

Programming World

All the stuffs related to Programming ...

MONDAY, AUGUST 12, 2013

Java WebService Tutorial - Part 11 (Writing Simple REST WebService using Netbeans & GlassFish)

In this part we will see how to write a Simple REST Style Web Service using NetBeans and GlassFish Server Server and we will test the service.

Requirements:

1. NetBeans ID
2. GlassFish Server.

Usually NetBeans comes along with the GlassFish Server and also in build support for generating REST Services using the Jersey Framework.

So , once we have installed the NetBeans ID and GlassFish we can create our first webservice now.

Note : This is a quick guide of creating REST using Jersey Framework, we are not going in detail about the annotations we are using in the REST Webservice , we will discuss the basic annotation right now ,in later part we will cover in detail about each annotations with respective example.

Step 1:

Open NetBeans ID.

File-->New ---> New Project --> Java Web-->Web Application and click the **"Next" Button.**

Give a name to the Project say **"MyFirstREST"** (I am using this name here for this example) and click the **"Next"** Button.

Select the Server as **"GlassFish Server"** and select the J2EE Version as **"Java EE 5"** click the **"Finish"** Button.

So , now our project folders are created .

About Me

SEARCH THIS BLOG

Search

TOTAL PAGEVIEWS

BLOG ARCHIVE

▼ 2013 (19)

► February (1)

► July (11)

▼ August (7)

[Java WebService Tutorial - Part 09 \(Writing a Sim...](#)

[Java WebService Tutorial - Part 10 \(REST WebServi...](#)

[Java WebService Tutorial - Part 11 \(Writing Simple...](#)

[Java WebService Tutorial - Part 12 \(Useful REST A...](#)

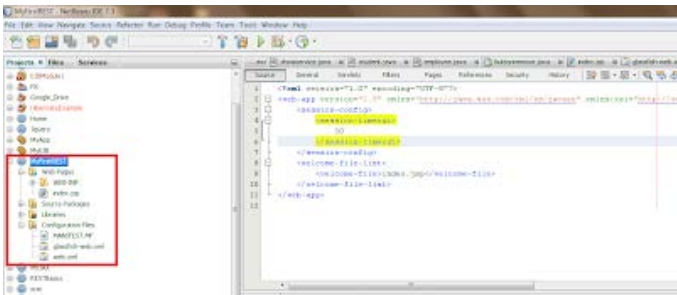
[A Simple Swing Application to Load Multiple Langua...](#)

[Java WebService Tutorial - Part 13 \(XML & JSON \)...](#)

[Calling / Accessing an EJB Stateless Bean Remotel...](#)

► 2015 (1)

FOLLOWERS



Step 2:

Creating Package.

Right click on the Project Node "MyFirstREST" and select :

New ----> Java Package and give name as "com.test" (You can provide your own name for this example i am using this name)

Now we have a package where all our service class will be kept.

Step 3:

Writing the REST Business Logic.

There are couple of annotations i liked to discussed here, these are the very basic and commonly used annotations.

@Path -- Mention the Path from which you want to access a REST Service either class level or method level.

@GET -- Performs HTTP get Operation useful for getting info read only.

@Produces - This Produces the Respective output in different format such as XML,JSON,TEXT,HTML etc to the client.

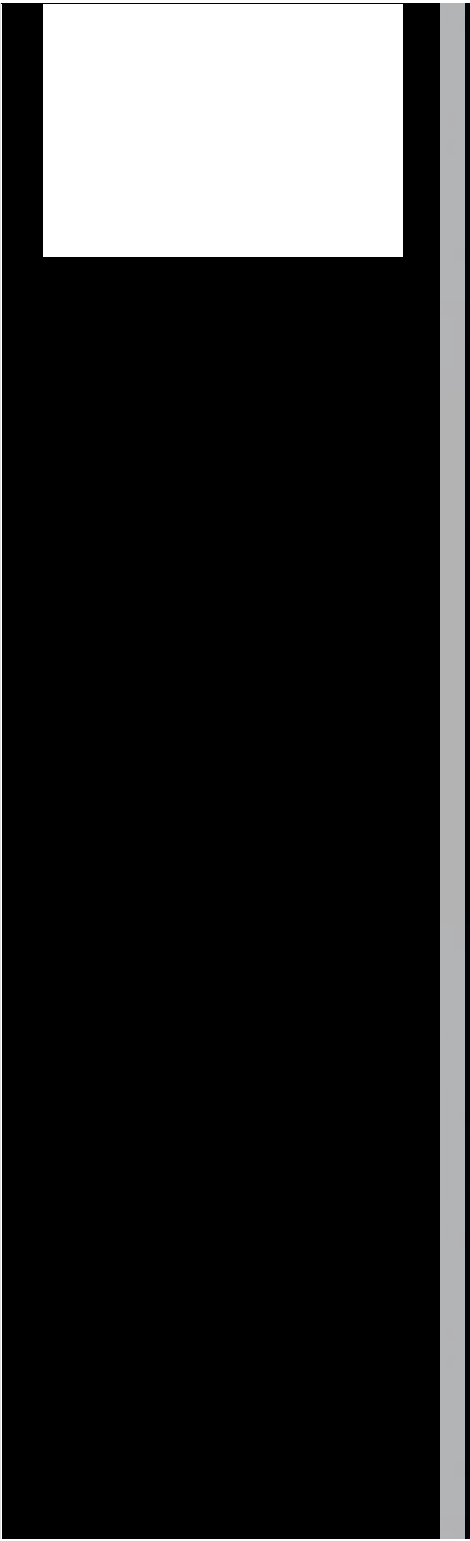
Step 4:

Implementing the Jersey Framework & Writing our First REST Service

NetBeans generally comes along with the Jersey Framework if you don't have the Framework install , manually you should have to download it and add the jars to the libraries.

But i am not going to discuss about that , i assume you all are smart guys , and you have NetBeans with J2EE Supported version , so basically you will have Jersey supported framework install in the NetBeans Itself.

So , We don't need actually write any code from implementing this frame work in **web.xml**, NetBeans automatically does

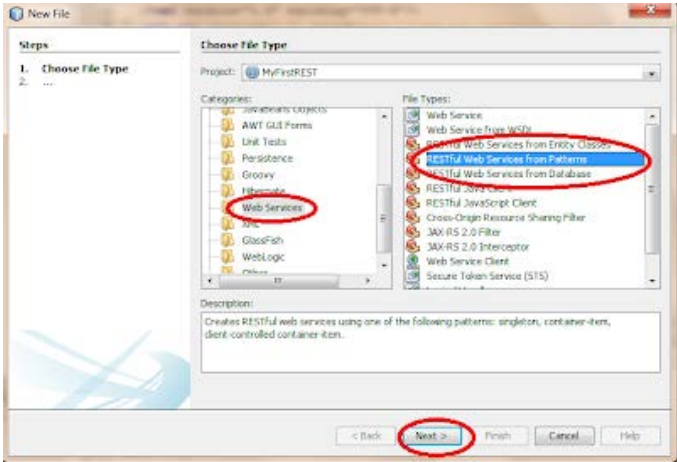


for us., that's the advantage of using NetBeans.

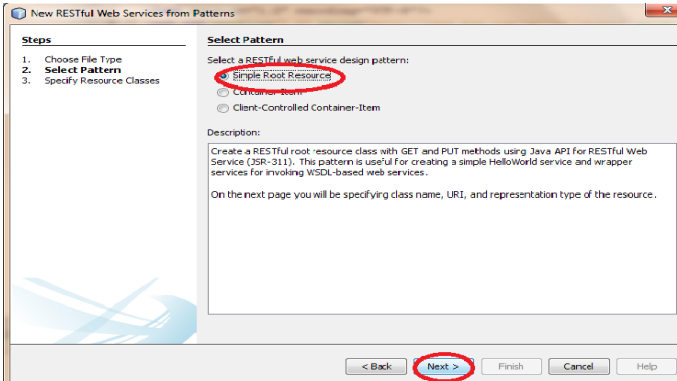
So now this can achieve by following these steps :

Right click on the "com.test"(Package i am using for this example, you can use your own)ans select

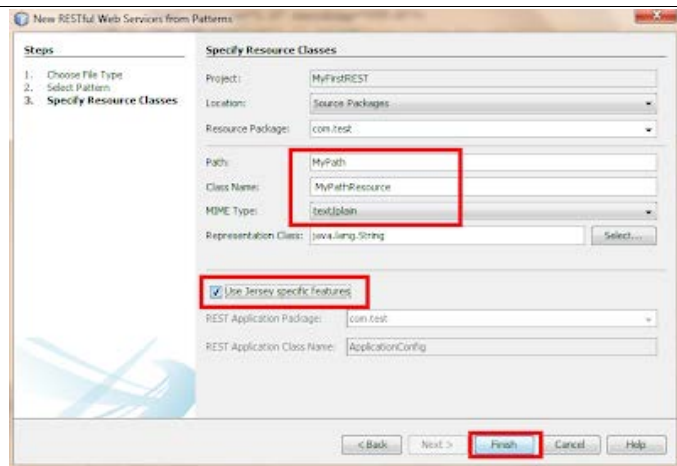
New--->Other--->WebService-->RESTful WebServices from Patterns.



Select patterns as "Simple Root Resource" and Press the "Next" Button



Give path Name as"MyPath" and class Name as"MypathResource" and select the MIME Type as "text/plain". and importantly select the check box for Jersey framework and click the "finish" Button.



Step 5:

Develop the code.

By default a class name called "MypathResource" will be created with some methods and instance variable define in it. Delete the code and replace with the following code.

```
package com.test;
import javax.ws.rs.core.Context;
import javax.ws.rs.core.UriInfo;
import javax.ws.rs.PathParam;
import javax.ws.rs.Consumes;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.GET;
import javax.ws.rs.Produces;
@Path("Mypath")
public class MypathResource {
    @GET
    @Produces("text/plain")
    public String getText() {
        return "My First RESTful Service....";
    }
}
```

This is the very basic and simplest RESTful Service , it will accept the Path "MyPath" in the URL and invoke the method getText which in turn return a Response as Text with the wording "My First RESTful Service..."

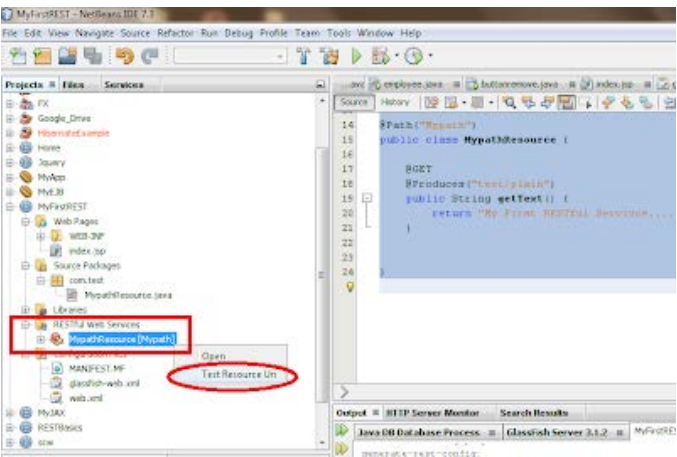
Step 6:

Clean & Build , Deploy to check the service.

Right Click on the Project Node "**MyFirstREST**" and select Clean and build this will clean the directory and compiles the Java files and creates a WAR Archive.

Once , the Clean and build is finish successfully again RightClick on the ProjectNode "**MyFirstREST**" and select "**Deploy**".

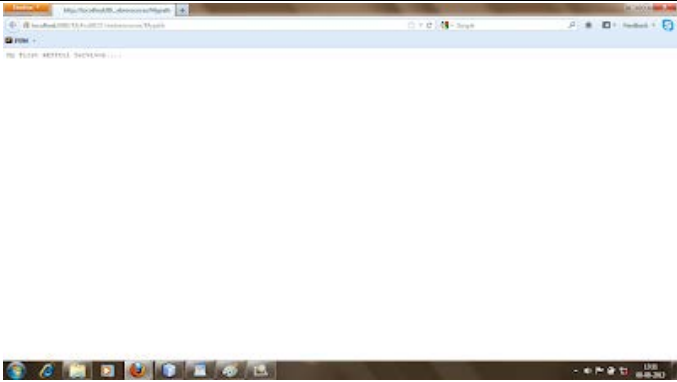
Once the Deployment is done we can check the service by:
Right click on the REST Service created under folder called "**Restful Web Service**" and select "**Test Resource Uri**"



or we can directly access using the URL
<http://localhost:8080/MyFirstREST/webresources/MyPath>

Note : The Port Number will differ based upon your Server Configuration Setting.

Result :



Main Components :

How all these is happening ?

The main Gate Keeper or the main Servlet for this is specified in the web.xml which performs the Jersey Mechanism , Please refer the image below for detail explanation of Jersey Servlet define in web.xml .

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="2.5" xmlns:xsi="http://java.sun.com/xml/jaxns"
xsi:schemaLocation="http://java.sun.com/xml/jaxns
http://java.sun.com/xml/jaxns/web-app-2.5.xsd">

  <servlet>
    <servlet-name>ServletAdapter</servlet-name>
    <servlet-class>com.sunersey.container.servlet.ServletContainer</servlet-class>
    <init-param>
      <description>Multiple packages, separated by semicolon(,), can be specified in param-
value</description>
      <param-name>com.sunersey.config.property.package</param-name>
      <param-value>com.sunersey</param-value>
    </init-param>
    <init-param>
      <param-name>com.sunersey.api.json.PODMapping</param-name>
      <param-value>true</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>

  <servlet-mapping>
    <servlet-name>ServletAdapter</servlet-name>
    <url-pattern>/webresources/*</url-pattern>
  </servlet-mapping>

  <session-config>
    <session-timeout>
      30
    </session-timeout>
  </session-config>
  <welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

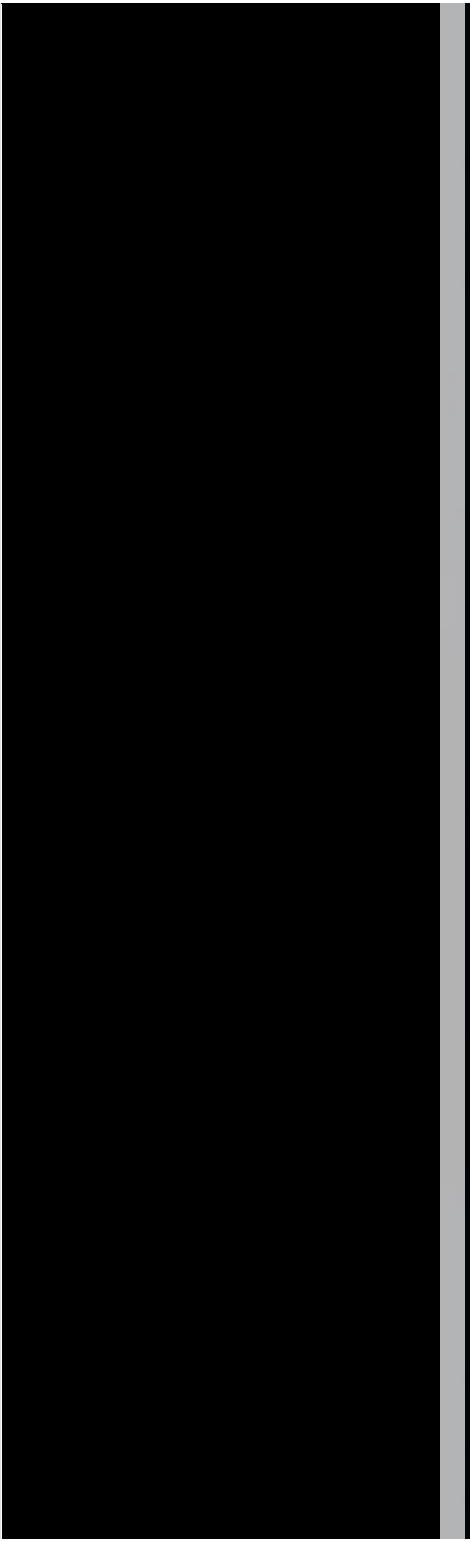
Jersey Servlet which checks each request.

Jersey Framework Init Param to map the package with Jersey

User define package name which contains Jersey Business Methods

An API Used for JSON Parsing

User define URI from Which all the Other URI should follow,



That's all for the day, In the Next Section we will be seeing different useful annotations and how to produce different form of Outputs.

Thanks for Reading.
Please provide your valuable suggestion and comments.

[<< Prev](#) [Index](#) [Next >>](#)



blogs

Posted by **Ayaz Roomy** at 7:17 AM     

11 comments:

Anonymous **June 9, 2014 at 10:45 PM**
Excellent information for beginners. Thanks for sharing. Keep sharing your knowledge.
Reply

Anonymous **July 25, 2014 at 3:56 AM**
thank you for your information
Reply

Anonymous **July 30, 2014 at 8:40 PM**
Thanks a lot buddy..
Reply

 **Narayan** **August 4, 2014 at 4:47 AM**

I am getting the following error

```
WEB0123: WebModule [RestProject1] failed to deploy and has been disabled
java.lang.NoClassDefFoundError: javax/servlet/Filter
at java.lang.ClassLoader.defineClass1(Native Method)
at java.lang.ClassLoader.defineClass(ClassLoader.java:620)
at java.security.SecureClassLoader.defineClass(SecureClassLoader.java:124)
at java.net.URLClassLoader.defineClass(URLClassLoader.java:260)
at java.net.URLClassLoader.access$000(URLClassLoader.java:56)
at java.net.URLClassLoader$1.run(URLClassLoader.java:195)
at java.security.AccessController.doPrivileged(Native Method).....
```

Reply

hollosys August 7, 2014 at 8:42 AM

N FOLL-BACK

I <http://hollosys.blogspot.com/>

C

E ARTIKEL

Reply



John Alert December 30, 2015 at 1:07 AM

Great Article

[Java Web Services Online Training](#) | [Web Services Course](#) | [Web Services Training Courses](#) | [Java Web Services Training in Chennai](#) | [Online Java Training](#) | [Java Training in Chennai](#)

Reply



JB Soft System July 12, 2016 at 2:18 AM

Thanks for sharinig this article.

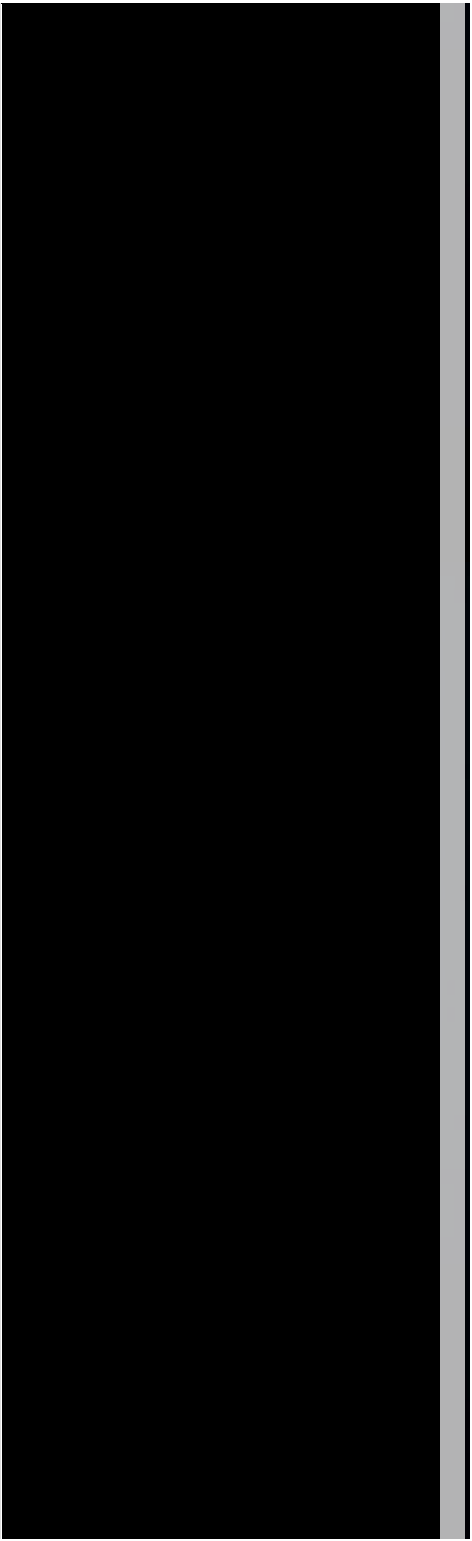
[ERP Software Developing Company Chennai](#) | [ERP Development Company in Chennai](#)

Reply

Anonymous July 28, 2016 at 11:09 AM

why have we selected java version as java EE 5, I tried with java EE 7 and netbeans doesnt create applicationconfig.java .

Reply



Durante August 28, 2017 at 6:35 AM

Excellent post. Thank you!

Reply



pooja January 6, 2019 at 9:55 PM

This is a nice article here with some useful tips for those who are not used-to comment that frequently. Thanks for this helpful information I agree with all points you have given to us. I will follow all of them.

[Devops Training courses](#)
[python Training in chennai](#)
[Devops Training in Bangalore](#)

Reply



pooja January 6, 2019 at 9:55 PM

Its really an Excellent post. I just stumbled upon your blog and wanted to say that I have really enjoyed reading your blog. Thanks for sharing....

[Devops Training courses](#)
[python Training in chennai](#)
[Devops Training in Bangalore](#)

Reply

Add comment

Comment with your Google account if you'd like to be able to manage your comments in the future. If you comment anonymously, you won't be able to edit or delete your comment. [Learn more](#)



Comment as:

Publish

Preview

☐ Notify me

[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)

TOPSITE

144 [BLOGTOPSITES](#)

[blogs](#)

Picture Window theme. Powered by Blogger.