|  |
| --- |
| ­­ |
| **CURSO:** Análise e Desenvolvimento de Sistemas |
| **DISCIPLINA:** Desenvolvimento de Sistemas Distribuídos |
| **PROFESSOR:** Msc. Edjalma Queiroz da Silva |
| **Material de Apoio sobre WebServices RESTful** |

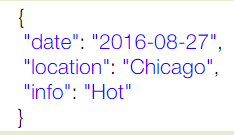
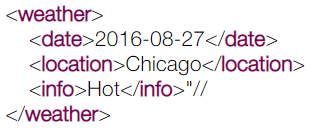
**Serviços da Web** geralmente fornecem os dados brutos que são difíceis de entender para a maioria dos usuários finais comuns e os **Serviços** da **Web** geralmente são retornados no formato XML ou JSON.

**O Serviço Web RESTful** é um **Serviço da Web** que usa a estrutura **REST** . **O REST** tem sido amplamente utilizado, substituindo **serviços da Web** baseados em **SOAP** e **WSDL** . **Os serviços da Web RESTful** são leves, fáceis de estender e manter.

**O REST** define um conjunto de princípios arquiteturais pelos quais você pode projetar serviços da Web que enfocam os recursos de um sistema, incluindo como os estados de recursos são endereçados e transferidos por **HTTP** por uma ampla gama de clientes escritos em diferentes idiomas. Se medido pelo número de serviços da Web que o utilizam, o **REST** surgiu apenas nos últimos anos como um modelo de design de serviço da Web predominante. Na verdade, o **REST** causou um impacto tão grande na Web que quase substituiu o design da interface baseada em **SOAP** e **WSDL** , porque é um estilo consideravelmente mais simples de usar.

**REST** é um conjunto de regras que visa criar um aplicativo de **serviço da Web de** acordo com quatro regras básicas abaixo:

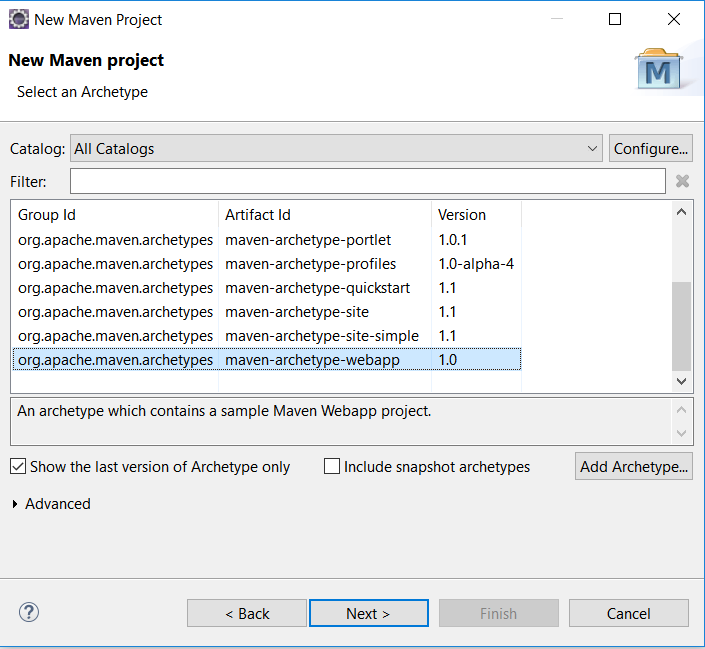
* Use métodos HTTP explicitamente.
* Seja sem estado.
* Expor URIs de estrutura de diretórios.
* Transferir XML, JavaScript Object Notation (JSON) ou ambos.
* Exemplos de saída para a requisição GET /weather/chicago/2016-08-27:

 ou 

JSON XML

***Vamos ao passo a passo para criação de um “Simple Restful WebService” usando o JAX-RS***

1. ***Comece criando um Projeto Maven com Archetype WebApp***



1. ***Em seguida nos seguintes campos preencha com os seguintes dados:***
   1. ***Grupo ID:*** *br.senai*
   2. ***Artifact ID:*** *JAXRS-WebService*
   3. ***Package:*** *br.senai.jaxrs*
2. ***Modifique o arquivo “pom.xml” para conter as seguintes configurações:***

<properties>

<jersey.version>1.18.3</jersey.version>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>com.sun.jersey</groupId>

<artifactId>jersey-servlet</artifactId>

<version>${jersey.version}</version>

</dependency>

<dependency>

<groupId>com.sun.jersey</groupId>

<artifactId>jersey-json</artifactId>

<version>${jersey.version}</version>

</dependency>

<dependency>

<groupId>commons-logging</groupId>

<artifactId>commons-logging</artifactId>

<version>1.2</version>

</dependency>

<!-- for web servlet -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>3.1.0</version>

<scope>provided</scope>

</dependency>

<!-- Some containers like Tomcat don't have jstl library -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

<scope>provided</scope>

</dependency>

</dependencies>

<build>

<finalName>JAXRS-WebService</finalName>

<plugins>

<!-- http://www.eclipse.org/jetty/documentation/current/jetty-maven-plugin.html -->

<plugin>

<groupId>org.eclipse.jetty</groupId>

<artifactId>jetty-maven-plugin</artifactId>

<version>9.4.12.v20180830</version>

</plugin>

<!-- Default is too old, update to latest to run the latest Spring 5 + jUnit 5 -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.0</version>

</plugin>

<!-- Default 2.2 is too old, update to latest -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-war-plugin</artifactId>

<version>3.2.2</version>

</plugin>

</plugins>

</build>

1. ***Modifique o arquivo “web.xml” localizado em src/main/webapp/WEB-INF conforme abaixo:***

<web-app>

<display-name>Archetype Created Web Application</display-name>

<servlet>

<servlet-name>jersey-serlvet</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>br.senai.controlador</param-value>**

</init-param>

<init-param>

<param-name>com.sun.jersey.api.json.POJOMappingFeature</param-name>

<param-value>true</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>

</web-app>

**Obs. O <param-value> em vermelho deve ser o mesmo nome do pacote onde estará a sua classe Controladora (Controller).**

1. ***Em seguida, crie a classe Pais com pacote “br.senai.modelo”.***

**package** br.senai.modelo;

**public** **class** Pais {

**int** id;

String nomePais;

**long** Populacao;

**public** Pais() {

}

**public** Pais(**int** id, String nomePais, **long** populacao) {

**super**();

**this**.id = id;

**this**.nomePais = nomePais;

Populacao = populacao;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getNomePais() {

**return** nomePais;

}

**public** **void** setNomePais(String nomePais) {

**this**.nomePais = nomePais;

}

**public** **long** getPopulacao() {

**return** Populacao;

}

**public** **void** setPopulacao(**long** populacao) {

Populacao = populacao;

}

}

1. ***Em seguida, crie a classe PaisControlador no pacote br.senai.controlador***

@Path("/paises")

**public** **class** PaisControlador {

PaisService paisService = **new** PaisService();

@GET

@Produces(MediaType.***APPLICATION\_JSON***)

**public** List getCountries() {

List listOfCountries = paisService.getAllCountries();

**return** listOfCountries;

}

@GET

@Path("/{id}")

@Produces(MediaType.***APPLICATION\_JSON***)

**public** Pais getPaisById(@PathParam("id") **int** id) {

**return** paisService.getPais(id);

}

@POST

@Produces(MediaType.***APPLICATION\_JSON***)

**public** Pais addPais(Pais Pais) {

**return** paisService.addPais(Pais);

}

@PUT

@Produces(MediaType.***APPLICATION\_JSON***)

**public** Pais updatePais(Pais Pais) {

**return** paisService.updatePais(Pais);

}

@DELETE

@Path("/{id}")

@Produces(MediaType.***APPLICATION\_JSON***)

**public** **void** deletePais(@PathParam("id") **int** id) {

paisService.deletePais(id);

}

}

1. ***Por fim, crie a classe PaisService no pacote br.senai.service***

**public** **class** PaisService {

**static** HashMap<Integer, Pais> *PaisIdMap* = *getPaisIdMap*();

**public** PaisService() {

**super**();

**if** (*PaisIdMap* == **null**) {

*PaisIdMap* = **new** HashMap<Integer, Pais>();

Pais indiaPais = **new** Pais(1, "India", 10000);

Pais chinaPais = **new** Pais(4, "China", 20000);

Pais nepalPais = **new** Pais(3, "Nepal", 8000);

Pais bhutanPais = **new** Pais(2, "Bhutan", 7000);

*PaisIdMap*.put(1, indiaPais);

*PaisIdMap*.put(4, chinaPais);

*PaisIdMap*.put(3, nepalPais);

*PaisIdMap*.put(2, bhutanPais);

}

}

**public** List getAllCountries() {

List countries = **new** ArrayList(*PaisIdMap*.values());

**return** countries;

}

**public** Pais getPais(**int** id) {

Pais Pais = *PaisIdMap*.get(id);

**return** Pais;

}

**public** Pais addPais(Pais Pais) {

Pais.setId(*PaisIdMap*.size() + 1);

*PaisIdMap*.put(Pais.getId(), Pais);

**return** Pais;

}

**public** Pais updatePais(Pais Pais) {

**if** (Pais.getId() <= 0)

**return** **null**;

*PaisIdMap*.put(Pais.getId(), Pais);

**return** Pais;

}

**public** **void** deletePais(**int** id) {

*PaisIdMap*.remove(id);

}

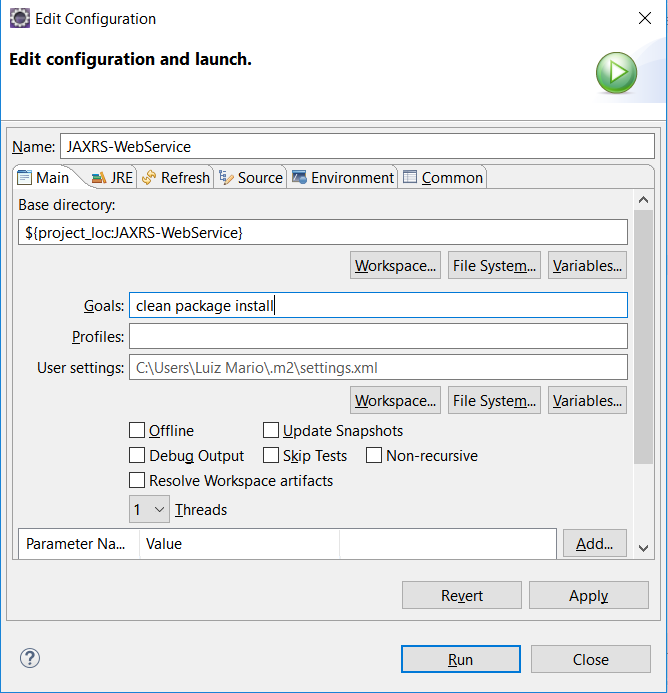
**public** **static** HashMap<Integer, Pais> getPaisIdMap() {

**return** *PaisIdMap*;

}

}

1. ***Para executar o programa, clique com botão direito no Projeto, vá em Run As -> Maven Build***
   1. ***Em Goals, digite “ clean package install “***



1. ***Após receber a mensagem de “BUILD SUCCESS”. Clique com Botão Direito no Projeto -> Run As -> Run on Server. Escolha o servidor de aplicação instalado (Tomcat), e execute.***
2. ***Vá no seu navegador e digite: “*** <http://localhost:8080/JAXRS-WebService/rest/paises> “ ou <http://localhost:8080/JAXRS-WebService/rest/paises/1>