

```

package CustomerResource;

import java.util.HashSet;
import java.util.Set;
import javax.ws.rs.core.Application;

/**
 *
 * @author soac-sh
 */
@javax.ws.rs.ApplicationPath("webresources")
public class ApplicationConfig extends Application {
    private Set<Object> singletons = new HashSet<Object>();
    private Set<Class<?>> empty = new HashSet<Class<?>>();
    public ApplicationConfig() {
        singletons.add(new CustomerClass());
    }
    @Override
    public Set<Class<?>> getClasses() {

        Set<Class<?>> resources = new java.util.HashSet<>();
        addRestResourceClasses(resources);
        return resources;
    }
    @Override
    public Set<Object> getSingletons() {
        return singletons;
    }

    /**
     * Do not modify addRestResourceClasses() method.
     * It is automatically populated with
     * all resources defined in the project.
     * If required, comment out calling this method in getClasses().
     */
    private void addRestResourceClasses(Set<Class<?>> resources) {
        resources.add(CustomerResource.CustomerClass.class);
    }
}

```

```

=====

package CustomerResource;

/**
 *
 * @author soac-sh
 */
public class Customer {
    private int id;
    private String firstName;
    private String lastName;
    private String street;
    private String city;
    private String state;
    private String zip;
    private String country;

    //@XmlAttribute
    public int getId()
    {
        return id;
    }

    public void setId(int id)
    {
        this.id = id;
    }
}

```

```
//@XmlElement(name = "first-name")
public String getFirstName()
{
    return firstName;
}

public void setFirstName(String firstName)
{
    this.firstName = firstName;
}

//@XmlElement(name = "last-name")
public String getLastName()
{
    return lastName;
}

public void setLastName(String lastName)
{
    this.lastName = lastName;
}

//@XmlElement
public String getStreet()
{
    return street;
}

public void setStreet(String street)
{
    this.street = street;
}

//@XmlElement
public String getCity()
{
    return city;
}

public void setCity(String city)
{
    this.city = city;
}

//@XmlElement
public String getState()
{
    return state;
}

public void setState(String state)
{
    this.state = state;
}

//@XmlElement
public String getZip()
{
    return zip;
}

public void setZip(String zip)
{
    this.zip = zip;
}

//@XmlElement
public String getCountry()
{
    return country;
}
```

```

    public void setCountry(String country)
    {
        this.country = country;
    }

    @Override
    public String toString()
    {
        return "Customer{" +
            "id=" + id +
            ", firstName='" + firstName + '\'' +
            ", lastName='" + lastName + '\'' +
            ", street='" + street + '\'' +
            ", city='" + city + '\'' +
            ", state='" + state + '\'' +
            ", zip='" + zip + '\'' +
            ", country='" + country + '\'' +
            '}';
    }
}

=====

package CustomerResource;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;

import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.WebApplicationException;
import javax.ws.rs.core.Response;
import javax.ws.rs.core.StreamingOutput;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.io.PrintStream;
import java.net.URI;
import java.util.Map;
import java.util.concurrent.ConcurrentHashMap;
import java.util.concurrent.atomic.AtomicInteger;
/**
 * REST Web Service
 *
 * @author stephenhole
 */
@Path("/customers")
public class CustomerClass {
    private Map<Integer, Customer> customerDB = new ConcurrentHashMap<Integer, Customer>();
    private AtomicInteger idCounter = new AtomicInteger();

    public CustomerClass() {
    }

    @POST
    @Consumes("application/xml")
    public Response createCustomer(InputStream is) {
        System.out.println("POST Method has been reached!");

        Customer customer = readCustomer(is);

```

```

customer.setId(idCounter.incrementAndGet());
//System.out.println(customer.getId());
customerDB.put(customer.getId(), customer);
//System.out.println("Created customer " + customer.getId());

//Note the additional elements in the path /ShoppingApplication/webresources
return Response.created(URI.create("/ShoppingApplication/webresources/customers/" +
customer.getId())).build();

}

@GET
@Path("/{id}")
@Produces("application/xml")
public StreamingOutput getCustomer(@PathParam("id") int id) {
    //System.out.println("GET part of the web service");
    //System.out.println("GET part of the web service" + customerDB.size());

    final Customer customer = customerDB.get(id);
    if (customer == null) {
        throw new WebApplicationException(Response.Status.NOT_FOUND);
    }
    return new StreamingOutput() {
        public void write(OutputStream outputStream) throws IOException,
            WebApplicationException {
            outputCustomer(outputStream, customer);
        }
    };
}

@PUT
@Path("/{id}")
@Consumes("application/xml")
public void updateCustomer(@PathParam("id") int id, InputStream is) {
    Customer update = readCustomer(is);
    Customer current = customerDB.get(id);
    if (current == null) throw new WebApplicationException(Response.Status.NOT_FOUND);

    current.setFirstName(update.getFirstName());
    current.setLastName(update.getLastName());
    current.setStreet(update.getStreet());
    current.setState(update.getState());
    current.setZip(update.getZip());
    current.setCountry(update.getCountry());
}

protected void outputCustomer(OutputStream os, Customer cust) throws IOException {
    PrintStream writer = new PrintStream(os);
    writer.println("<customer id=\"" + cust.getId() + "\">");
    writer.println("    <first-name>" + cust.getFirstName() + "</first-name>");
    writer.println("    <last-name>" + cust.getLastName() + "</last-name>");
    writer.println("    <street>" + cust.getStreet() + "</street>");
    writer.println("    <city>" + cust.getCity() + "</city>");
    writer.println("    <state>" + cust.getState() + "</state>");
    writer.println("    <zip>" + cust.getZip() + "</zip>");
    writer.println("    <country>" + cust.getCountry() + "</country>");
    writer.println("</customer>");
}

protected Customer readCustomer(InputStream is) {
    try {
        DocumentBuilder builder = DocumentBuilderFactory.newInstance().newDocumentBuilder();
        Document doc = builder.parse(is);
        Element root = doc.getDocumentElement();
        Customer cust = new Customer();
        if (root.getAttribute("id") != null && !root.getAttribute("id").trim().equals(""))
            cust.setId(Integer.valueOf(root.getAttribute("id")));
        NodeList nodes = root.getChildNodes();
        for (int i = 0; i < nodes.getLength(); i++) {
            Element element = (Element) nodes.item(i);

```

```

        if (element.getTagName().equals("first-name")) {
            cust.setFirstName(element.getTextContent());
        }
        else if (element.getTagName().equals("last-name")) {
            cust.setLastName(element.getTextContent());
        }
        else if (element.getTagName().equals("street")) {
            cust.setStreet(element.getTextContent());
        }
        else if (element.getTagName().equals("city")) {
            cust.setCity(element.getTextContent());
        }
        else if (element.getTagName().equals("state")) {
            cust.setState(element.getTextContent());
        }
        else if (element.getTagName().equals("zip")) {
            cust.setZip(element.getTextContent());
        }
        else if (element.getTagName().equals("country")) {
            cust.setCountry(element.getTextContent());
        }
    }
    return cust;
}
catch (Exception e) {
    throw new WebApplicationException(e, Response.Status.BAD_REQUEST);
}
}
}

```

```
package CustomerResource;
```

```

/**
 *
 * @author soac-sh
 */

```

```

import javax.ws.rs.client.ClientBuilder;
import javax.ws.rs.client.Client;
import javax.ws.rs.core.Response;
import javax.ws.rs.client.Entity;

```

```
public class MyClient {
```

```

    public static void main(String[] args) {
        Client client = ClientBuilder.newClient();
        try {
            System.out.println("Creating a New Customer");

```

```

        String xml = "<customer>"
            + "<first-name>Bill</first-name>"
            + "<last-name>Burke</last-name>"
            + "<street>256 Clarendon Street</street>"
            + "<city>Boston</city>"
            + "<state>MA</state>"
            + "<zip>02115</zip>"
            + "<country>USA</country>"
            + "</customer>";

```

```
System.out.println(xml);
```

```

Response response =
client.target("http://localhost:8080/ShoppingApplication/webresources/customers")
    .request().post(Entity.xml(xml));

```

```

if (response.getStatus() != 201)
    throw new RuntimeException("Failed to Create");

```

```
String location = response.getLocation().toString();
```

```

System.out.println("Location: " + location);
response.close();

System.out.println("Location: " + location);
System.out.println("*** GET Created Customer ***");
String customer = client.target(location).request().get(String.class);

System.out.println(customer);

String updateCustomer = "<customer>"
    + "<first-name>William</first-name>"
    + "<last-name>Burke</last-name>"
    + "<street>256 Clarendon Street</street>"
    + "<city>Boston</city>"
    + "<state>MA</state>"
    + "<zip>02115</zip>"
    + "<country>USA</country>"
    + "</customer>";

System.out.println(xml);

response = client.target(location).request().put(Entity.xml(updateCustomer));
if(response.getStatus() != 204)
    throw new RuntimeException("Failed to update");
response.close();

System.out.println("*** After Update ***");
customer = client.target(location).request().get(String.class);
System.out.println(customer);

} finally {
    client.close();
}
}
}

```

Output x

Java DB Database Process x GlassFish Server 4.1 x ShoppingApplication (run) x

```

run:
Creating a New Customer
<customer><first-name>Bill</first-name><last-name>Burke</last-name><street>256 Clarendon Street</street><city>Boston</city><state>MA</state><zip>02115</zip><country>USA</country></customer>
Location: http://localhost:8080/ShoppingApplication/webresources/customers/1
Location: http://localhost:8080/ShoppingApplication/webresources/customers/1
*** GET Created Customer **
<customer id="1">
  <first-name>Bill</first-name>
  <last-name>Burke</last-name>
  <street>256 Clarendon Street</street>
  <city>Boston</city>
  <state>MA</state>
  <zip>02115</zip>
  <country>USA</country>
</customer>

<customer><first-name>Bill</first-name><last-name>Burke</last-name><street>256 Clarendon Street</street><city>Boston</city><state>MA</state><zip>02115</zip><country>USA</country></customer>
*** After Update ***
<customer id="1">
  <first-name>William</first-name>
  <last-name>Burke</last-name>
  <street>256 Clarendon Street</street>
  <city>Boston</city>
  <state>MA</state>
  <zip>02115</zip>
  <country>USA</country>
</customer>

BUILD SUCCESSFUL (total time: 30 seconds)

```