JAX RS Web Services and Clients

Prerequisites

This lab explores JSR-311, a specification that is not supported by default in Rational Application Developer 8 environment or the WebSphere Application Server. In order to run this lab, an implementation of the JSR-311 specification must be obtained, and both the development and runtime environments must be able to link to it. Apache CXF provides an open source implementation of JSR-311. By downloading this implementation, and placing the associated ZIP and JAR files into either a shared directory, or the WEB-INF\lib folder of the web project, the labs will function as expected.

Goal

The goal of this lab is to demonstrate the basic concepts behind developing and testing RESTful web services with an implementation of the JSR-311 specification

Creating the ParameterData Service Bean

1. Create a new class named com.mcnz.rest.ParameterData with the following body implementation:

```
@Path("/pd/{id}")
@GET
@Produces("text/plain")
public String pathParameter(@PathParam("id") String id) {
        String result = "Id: " + id;
        return result;
}

@Path("/pd/")
@GET
@Produces("text/plain")
public String pathParameters(@PathParam("name")String name, @PathParam("id") String id) {
        String result = "Name: " + name +  " <br/>Id: " + id;
        return result;
}


@Path("/pd/")
@PUT
@Produces("text/plain")
public String queryParameters(@QueryParam("name") String name,@QueryParam("id")
                                                                        String id) {
        String result = "Name: " + name +  " <br/>Id: " + id;
        return result;
}

@Path("/pd/matrix")
@DELETE
@Produces("text/plain")
public String matrixParameters(@MatrixParam("name") String name, @MatrixParam("id")
                                                                        String id) {
        String result = "Name: " + name +  " <br/>Id: " + id;
        return result;
}

@Path("/pd/")
@POST
@Produces("text/plain")
public String cookieParameters(@CookieParam("name") String name, @CookieParam("id")
                                                                        String id) {
        String result = "Name: " + name +  " <br/>Id: " + id;
        return result;
}

@Path("/pd/")
@DELETE
@Produces("text/plain")
public String headerParameters(@HeaderParam("User-Agent") String browser,
                                        @HeaderParam("Accept") String accepts) {
        String result = "Browser: " + browser +  " <br/>Accepts: " + accepts;
        return result;
}
```

Writing and Running REST Client Tests

2. Create a new class called com.mcnz.rest.TestRest that contains a main method. Create tests for each method in the ParameterData RESTful web service by following the following basic steps:
    - Declare the URL for invoking the RESTful web service method as a String
    - Obtain an instance of the WebClient using the URL String
    - Set the accept property on the WebClient appropriately
    - Set any required properties on the WebClient for representing cookies, query data, matrix parameters, etc.
    - Invoke the web service using the WebClient instance and obtain the result as a String
    - Print out the returned String along with the name of the method being invoked to the console
3. When the TestRest class is created, restart the WebSphere Test environment, ensuring the RestWeb project is part of the deployment, and then run the main method of the TestRest class, monitoring the console output.