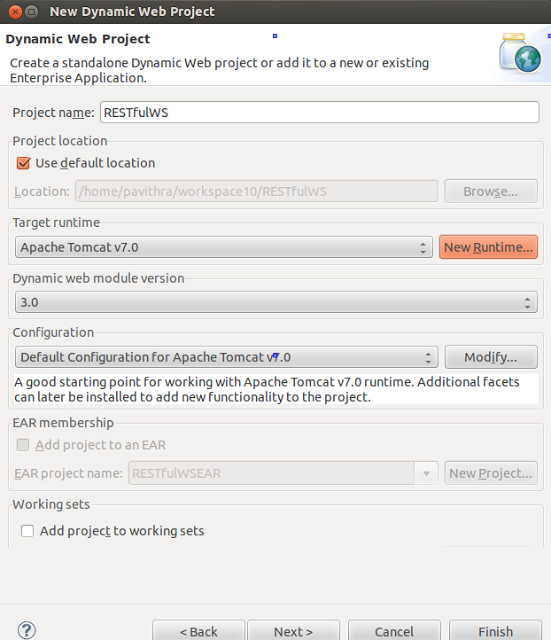
<http://java.dzone.com/articles/restful-web-services-java>

REST is an architectural style. HTTP is a protocol which contains the set of REST architectural constraints.  
  
**REST fundamentals**

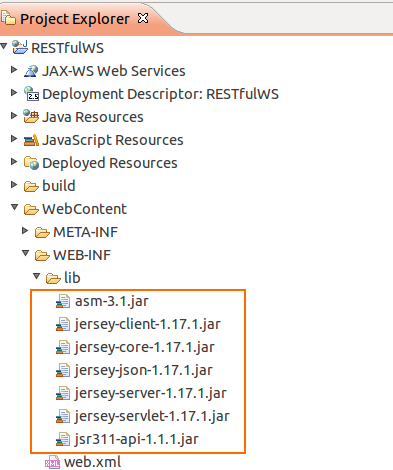
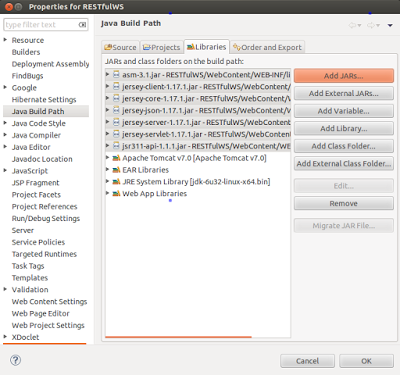
* Everything in REST is considered as a resource.
* Every resource is identified by an URI.
* Uses uniform interfaces. Resources are handled using POST, GET, PUT, DELETE operations which are similar to Create, Read, update and Delete(CRUD) operations.
* Be stateless. Every request is an independent request. Each request from client to server must contain all the information necessary to understand the request.
* Communications are done via representations. E.g. XML, JSON

**RESTful Web Services**  
  
RESTful Web Services have embraced by large service providers across the web as an alternative to SOAP based Web Services due to its simplicity. This post will demonstrate how to create a RESTful Web Service and client using Jersey framework which extends JAX-RS API. Examples are done using Eclipse IDE and Java SE 6.  
  
**Creating RESTful Web Service**

* In Eclipse, create a new dynamic web project called "RESTfulWS"

[](http://2.bp.blogspot.com/-5pum1LUoe7c/UnwPR8gIENI/AAAAAAAAAw0/jTbstCe73O8/s1600/Screenshot+from+2013-11-07+17:06:05.png)

* Download Jersey zip bundle from [here](https://jersey.java.net/download.html). Jersey version used in these examples is 1.17.1. Once you unzip it you'll have a directory called "jersey-archive-1.17.1". Inside it find the lib directory. Copy following jars from there and paste them inside WEB-INF -> lib folder in your project. Once you've done that, add those jars to your project build path as well.
  1. asm-3.1.jar
  2. jersey-client-1.17.1.jar
  3. jersey-core-1.17.1.jar
  4. jersey-server-1.17.1.jar
  5. jersey-servlet-1.17.1.jar
  6. jsr311-api-1.1.1.jar

[](http://2.bp.blogspot.com/-RKIR9umcmJ4/UnwPulV_jvI/AAAAAAAAAxM/HSqsW0H6x1g/s1600/Screenshot+from+2013-11-08+02:54:15.png) [](http://1.bp.blogspot.com/-JipViztlRwc/UnwPfgdmQhI/AAAAAAAAAxE/kSWM0ODHcDY/s1600/Screenshot+from+2013-11-07+17:17:36.png)

* In your project, inside Java Resources -> src create a new package called "com.eviac.blog.restws". Inside it create a new java class called "UserInfo". Also include the given web.xml file inside WEB-INF folder.

**UserInfo.java**

[view source](http://java.dzone.com/articles/restful-web-services-java#viewSource)

[print](http://java.dzone.com/articles/restful-web-services-java#printSource)[?](http://java.dzone.com/articles/restful-web-services-java#about)

01.

02.package com.eviac.blog.restws;

03.

04.

05.

06.

07.import javax.ws.rs.GET;

08.import javax.ws.rs.Path;

09.import javax.ws.rs.PathParam;

10.import javax.ws.rs.Produces;

11.import javax.ws.rs.core.MediaType;

12.

13.

14.

15.

16./\*\*

17.\*

18.\* @author pavithra

19.\*

20.\*/

21.

22.

23.

24.

25.// @Path here defines class level path. Identifies the URI path that

26.// a resource class will serve requests for.

27.@Path("UserInfoService")

28.public class UserInfo {

29.

30.

31.

32.

33.// @GET here defines, this method will method will process HTTP GET

34.// requests.

35.@GET

36.// @Path here defines method level path. Identifies the URI path that a

37.// resource class method will serve requests for.

38.@Path("/name/{i}")

39.// @Produces here defines the media type(s) that the methods

40.// of a resource class can produce.

41.@Produces(MediaType.TEXT\_XML)

42.// @PathParam injects the value of URI parameter that defined in @Path

43.// expression, into the method.

44.public String userName(@PathParam("i") String i) {

45.

46.

47.

48.

49.String name = i;

50.return "<User>" + "<Name>" + name + "</Name>" + "</User>";

51.}

52.

53.

54.

55.

56.@GET

57.@Path("/age/{j}")

58.@Produces(MediaType.TEXT\_XML)

59.public String userAge(@PathParam("j") int j) {

60.

61.

62.

63.

64.int age = j;

65.return "<User>" + "<Age>" + age + "</Age>" + "</User>";

66.}

67.}

**web.xml**

[view source](http://java.dzone.com/articles/restful-web-services-java#viewSource)

[print](http://java.dzone.com/articles/restful-web-services-java#printSource)[?](http://java.dzone.com/articles/restful-web-services-java#about)

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

02.<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> <a href="<http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd>"><http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd>"</a> id="WebApp\_ID" version="2.5">

03.<display-name>RESTfulWS</display-name>

04.<servlet>

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

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

07.<init-param>

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

09.<param-value>com.eviac.blog.restws</param-value>

10.</init-param>

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

12.</servlet>

13.<servlet-mapping>

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

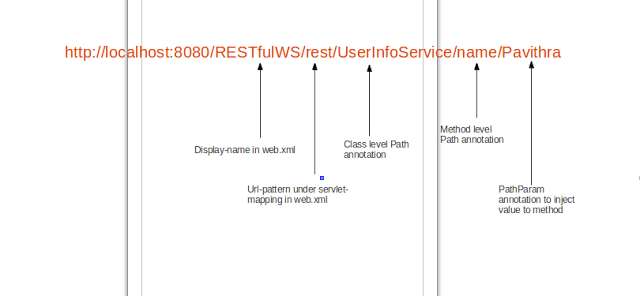
15.<url-pattern>/rest/\*</url-pattern>

16.</servlet-mapping>

17.</web-app>

To run the project, right click on it and click on run as ->run on server.

* Execute the following URL in your browser and you'll see the output.[view plainprint?](http://blog.eviac.com/2013/11/restful-web-services-with-java.html)
  1. <http://localhost:8080/RESTfulWS/rest/UserInfoService/name/Pavithra>

[](http://1.bp.blogspot.com/-l9FQ4eFHRNU/UnwR3Y4EA8I/AAAAAAAAAyE/Ti1iCqpXZCI/s1600/Screenshot+from+2013-11-08+02:04:07.png)**output**[](http://4.bp.blogspot.com/-1C2CHUwDzFg/UnwRkrGyIDI/AAAAAAAAAx8/Vq_5zZlcdus/s1600/Screenshot+from+2013-11-08+03:09:57.png)

**Creating Client**

* Create a package called "com.eviac.blog.restclient". Inside it create a java class called "UserInfoClient".

**UserInfoClient.java**

[view source](http://java.dzone.com/articles/restful-web-services-java#viewSource)

[print](http://java.dzone.com/articles/restful-web-services-java#printSource)[?](http://java.dzone.com/articles/restful-web-services-java#about)

001.package com.eviac.blog.restclient;

002.

003.

004.

005.

006.import javax.ws.rs.core.MediaType;

007.

008.

009.

010.

011.import com.sun.jersey.api.client.Client;

012.import com.sun.jersey.api.client.ClientResponse;

013.import com.sun.jersey.api.client.WebResource;

014.import com.sun.jersey.api.client.config.ClientConfig;

015.import com.sun.jersey.api.client.config.DefaultClientConfig;

016.

017.

018.

019.

020./\*\*

021.\*

022.\* @author pavithra

023.\*

024.\*/

025.public class UserInfoClient {

026.

027.

028.

029.

030.public static final String BASE\_URI = "[http://localhost](http://localhost/):8080/RESTfulWS";

031.public static final String PATH\_NAME = "/UserInfoService/name/";

032.public static final String PATH\_AGE = "/UserInfoService/age/";

033.

034.

035.

036.

037.public static void main(String[] args) {

038.

039.

040.

041.

042.String name = "Pavithra";

043.int age = 25;

044.

045.

046.

047.

048.ClientConfig config = new DefaultClientConfig();

049.Client client = Client.create(config);

050.WebResource resource = client.resource(BASE\_URI);

051.

052.

053.

054.

055.WebResource nameResource = resource.path("rest").path(PATH\_NAME + name);

056.System.out.println("Client Response \n"

057.+ getClientResponse(nameResource));

058.System.out.println("Response \n" + getResponse(nameResource) + "\n\n");

059.

060.

061.

062.

063.WebResource ageResource = resource.path("rest").path(PATH\_AGE + age);

064.System.out.println("Client Response \n"

065.+ getClientResponse(ageResource));

066.System.out.println("Response \n" + getResponse(ageResource));

067.}

068.

069.

070.

071.

072./\*\*

073.\* Returns client response.

074.\* e.g :

075.\* GET [http://localhost](http://localhost/):8080/RESTfulWS/rest/UserInfoService/name/Pavithra

076.\* returned a response status of 200 OK

077.\*

078.\* @param service

079.\* @return

080.\*/

081.private static String getClientResponse(WebResource resource) {

082.return resource.accept(MediaType.TEXT\_XML).get(ClientResponse.class)

083..toString();

084.}

085.

086.

087.

088.

089./\*\*

090.\* Returns the response as XML

091.\* e.g : <User><Name>Pavithra</Name></User>

092.\*

093.\* @param service

094.\* @return

095.\*/

096.private static String getResponse(WebResource resource) {

097.return resource.accept(MediaType.TEXT\_XML).get(String.class);

098.}

099.}

* Once you run the client program, you'll get following output.

[view source](http://java.dzone.com/articles/restful-web-services-java#viewSource)

[print](http://java.dzone.com/articles/restful-web-services-java#printSource)[?](http://java.dzone.com/articles/restful-web-services-java#about)

01.Client Response

02.GET [http://localhost](http://localhost/):8080/RESTfulWS/rest/UserInfoService/name/Pavithra returned a response status of 200 OK

03.Response

04.<User><Name>Pavithra</Name></User>

05.

06.

07.

08.

09.

10.

11.

12.

13.Client Response

14.GET [http://localhost](http://localhost/):8080/RESTfulWS/rest/UserInfoService/age/25 returned a response status of 200 OK

15.Response

16.<User><Age>25</Age></User>