PA165 - Lab session – RESTful Webservices 25.11.2014

Goals
a) set-up a RESTful webservice in Spring, b) look at the differences with Jersey (JAX-RS reference implementation), as a side-goal c) look also at Spring-boot for Spring configuration.

Prerequisites
Netbeans 7.3.x, Tomcat 7, Java 7, Maven 3

Scenario
There is an old RESTful application written with the support of Jersey (JAX-RS) that allows to manage Customers resources. Your goal is to rewrite the REST methods and port it to Spring, possibly improving the RESTful design.

Applications
In the IS you can find two applications:

- **PA165-Fall2014-Seminar11_JerseyREST.zip**: this is the application you are needed to port -> *No need to modify it, you can look at the REST methods and observe how it behaves*;
- **PA165-Fall2014-Seminar11_Spring-REST-Start.zip**: this is a pre-configured Spring application that already contains most that is needed -> *you can use it at as a base for the tasks*;

Task 1
Check the original Jersey application (from **PA165-Fall2014-Seminar11_JerseyREST.zip**). Look at how Jersey has been configured and how the REST methods work.
You should be able to import the project in Netbeans and run from it. Alternatively you can compile & run it from the command line from the root of the project:

```
module add maven-3.0.5
mvn clean install && mvn tomcat7:run
```

then check that methods are accessible http://localhost:8080/JerseyREST/customers.

You can use curl to interrogate the REST methods (just some sample commands you can use):

GET
curl -i -X GET http://localhost:8080/JerseyREST/customers

DELETE
curl -i -X DELETE http://localhost:8080/JerseyREST/customers/1

POST
curl -X POST -i -H "Content-Type: application/json" --data '{"id":"99","invention":"mass production","name":"Henry","occupation":"Industrialist","surname":"Ford"}' http://localhost:8080/JerseyREST/customers

PUT

curl -X PUT -i -H "Content-Type: application/json" --data '{"id":"99","invention":"mass production","name":"Henry","occupation":"Industrialist","surname":"Ford"}' http://localhost:8080/JerseyREST/customers/99

Task 2

Open the Spring project (from **PA165-Fall2014-Seminar11_Spring-REST-Start.zip**). You should be able to compile and run it from Netbeans. If you need to run it from command line, you can do so with

```
module add maven-3.0.5      (if not done previously in the same shell)
mvn clean install && mvn tomcat7:run
```

You do not have yet a REST controller, but once you have added it, you should be able to access it from http://localhost:8080/spring-rest/ plus the mapping - for the moment you should see a page reporting that we do not have any explicit mapping for /error, this is fine since we do not need views in this project.

Take a look at the project: both Spring-boot configuration and the classes. In particular, most of your changes will be done in `cz.muni.fi.pa165.rest.controllers.CustomerController` class.

Task 3

Start following the TODOs for implementation of the REST methods in `cz.muni.fi.pa165.rest.controllers.CustomerController` class. You can also implement exception handling/code return messages according to what seen during lecture. You can use curl to invoke the different methods.

Task 4

If you complete successfully, follow the tutorial at the page http://spring.io/guides/gs/consuming-rest/ to write a REST client and adapt it to interrogate the RESTful application developed.

Task 5 (extra)

Add Integration testing for your REST application: ideally this should have been the first step before rewriting the original application. Follow templates from http://spring.io/guides/tutorials/bookmarks/#_testing_a_rest_service

Additional References

- Useful information for implementing a HATEOAS REST Service and for Securing a REST Service can be found in the Spring guides: http://spring.io/guides/tutorials/bookmarks/#_testing_a_rest_service
- For useful information about expected behaviour of REST methods, refer to *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content* https://www.rfc-editor.org/rfc/rfc7231.txt
- If interested in using Jersey for your project, you can see last year's seminar session for additional information about Jersey (e.g. how to secure REST methods): https://kore.fi.muni.cz/wiki/index.php/PA165/Lab_session_Webservices_REST