Introduction to Web Services (SOAP & REST)

Working with JAX-RS

Lesson Objectives

- What is REST
- Working with JAX - RS
- JAX – RS Annotations
- Creating JAX – RS web service
- Consuming RESTful service

What is REST

- **RESTful web services** are built to work best on the Web
- Representational State Transfer (REST) is an architectural style that specifies constraints, such as the uniform interface, that if applied to a web service induce desirable properties, such as performance, scalability, and modifiability, that enable services to work best on the Web
- In the REST architectural style, data and functionality are considered resources and are accessed using **Uniform Resource Identifiers (URIs)**, typically links on the Web
- In the REST architecture style, clients and servers exchange representations of resources by using a standardized interface like a URI

SOAP and REST

- Consider a scenario where a web service is going to query a phonebook application for the details of a given user when the user's ID is known
- A SOAP message request looks like:

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">

<soap:Body xmlns:m="http://webserice.learning.cg.com/">
<m:getUserDetails>
        <m:userId>12345</m:userId>
</m:getUserDetails>
</soap:Body>
</soap:Envelope>
```

- Consider a scenario where a web service is going to query a phonebook application for the details of a given user when the user's ID is known
- A REST request URI looks like:
- http://localhost/phonebook/UserDetails/12345

Working with JAX - RS

- JAX-RS is a Java programming language API designed to make it easy to develop applications that use the REST architecture
- Moreover annotations are used to simplify the development of RESTful web services
- JAX-RS annotations define resources and the actions that can be performed on those resources
- Creating a RESTful Root Resource Class
  - **Root Resource classes** are POJOs that are annotated with @Path or have at least one method annotated with @Path
  - **Resource methods** are methods of a resource class annotated with a request method designator such as @GET, @PUT, @POST, and @DELETE

JAX – RS Annotations

- These are the few annotations that can be used with REST
- @Path
  - The @Path annotation's value is a relative URI path indicating where the Java class will be hosted, for example, /helloworld.
- @GET
  - The Java method annotated with this request method designator will process HTTP GET requests.
- @POST
  - The Java method annotated with this request method designator will process HTTP POST requests.
- @PUT
  - The Java method annotated with this request method designator will process HTTP PUT requests.

JAX – RS Annotations

- @DELETE
  - The Java method annotated with this request method designator will process HTTP DELETE requests
- @FormParam
  - To bind HTML form parameters value to a Java method
- @Consumes
  - The @Consumes annotation is used to specify the MIME media types of representations a resource can consume that were sent by the client
- @Produces
  - The @Produces annotation is used to specify the MIME media types of representations a resource can produce and send back to the client: for example, "text/plain“ or application/json or application / xml
- @PathParam
  - The @PathParam annotation is a type of parameter that you can extract for use in your resource class

Creating JAX – RS Service

- Following code sample illustrates a simple example of a root resource class that uses JAX – RS annotations

```
import javax.ws.rs.GET;
import javax.ws.rs.Produces;
import javax.ws.rs.Path;

// The Java class will be hosted at the URI path "/helloworld"
@Path("/helloworld")
public class HelloWorldResource {

    // The Java method will process HTTP GET requests
    @GET
    // The Java method will produce content identified by the MIME Media
    // type "text/plain"
    @Produces("text/plain")
    public String getClichedMessage() {
        // Return some cliched textual content
        return "Hello World";
    }
}
```

Creating JAX – RS Service

- Following mapping is needed in web.xml :

```
<servlet>
    <servlet-name>jersey-serlvet</servlet-name>
    <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>
    <init-param>
        <param-name>jersey.config.server.provider.packages</param-name>
        <param-value>com.cg.learning.controller</param-value>
    </init-param>

    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    <servlet-name>jersey-serlvet</servlet-name>
    <url-pattern>/rest/*</url-pattern>
</servlet-mapping>
```

Creating JAX – RS Service

- More on @Path annotation and URI Path Templates
- Look at below @Path annotation:

```
@Path("/users/{username}")
```

In this kind of example, a user is prompted to type his or her name, and then a JAX-RS web service configured to respond to requests to this URI path template responds. For example, if the user types the user name "Galileo," the web service responds to the following URL:

```
http://example.com/users/Galileo
```

To obtain the value of the user name, the `@PathParam` annotation may be used on the method parameter of a request method, as shown in the following code example:

```
@Path("/users/{username}")
public class UserResource {

    @GET
    @Produces("text/xml")
    public String getUser(@PathParam("username") String userName) {
        ...
    }
}
```

Creating JAX – RS Service

URI Path Template	URI After Substitution
http://example.com/{name1}/{name2}/	http://example.com/james/gatz/
http://example.com/{question}/{question}/{question}	http://example.com/why/why/why/
http://example.com/maps/{location}	http://example.com/maps/MainStreet
http://example.com/{name3}/home/	http://example.com/james/home/

Consuming JAX – RS Service

- Here we would need a web client to consume a resource
- An jsp page can be created that can navigate to the specific URI.
  - The dynamic web project is created by name JAX-RS-HelloApp and URI's are mapped through /rest/*

```
<%@ page language="java" contentType="text/html; charset=ISO-
8859-1"
    pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>

<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
        <c:redirect url="http://localhost:9090/JAX-RS-
HelloApp/rest/helloworld"></c:redirect>
        <br>

</body>
</html>
```

Consuming JAX – RS Service

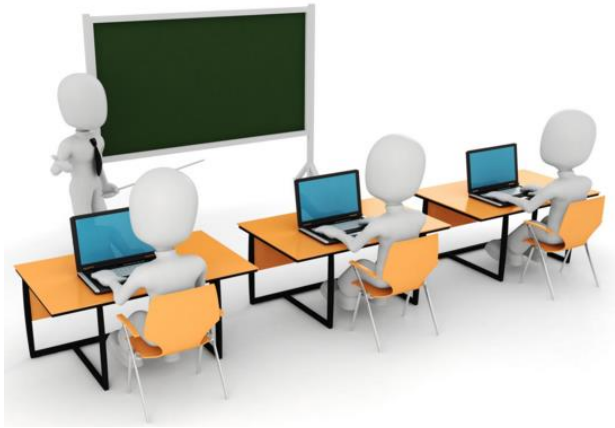
- JAX-RS-CRUD

Summary

- So far we have learnt:
  - What is REST architecture
  - How REST is any easy alternative to create a web service
  - REST annotations
  - Creating a RESTful service
  - Consuming a RESTful service

Lab

- Lab 2



Review Question

- Question1: Which of the following are true?
  - **Option1 :** Resource classes are POJOs that have at least one method annotated with @Path
  - **Option 2:** Resource methods are methods of a resource class annotated with a request method designator such as @GET, @PUT, @POST, or @DELETE
  - **Option 3:** @FormParam binds query parameters value to a Java method
- Question 2: The @Path annotation's value is a relative URI path indicating where the Java class will be hosted?
  - True
  - False

Review Question

- Question 3: _____ specifies a media type a resource can generate.
  - @PUT
  - @POST
  - @Produces
  - @Consumes