

TDM SOAP – Correction

serveurEntiers

```
CalculatriceWebService.java ✕ CalculatriceWSImpl.java ✕ CalculatriceWSPublisher.java ✕ EntierNegatif.java ✕
1  package      calculatriceWebService;
2
3  import javax.xml.ws.WebService;
4  import javax.xml.ws.WebMethod;
5  import javax.xml.ws.WebParam;
6  import javax.xml.ws.soap.SOAPBinding;
7
8  @WebService(name="CalculatriceWebService")
9  @SOAPBinding(style=SOAPBinding.Style.RPC)
10 public interface CalculatriceWebService {
11     @WebMethod(action="additionner")
12     public int additionner(@WebParam(name="z1") int z1, @WebParam(name="z2") int z2) throws EntierNegatif;
13     @WebMethod(action="soustraire")
14     public int soustraire(@WebParam(name="z1") int z1, @WebParam(name="z2") int z2) throws EntierNegatif;
15     @WebMethod(action="multiplier")
16     public int multiplier(@WebParam(name="z1") int z1, @WebParam(name="z2") int z2) throws EntierNegatif;
17     @WebMethod(action="diviser")
18     public int diviser(@WebParam(name="z1") int z1, @WebParam(name="z2") int z2) throws EntierNegatif;
19 }
20
```

```
CalculatriceWSImpl.java X CalculatriceWebService.java X CalculatriceWSPublisher.java X EntierNegatif.java X
1 package calculatriceWebService;
2
3 import javax.jws.WebService;
4
5 @WebService(endpointInterface="calculatriceWebService.CalculatriceWebService")
6 public class CalculatriceWSImpl implements CalculatriceWebService {
7
8     @Override
9     public int additionner(int e1, int e2) throws EntierNegatif {
10         int resultat = e1+e2;
11         if (e1<0 || e2<0)
12             throw new EntierNegatif();
13         else
14             return resultat;
15     }
16
17     @Override
18     public int soustraire(int e1, int e2) throws EntierNegatif {
19         int resultat = e1-e2;
20         if (e1<0 || e2<0 || resultat<0)
21             throw new EntierNegatif();
22         else
23             return resultat;
24     }
25
26     @Override
27     public int multiplier(int e1, int e2) throws EntierNegatif {
28         int resultat = e1*e2;
29         if (e1<0 || e2<0)
30             throw new EntierNegatif();
31         else
32             return resultat;
33     }
34
35     @Override
36     public int diviser(int e1, int e2) throws EntierNegatif {
37         int resultat = e1/e2;
38         if (e1<0 || e2<0)
39             throw new EntierNegatif();
40         else
41             return resultat;
42     }
43 }
44
```

```
EntierNegatif.java X CalculatriceWSImpl.java X CalculatriceWebService.java X CalculatriceWSPublisher.java X
1 package calculatriceWebService;
2
3 public class EntierNegatif extends Exception {
4     public EntierNegatif(){
5         super("Entier non positif !");
6     }
7 }
8
```

```

CalculatriceWSPublisher.java ✕ EntierNegatif.java ✕ CalculatriceWSImpl.java ✕ CalculatriceWebService.java ✕
1 package calculatriceWebService;
2
3 import javax.xml.ws.Endpoint;
4
5 public class CalculatriceWSPublisher {
6     public static void main(String[] args) {
7         int port = 8888;
8         String url = "http://localhost:" + port + "/calculatriceWS";
9         System.out.println("Publishing CalculatriceWS on port " + port);
10        Endpoint.publish(url, new CalculatriceWSImpl());
11    }
12 }
13

```

Mise en œuvre

- Compilation classique
javac -d . *.java
- Utilisation de wsgen pour générer les classes et fichiers requis pour l'exécution du service web
wsgen -cp . calculatriceWebService.CalculatriceWSImpl
- Publication du service
java calculatriceWebService.CalculatriceWSPublisher

serveurComplexes

```

CalculatriceWebService.java ✕ CalculatriceWSImpl.java ✕ CalculatriceWSPublisher.java ✕ Complexe.java ✕
1 package calculatriceComplexesWebService;
2
3 import javax.jws.WebService;
4 import javax.jws.WebMethod;
5 import javax.jws.WebParam;
6 import javax.jws.soap.SOAPBinding;
7
8 @WebService(name="CalculatriceComplexesWebService")
9 @SOAPBinding(style=SOAPBinding.Style.RPC)
10 public interface CalculatriceWebService {
11     @WebMethod(action="additionner")
12     public Complexe additionner(@WebParam(name="c1") Complexe c1, @WebParam(name="c2") Complexe c2);
13     @WebMethod(action="soustraire")
14     public Complexe soustraire(@WebParam(name="c1") Complexe c1, @WebParam(name="c2") Complexe c2);
15     @WebMethod(action="multiplier")
16     public Complexe multiplier(@WebParam(name="c1") Complexe c1, @WebParam(name="c2") Complexe c2);
17     @WebMethod(action="diviser")
18     public Complexe diviser(@WebParam(name="c1") Complexe c1, @WebParam(name="c2") Complexe c2);
19 }
20

```

```

CalculatriceWSImpl.java  CalculatriceWebService.java  CalculatriceWSPublisher.java  Complexe.java
1  package calculatriceComplexesWebService;
2
3  import javax.ws.WebService;
4
5  @WebService(endpointInterface="calculatriceComplexesWebService.CalculatriceWebService")
6  public class CalculatriceWSImpl implements CalculatriceWebService {
7      @Override
8      public Complexe additionner(Complexe c1, Complexe c2) {
9          return new Complexe(c1.getRe()+c2.getRe(), c1.getIm()+c2.getIm());
10     }
11
12     @Override
13     public Complexe soustraire(Complexe c1, Complexe c2) {
14         return new Complexe(c1.getRe()-c2.getRe(), c1.getIm()-c2.getIm());
15     }
16
17     @Override
18     public Complexe multiplier(Complexe c1, Complexe c2) {
19         return new Complexe(c1.getRe()*c2.getRe()-c1.getIm()*c2.getIm(), c1.getRe()*c2.getIm()+c1.getIm()*c2.getRe());
20     }
21
22     @Override
23     public Complexe diviser(Complexe c1, Complexe c2) {
24         float diviseur = c2.getRe()*c2.getRe()+c2.getIm()*c2.getIm();
25         if(diviseur == 0)
26             throw new ArithmeticException("Division par zero !");
27         return new Complexe((c1.getRe()*c2.getRe()+c1.getIm()*c2.getIm())/diviseur, (c1.getIm()*c2.getRe()-c1.getRe()*c2.getIm())/diviseur);
28     }
29 }
30

```

```

Complexe.java  CalculatriceWSImpl.java  CalculatriceWebService.java  CalculatriceWSPublisher.java
1  package calculatriceComplexesWebService;
2
3  public class Complexe {
4      private float re,im;
5
6      public Complexe() {
7          this.re = (float)0;
8          this.im = (float)0; }
9
10     public Complexe(int re,int im) {
11         this.re = (float)re;
12         this.im = (float)im; }
13
14     public Complexe(float re,float im) {
15         this.re = re;
16         this.im = im; }
17
18     public float getRe() {
19         return this.re; }
20
21     public void setRe(float re) {
22         this.re = re; }
23
24     public float getIm() {
25         return this.im; }
26
27     public void setIm(float im) {
28         this.im = im; }
29
30     public String toString(){
31         String resultat="";
32         if (this.re!=0)
33             resultat=String.valueOf(this.re);
34         if (this.im!=0) {
35             if (this.im>0){
36                 if (resultat!="")
37                     resultat=resultat+"+"+String.valueOf(this.im);
38                 else
39                     resultat=String.valueOf(this.im);
40             }else
41                 resultat=resultat+String.valueOf(this.im);
42             resultat=resultat+"i"; }
43         return resultat; }
44 }

```

```
CalculatriceWSPublisher.java ✕
1  package calculatriceComplexesWebService;
2
3  import javax.xml.ws.Endpoint;
4
5  public class CalculatriceWSPublisher {
6      public static void main(String[] args) {
7          int port = 8880;
8          String url = "http://localhost:" + port + "/calculatriceWSComplexe";
9          System.out.println("Publishing CalculatriceWSComplexe on port " + port);
10         Endpoint.publish(url, new CalculatriceWSImpl());
11     }
12 }
13
```

Mise en œuvre

- Compilation classique
javac -d . *.java
- Utilisation de wsgen pour générer les classes et fichiers requis pour l'exécution du service web
wsimport -cp . calculatriceComplexesWebService.CalculatriceWSImpl
- Publication du service
java calculatriceComplexesWebService.CalculatriceWSPublisher

Clients

Préparation

Il faut générer le code nécessaire aux clients à partir des serveurs respectifs

```
wsimport -p calculatriceWebService -keep http://localhost:8888/calculatriceWS?wsdl
```

```
wsimport -p calculatriceComplexesWebService -keep  
http://localhost:8880/calculatriceWSComplexe?wsdl
```

Puis écrire le code des clients, compiler et exécuter

```

ClientEntiersPositifs.java ClientComplexes.java
1 import calculatriceWebService.CalculatriceWebService;
2
3
4 public class ClientEntiersPositifs {
5     public static void main(String[] args) {
6         try {
7             CalculatriceWebService calculatrice = (new calculatriceWebService.CalculatriceWSImplService()).getCalculatriceWSImplPort();
8             int operande1 = Integer.parseInt(args[args.length-3]),
9                 operande2 = Integer.parseInt(args[args.length-1]),
10                resultat=-1;
11
12             if(args[args.length-2].equals("+"))
13                 resultat = calculatrice.additionner(operande1, operande2);
14             if(args[args.length-2].equals("-"))
15                 resultat = calculatrice.soustraire(operande1, operande2);
16             if(args[args.length-2].equals("x"))
17                 resultat = calculatrice.multiplier(operande1, operande2);
18             if(args[args.length-2].equals("/"))
19                 resultat = calculatrice.diviser(operande1, operande2);
20
21             System.out.println(operande1 + args[args.length-2] + operande2 + " = " + resultat);
22         } catch (Exception e) {
23             System.out.println("Client exception: " + e);
24         }
25     }
26 }
27

```

```

ClientComplexes.java ClientEntiersPositifs.java
1 import calculatriceComplexesWebService.CalculatriceComplexesWebService;
2 import calculatriceComplexesWebService.Complexe;
3
4
5 public class ClientComplexes {
6     public static void main(String[] args) throws Exception {
7         try {
8             Complexe c1=new Complexe(), c2=new Complexe(), un=new Complexe(), zero=new Complexe(), result;
9             c1.setRe(2); c1.setIm(4);
10            c2.setRe(1); c2.setIm(2);
11            un.setRe(1); un.setIm(0);
12            zero.setRe(0); zero.setIm(0);
13            CalculatriceComplexesWebService calculatrice = (new calculatriceComplexesWebService.CalculatriceWSImplService()).getCalculatriceWSImplPort();
14
15            result = calculatrice.additionner(c1,c2);
16            System.out.println("(2+4i) + (1+2i) = " + result.getRe() + " + " + result.getIm() + "i");
17
18            result = calculatrice.soustraire(c1,c2);
19            System.out.println("(2+4i) - (1+2i) = " + result.getRe() + " + " + result.getIm() + "i");
20
21            result = calculatrice.multiplier(c1,c2);
22            System.out.println("(2+4i) x (1+2i) = " + result.getRe() + " + " + result.getIm() + "i");
23
24            result = calculatrice.diviser(c1,c2);
25            System.out.println("(2+4i) / (1+2i) = " + result.getRe() + " + " + result.getIm() + "i");
26
27            result = calculatrice.diviser(c1,un);
28            System.out.println("(2+4i) / 1 = " + result.getRe() + " + " + result.getIm() + "i");
29
30            result = calculatrice.diviser(c1,zero);
31            System.out.println("(2+4i) / 0 = " + result.getRe() + " + " + result.getIm() + "i");
32        } catch (Exception e) {
33            System.out.println("Client exception: " + e);
34        }
35    }
36 }
37

```