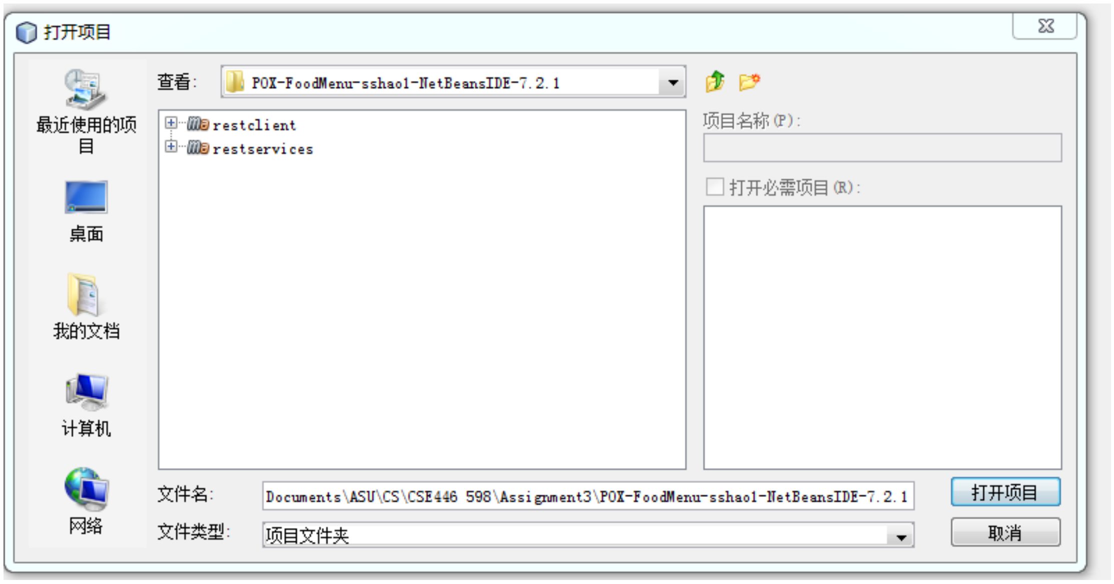
CSE446/598 Assignment3

Readme

Shihuan Shao

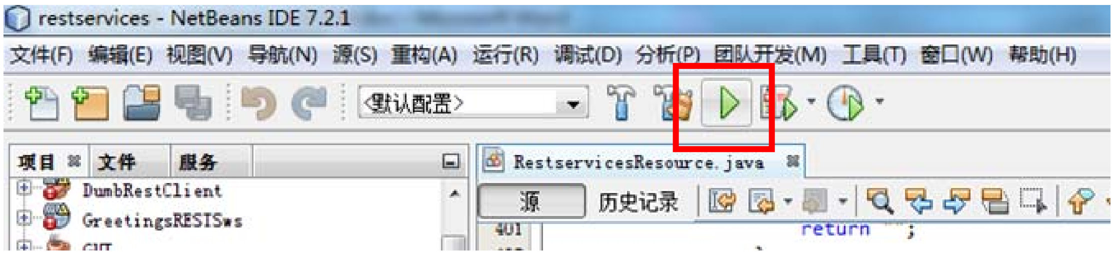
1. Download the zipped package POX-FoodMenu-sshao1-NetBeansIDE-7.2.1 and unzip. Copy the XML file FoodItemData.xml inside the folder to C:\, since the path for the server to access this file is C:\FoodItemData.xml.

2. Open NetBeans IDE 7.2.1. Click File -> Open Project. Go to the folder, and open the project restservices and restclient.



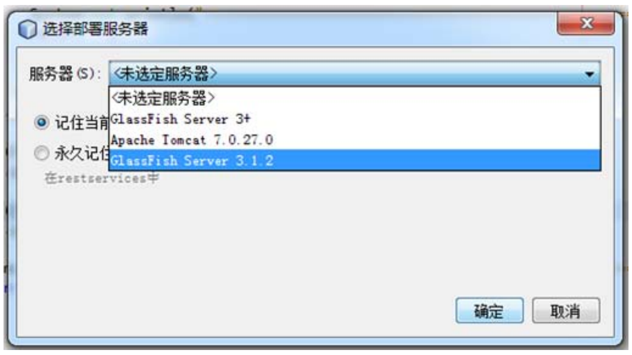
打开项目
查看: POX-FoodMenu-sshao1-NetBeansIDE-7.2.1
restclient
restservices
项目名称 (P):
打开必需项目 (R):
最近使用的项目
桌面
我的文档
计算机
网络
文件名: Documents\ASU\CS\CSE446 598\Assignment3\POX-FoodMenu-sshao1-NetBeansIDE-7.2.1
文件类型: 项目文件夹
打开项目
取消

3. Open the server file called RestservicesResource.java in restservices project. Run the program by clicking the green arrow button showed in the red frame.

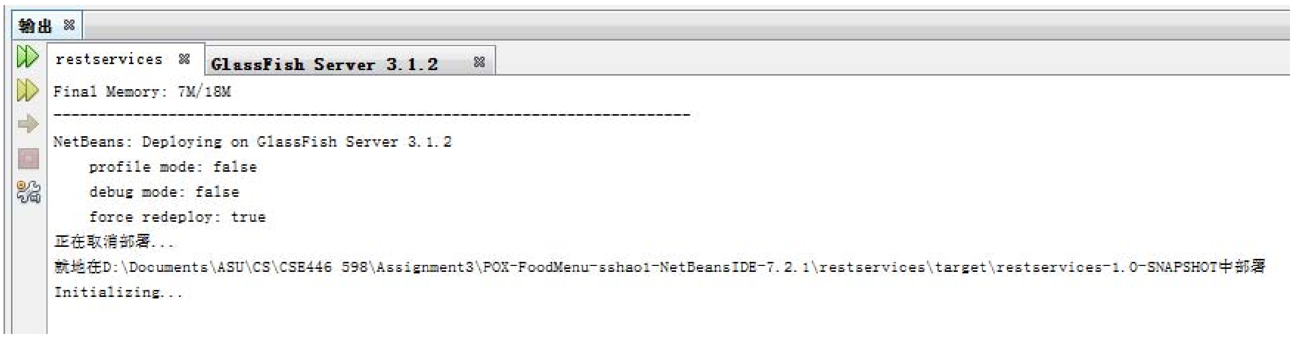


restservices - NetBeans IDE 7.2.1
文件(F) 编辑(E) 视图(V) 导航(N) 源(S) 重构(A) 运行(R) 调试(D) 分析(P) 团队开发(M) 工具(T) 窗口(W) 帮助(H)
<默认配置>
项目
文件
服务
DumbRestClient
GreetingsRESTSws
RestservicesResource.java
源
历史记录
return

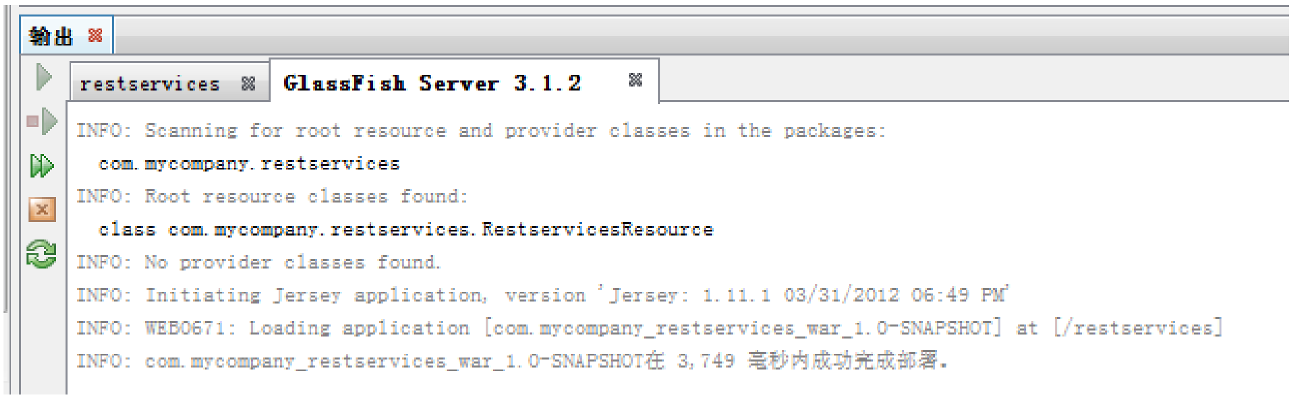
The user might be asked to choose a server for deployment. Just select GlassFish Server 3.1.2 and press Confirm.



4. The output panel will be open and show following contents:

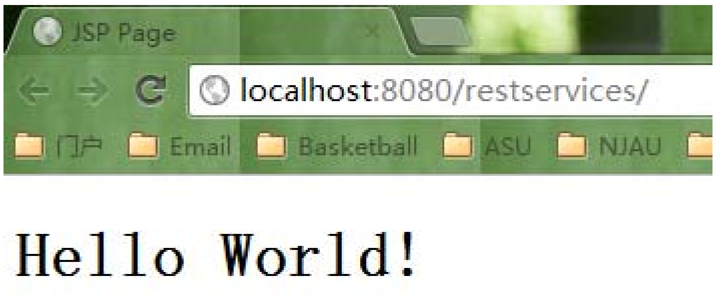


```
Final Memory: 7M/18M
------------------------------------------------------------------------
NetBeans: Deploying on GlassFish Server 3.1.2
    profile mode: false
    debug mode: false
    force redeploy: true
正在取消部署...
就地在D:\Documents\ASU\CS\CSE446 598\Assignment3\POX-FoodMenu-sshao1-NetBeansIDE-7.2.1\restservices\target\restservices-1.0-SNAPSHOT中部署
Initializing...
```



```
INFO: Scanning for root resource and provider classes in the packages:
  com.mycompany.restservices
INFO: Root resource classes found:
  class com.mycompany.restservices.RestservicesResource
INFO: No provider classes found.
INFO: Initiating Jersey application, version 'Jersey: 1.11.1 03/31/2012 06:49 PM'
INFO: WEB0671: Loading application [com.mycompany_restservices_war_1.0-SNAPSHOT] at [/restservices]
INFO: com.mycompany_restservices_war_1.0-SNAPSHOT在 3,749 毫秒内成功完成部署。
```

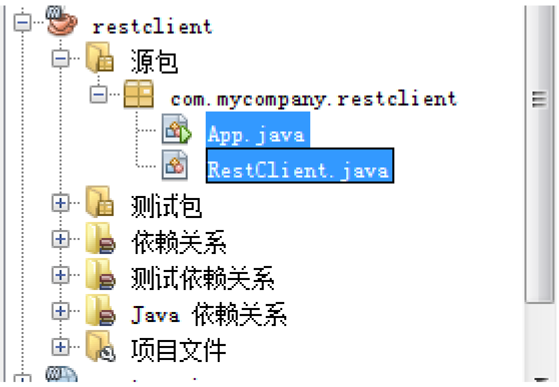
A JSP page is also automatically opened and show “Hello World!”:



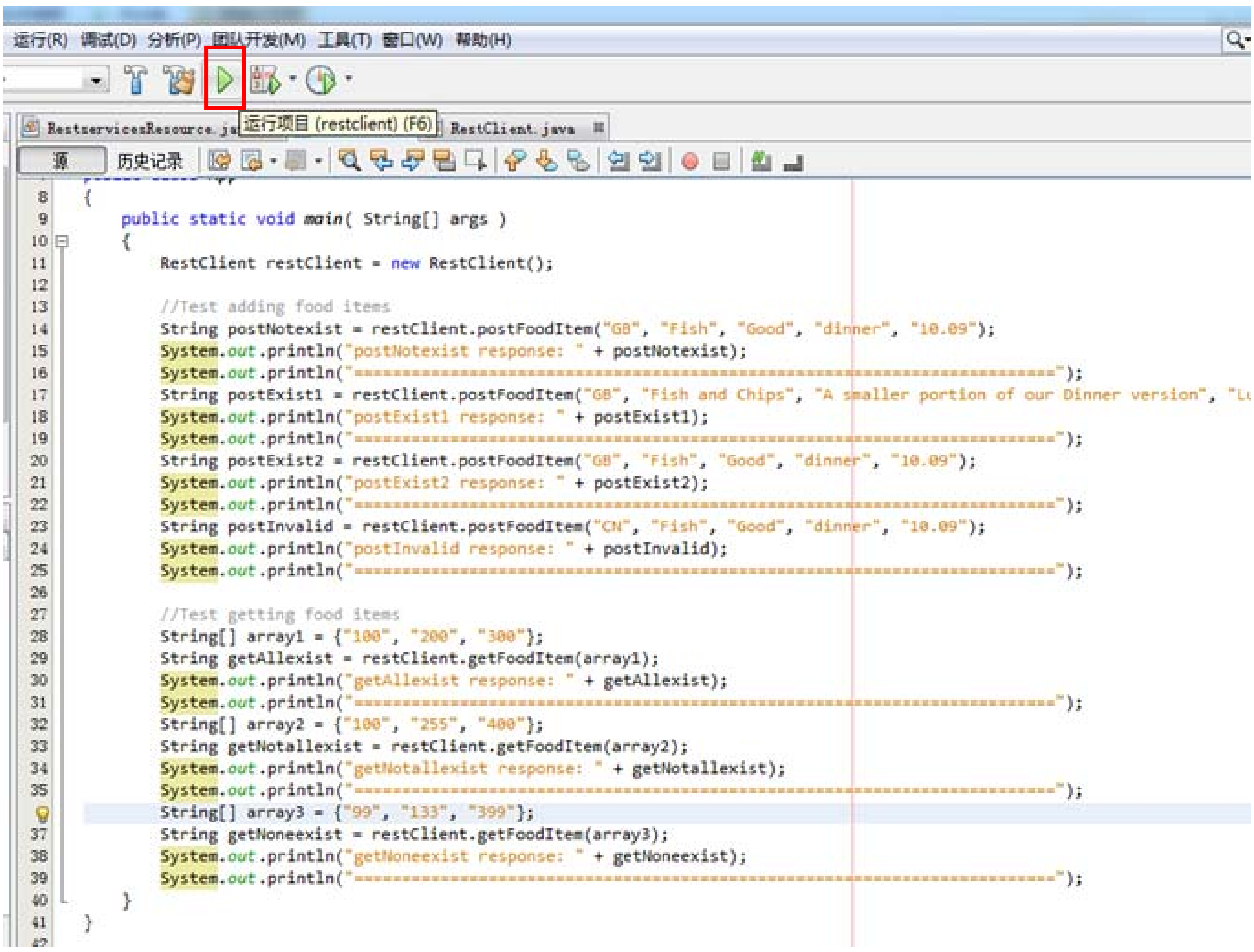
Hello World!

Now the service is deployed on GlassFish Server 3.1.2.

5. Open the driver file App.java and client file RestClient.java in the restclient project.



6. Choose the App.java and run it by clicking the green arrow button.

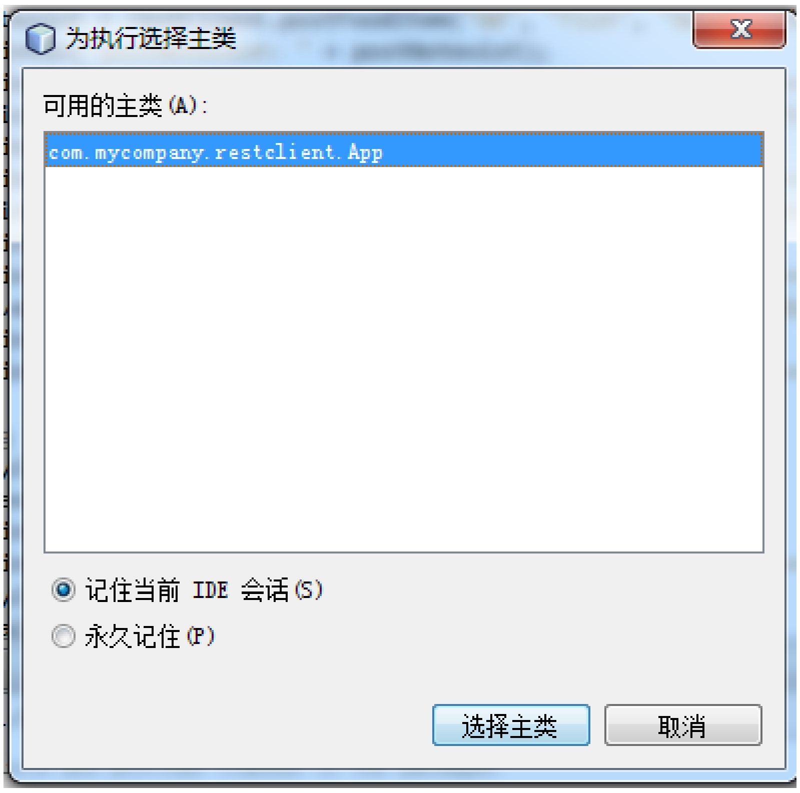

运行(R) 调试(D) 分析(P) 团队开发(M) 工具(T) 窗口(W) 帮助(H)
运行项目 (restclient) (F6)
RestservicesResource.ja
RestClient.java
源
历史记录

```java
{
    public static void main( String[] args )
    {
        RestClient restClient = new RestClient();

        //Test adding food items
        String postNotexist = restClient.postFoodItem("GB", "Fish", "Good", "dinner", "10.09");
        System.out.println("postNotexist response: " + postNotexist);
        System.out.println("=========================================================================");
        String postExist1 = restClient.postFoodItem("GB", "Fish and Chips", "A smaller portion of our Dinner version", "Lu
        System.out.println("postExist1 response: " + postExist1);
        System.out.println("=========================================================================");
        String postExist2 = restClient.postFoodItem("GB", "Fish", "Good", "dinner", "10.09");
        System.out.println("postExist2 response: " + postExist2);
        System.out.println("=========================================================================");
        String postInvalid = restClient.postFoodItem("CN", "Fish", "Good", "dinner", "10.09");
        System.out.println("postInvalid response: " + postInvalid);
        System.out.println("=========================================================================");

        //Test getting food items
        String[] array1 = {"100", "200", "300"};
        String getAllexist = restClient.getFoodItem(array1);
        System.out.println("getAllexist response: " + getAllexist);
        System.out.println("=========================================================================");
        String[] array2 = {"100", "255", "400"};
        String getNotallexist = restClient.getFoodItem(array2);
        System.out.println("getNotallexist response: " + getNotallexist);
        System.out.println("=========================================================================");
        String[] array3 = {"99", "133", "399"};
        String getNoneexist = restClient.getFoodItem(array3);
        System.out.println("getNoneexist response: " + getNoneexist);
        System.out.println("=========================================================================");
    }
}
```

As you can see in the above figure, the App calls the functions postFoodItem() and getFoodItem() in RestClient class to add fooditems to the server and also retrieve fooditems from the server, and also print the response message from the server.

The test cases I used are listed below:
Test case 1: try to add a food item that does not exist in the list.
Test case 2: try to add a food item that does exist in the list.
Test case 3: try to add the food item that is added in the Test case 1.
Test case 4: try to add an illegal food item. In my program, one type of illegal food items is the one whose “country” attribute is not “GB”, ”US” or ”IN”.
Test case 5: try to retrieve three food items which all exist in list with there ids.
Test case 6: try to retrieve three food items, one of which does exist and the rest two do not exist in list.
Test case 7: try to retrieve one food item which does not exist in the list.

7. The user might be asked to choose the main class to run, just choose the App and click confirm.



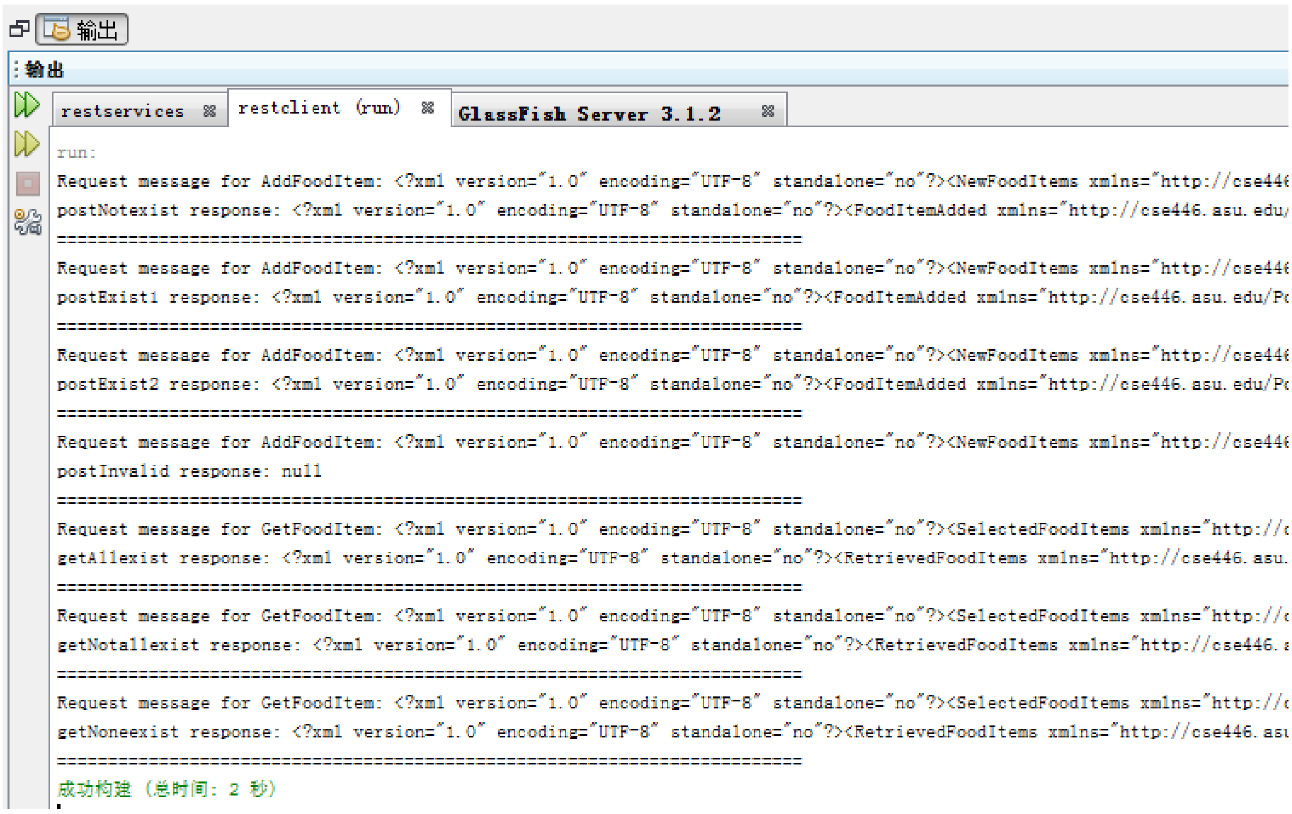
为执行选择主类

可用的主类 (A):

com.mycompany.restclient.App

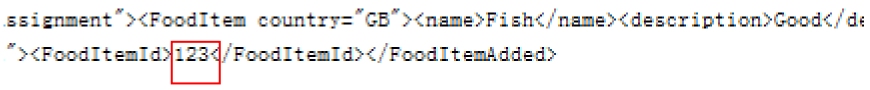
[✓ 记住当前 IDE 会话(S)] 永久记住(P)

8. In the output panel of restclient, you can see the both the request and response messages in XML format.



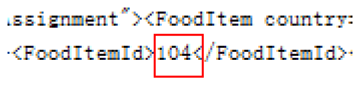
```
run:
Request message for AddFoodItem: <?xml version="1.0" encoding="UTF-8" standalone="no"?><NewFoodItems xmlns="http://cse446
postNotexist response: <?xml version="1.0" encoding="UTF-8" standalone="no"?><FoodItemAdded xmlns="http://cse446.asu.edu/
==========================================================================
Request message for AddFoodItem: <?xml version="1.0" encoding="UTF-8" standalone="no"?><NewFoodItems xmlns="http://cse446
postExist1 response: <?xml version="1.0" encoding="UTF-8" standalone="no"?><FoodItemAdded xmlns="http://cse446.asu.edu/Pc
==========================================================================
Request message for AddFoodItem: <?xml version="1.0" encoding="UTF-8" standalone="no"?><NewFoodItems xmlns="http://cse446
postExist2 response: <?xml version="1.0" encoding="UTF-8" standalone="no"?><FoodItemAdded xmlns="http://cse446.asu.edu/Pc
==========================================================================
Request message for AddFoodItem: <?xml version="1.0" encoding="UTF-8" standalone="no"?><NewFoodItems xmlns="http://cse446
postInvalid response: null
==========================================================================
Request message for GetFoodItem: <?xml version="1.0" encoding="UTF-8" standalone="no"?><SelectedFoodItems xmlns="http://c
getAllexist response: <?xml version="1.0" encoding="UTF-8" standalone="no"?><RetrievedFoodItems xmlns="http://cse446.asu.
==========================================================================
Request message for GetFoodItem: <?xml version="1.0" encoding="UTF-8" standalone="no"?><SelectedFoodItems xmlns="http://c
getNotallexist response: <?xml version="1.0" encoding="UTF-8" standalone="no"?><RetrievedFoodItems xmlns="http://cse446.a
==========================================================================
Request message for GetFoodItem: <?xml version="1.0" encoding="UTF-8" standalone="no"?><SelectedFoodItems xmlns="http://c
getNoneexist response: <?xml version="1.0" encoding="UTF-8" standalone="no"?><RetrievedFoodItems xmlns="http://cse446.asu
==========================================================================
成功构建 (总时间: 2 秒)
```

The response for Test case 1 shows a new id allocated for the new food item.

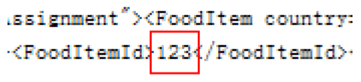


```
ssignment"><FoodItem country="GB"><name>Fish</name><description>Good</de
"><FoodItemId>123</FoodItemId></FoodItemAdded>
```

The response for Test case 2 and 3 show the ids of the existed food item.

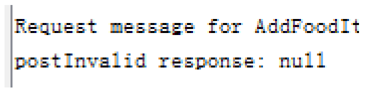


```
ssignment"><FoodItem country=
<FoodItemId>104</FoodItemId>
```



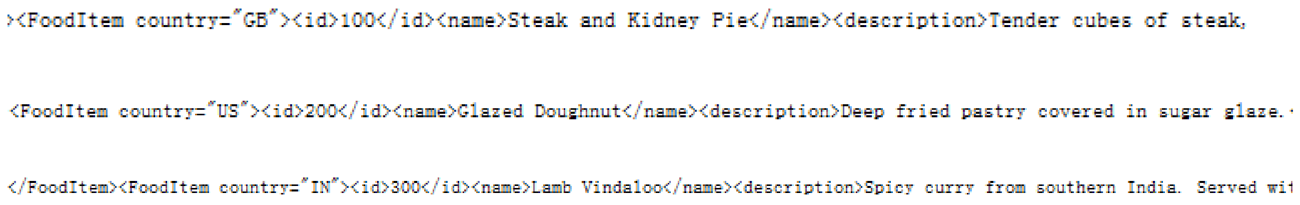
```
ssignment"><FoodItem country=
<FoodItemId>123</FoodItemId>
```

The response for Test case 4 shows null since that is an illegal food item.



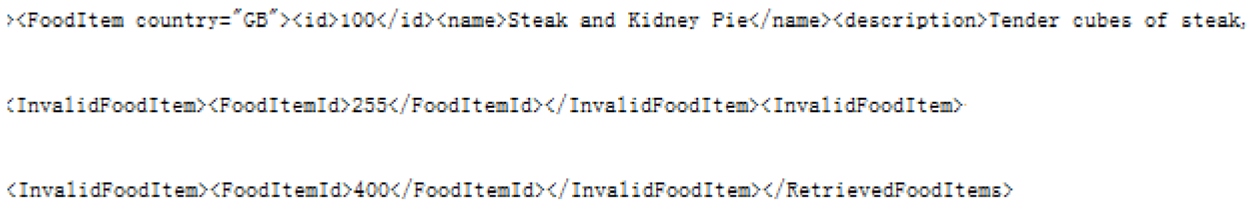
```
Request message for AddFoodIt
postInvalid response: null
```

The response for Test case 5 shows the retrieved food items with their country, id, name, description, category and price.



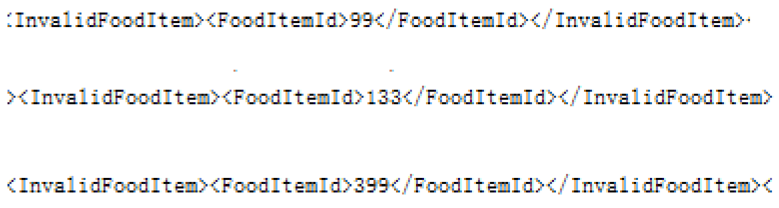
```
><FoodItem country="GB"><id>100</id><name>Steak and Kidney Pie</name><description>Tender cubes of steak,

<FoodItem country="US"><id>200</id><name>Glazed Doughnut</name><description>Deep fried pastry covered in sugar glaze.

</FoodItem><FoodItem country="IN"><id>300</id><name>Lamb Vindaloo</name><description>Spicy curry from southern India. Served wi
```

The response for Test case 6 shows corresponding information for different request items according to their existance.



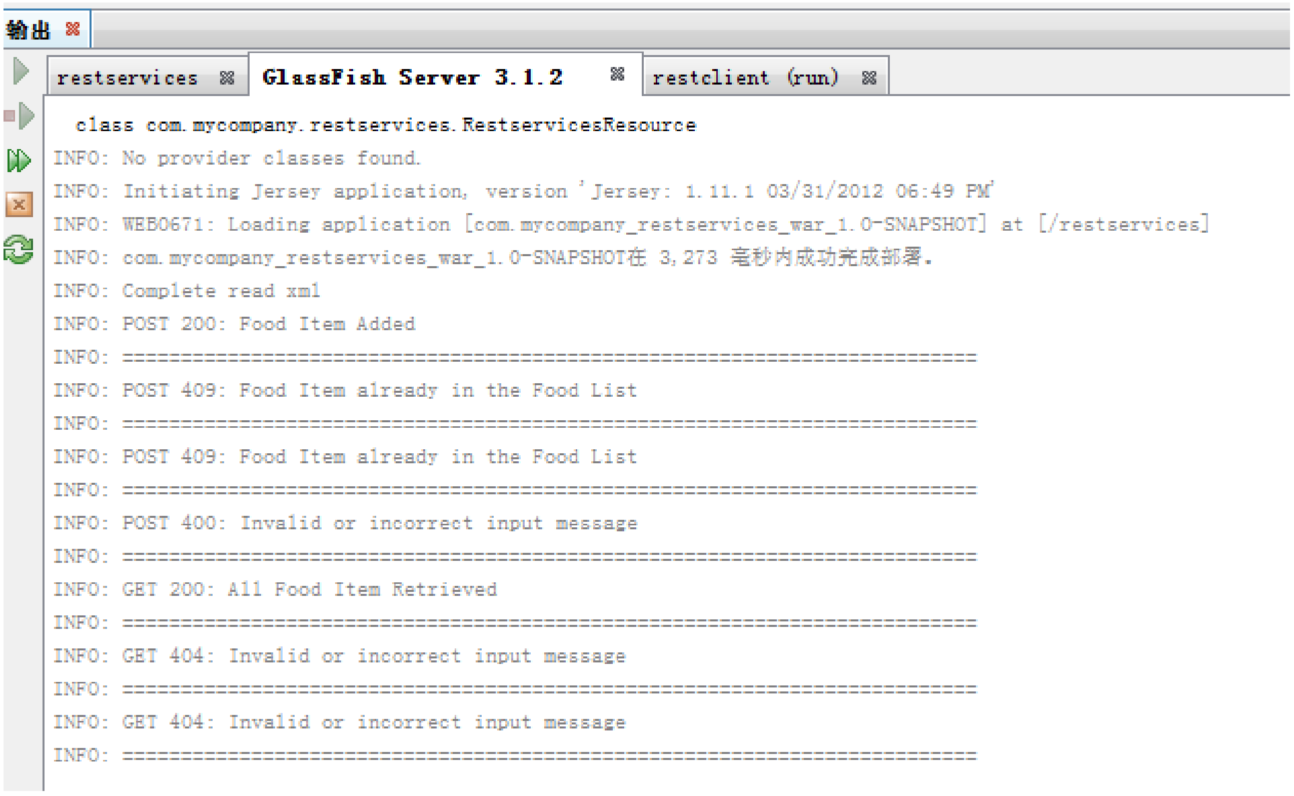
```
><FoodItem country="GB"><id>100</id><name>Steak and Kidney Pie</name><description>Tender cubes of steak,

<InvalidFoodItem><FoodItemId>255</FoodItemId></InvalidFoodItem><InvalidFoodItem>

<InvalidFoodItem><FoodItemId>400</FoodItemId></InvalidFoodItem></RetrievedFoodItems>
```

The response for Test case 7 shows invalid information for all the three non-existent items.



```
InvalidFoodItem><FoodItemId>99</FoodItemId></InvalidFoodItem>

><InvalidFoodItem><FoodItemId>133</FoodItemId></InvalidFoodItem>

<InvalidFoodItem><FoodItemId>399</FoodItemId></InvalidFoodItem><
```

9. Change to the GlassFish Server 3.1.2 output panel, the user can see the code and interpretation of every request.



```
  class com.mycompany.restservices.RestservicesResource
INFO: No provider classes found.
INFO: Initiating Jersey application, version 'Jersey: 1.11.1 03/31/2012 06:49 PM'
INFO: WEB0671: Loading application [com.mycompany_restservices_war_1.0-SNAPSHOT] at [/restservices]
INFO: com.mycompany_restservices_war_1.0-SNAPSHOT在 3,273 毫秒内成功完成部署.
INFO: Complete read xml
INFO: POST 200: Food Item Added
INFO: ==========================================================================
INFO: POST 409: Food Item already in the Food List
INFO: ==========================================================================
INFO: POST 409: Food Item already in the Food List
INFO: ==========================================================================
INFO: POST 400: Invalid or incorrect input message
INFO: ==========================================================================
INFO: GET 200: All Food Item Retrieved
INFO: ==========================================================================
INFO: GET 404: Invalid or incorrect input message
INFO: ==========================================================================
INFO: GET 404: Invalid or incorrect input message
INFO: ==========================================================================
```

Test case 1: Food item is added. Response code 200.
Test case 2 and 3: Food item is existed. Response code 409.
Test case 4: Food item is invalid. Response code 400.
Test case 5: All the food items can be retrieved. Response code 200.
Test case 6: Two of the three food items cannot be retrieved. Response code 404.
Test case 7: All of the food items cannot be retrieved. Response code 404.