# TDM10 d' Informatique Répartie
# EJB et Rest

ASI4 - INSA Rouen
CORRECTION

## Calculatrice distribuée sur Complexes en EJB

Le but de cet exercice est de réaliser une calculatrice distribuée sur Complexes en EJB. Pour cela, vous respecterez les contraintes suivantes :

- Votre calculatrice supportera 4 types d'opération : additions, soustractions, multiplications et divisions. Le serveur proposera donc 4 services distribués et ne travaillera que sur des Complexes (Objet à créer).
- Vous créerez un répertoire spécifique pour le lancement du client contenant uniquement les fichiers nécessaires à son lancement.

### *Correction*

#### *Complexe.java*

```
package calculatriceEJB;

import java.io.Serializable;

public class Complexe implements Serializable {
  public float re,im;

  public Complexe() {
    this.re=(float)0;
    this.im=(float)0;
  }

  public Complexe(int re,int im) {
    this.re=(float)re;
    this.im=(float)im;
  }

  public Complexe(float re,float im) {
    this.re=re;
    this.im=im;
  }

  public float getRe() {
    return this.re;
  }

  public void setRe(float re) {
    this.re = re;
  }

  public float getIm() {
    return this.im;
  }

  public void setIm(float im) {
    this.im = im;
  }

  public String toString(){
    String resultat="";
    if (this.re!=0)
      resultat=String.valueOf(this.re);
    if (this.im!=0) {
      if (this.im>0){
        if (resultat!="")
          resultat=resultat+"+"+String.valueOf(this.im);
        else
          resultat=String.valueOf(this.im);
      }else
        resultat=resultat+String.valueOf(this.im);
      resultat=resultat+"i";
    }
    return resultat;
  }
```

```
}
```

### *CalculatriceRemote.java*

```
package calculatriceEJB;

import javax.ejb.Remote;
import calculatriceEJB.Complexe;

@Remote
public interface CalculatriceRemote {
   public Complexe additionner(Complexe c1, Complexe c2);
   public Complexe soustraire(Complexe c1, Complexe c2);
   public Complexe multiplier(Complexe c1, Complexe c2);
   public Complexe diviser(Complexe c1, Complexe c2) throws ArithmeticException;
}
```

### *CalculatriceBean.java*

```
package calculatriceEJB;

import javax.ejb.Stateless;

@Stateless
public class CalculatriceBean implements CalculatriceRemote {
  public Complexe additionner(Complexe c1, Complexe c2) {
    Complexe z = new Complexe();
    z.re=c1.re+c2.re;
    z.im=c1.im+c2.im;
    return z;
  }

  public Complexe soustraire(Complexe c1, Complexe c2) {
    Complexe z = new Complexe();
    z.re=c1.re-c2.re;
    z.im=c1.im-c2.im;
    return z;
  }

  public Complexe multiplier(Complexe c1, Complexe c2) {
    Complexe z = new Complexe();
    z.re = c1.re * c2.re - c1.im * c2.im;
    z.im = c1.re * c2.im + c1.im * c2.re;
    return z;
  }

  public Complexe diviser(Complexe c1, Complexe c2) throws ArithmeticException {
    float r = c2.re*c2.re + c2.im*c2.im;
    Complexe z = new Complexe();
    if (r==0) {
      throw new ArithmeticException("Division par zero !");
    }else {
      z.re = (c1.re * c2.re + c1.im * c2.im) / r;
      z.im = (c1.im * c2.re - c1.re * c2.im) / r;
      return z;
    }
  }
}
```

### *Client.java*

```
import javax.naming.InitialContext;
import calculatriceEJB.CalculatriceRemote;
import calculatriceEJB.Complexe;

public class Client {
  public static void main(String[] args) {
    try {
/*     Complexe c1=new Complexe(2,4), c2=new Complexe(1,2), un=new Complexe(1,0), zero=new Complexe(0,0);
      CalculatriceRemote calculatrice = (CalculatriceRemote)InitialContext.doLookup("CalculatriceEntiers/
           CalculatriceBean!calculatriceEJB.CalculatriceRemote");
      System.out.println(calculatrice.additionner(c1,c2));
      System.out.println(calculatrice.soustraire(c1,c2));
      System.out.println(calculatrice.multiplier(c1,c2));
      System.out.println(calculatrice.diviser(c1,c2));
      System.out.println(calculatrice.diviser(c1,un));
      System.out.println(calculatrice.diviser(c1,zero));*/

      Complexe c1=new Complexe(), c2=new Complexe(), un=new Complexe(), zero=new Complexe(), result;
      c1.setRe(2); c1.setIm(4);
      c2.setRe(1); c2.setIm(2);
      un.setRe(1); un.setIm(0);
      zero.setRe(0); zero.setIm(0);
      CalculatriceRemote calculatrice = (CalculatriceRemote)InitialContext.doLookup("CalculatriceEntiers/
           CalculatriceBean!calculatriceEJB.CalculatriceRemote");

      result = calculatrice.additionner(c1,c2);
      System.out.println("(2+4i) + (1+2i) = " + result.getRe() + " " + result.getIm() + "i");

      result = calculatrice.soustraire(c1,c2);
```

```
      System.out.println("(2+4i) - (1+2i) = " + result.getRe() + " " + result.getIm() + "i");

      result = calculatrice.multiplier(c1,c2);
      System.out.println("(2+4i) x (1+2i) = " + result.getRe() + " " + result.getIm() + "i");

      result = calculatrice.diviser(c1,c2);
      System.out.println("(2+4i) / (1+2i) = " + result.getRe() + " " + result.getIm() + "i");

      result = calculatrice.diviser(c1,un);
      System.out.println("(2+4i) / 1 = " + result.getRe() + " " + result.getIm() + "i");

      result = calculatrice.diviser(c1,zero);
      System.out.println("(2+4i) / 0 = " + result.getRe() + " " + result.getIm() + "i");
    } catch (Exception e) {
      System.out.println(e);
      e.printStackTrace();
    }
  }
}
```

***compile.sh***

```
CLASSPATH=.:$JBOSS_HOME/modules/javax/ejb/api/main/jboss-ejb-api_3.1_spec-1.0.1.Final.jar

javac -cp $CLASSPATH calculatriceEJB/*.java
jar cvf CalculatriceEntiers.jar calculatriceEJB/*.class
cp CalculatriceEntiers.jar $JBOSS_HOME/standalone/deployments

cp calculatriceEJB/CalculatriceRemote.class Client/calculatriceEJB/
cp calculatriceEJB/Complexe.class Client/calculatriceEJB/
javac Client/Client.java
```

***run.sh***

```
CLASSPATH=$JBOSS_HOME/bin/client/jboss-client.jar:.
cd Client
java -cp $CLASSPATH Client
cd ..
```

**NB : il s'agit d'Informatique Répartie donc pensez à tester votre programme avec serveur et client sur des machines différentes !**

# Service d'annuaire en Rest

Le but de cet exercice est de réaliser un service d'annuaire en Rest. Celui-ci permet de connaître le bureau d'appartenance d'une personne et réciproquement. La logique métier est fournie en annexe (fichier `annuaire.tar.gz`), il ne reste qu'à distribuer le service et créer un client. Pour cela, vous respecterez les contraintes énumérées ci-dessous.

- Le serveur respectera l'API suivante :
  - GET, `annuaire/personnes` → retourne la liste des personnes et leur bureau associé ;
  - GET, `annuaire/personnes/{bureau}` → retourne la personne associée à un bureau ;
  - GET, `annuaire/bureaux/{nom}` → retourne le bureau associé à une personne ;
  - POST, `annuaire/personnes` → ajoute une nouvelle personne à l'annuaire ;
  - PUT, `annuaire/personnes` → met à jour les informations d'une personne de l'annuaire ;
  - DELETE, `annuaire/personnes/{nom}` → supprime une personne de l'annuaire ;
- Le client sera accessible en ligne de commande, avec comme premier paramètre la méthode Http appelée (GET, POST, PUT ou DELETE), suivi éventuellement des paramètres nécessaires à l'appel des différentes méthodes de l'API Rest proposée.

**NB : il s'agit d'Informatique Répartie donc pensez à tester votre programme avec serveur et client sur des machines différentes !**

### *Correction*

***annuaire/Personne.java***

```
package annuaire;

import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement(name="personne")
public class Personne {
    private String nom, bureau;

    public Personne() {}

    public Personne(String name, String office) {
        this.nom = name;
        this.bureau = office;
```

```
    }

    public String getNom() {
        return this.nom;
    }

    @XmlElement
    public void setNom(String name) {
        this.nom = name;
    }

    public String getBureau() {
        return this.bureau;
    }

    @XmlElement
    public void setBureau(String office) {
        this.bureau = office;
    }

    @Override
    public String toString() {
        return this.nom + " : " + this.bureau;
    }
}
```

### *annuaire/AnnuaireModule.java*

```
package annuaire;

import com.google.inject.Binder;
import com.google.inject.Module;

public class AnnuaireModule implements Module
{
   public void configure(final Binder binder)
   {
      binder.bind(annuaire.AnnuaireRessource.class);
   }
}
```

### *annuaire/AnnuaireRessource.java*

```
package annuaire;

import javax.ws.rs.*;
import javax.ws.rs.core.Response;

import java.util.HashMap;
import java.util.Iterator;

import annuaire.Personne;

@Path("annuaire")
public class AnnuaireRessource
{
    private static HashMap<String,String> listeDePersonnes = new HashMap<String,String>();
    private static HashMap<String,String> listeDeBureaux = new HashMap<String,String>();

    @GET
    @Path("bureaux/{nom}")
    @Produces("text/plain")
    public String getBureau(@PathParam("nom") final String nom) {
        return AnnuaireRessource.listeDeBureaux.get(nom);
    }

    @GET
    @Path("personnes/{bureau}")
    @Produces("text/plain")
    public String getPersonne(@PathParam("bureau") final String bureau) {
        return AnnuaireRessource.listeDePersonnes.get(bureau);
    }

    @GET
    @Path("personnes")
    @Produces("text/plain")
    public String toString() {
        String annuaire;
        if(AnnuaireRessource.listeDePersonnes.size() == 0)
            annuaire = "Liste vide";
        else {
            annuaire = "";
            for(Iterator i=AnnuaireRessource.listeDeBureaux.keySet().iterator();i.hasNext();){
                String nom = (String)i.next();
                annuaire += nom + " : " + AnnuaireRessource.listeDeBureaux.get(nom);
                if(i.hasNext())
                    annuaire += "\n";
            }
        }
        return annuaire;
```

```
    }

    @POST
    @Path("personnes")
    @Consumes("application/xml")
    public Response addPersonne(Personne personne) {
        AnnuaireRessource.listeDeBureaux.put(personne.getNom(), personne.getBureau());
        AnnuaireRessource.listeDePersonnes.put(personne.getBureau(), personne.getNom());
        return Response.status(200).entity(personne.toString()).build();
    }

    @PUT
    @Path("personnes")
    @Consumes("application/xml")
    public Response updateTerm(Personne personne) {
        String ancienBureau = this.getBureau(personne.getNom());
        AnnuaireRessource.listeDeBureaux.remove(personne.getNom());
        AnnuaireRessource.listeDeBureaux.put(personne.getNom(), personne.getBureau());
        AnnuaireRessource.listeDePersonnes.remove(ancienBureau);
        AnnuaireRessource.listeDePersonnes.put(personne.getBureau(), personne.getNom());
        return Response.status(200).entity(personne.toString()).build();
    }

    @DELETE
    @Path("personnes/{nom}")
    public Response removeTerm(@PathParam("nom") final String nom) {
        String bureau = AnnuaireRessource.listeDeBureaux.get(nom);
        AnnuaireRessource.listeDeBureaux.remove(nom);
        AnnuaireRessource.listeDePersonnes.remove(bureau);
        return Response.status(200).entity(nom).build();
    }
}
```

### *webapp/WEB-INF/web.xml*

```
<?xml version="1.0"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
        "http://java.sun.com/dtd/web-app_2_3.dtd">

<web-app>
    <display-name>Dictionnaire REST</display-name>

    <context-param>
        <param-name>resteasy.guice.modules</param-name>
        <param-value>annuaire.AnnuaireModule</param-value>
    </context-param>

    <listener>
        <listener-class>
            org.jboss.resteasy.plugins.guice.GuiceResteasyBootstrapServletContextListener
        </listener-class>
    </listener>

    <servlet>
        <servlet-name>Resteasy</servlet-name>
        <servlet-class>
            org.jboss.resteasy.plugins.server.servlet.HttpServletDispatcher
        </servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>Resteasy</servlet-name>
        <url-pattern>/*</url-pattern>
    </servlet-mapping>

</web-app>
```

### *client/AnnuaireClient.java*

```
package client;

import org.jboss.resteasy.client.jaxrs.ResteasyClient;
import org.jboss.resteasy.client.jaxrs.ResteasyClientBuilder;
import org.jboss.resteasy.client.jaxrs.ResteasyWebTarget;
import javax.ws.rs.core.Response;
import javax.ws.rs.client.Entity;

import annuaire.Personne;

public class AnnuaireClient {
  public static void main(String[] args) {
    ResteasyClient client = new ResteasyClientBuilder().build();
    ResteasyWebTarget target;
    Response response;
    String baseURL = "http://localhost:8080/AnnuaireRest-1.0/annuaire";
    Personne personne;

    if(args[0].equals("GET")) {
        if(args.length==1) {
            target = client.target(baseURL + "/personnes");
```

```
            response = target.request().get();
            System.out.println(response.readEntity(String.class));
            response.close();
        } else {
            if(args[1].equals("bureau"))
                target = client.target(baseURL + "/bureaux/" + args[2]);
            else
                target = client.target(baseURL + "/personnes/" + args[2]);
            response = target.request().get();
            System.out.println(args[2] + " : " + response.readEntity(String.class));
            response.close();
        }
    } else if(args[0].equals("POST")) {
        personne = new Personne(args[1], args[2]);
        target = client.target(baseURL + "/personnes");
        response =target.request().post(Entity.entity(personne, "application/xml;charset=UTF-8"));
        System.out.println("POST : " + response.getStatus());
        response.close();
    } else if(args[0].equals("PUT")) {
        personne = new Personne(args[1], args[2]);
        target=client.target(baseURL + "/personnes");
        response = target.request().put(Entity.entity(personne, "application/xml;charset=UTF-8"));
        System.out.println("PUT : " + response.getStatus());
        response.close();
    } else if(args[0].equals("DELETE")) {
        target = client.target(baseURL + "/personnes/" + args[1] + "");
        response = target.request().delete();
        System.out.println("DELETE : " + response.getStatus());
        response.close();
    } else {
        System.out.println("Méthode non supportée");
    }
  }
}
```

***runClient.sh***

```
java -cp target/classes:lib/resteasy-client-3.0.9.Final.jar:lib/jaxrs-api-3.0.9.Final.jar:lib/httpclient-4.2.3.jar:
     lib/httpcore-4.2.3.jar:lib/commons-logging-1.1.1.jar:lib/commons-io-2.3.jar:lib/resteasy-jaxrs-3.0.9.Final.jar
     :lib/resteasy-jaxb-provider-3.0.9.Final.jar client.AnnuaireClient $1 $2 $3
```

# Remarques

— Vous aurez alors accès à la correction du TDM durant une semaine à compter de la fin du TDM.

— **Déposez un compte-rendu de 2 pages** `TDMEJBetREST-NomPrenom.pdf` **sur moodle chez TOUTES les personnes du binôme. Ce CR contiendra les informations que vous jugerez nécessaires.**

— Votre CR sera disponible pour vous lors de l'examen machine.