RESTful Web Services Developer's Guide

The HelloWorld-WebApp Application

This section discusses the `HelloWorld-WebApp` application that ships with Jersey. The `HelloWorld-WebApp` application is a “Hello, world” application that demonstrates the basics of developing a resource. There is a single class, `HelloWorldResource` that contains one method, `getClichedMessage` that produces a textual response to an HTTP `GET` request with a greeting that is sent back as plain text.

Annotating the Resource Class

The following code is the contents of the `com.sun.jersey.samples.helloworld.resources.HelloWorldResource` class:

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

@Path("/helloworld")
public class HelloWorldResource {

                @GET
                @Produces("text/plain")
                public String getClichedMessage() {
                        return "Hello World";
        }
}
```

In this example, the following annotations are processed at runtime:

- The `@Path` annotation specifies that the Java class will be hosted at the URI path `/helloworld` .
- The `@GET` annotation specifies that the Java method will process HTTP GET requests.
- The `@Produces` annotation specifies that the Java method will produce content identified by the MIME media type `text/plain` .

Configuring the Resource with the Runtime

The `helloworld-webapp/src/main/webapp/WEB-INF/web.xml` deployment descriptor for `HelloWorld-Webapp` contains the settings for configuring your resource with the JAX-RS API runtime:

```
<servlet>
     <servlet-name>Jersey Web Application</servlet-name>
     <servlet-class>
         com.sun.jersey.spi.container.servlet.ServletContainer
     </servlet-class>
                        <init-param>
                                        <param-name>
                                                        com.sun.jersey.config.property.packages
                                        </param-name>
                                        <param-value>
                                                        com.sun.jersey.samples.helloworld.resources
```

```
                                </param-value>
                        </init-param>
                        <load-on-startup>1</load-on-startup>
        </servlet>
 <servlet-mapping>
      <servlet-name>Jersey Web Application</servlet-name>
          <url-pattern>/*</url-pattern>
 </servlet-mapping>
```

The `com.sun.jersey.spi.container.servlet.ServletContainer` servlet is part of the JAX-RS API runtime, and works with the generated `HelloWorldResource` class to get the resources provided by your application. The `<servlet-mapping>` elements specify which URLs your application responds to relative to the context root of your WAR. Both the context root and the specified URL pattern prefix the URI Template specified in the `@Path` annotation in the resource class file. In the case of the `HelloWorld-WebApp` sample application, `@Path` is set to `/helloworld`, the URL pattern is set to the wild card character, and the context root specified in `sun-web.xml` is `/helloworld-webapp`, so the resource will respond to requests of the form:

```
http://<server>:<server port>/helloworld-webapp/helloworld
```

▼ Building and Running the HelloWorld-WebApp Application in NetBeans

1. Make sure that Maven is installed and configured, as described in Running the Jersey Examples.
2. Select File->Open Project in NetBeans IDE 6.5.
3. Navigate to `jersey/samples`, select `HelloWorld-WebApp`, and click OK.
4. Right click the `HelloWorld-WebApp` project in the Projects pane and select Run.

   This will generate the helper classes and artifacts for your resource, compile the classes, package the files into a WAR file, and deploy the WAR to your GlassFish instance.
5. If a web browser doesn't open automatically to display the output, you may need to open a web browser and enter the URL for the application.

   ```
   http://localhost:8080/helloworld-webapp/helloworld
   ```

   You will see the following output in your web browser:

   ```
   Hello World
   ```

▼ Building and Running the HelloWorld-WebApp Application with Maven

1. Make sure that Maven is installed and configured, as described in Running the Jersey Examples.
2. Open a terminal prompt and navigate to `jersey.home/samples/HelloWorld-WebApp`.
3. Enter `mvn glassfish:run` and press Enter.

   This will build, package, deploy, and run the web application. It will also start GlassFish if it is not running.
4. In a web browser navigate to:

   ```
   http://localhost:8080/helloworld-webapp/helloworld
   ```

   You will see the following output in your web browser:

   ```
   Hello World
   ```

Previous: The Jersey Sample Applications   **Next**: The Storage-Service Application