# Lab 3 – JAX-RS

## JAX-RS Server Configuration

* ~~Choose JEE perspective.~~
* ~~New – Dynamic Web Project.~~
  + ~~Target Runtime – accept default~~
  + ~~Configuration - accept default~~
  + ~~Choose ‘Next’~~
  + **~~Generate web.xml deployment descriptor - Yes~~**
* ~~Copy jersey-bundle-1.15 into lib folder and add to your BuildPath.~~
* ~~Copy into WEB-INF/lib folder~~
  + ~~jersey-bundle-1.15~~
  + ~~asm-3.3.1.jar~~
* ~~Modify web.xml (Add the blue section below)~~

**Web.xml:**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<web-app xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns=*"http://java.sun.com/xml/ns/javaee"* xmlns:web=*"http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"* xsi:schemaLocation=*"http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"* id=*"WebApp\_ID"* version=*"3.0"*>

<!-- more stuff here –>

<servlet>

<servlet-name>Jersey REST Service</servlet-name>

<servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>

<init-param>

<param-name>com.sun.jersey.config.property.packages</param-name>

<param-value>myApp</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>Jersey REST Service</servlet-name>

<url-pattern>/rest/\*</url-pattern>

</servlet-mapping>

</web-app>

## ~~Q1.~~

~~Type in the first ‘Hello’ example from the notes and run on Apache Tomcat 8.0~~

## ~~Q2. (Chrome Postman)~~

~~Make Http Get request using Postman.~~

* ~~Install Chrome Postman~~
* ~~Make Html Request~~
* ~~Make XML Request~~
* ~~Make Plain Text Request~~

## Q3. (JAX-RS Books Server)

Create a Dynamic Web Project called j2BookServer.

1. ~~Copy the web.xml from the first project.~~
2. ~~Create a package called myApp containing the following classes.~~
3. ~~Create a class called Book with attributes id, title, author, year. Generate getters and setters for the attributes. Add JAXB annotations.~~
4. ~~Create a class called BookDAO as given in the notes. You need to finish the constructor. Create a second Book object there.~~
5. ~~Create a class called BookResource.~~
6. ~~Run the application on the apache tomcat server.~~
7. ~~In the browser send a request for~~ 
   * ~~http://localhost:8080/j2bookServer/rest/books~~

## ~~Q4.~~

1. ~~Update the server application by adding the method getBook() as outlined in the notes.~~
2. ~~Test with a browser~~