Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχ. Και Μηχ. Υπολογιστών

Διαδίκτυο & Εφαρμογές

8ο εξάμηνο, Ροή Δ

Ακαδημαϊκή Περίοδος : 2011-2012

3η Εργαστηριακή Άσκηση

Γερακάρης Βασίλης
<vgerak@gmail.com>
Α.Μ. :03108092

08/06/2012

Εισαγωγικά:

Σε αυτή την εργαστηριακή άσκηση κληθήκαμε να γράψουμε μια υπηρεσία SOAP - RPC, η οποία διαχειρίζεται εγγραφές σε μορφή JavaBean, οι οποίες θα περιέχουν και άλλα JavaBean ως ιδιότητες.

Για την ανάπτυξη αυτής της εφαρμογής χρησιμοποιήθηκε το IDE Eclipse Java EE Indigo για τη συγγραφή του κώδικα, ο Apache Tomcat 6.0.35 web server, πάνω σε λειτουργικό MacOS X.

Source Code:

Παρατίθενται τα αρχεία του πηγαίου κώδικα που συγγράφηκαν για τις ανάγκες της άσκησης.

BVCatalog.Java

Έχει γίνει προσθήκη της συνάρτησης delV(string model) όπως ζητείται, και οι εγγραφές πλέον εμπεριέχουν ένα bean

```
1  package BVShop;
2  /*if present 'package MUST be the first statement,*/
3  /* (possibly) followed by 'import'*/
4  import java.util.Hashtable;
5
6  public class BVCatalog {
7      /**The vehicles as Hashtable*/
8      /** the car Model is the key for finding entries  */
9      private Hashtable catalog;
10
11     public BVCatalog(){
12         /** the old catalog remains*/
13         /** but now the catalog (Hashtable) will contain also beans !!!*/
14         catalog=new Hashtable();
15
16         /**Some content - we NOW populate with some objects!!*/
17         addV(new VehicleBean("Buick","GM","1948",new MotorBean("2000", "V12", "620")));
18         addV(new VehicleBean("Mustang","Ford","1960",new MotorBean("4000", "V8", "400")));
19         addV(new VehicleBean("4CV","Citroen","1950",new MotorBean("1600", "6", "110")));
20         addV(new VehicleBean("Jeep","GM", "1942",new MotorBean("1800", "4", "100")));
21         addV(new VehicleBean("Beatle","Volkswagen","1938",new MotorBean("5000", "V6", "380")));
22     }
23     /** FIRST METHOD TO BE EXPOSED - addV */
24     /** input argument is now a single ('complex') object !!*/
25     /** nothing is returned*/
26     public void addV(VehicleBean  vObj) {
27         if (vObj == null){
28             throw new IllegalArgumentException("The object provided cannot be null.");
29         }
30         /** entry format for the Hashtable: key,data */
31         /** vObj.getVModel gets the 'key' out of the bean object*/
32         catalog.put(vObj.getVModel(),vObj);
33         System.out.println("Addition at server side: " + vObj.getVModel());
34     }
35
```

```
36      /** SECOND METHOD TO BE EXPOSED - getVehicleBean */
37      /** input argument is a String */
38      /** a ('complex') object is returned */
39      public VehicleBean getVehicleBean(String model) {
40          if (model==null){
41              throw new IllegalArgumentException("carModel cannot be null.");
42          }
43
44          //Return the requested vehicle - NOW AS AN OBJECT !!!!
45          return (VehicleBean)catalog.get(model);
46      }
47
48      /** THIRD METHOD TO BE EXPOSED - listV */
49      /** NO input argument - a whole table is returned */
50      /** ... , and that with ('complex') objects as entries !!!!*/
51      public Hashtable listV(){
52          return catalog;
53      }
54
55      /** FOURTH METHOD TO BE EXPOSED - delV */
56      /** input argument is a String */
57      public void delV(String model){
58          catalog.remove(model);
59      }
60  }
```

MotorBean.java

Η περιγραφή του MotorBean

```
1  package BVShop;
2  //if present 'package' MUST be the first statement (possibly) followed by 'import'
3  // The following class follows the structure of a 'Java Bean'
4
5  public class MotorBean {
6      /**The properties of the MotorBean*/
7      String MCc;
8      String MNo_cylinders;
9      String MPs;
10
11     /** The following non-argument constructor MUST be always present*/
12     /** for this class to be a Java Bean*/
13     public MotorBean(){}
14
15     /** Another constructor CAN also be always present*/
16     /**  the following initializes all properties whenever a new object is instantiated (with
'new') - */
17     public MotorBean(String mcc,String cyli, String mps){
18         this.MCc= mcc; this.MNo_cylinders = cyli; this.MPs= mps;
19     }
20
21     /** get   set methods for all properties MUST be present*/
22     public String getMCc(){return MCc;}
23     public void setMCc(String mcc){this.MCc= mcc;}
24     public String getMNo_cylinders(){return MNo_cylinders;}
25     public void setMNo_cylinders(String cyli){this.MNo_cylinders= cyli;}
26     public String getMPs(){return MPs;}
27     public void setMPs(String mps){this.MPs= mps;}
28
29     //toString is an in-built method for every Java object. Outputs an informative string
30     public String toString(){
```

```
31          return  MCc + " cc has " + MNo_cylinders + " and offers " + MPs + " horsepower";
32      }
33  }
```

VehicleBean.java

Η νεα περιγραφή του VehicleBean

```
 1  package BVShop;
 2  //if present 'package' MUST be the first statement (possibly) followed by 'import'
 3  // The following class follows the structure of a 'Java Bean'
 4
 5  public class VehicleBean {
 6      /**The properties of the VehicleBean*/
 7      String VModel;
 8      String VManufacturer;
 9      String VYear;
10      MotorBean Motor;
11
12      /** The following non-argument constructor MUST be always present*/
13      /** for this class to be a Java Bean*/
14      public VehicleBean(){}
15
16      /** Another constructor CAN also be always present*/
17      /**  the following initializes all properties whenever a new object is instantiated (with
'new') - */
18      public VehicleBean(String model,String manu,String year, MotorBean motor){
19          this.VModel= model; this.VManufacturer = manu; this.VYear= year; this.Motor = motor;
20      }
21
22      /** get   set methods for all properties MUST be present*/
23      public String getVModel(){return VModel;}
24      public void setVModel(String model){this.VModel= model;}
25      public String getVManufacturer(){return VManufacturer;}
26      public void setVManufacturer(String manu){this.VManufacturer= manu;}
27      public String getVYear(){return VYear;}
28      public void setVYear(String year){this.VYear= year;}
29      public MotorBean getMotor(){return Motor;}
30      public void setMotor(MotorBean motor){this.Motor = motor;}
31
32      //toString is an in-built method for every Java object. Outputs an informative string
33      public String toString(){
34          return "'" + VModel + "' by " + VManufacturer + " (" + VYear + ")";
35      }
36  }
```

BVAdderLister.java

```
 1  package BVShop;
 2  import java.net.URL;
 3  import java.util.Enumeration;
 4  import java.util.Hashtable;
 5  import java.util.Vector;
 6  import org.apache.soap.Constants;
 7  import org.apache.soap.Fault;
 8  import org.apache.soap.SOAPException;
 9  import org.apache.soap.encoding.SOAPMappingRegistry;
```

```
10 import org.apache.soap.encoding.soapenc.BeanSerializer;
11 import org.apache.soap.encoding.soapenc.StringDeserializer;
12 import org.apache.soap.rpc.Call;
13 import org.apache.soap.rpc.Parameter;
14 import org.apache.soap.rpc.Response;
15 import org.apache.soap.util.xml.QName;
16
17 public class BVAdderLister {
18
19     public void addlist(URL url, String model, String manufacturer, String year, String
modelDel)
20             throws SOAPException{
21
22         // Build the object with the data given by user
23         VehicleBean vObj = new VehicleBean(model, manufacturer, year);
24
25         //VehicleBean must now be mapped ... so SOAP can use it
26         SOAPMappingRegistry reg = new SOAPMappingRegistry();
27         BeanSerializer serializer = new BeanSerializer();
28         reg.mapTypes(Constants.NS_URI_SOAP_ENC,
29                 new QName("urn:VBean_xmlns","vObj"),
30                 VehicleBean.class, serializer, serializer);
31
32         //Build the Call object
33         Call call = new Call();
34         //How to map, where to send, method to call, encoding "style"
35         call.setSOAPMappingRegistry(reg);
36         call.setTargetObjectURI("urn:BVehicleCatalog");
37         call.setMethodName("addV");
38         call.setEncodingStyleURI(Constants.NS_URI_SOAP_ENC);
39
40
41         //------------  A D D I N G ----------------------------------------------------
42         System.out.println("Adding vehicle model '" + model + "' by " + manufacturer + " of
year " + year);
43
44         // Set up the parameters of the call
45         Vector params = new Vector();
46
47         //in the instructions given to the 'Serializer' - see Depl. Descr.
48         params.addElement(new Parameter("vObj", VehicleBean.class, vObj, null));
49         call.setParams(params);
50
51         // Invoke the call
52         Response response;
53         response = call.invoke(url, "");
54         //We do not expect something back, unless there is a fault!!
55         if (!response.generatedFault())
56         {        System.out.println("Server reported NO FAULT while adding vehicle");}
57         else {        Fault fault = response.getFault();
58         System.out.println("Server reported FAULT while adding:");
59         System.out.println(fault.getFaultString());}
60
61
62         //------------  D E L E T I N G ------------------------------------------------
63         //We use the same Call Object and change this as appropriate
64         /* Another method is now called*/
65         call.setMethodName("delV");
66         Vector par = new Vector();
67         par.addElement(new String(modelDel));
```

```
68          call.setParams(par);
69          // Invoke the call;
70          response = call.invoke(url, "");
71          if (!response.generatedFault())
72          {        System.out.println("Server reported NO FAULT while removing vehicle");}
73          else {        Fault fault = response.getFault();
74          System.out.println("Server reported FAULT while removing:");
75          System.out.println(fault.getFaultString());}
76
77
78          //------------  L I S T I N G ----------------------------------------------------
79          //We use the same Call Object and change this as appropriate
80          /* Another method is now called*/
81          call.setMethodName("listV");
82          /* NO parameters here !!*/
83          /*(we cannot have a call with arguments as before)*/
84          call.setParams(null);
85          // Invoke the call; here we expect something back !!
86          response = call.invoke(url, "");
87
88          /*Extract the value returned in the form of a 'Parameter' Object*/
89          Parameter returnValue = response.getReturnValue();
90          /*Cast the 'Parameter' Object onto a Hashtabel Object*/
91          Hashtable catalog = (Hashtable)returnValue.getValue();
92          Enumeration e = catalog.keys();
93          while (e.hasMoreElements()) {
94              String VModel = (String)e.nextElement();
95              VehicleBean vo = (VehicleBean)catalog.get(VModel);
96              System.out.println("  '" + vo.getVModel() + "' by " + vo.getVManufacturer() +
97                      ", year " + vo.getVYear());
98          }
99      }
100
101
102
103     public static void main(String[] args) {
104         if (args.length != 4) {
105             for (int i = 0; i < args.length; ++i) {
106                 System.out.println("arg =" + args[i]);
107             }
108             System.out.println("Put model, manufacturer, year and model to delete as
arguments !!");
109             return;
110         }
111
112         try {
113             // URL for SOAP server to connect to
114             URL urlink = new URL("http://localhost:8080/webApp3/services/BVCatalog");
115
116             // Get values for the new vehicle
117             String model = args[0];
118             String manufacturer = args[1];
119             String year = args[2];
120             String modelDel = args[3];
121             //Add the new vehicle - Delete one aswell
122             BVAdderLister adderlister = new BVAdderLister();
123             adderlister.addlist(urlink, model, manufacturer, year, modelDel);
124         }  catch (Exception e) {e.printStackTrace();}
125     }
126 }
```