header to determine the content type the client is expecting.  
  
It will then try to find a registered HTTPMessageConverter that is capable of handling that specific content-type and use it to convert the response into that format before sending it to the client.

**9. How to create a custom implementation of HttpMessageConverter to support a new type of request/response? (answer)**

You just need to create an implementation of AbstractHttpMessageConverter and register it using the WebMvcConfigurerAdapter#extendMessageConverters() method with the classes which generate a new type of request/response.

**10. Is REST normally stateless? (**[**answer**](http://javarevisited.blogspot.sg/2015/08/difference-between-soap-and-restfull-webservice-java.html)**)**

Yes, REST API should be stateless because it is based on HTTP which is also stateless. A Request in REST API should contain all the details required it to process i.e. it should not rely on previous or next request or some data maintained at the server end e.g. Sessions. REST specification puts a constraint to make it stateless and you should keep that in mind while designing your REST API.

**11. What does @RequestMapping annotation do? (**[**answer**](http://javarevisited.blogspot.sg/2017/06/how-spring-mvc-framework-works-web-flow.html#axzz55vF5ugU8)**)**

The @RequestMapping annotation is used to map web requests to Spring Controller methods. You can map requests based upon HTTP methods like the GET and POST and various other parameters. For examples, if you are developing RESTful Web Service using Spring then you can use produces and consumes property along with media type annotation to indicate that this method is only used to produce or consumers JSON as shown below:

@RequestMapping (**method** **=** RequestMethod.POST, consumes**=**"application/json")

**public** Book save(@RequestBody Book aBook) {

**return** bookRepository.save(aBook);

}

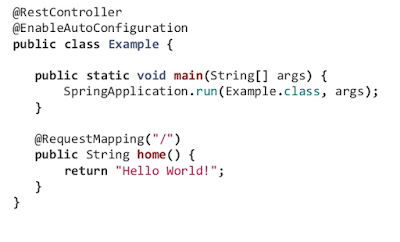
You can similarly create other handler methods to produce JSON or XML. If you are not familiar with these annotations then I suggest you join [**Spring MVC For Beginners**](https://click.linksynergy.com/fs-bin/click?id=JVFxdTr9V80&subid=0&offerid=323058.1&type=10&tmpid=14538&RD_PARM1=https%3A%2F%2Fwww.udemy.com%2Fspring-mvc-tutorial-for-beginners-step-by-step%2F) course on Udemy to learn from scratch.

**12. Is @Controller a stereotype? Is @RestController a stereotype? (**[**answer**](http://javarevisited.blogspot.sg/2017/08/difference-between-restcontroller-and-controller-annotations-spring-mvc-rest.html)**)**

Yes, both @Controller and @RestController are stereotypes. The @Controller is actually a specialization of Spring's @Component stereotype annotation. This means that class annotated with @Controller will also be automatically be detected by Spring container as part of the container's component scanning process.  
  
And, @RestController is a specialization of @Controller for RESTful web service. It not only combines @ResponseBody and @Controller annotation but also gives more meaning to your controller class to clearly indicate that it deals with RESTful requests.  
  
Spring Framework may also use this annotation to provide some more useful features related to REST API development in the future.

**13. What is the difference between @Controller and @RestController?  (**[**answer**](http://javarevisited.blogspot.sg/2017/08/difference-between-restcontroller-and-controller-annotations-spring-mvc-rest.html)**)**

There are many differences between @Controller and @RestController as discussed in my earlier article (see the answer) but the most important one is that with @RestController you get the @ResponseBody annotation automatically, which means you don't need to separately annotate your handler methods with @ResponseBody annotation. This makes the development of RESTful web service easier using Spring. You can see [here](https://javarevisited.blogspot.com/2017/08/difference-between-restcontroller-and-controller-annotations-spring-mvc-rest.html) learn more about the difference between @RestController and @Controller annotations in Spring API.

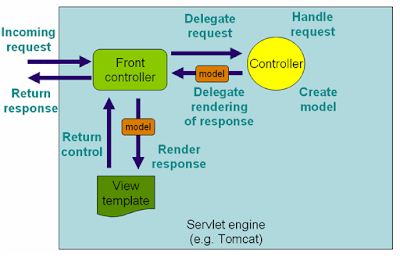
[](http://javarevisited.blogspot.sg/2016/12/top-5-spring-and-hibernate-training-courses-java-jee-programmers.html)

**14. When do you need @ResponseBody annotation in Spring MVC? (**[**answer**](http://javarevisited.blogspot.sg/2018/01/7-reasons-for-using-spring-to-develop-RESTful-web-service.html#axzz55a8rTeu7)**)**

The @ResponseBody annotation can be put on a method to indicates that the return type should be written directly to the HTTP response body (and not placed in a Model, or interpreted as a view name).  
  
For example:  
  
@RequestMapping(path = "/hello", method = RequestMethod.PUT)  
@ResponseBody  
public String helloWorld() {  
   return "Hello World";  
}  
  
Alternatively, you can also use @RestController annotation instead of @Controller annotation. This will remove the need for using @ResponseBody because as discussed in the previous answer, it comes automatically with @RestController annotation.

**15. What does @PathVariable do in Spring MVC? Why it's useful in REST with Spring? (**[**answer**](http://javarevisited.blogspot.sg/2017/10/differences-between-requestparam-and-pathvariable-annotations-spring-mvc.html)**)**

It's one of the useful annotations from Spring MVC which allows you to read values from URI like query parameter. It's particularly useful in case of creating RESTful web service using Spring because in REST resource identifiers are part of URI. This question is normally asked experienced Spring MVC developers e.g. 4 to 6 years of experience.  
  
For example, in the URL http://myapp.com/books/101 if you want to extract 101 the id, then you can use @PathVariable annotation of Spring MVC.  If you are not familiar with Spring MVC annotations then [**Spring MVC For Beginners: Build Java Web App in 25 Steps**](https://click.linksynergy.com/fs-bin/click?id=JVFxdTr9V80&subid=0&offerid=323058.1&type=10&tmpid=14538&RD_PARM1=https%3A%2F%2Fwww.udemy.com%2Fspring-mvc-tutorial-for-beginners-step-by-step%2F) is a good place to start with.

[](https://click.linksynergy.com/fs-bin/click?id=JVFxdTr9V80&subid=0&offerid=323058.1&type=10&tmpid=14538&RD_PARM1=https%3A%2F%2Fwww.udemy.com%2Fspring-mvc-tutorial-for-beginners-step-by-step%2F)

**16. What is the HTTP status return code for a successful DELETE statement? (**[**answer**](http://www.java67.com/2015/09/top-10-restful-web-service-interview-questions-answers.html)**)**

There is no strict rule with respect to what status code your REST API should return after a successful DELETE i.e it can return 200 Ok or 204 No Content. In general, if the DELETE operation is successful and the response body is empty return 204. If the DELETE request is successful and the response body is NOT empty, return 200

**17. What does CRUD mean? (answer)**

CRUD is a short form of Create, Read, Update and Delete. In REST API, the POST is used to create a resource, GET is used to read a resource, [PUT](http://javarevisited.blogspot.sg/2016/10/difference-between-put-and-post-in-restful-web-service.html) is used to updated a resource and DELETE is used to remove a resource from the server. This one is another beginner level Spring MVC questions for 1 to 3 years experienced programmers

**18. Where do you need @EnableWebMVC? (answer)**

The @EnableWebMvc annotation is required to enable Spring MVC when Java configuration is used to configure Spring MVC instead of XML. It is equivalent to <mvc: annotation-driven>  in XML configuration.  
  
It enables support for @Controller-annotated classes that use @RequestMapping to map incoming requests to handler methods not already familiar with Spring's support for Java configuration, [**Spring Master Class**](https://click.linksynergy.com/fs-bin/click?id=JVFxdTr9V80&subid=0&offerid=323058.1&type=10&tmpid=14538&RD_PARM1=https%3A%2F%2Fwww.udemy.com%2Fspring-tutorial-for-beginners%2F) on Udemy is a good place to start.

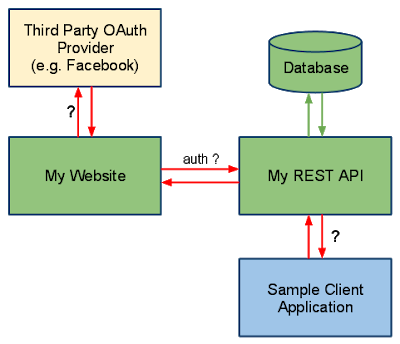
[](https://click.linksynergy.com/fs-bin/click?id=JVFxdTr9V80&subid=0&offerid=323058.1&type=10&tmpid=14538&RD_PARM1=https%3A%2F%2Fwww.udemy.com%2Fspring-tutorial-for-beginners%2F)

**19. When do you need @ResponseStatus annotation in Spring MVC? (**[**answer**](http://javarevisited.blogspot.sg/2018/01/7-reasons-for-using-spring-to-develop-RESTful-web-service.html#axzz55a8rTeu7)**)**

A good questions for 3 to 5 years experienced spring developers. The @ResponseStatus annotation is required during error handling in Spring MVC and REST. Normally when an error or exception is thrown at the server-side, web server returns a blanket HTTP status code 500 - Internal server error.  
  
This may work for a human user but not for REST clients. You need to send them a proper status code like 404 if the resource is not found. That's where you can use @ResponseStatus annotation, which allows you to send custom HTTP status code along with proper error message in case of Exception.  
  
In order to use it, you can create custom exceptions and annotated them using @ResponseStatus annotation and proper HTTP status code and reason.  
  
When such exceptions are thrown from controller's handler methods and not handled anywhere else, then appropriate HTTP response with the proper HTTP status code, which you have set is sent to the client.  
  
For example, if you are writing a RESTful Web Service for a library which provides book information then you can use @ResponseStatus to create Exception which returns HTTP response code 404 when a book is not found instead of Internal Server Error (500), as shown below:  
  
 @ResponseStatus(value=HttpStatus.NOT\_FOUND, reason="No such Book")  // 404  
 public class BookNotFoundException extends RuntimeException {  
     // ...  
 }  
  
If this Exception is thrown from any handler method then HTTP error code 404 with reason "No such Book" will be returned to the client.

**20. Is REST secure? What can you do to secure it? (**[**answer**](http://www.java67.com/2017/04/3-great-books-to-learn-java-web-services-soap-and-restful.html)**)**

This question is mostly asked with experienced Java programmers e.g. 2 to 5 years experience with both REST and Spring. Security is a broad term, it could mean security of message which is provided by encryption or access restriction which is provided using authentication and authorization. REST is normally not secure but you can secure it by using Spring security.  
  
At the very least you can enable HTTP basic authentication by using HTTP in your Spring security configuration file. Similarly, you can expose your REST API using [HTTPS](http://javarevisited.blogspot.sg/2013/07/how-ssl-https-and-certificates-works-in-java-web-application.html) if the underlying server supports HTTPS.

[](http://javarevisited.blogspot.sg/2018/01/how-to-enable-http-basic-authentication-spring-security-java-xml-configuration.html)

**21. Does REST work with transport layer security (TLS)? (**[**answer**](http://javarevisited.blogspot.sg/2012/01/rest-web-services-framework-interview.html)**)**

TLS or Transport Layer Security is used for secure communication between client and server. It is the successor of SSL (Secure Socket Layer). Since HTTPS can work with both SSL and TLS, REST can also work with TLS.  
  
Actually, REST says anything about Security, it's up to the server which implements that. The same RESTful Web Service can be accessed using HTTP and HTTPS if the server supports [SSL](http://javarevisited.blogspot.sg/2013/07/how-to-configure-https-ssl-in-tomcat-6-7-web-server-java.html#axzz56WXxxAC0).  
  
If you are using Tomcat, you can see here to learn more about how to enable SSL in Tomcat.

**22. Do you need Spring MVC in your classpath for developing RESTful Web Service? (**[**answer**](http://javarevisited.blogspot.sg/2017/01/where-and-how-to-download-spring-JAR-Files-Spring4-without-Maven-Gradle.html#axzz4pp42TeHu)**)**

This question is often asked Java programmers with 1 to 2 years of experience in Spring. The short answer is Yes, you need Spring MVC in your Java application's classpath to develop RESTful web services using the Spring framework. It's actually Spring MVC which provides all useful annotations e.g. @RestController, @ResponseCode, @ResponseBody, @RequestBody, and @PathVariable, hence you must spring-mvc.jar or appropriate Maven entry in your pom.xml  
  
  
Read more: <https://javarevisited.blogspot.com/2018/02/top-20-spring-rest-interview-questions-answers-java.html#ixzz6QeEecKn6>