UNICO

Assignment Documentation

2/22/2018

VERSION HISTORY

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| 1.0 | Parampreet Kaur | 02/22/18 | Parampreet Kaur | 02/22/2018 |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Table of Contents

1.1 **INTRODUCTION**

1.2 **TECHNOLOGIES USED**

1.3 **OPEN SOURCE LIBRARIES**

1.4 **PROJECT CONFIGURATION**

1.4.1 **JMS Resource**

1.4.2 **Database Setup**

1.5 **WORKING PROJECT**

**1.1 INTRODUCTION**

The assignment has two web services implemented as REST and SOAP.

The REST service will specifically take the input as integers from the client (Postman, chrome extension) and pass it on to a JMS queue for the SOAP service to receive. Rest service also provides functionality to get the list of integers sent so far to the JMS queue.

The SOAP service on the other side takes the JMS queue and passes it on to calculate GCD of the values. These records are then inserted into a MySQL database. The Soap service also provides implementation for fetching the list and sum of GCDs from database.

**1.2 TECHNOLOGIES USED**

|  |  |  |
| --- | --- | --- |
| 1. | Java | Jdk1.7.0\_15 |
| 2. | Eclipse | Eclipse Kepler |
| 3. | Database | MySQL server 5.6 |
| 4. | App Server | Glassfish |
| 5. | REST service test client | Postman |
| 6. | SOAP service test client | Glassfish Admin console |

**1.3 OPEN SOURCE LIBRARIES**

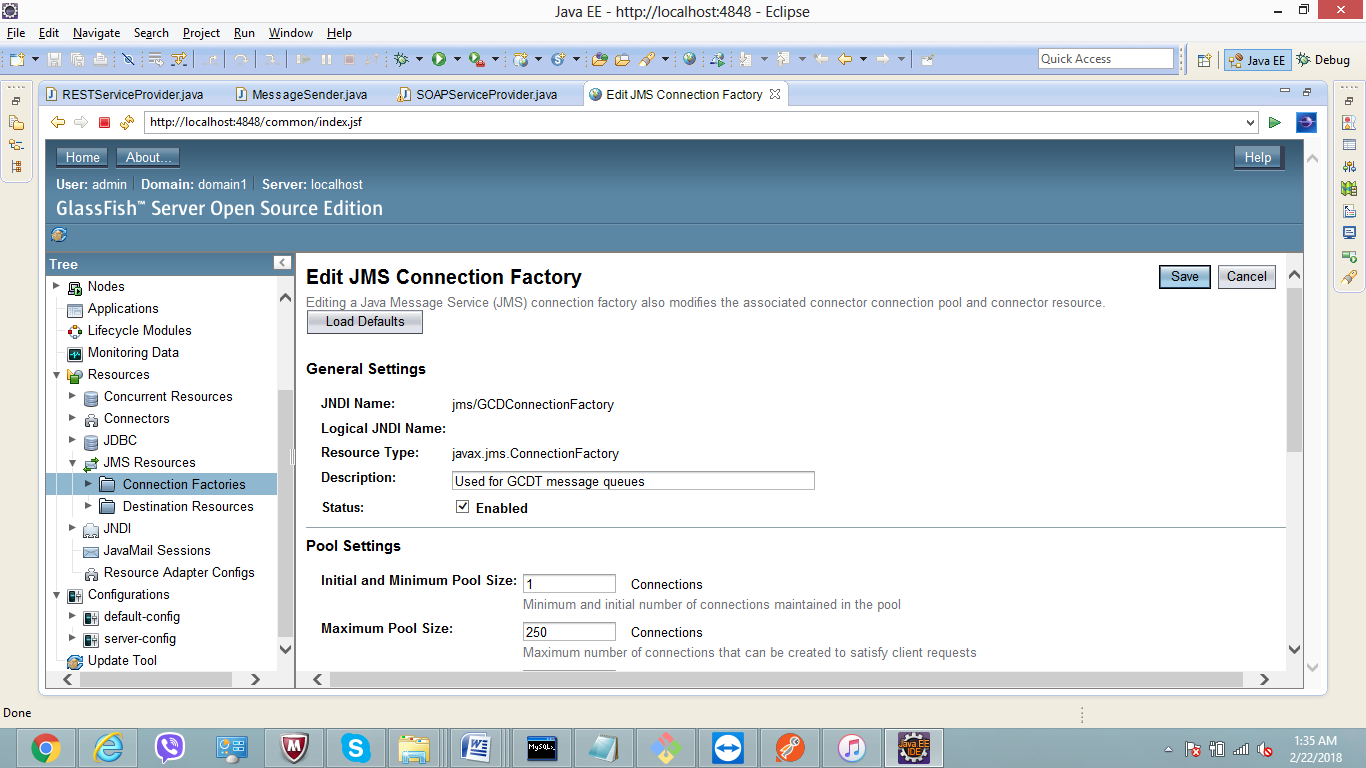
|  |  |  |
| --- | --- | --- |
| 1. | MySQL connector | Mysql-connector.jar 5.1.18 |
| 2. | JMS | Javax-jms.jar |
| 3. | Jersey | Jersey package 2.13 |
| 4. | Java web services | Javax.ws.rs-api.jar |

**1.4 PROJECT CONFIGURATION**

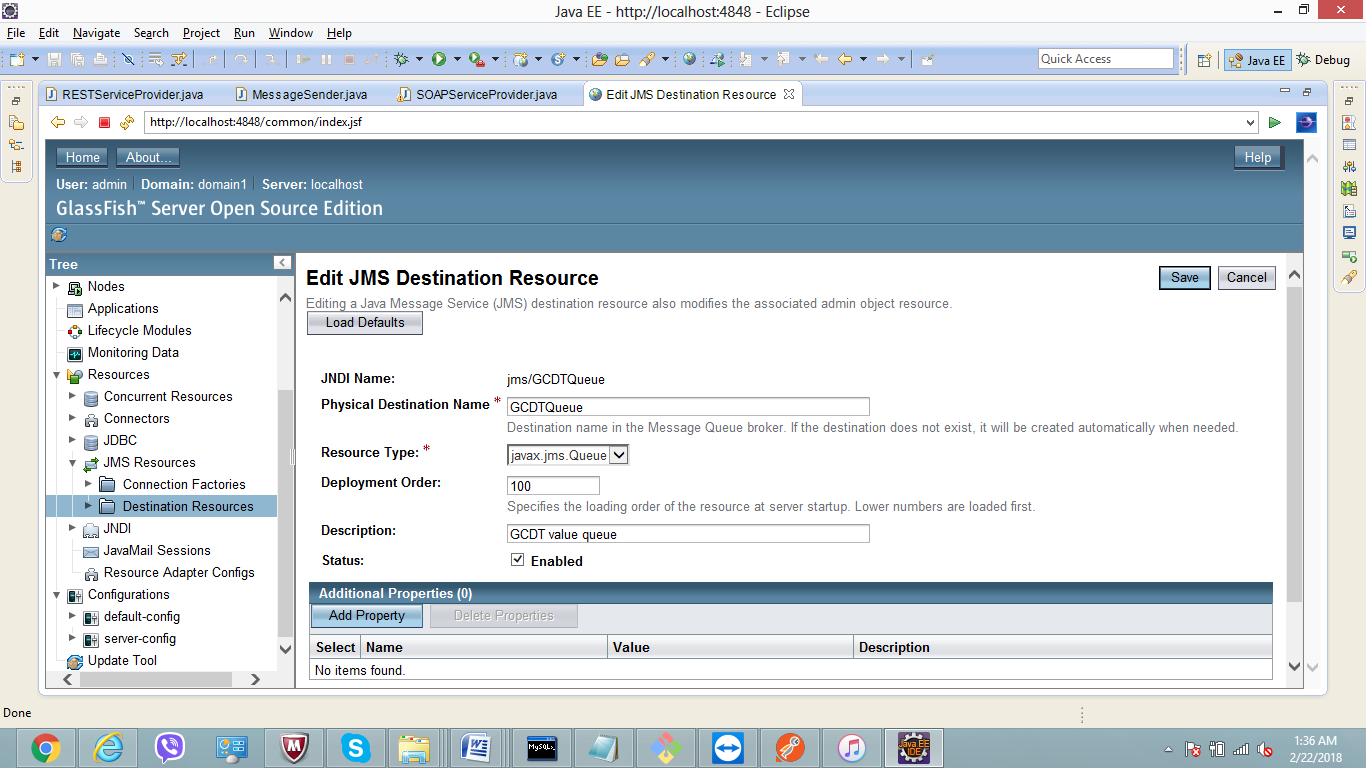
**1.4.1 JMS Resource**

In order to use the JMS queue, the JMS resources need to be set up in the glassfish application server. For this following steps are needed:

1. Create Connection Factory in JMS Resources tab



1. Create JMS queue in Destination Resources tab



**1.4.2 Database Setup**

MySQL database is used in this project. The sample connection string is:

("jdbc:mysql://localhost:3308/unico", "root", "root")

1. Database port : 3308
2. Database name : UNICO
3. Username : root
4. Password : root

Below is the set of queries run to create Database table named GCDT:

mysql> CREATE DATABASE UNICO;

Query OK, 1 row affected (0.00 sec)

mysql> USE DATABASE UNICO;

ERROR 1049 (42000): Unknown database 'database'

mysql> USE UNICO;

Database changed

mysql> CREATE TABLE GCDT(S\_NO INT PRIMARY KEY, GCD INT, INTEGER1 INT, INTEGER2 INT);

Query OK, 0 rows affected (0.37 sec)

mysql> INSERT INTO GCDT VALUES(1, 4, 12, 16);

Query OK, 1 row affected (0.08 sec)

mysql> INSERT INTO GCDT VALUES(2, 3, 6, 9);

Query OK, 1 row affected (0.07 sec)

mysql> COMMIT;

Query OK, 0 rows affected (0.00 sec)

**1.5 SOLUTION PROVIDED**

The project is a Maven Project. The maven dependencies are defined in POM.xml along with some referenced libraries.

Below is description about each source file developed:

1. **RESTServiceProvider:**

This class acts as a REST resource when requested by the client. It has two http methods POST and GET implemented.

@POST method, receives the input from client and passes it to a Message sender to send across JMS queue. It also returns the status of request.

@GET method, returns the list of all the values that were passed on the jms queue after fetching from database.

1. **MessageSender:**

This class creates a session to create JMS queue message. The value of JNDI connection factory and queue is given to initialize the session. The integer values are passed as an ArrayList at the head of the queue.

1. **SOAPServiceProvider:**

This class acts a SOAP resource. SOAP service methods are identified by a @WebMethod annotation. Below is the implementation detail of each method:

gcd(), uses Message Receiver to get the values passed on JMS queue. Also, calculates the gcd and returns the value to client. This method also inserts the value of received integers and calculated gcds in the mysql database.

gcdList(), fetches the list of gcds stored in the database.

gcdSum(), calculates the sum of all gcds stored in the database.

1. **MessageReceiver:**

This class creates a session to receive JMS queue message. The value of JNDI connection factory and queue is given to initialize the session. The values present at the head of the list are obtained and set to a local List variable.

1. **DBConnection:**

This class creates and maintains a database connection with the local mysql database hosted at port number 3308.

1. **GCDRepository:**

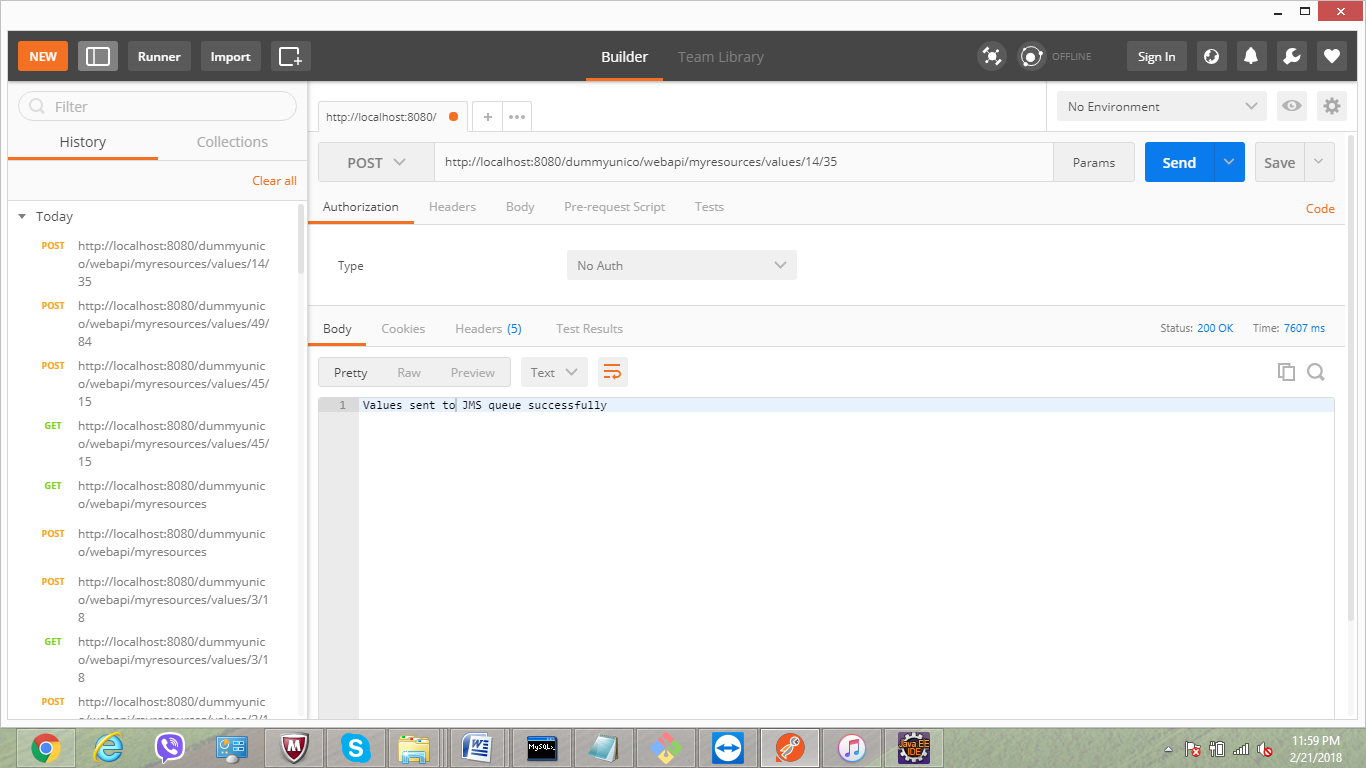
This class provides simple database manipulation operations, such as to fetch the values from database and to insert the values.

**1.5 WORKING PROJECT**

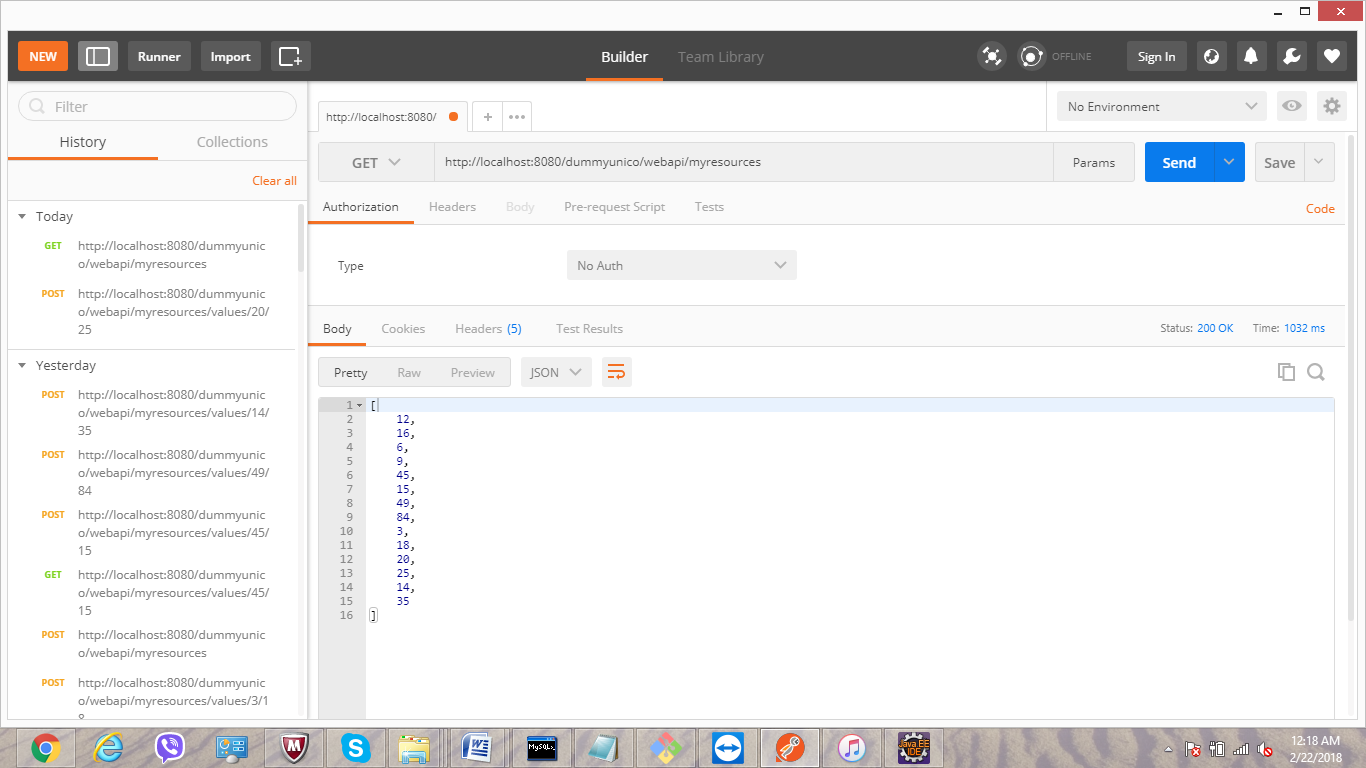
Below are some screenshots of the working application:

1. **Rest service** testing through Postman:

@POST

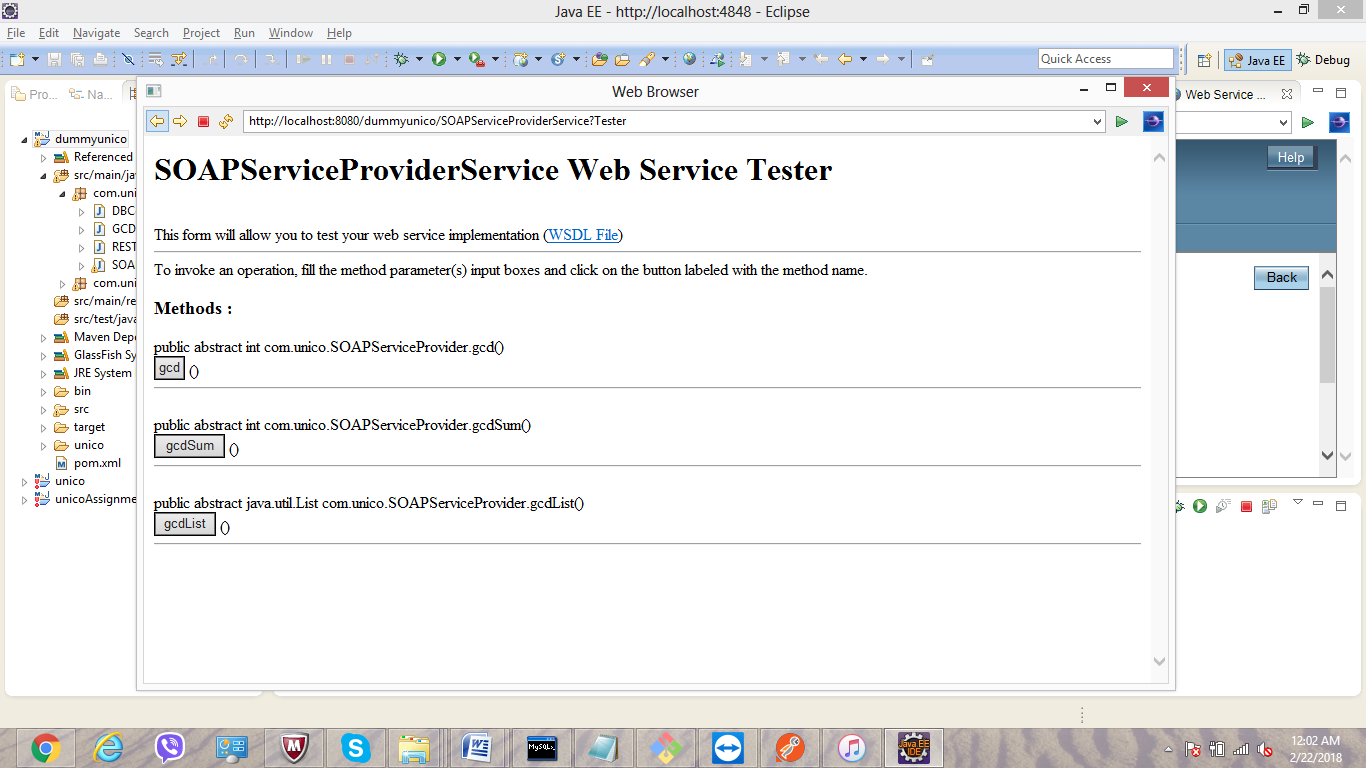


@GET

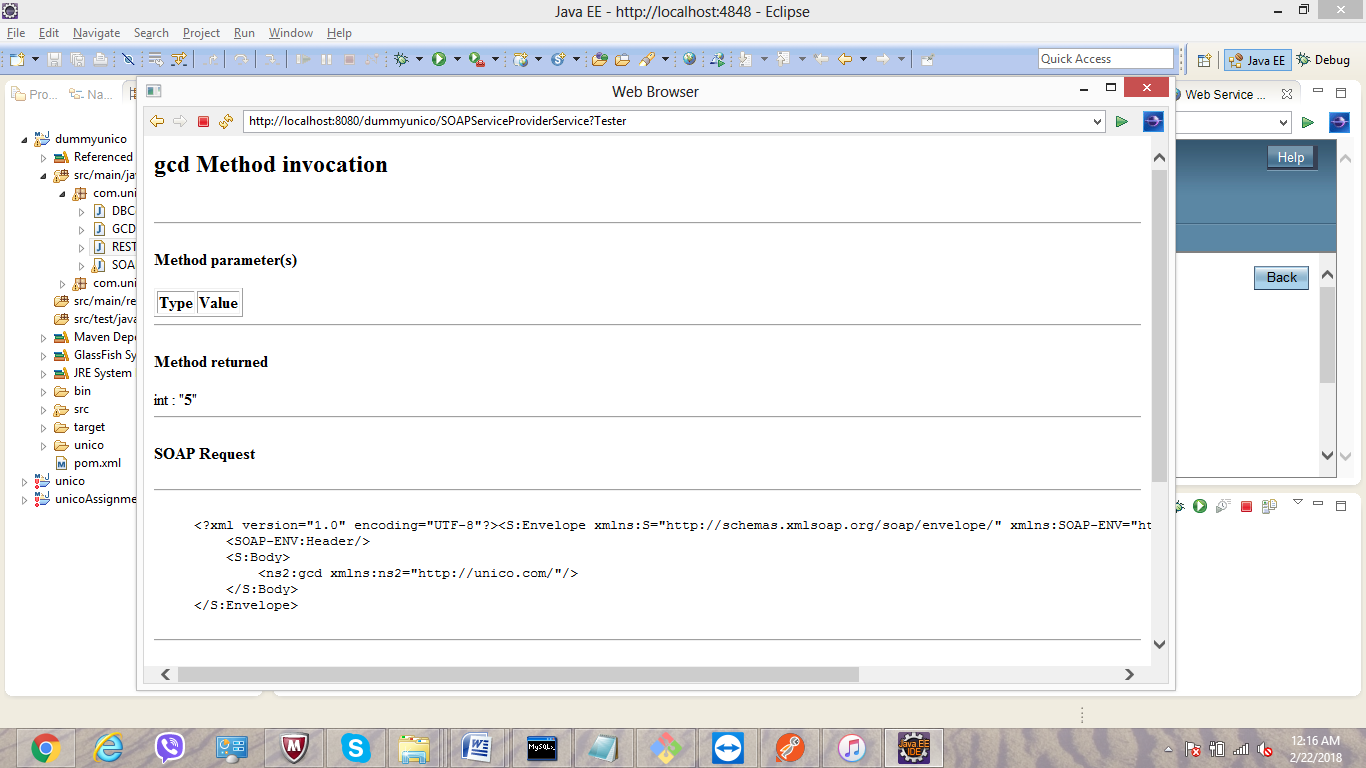


1. **SOAP service** testing through Glassfish Admin console

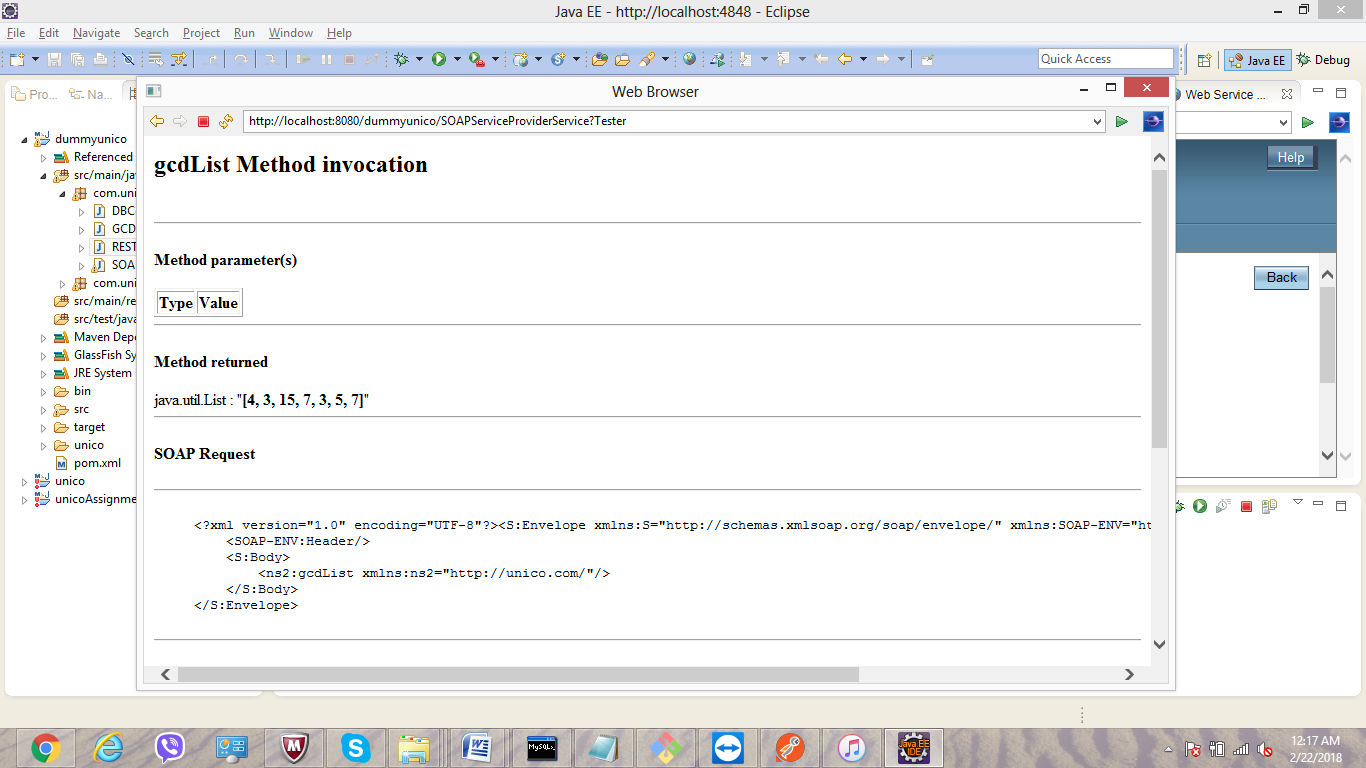
Main page



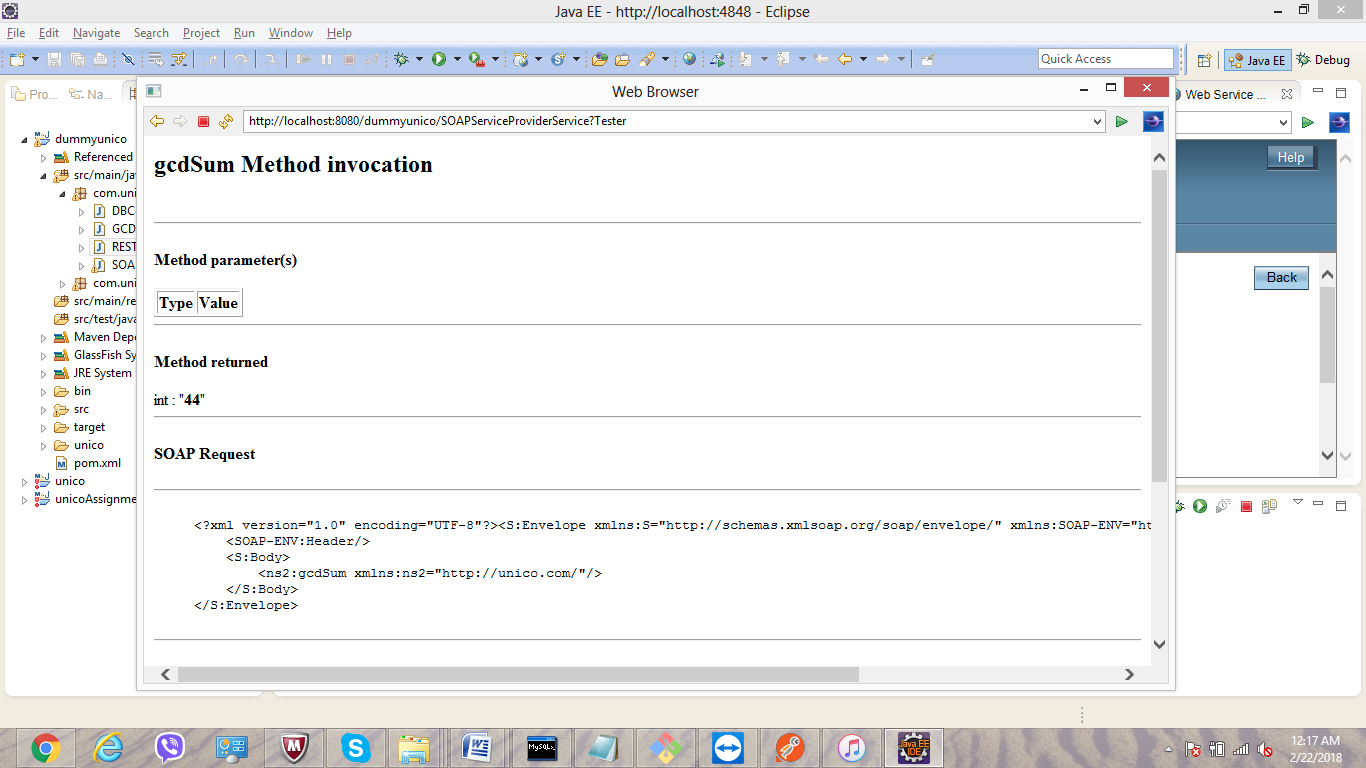
gcd();



gcdList()



gcdSum()

****

THANK YOU