« RESTFUL service »

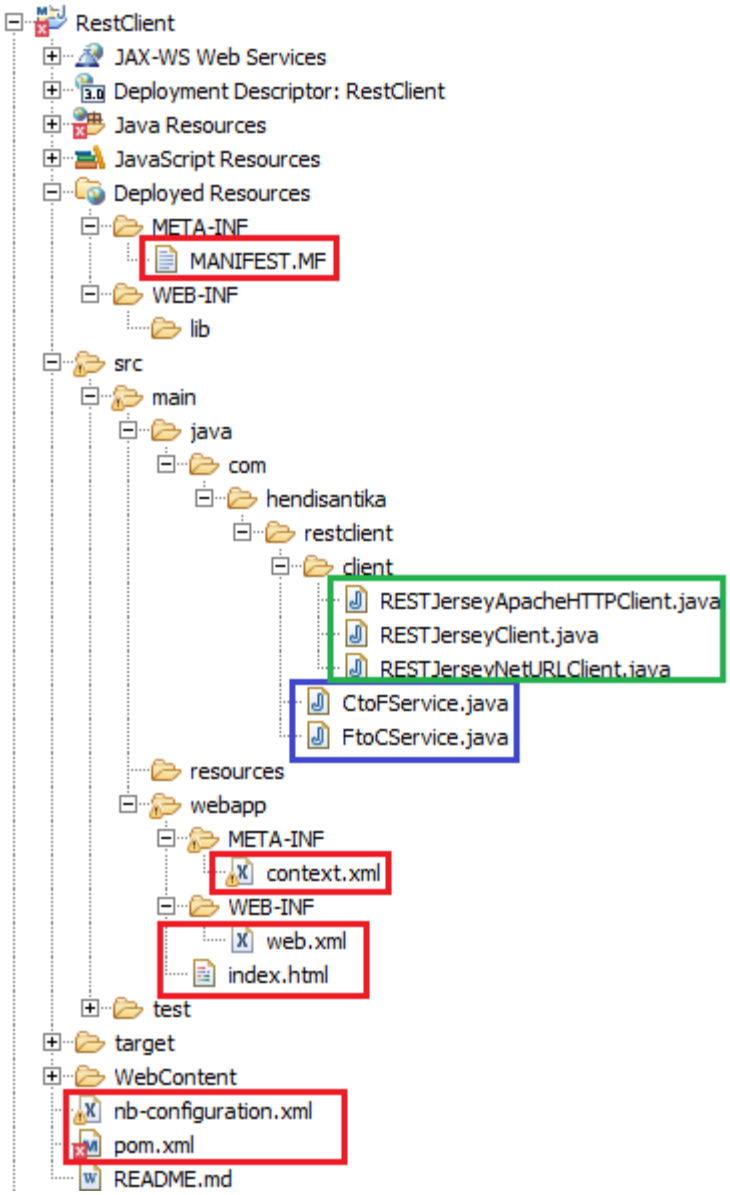
11/04/2018

By : REGRAG mouhcine

Table des matières

1- Arborescence du projet :

La structure de base d'un projet RESTFUL est comme suit :

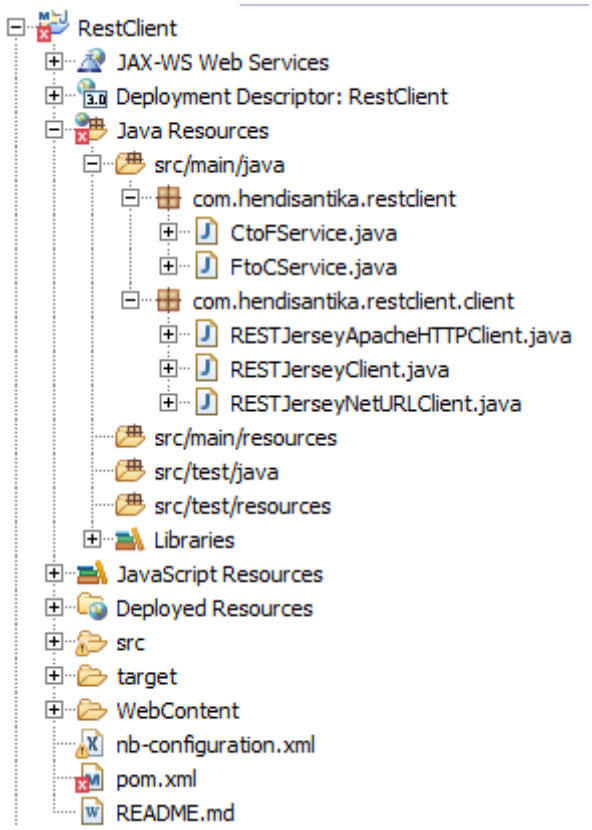


On trouve les fichiers suivants :

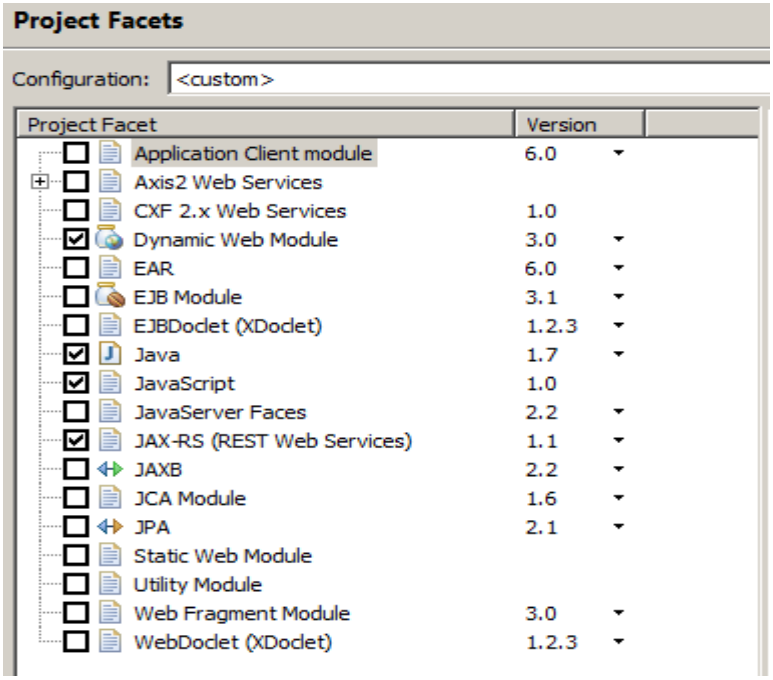
- Pour la configuration : web.xml, context.xml, nb-configuration.xml, MANIFEST.MF.
- Pour le service : CtoFService.java, FtoCSevice.java.
- Pour l'appel de service : RESTJerseyApacheHTTPClient.java, RESTJerseyClient.java, RESTJerseyNetURLClient.java.
- Pour appeler le service à partir de lien d'une page : index.html

Il faut respecter cette organisation, en créant les deux package :

com.hendisantika.restclient.client et com.hendisantika.restclient comme suit :

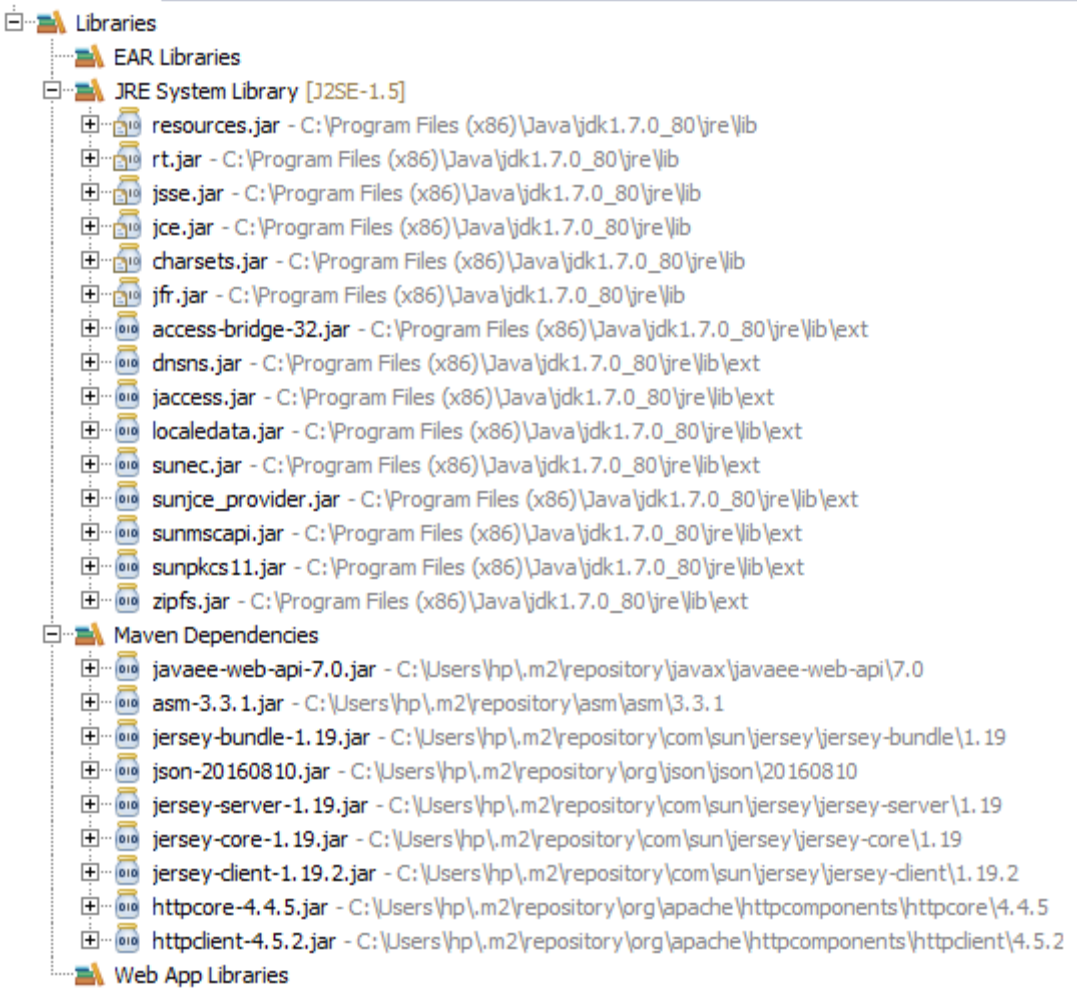


Activez dans les propriétés du projet → projet facets : JAX-RS (REST Web Services) comme indiqué en bas :



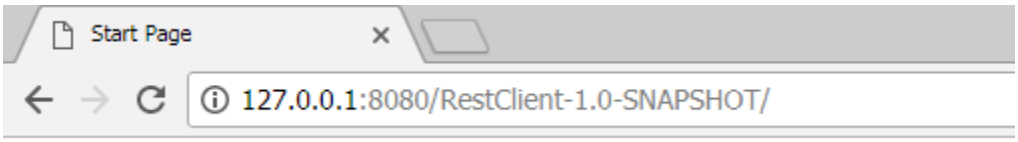
2- Dépendances en API :

Il faut bien sur ajouter l'ensemble des API comme indiqué en bas :



3- Exécution de l'exemple :

a- L'exemple consiste à réaliser un web service et pour pouvoir afficher le résultat dans une Page web par billet d'angulairejs, voir en bas :

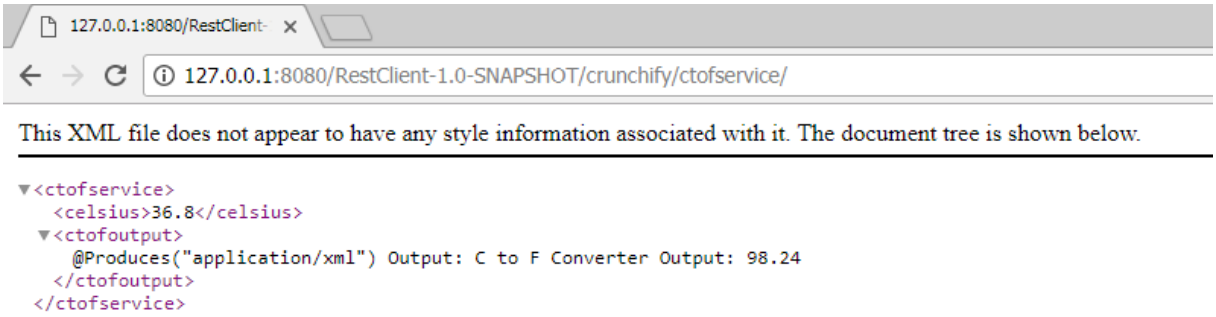


Hello World!

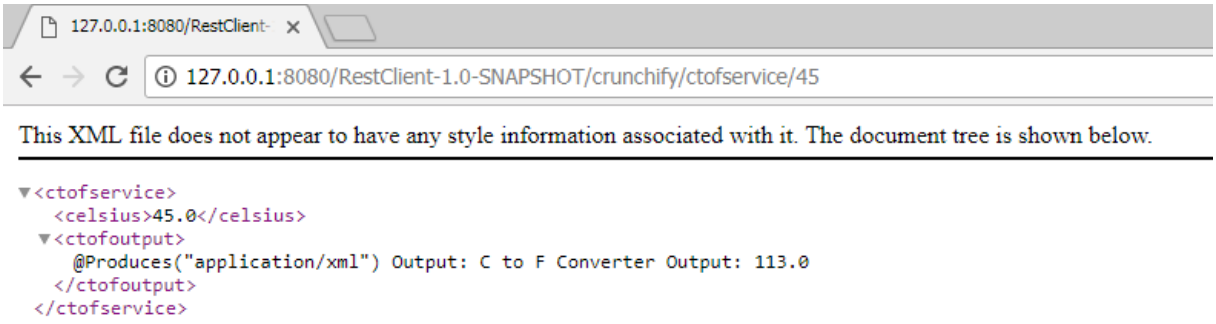
Json Response
Xml Response

Les deux liens sont tout simplement des appels à nos deux services web.

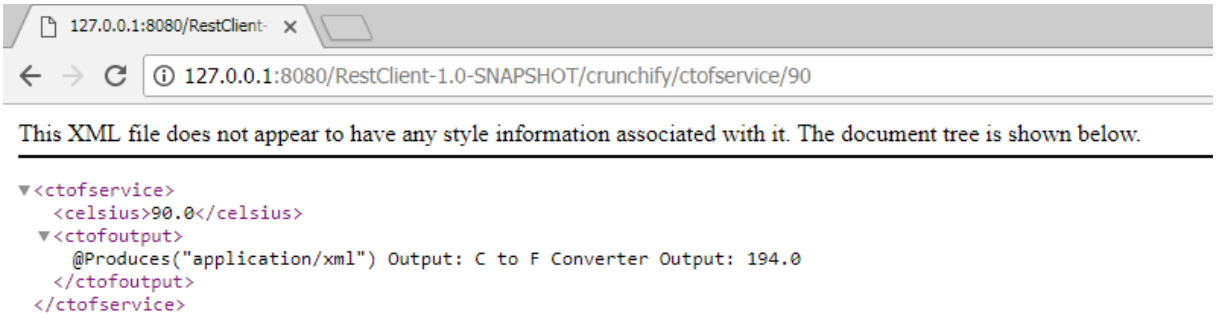
b- Le premier lien Json Reponse donne ce qui suit:



c- Le deuxième lien Xml Response donne ce qui suit :



On peut passer en paramètre une autre valeur voir en bas : avec 90



4- Fichiers sources :

1- CtoFService.java :

```
package com.hendisantika.restclient;

import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;

/**
 *
 * @author hendisantika
 * http://crunchify.com/how-to-build-restful-service-with-java-using-jax-rs-and-
jersey/
 */
@Path("/ctofservice")
public class CtoFService {

    @GET
    @Produces("application/xml")
    public String convertCtoF() {

        Double fahrenheit;
        Double celsius = 36.8;
        fahrenheit = ((celsius * 9) / 5) + 32;

        String result = "@Produces(\"application/xml\") Output: \n\nC to F
Converter Output: \n\n" + fahrenheit;
        return "<ctofservice>" + "<celsius>" + celsius + "</celsius>" +
"<ctofoutput>" + result + "</ctofoutput>" + "</ctofservice>";
    }

    @Path("{c}")
    @GET
    @Produces("application/xml")
    public String convertCtoFfromInput(@PathParam("c") Double c) {
        Double fahrenheit;
        Double celsius = c;
        fahrenheit = ((celsius * 9) / 5) + 32;

        String result = "@Produces(\"application/xml\") Output: \n\nC to F
Converter Output: \n\n" + fahrenheit;
        return "<ctofservice>" + "<celsius>" + celsius + "</celsius>" +
"<ctofoutput>" + result + "</ctofoutput>" + "</ctofservice>";
    }
}
```

2- FtoCService.java :

```
package com.hendisantika.restclient;

import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Response;
import org.json.JSONException;
import org.json.JSONObject;

/**
 *
 * @author hendisantika
 */
@Path("/ftocservice")
public class FtoCService {

    @GET
    @Produces("application/json")
    public Response convertFtoC() throws JSONException {

        JSONObject jsonObject = new JSONObject();
        Double fahrenheit = 98.24;
        Double celsius;
        celsius = (fahrenheit - 32) * 5 / 9;
        jsonObject.put("F Value", fahrenheit);
        jsonObject.put("C Value", celsius);

        String result = "@Produces(\"application/json\") Output: \n\nF to C
Converter Output: \n\n" + jsonObject;
        return Response.status(200).entity(result).build();
    }

    @Path("{f}")
    @GET
    @Produces("application/json")
    public Response convertFtoCfromInput(@PathParam("f") float f) throws
JSONException {

        JSONObject jsonObject = new JSONObject();
        float celsius;
        celsius = (f - 32) * 5 / 9;
        jsonObject.put("F Value", f);
        jsonObject.put("C Value", celsius);

        String result = "@Produces(\"application/json\") Output: \n\nF to C
Converter Output: \n\n" + jsonObject;
        return Response.status(200).entity(result).build();
    }
}
```

3- RESTJerseyApacheHTTPClient.java :

```
package com.hendisantika.restclient.client;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.HttpClientBuilder;

/**
 *
 * @author hendisantika
 * http://crunchify.com/how-to-create-restful-java-client-using-apache-httpclient-
example/
 */
public class RESTJerseyApacheHTTPClient {

    public static void main(String[] args) {

        try {

            // create HTTP Client
            HttpClient httpClient = HttpClientBuilder.create().build();

            // Create new getRequest with below mentioned URL
            HttpGet getRequest = new HttpGet("http://localhost:8080/RestClient-
1.0-SNAPSHOT/crunchify/ctofservice/");

            // Add additional header to getRequest which accepts application/xml
data
            getRequest.addHeader("accept", "application/xml");

            // Execute your request and catch response
            HttpResponse response = httpClient.execute(getRequest);

            // Check for HTTP response code: 200 = success
            if (response.getStatusLine().getStatusCode() != 200) {
                throw new RuntimeException("Failed : HTTP error code : " +
response.getStatusLine().getStatusCode());
            }

            // Get-Capture Complete application/xml body response
            BufferedReader br = new BufferedReader(new
InputStreamReader((response.getEntity().getContent())));
            String output;
            System.out.println("============Output:============");

            // Simply iterate through XML response and show on console.
            while ((output = br.readLine()) != null) {
                System.out.println(output);
            }

        } catch (ClientProtocolException e) {
            e.printStackTrace();
```

```
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

4- RESTJerseyClient.js :

```
package com.hendisantika.restclient.client;

import com.sun.jersey.api.client.Client;
import com.sun.jersey.api.client.ClientResponse;
import com.sun.jersey.api.client.WebResource;

/**
 *
 * @author hendisantika
 */
public class RESTJerseyClient {

    public static void main(String[] args) {

        RESTJerseyClient client = new RESTJerseyClient();
        client.getCtoFResponse();
        client.getFtoCResponse();
    }

    private void getFtoCResponse() {
        try {

            Client client = Client.create();
            WebResource webResource2 =
client.resource("http://localhost:8080/RestClient-1.0-
SNAPSHOT/crunchify/ftocservice/90");
            ClientResponse response2 =
webResource2.accept("application/json").get(ClientResponse.class);
            if (response2.getStatus() != 200) {
                throw new RuntimeException("Failed : HTTP error code : " +
response2.getStatus());
            }

            String output2 = response2.getEntity(String.class);
            System.out.println("\n============getFtoCResponse============");
            System.out.println(output2);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private void getCtoFResponse() {
        try {

            Client client = Client.create();
            WebResource webResource =
client.resource("http://localhost:8080/RestClient-1.0-
SNAPSHOT/crunchify/ctofservice/40");
            ClientResponse response =
webResource.accept("application/xml").get(ClientResponse.class);
            if (response.getStatus() != 200) {
                throw new RuntimeException("Failed : HTTP error code : " +
response.getStatus());
            }

            String output = response.getEntity(String.class);
```

```
            System.out.println("============getCtoFResponse============");
            System.out.println(output);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

5- RESTJerseyNetURLClient.java :

```
package com.hendisantika.restclient.client;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;
import java.nio.charset.Charset;

/**
 *
 * @author hendisantika
 */
public class RESTJerseyNetURLClient {

    public static void main(String[] args) {
        System.out.println("\n============Output:============ \n"
                + callURL("http://localhost:8080/RestClient-1.0-
SNAPSHOT/crunchify/ctofservice/"));
    }

    public static String callURL(String myURL) {
        System.out.println("Requested URL: " + myURL);
        StringBuilder sb = new StringBuilder();
        URLConnection urlConn = null;
        InputStreamReader in = null;
        try {
            URL url = new URL(myURL);
            urlConn = url.openConnection();
            if (urlConn != null) {
                urlConn.setReadTimeout(60 * 1000);
            }
            if (urlConn != null && urlConn.getInputStream() != null) {
                in = new InputStreamReader(urlConn.getInputStream(),
Charset.defaultCharset());
                BufferedReader bufferedReader = new BufferedReader(in);
                if (bufferedReader != null) {
                    int cp;
                    while ((cp = bufferedReader.read()) != -1) {
                        sb.append((char) cp);
                    }
                    bufferedReader.close();
                }
            }
            in.close();
        } catch (Exception e) {
            throw new RuntimeException("Exception while calling URL:" + myURL, e);
        }

        return sb.toString();
    }
}
```

6- context.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
<Context path="/RestClient"/>
```

7- web.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
To change this license header, choose License Headers in Project Properties.
To change this template file, choose Tools | Templates
and open the template in the editor.
-->

<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"
         version="3.0">
    <display-name>CrunchifyRESTJerseyExample</display-name>
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
        <welcome-file>index.htm</welcome-file>
        <welcome-file>index.jsp</welcome-file>
        <welcome-file>default.html</welcome-file>
        <welcome-file>default.htm</welcome-file>
        <welcome-file>default.jsp</welcome-file>
    </welcome-file-list>

    <servlet>
        <servlet-name>Jersey Web Application</servlet-name>
        <servlet-
class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>Jersey Web Application</servlet-name>
        <url-pattern>/crunchify/*</url-pattern>
    </servlet-mapping>
</web-app>
```

8- index.html :

```
<!DOCTYPE html>
<html>
    <head>
        <title>Start Page</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    </head>
    <body>
    <h1>Hello World!</h1>
    <a href="../RestClient-1.0-SNAPSHOT/crunchify/ctofservice/">Json Response</a>
      <br />
      <a href="../RestClient-1.0-SNAPSHOT/crunchify/ctofservice/45">Xml
Response</a>
    </body>
</html>
```

9- nb-configuration.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project-shared-configuration>
    <!--
This file contains additional configuration written by modules in the NetBeans
IDE.
The configuration is intended to be shared among all the users of project and
therefore it is assumed to be part of version control checkout.
Without this configuration present, some functionality in the IDE may be limited
or fail altogether.
-->
    <properties xmlns="http://www.netbeans.org/ns/maven-properties-data/1">
        <!--
Properties that influence various parts of the IDE, especially code formatting and
the like.
You can copy and paste the single properties, into the pom.xml file and the IDE
will pick them up.
That way multiple projects can share the same settings (useful for formatting
rules for example).
Any value defined here will override the pom.xml file value but is only applicable
to the current project.
-->
        <org-netbeans-modules-maven-j2ee.netbeans_2e_hint_2e_j2eeVersion>1.7-
web</org-netbeans-modules-maven-j2ee.netbeans_2e_hint_2e_j2eeVersion>
        <org-netbeans-modules-maven-
j2ee.netbeans_2e_hint_2e_deploy_2e_server>Tomcat</org-netbeans-modules-maven-
j2ee.netbeans_2e_hint_2e_deploy_2e_server>
        <org-netbeans-modules-maven-jaxws.rest_2e_config_2e_type>ide</org-
netbeans-modules-maven-jaxws.rest_2e_config_2e_type>
    </properties>
</project-shared-configuration>
```

10- pom.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.hendisantika</groupId>
    <artifactId>RestClient</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>war</packaging>

    <name>RestClient</name>

    <properties>
        <endorsed.dir>${project.build.directory}/endorsed</endorsed.dir>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>

    <dependencies>
        <dependency>
            <groupId>javax</groupId>
            <artifactId>javaee-web-api</artifactId>
            <version>7.0</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>asm</groupId>
            <artifactId>asm</artifactId>
            <version>3.3.1</version>
        </dependency>
        <dependency>
            <groupId>com.sun.jersey</groupId>
            <artifactId>jersey-bundle</artifactId>
            <version>1.19</version>
        </dependency>
        <dependency>
            <groupId>org.json</groupId>
            <artifactId>json</artifactId>
            <version>20160810</version>
        </dependency>
        <dependency>
            <groupId>com.sun.jersey</groupId>
            <artifactId>jersey-server</artifactId>
            <version>1.19</version>
        </dependency>
        <dependency>
            <groupId>com.sun.jersey</groupId>
            <artifactId>jersey-core</artifactId>
            <version>1.19</version>
        </dependency>
        <dependency>
            <groupId>com.sun.jersey</groupId>
            <artifactId>jersey-client</artifactId>
            <version>1.19.2</version>
        </dependency>
        <dependency>
```

```
        <groupId>org.apache.httpcomponents</groupId>
        <artifactId>httpcore</artifactId>
        <version>4.4.5</version>
    </dependency>
    <dependency>
        <groupId>org.apache.httpcomponents</groupId>
        <artifactId>httpclient</artifactId>
        <version>4.5.2</version>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.1</version>
            <configuration>
                <source>1.7</source>
                <target>1.7</target>
                <compilerArguments>
                    <endorseddirs>${endorsed.dir}</endorseddirs>
                </compilerArguments>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-war-plugin</artifactId>
            <version>2.3</version>
            <configuration>
                <failOnMissingWebXml>false</failOnMissingWebXml>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-dependency-plugin</artifactId>
            <version>2.6</version>
            <executions>
                <execution>
                    <phase>validate</phase>
                    <goals>
                        <goal>copy</goal>
                    </goals>
                    <configuration>
                        <outputDirectory>${endorsed.dir}</outputDirectory>
                        <silent>true</silent>
                        <artifactItems>
                            <artifactItem>
                                <groupId>javax</groupId>
                                <artifactId>javaee-endorsed-api</artifactId>
                                <version>7.0</version>
                                <type>jar</type>
                            </artifactItem>
                        </artifactItems>
                    </configuration>
                </execution>
            </executions>
        </plugin>
    </plugins>
```

```
    </build>
</project>
```

11- MANIFEST.MF :

```
Manifest-Version: 1.0
Class-Path:
```