

# Edition d'un état jasper à partir d'une application ASP.NET en utilisant un web service java

par Faisel Chabli ([Accueil](#))

Date de publication : 30/05/2008

Dernière mise à jour : 30/05/2008

Cet article a pour objectif de vous présenter comment éditer un état jasper à partir d'une application ASP.NET en utilisant un web service java.

I - Introduction.....	3
II - Création du service web.....	3
2.1 - Présentation.....	3
2.2 - Création du service web.....	3
2.3 - Déploiement du fichier java.....	4
2.4 - Test du service web.....	4
III - Création de la partie client.....	6
3.1 - Ajouter la référence web.....	6
3.2 - L'appel du service web.....	7
3.3 - Exécution du projet.....	8
IV - Télécharger.....	9
V - Conclusion.....	9
VI - Remerciements.....	9

## I - Introduction

Cet article entre dans le cadre de l'interopérabilité entre .NET et J2EE. En effet, les états jasper ne s'éditent, généralement, qu'à partir des applications java via les API de JasperReports. Toutefois, on peut bénéficier de cette API côté serveur et donc dans une application java pour éditer l'état à partir d'une application .NET en utilisant les **web services** qui assurent une communication multi-plateforme.

Les exemples proposés seront écrits en java, pour la partie web service, et en vb.net pour l'application qui consommera ce web service.

Cet article vous aidera à accomplir cette tâche.

## II - Création du service web

### 2.1 - Présentation

Je ne vais pas rentrer dans les détails pour créer un web service en java. Vous pouvez vous référer au tutoriel disponible [ici](#). Il décrit parfaitement comment créer un web service en utilisant **AXIS**. Quant aux explications des différentes méthodes pour éditer un état jasper à partir d'une application java, vous pouvez toujours vous référer à mon tutoriel déjà publié [ici](#).

### 2.2 - Création du service web

Notre service web ne contiendra qu'une seule classe avec une seule méthode, celle qui permet de se connecter à la base de données et générer un tableau de bytes binaires contenant les données du fichier pdf à créer à partir du fichier jasper.

Cette classe se présente comme suit :

#### Code1 - Visualisation PDF

```
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JasperExportManager;
import net.sf.jasperreports.engine.JasperFillManager;
import net.sf.jasperreports.engine.JasperPrint;

import com.mysql.jdbc.Driver;

public class InitPageBean {

    private static final long serialVersionUID = -3581065399788581255L;

    /** La fonction pour la la visualisation PDF * */

    public byte[] viewReportPDF() throws SQLException, JRException, IOException {
        String reportId = "Person";
        Driver mDriver = new Driver();
        DriverManager.registerDriver(mDriver);
        Connection connection = DriverManager.getConnection(
            "jdbc:mysql://localhost/mytest",
            "root",
            "root");
        File file = new File("C:\\reports");
        JasperPrint jasperPrint = JasperFillManager.fillReport(
```

### Code1 - Visualisation PDF

```

new FileInputStream(new File(file, reportId + ".jasper")),
null, connection);
byte[] bytes = JasperExportManager.exportReportToPdf(jasperPrint);
return bytes;
}
}
}


```

Mon fichier jasper est nommé **Person**, il est sous : **C:\reports**. Je ne reviens pas sur la manière dont ce fichier a été créé mais vous pouvez vous référer au tutoriel [ici](#) pour comprendre les étapes de création d'un tel fichier.

La méthode **viewReportPDF()** se contentera juste de nous retourner un tableau de bytes à partir de notre fichier jasper.

## 2.3 - Déploiement du fichier java

On peut créer un dossier *testJasper* sous **TOMCAT\_HOME/webapps/axis/** et copier la source java précédemment créée sans compilation. Après l'avoir copiée, on doit renommer le fichier **InitPageBean.java** en **InitPageBean.jws**. On aura donc une arborescence qui ressemble à : **TOMCAT\_HOME/webapps/axis/testJasper/InitPageBean.jws**.

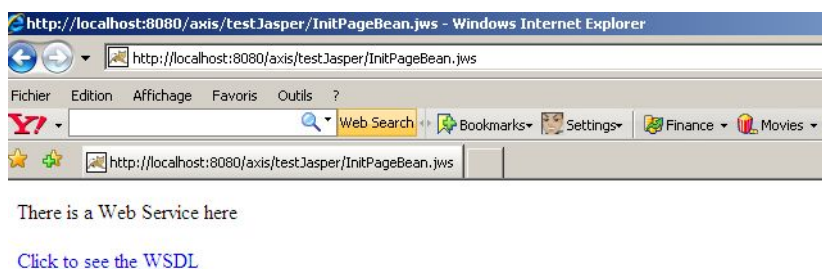
 *Sur l'article expliquant l'utilisation des web services sous AXIS on trouve bien qu'il y a deux manières pour déployer un web service. Pour moi j'ai choisi la plus simple, qu'est la modification de l'extension du fichier java en fichier jws.*

## 2.4 - Test du service web

Après avoir déployé le service web, on doit vérifier si tout est dans l'ordre.

Pour ce faire, démarrer TOMCAT et taper dans votre navigateur: <http://localhost:8080/axis/testJasper/InitPageBean.jws>.

Vous devez avoir une page qui ressemble à :



En cliquant sur le lien **Click to see the WSDL**, une page s'affiche comme suit :

### Code1- InitPageBean.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://localhost:8080/axis/testJasper/InitPageBean.jws"
xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="http://localhost:8080/axis/testJasper/InitPageBean.jws"
xmlns:intf="http://localhost:8080/axis/testJasper/InitPageBean.jws"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns1="http://engine.jasperreports.sf.net"
xmlns:tns2="http://lang.java"

```

**Code1- InitPageBean.xml**

```

xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
xmlns:wSDLsoap="http://schemas.xmlsoap.org/wSDL/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
<!--WSDL created by Apache Axis version: 1.4
Built on Apr 22, 2006 (06:55:48 PDT)-->
<wSDL:types>
  <schema targetNamespace="http://engine.jasperreports.sf.net" xmlns="http://www.w3.org/2001/
XMLSchema">
    <import namespace="http://lang.java"/>
    <import namespace="http://schemas.xmlsoap.org/soap/encoding"/>
    <complexType name="JRException">
      <sequence>
        <element name="cause" nillable="true" type="xsd:anyType"/>
      </sequence>
    </complexType>
  </schema>
</wSDL:types>

  <wSDL:message name="viewReportPDFResponse">
    <wSDL:part name="viewReportPDFReturn" type="xsd:base64Binary"/>
  </wSDL:message>

  <wSDL:message name="JRException">
    <wSDL:part name="fault" type="tns1:JRException"/>
  </wSDL:message>

  <wSDL:message name="viewReportPDFRequest">
  </wSDL:message>

  <wSDL:portType name="InitPageBean">
    <wSDL:operation name="viewReportPDF">
      <wSDL:input message="impl:viewReportPDFRequest" name="viewReportPDFRequest"/>
      <wSDL:output message="impl:viewReportPDFResponse" name="viewReportPDFResponse"/>
      <wSDL:fault message="impl:JRException" name="JRException"/>
    </wSDL:operation>
  </wSDL:portType>

  <wSDL:binding name="InitPageBeanSoapBinding" type="impl:InitPageBean">
    <wSDLsoap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wSDL:operation name="viewReportPDF">
      <wSDLsoap:operation soapAction=""/>
      <wSDL:input name="viewReportPDFRequest">
        <wSDLsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="http://DefaultNamespace" use="encoded"/>
      </wSDL:input>
      <wSDL:output name="viewReportPDFResponse">
        <wSDLsoap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="http://localhost:8080/axis/testJasper/InitPageBean.jws" use="encoded"/>
      </wSDL:output>
      <wSDL:fault name="JRException">

```

**Code1- InitPageBean.xml**

```

        <wsdlsoap:fault encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" name="JRException"
        namespace="http://localhost:8080/axis/testJasper/InitPageBean.jws"
        use="encoded"/>

        </wsdl:fault>

    </wsdl:operation>

</wsdl:binding>

<wsdl:service name="InitPageBeanService">

    <wsdl:port binding="impl:InitPageBeanSoapBinding"
    name="InitPageBean">

        <wsdlsoap:address location="http://localhost:8080/axis/testJasper/InitPageBean.jws"/>

    </wsdl:port>

</wsdl:service>

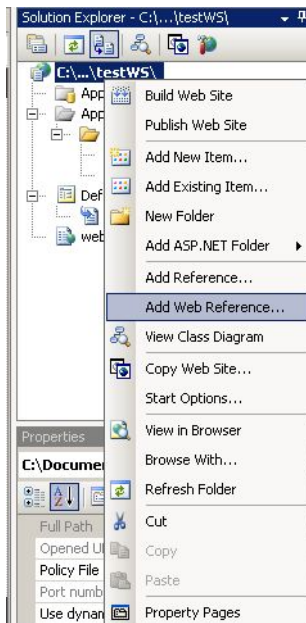
</wsdl:definitions>
    
```

Si ces deux affichages s'effectuent correctement c'est que notre service web a été bien déployé et est prêt pour être testé.

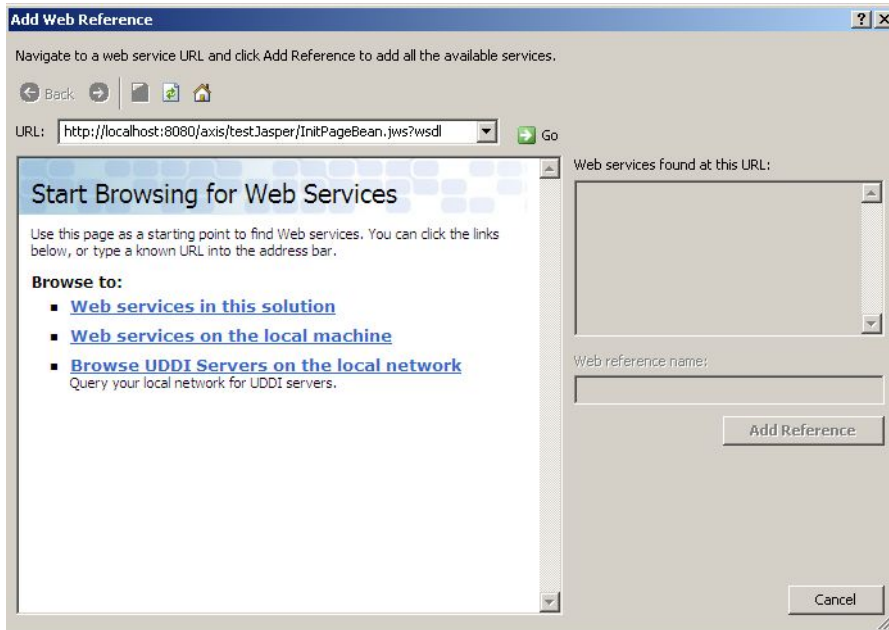
**III - Création de la partie client**

**3.1 - Ajouter la référence web**

Notre service web a été déployé. Maintenant on peut y faire référence à partir de notre application ASP.NET en l'ajoutant en tant que référence web comme suit :



Sur l'écran qui s'affiche on doit saisir l'adresse de notre service web comme suit :

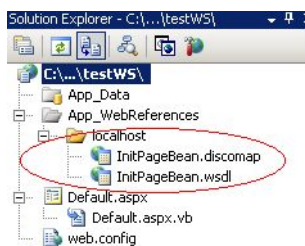


Ensuite, cliquer sur **Go**. Les différentes méthodes de notre service web s'affichent comme suit :



Enfin, il faut cliquer sur le bouton **Add reference**.

Vous remarquez que le service web a été ajouté à votre projet.



### 3.2 - L'appel du service web

Maintenant que notre service web a été référencé, on peut y faire appel en utilisant ses différentes méthodes.

On peut ajouter une nouvelle forme au projet ou utiliser une ancienne et y ajouter un bouton. Dans mon cas, ma page ressemble tout simplement à :



Le code qui s'exécute lors du clique sur ce bouton est comme suit :

**Code1 - Visualisation PDF**

```

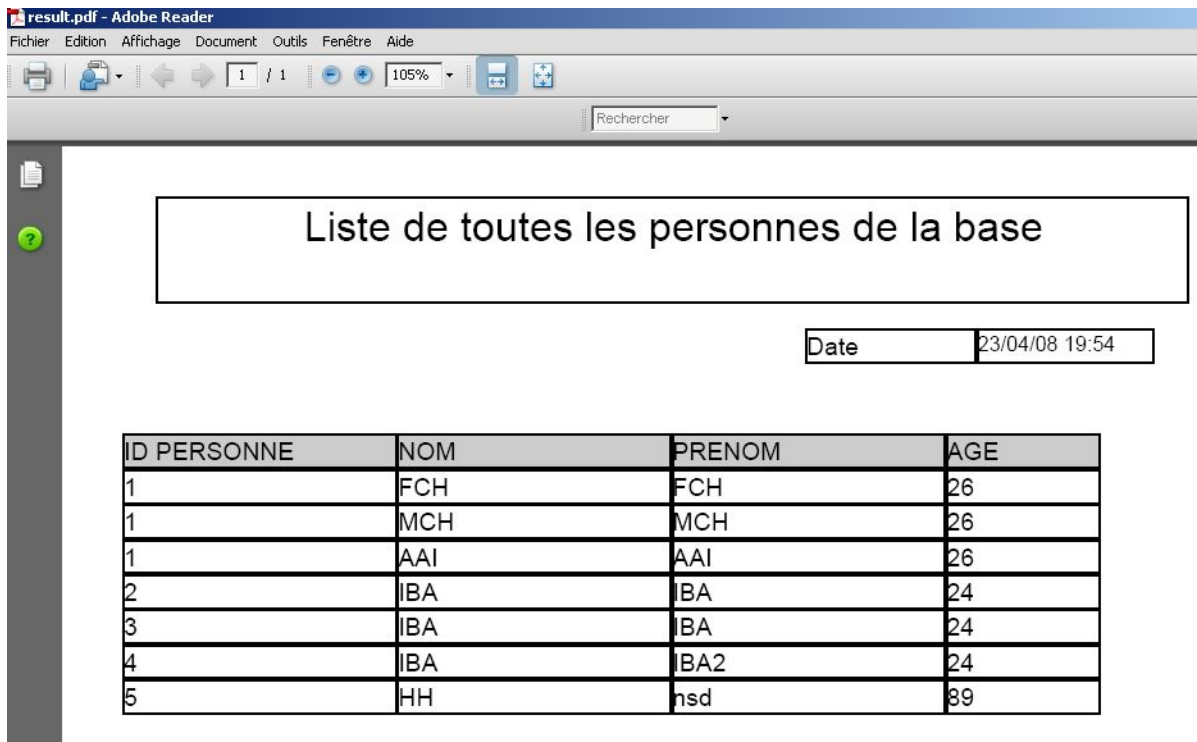
Protected Sub Button1_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Button1.Clic
    Dim ws As New localhost.InitPageBeanService
    Try
        'Déclaration d'un tableau de byte qui contiendra le résultat du retour de la méthode
        viewReportPDF du service web
        Dim b() As Byte
        'Instanciatioin du service web
        Dim ws As New localhost.InitPageBeanService
        'récupération du résultat de la méthode du service web
        b = ws.viewReportPDF()
        'L'export du résultat dans un fichier result.pdf qui sera déposé sous C:\
        Dim fs As New System.IO.FileStream("C:\result.pdf", System.IO.FileMode.OpenOrCreate,
        System.IO.FileAccess.Write)
        'L'écriture du contenu
        fs.Write(b, 0, b.Length)
        fs.Close()
        Catch ex As Exception
            Response.Write(ex.Message)
        End Try
    End Sub

```

**3.3 - Exécution du projet**

En exécutant le projet et en cliquant sur le bouton **Test Edition**, un fichier **result.pdf** sera créé sous C:\.

Exemple d'exécution :



## IV - Télécharger

Les codes sources sont disponibles en téléchargement dans l'archive suivant : [Source](#) [jasper-asp.net.rar](#).

## V - Conclusion

Cet article vous a montré comment éditer un état jasper, déployé sous un service web java, à partir d'une application ASP.NET. Je n'ai pas traité le cas de l'exportation dans un format Excel, Word mais c'est facile à faire. Je n'ai pas traité aussi le cas du passage de paramètres, mais, une fois mis à jour mon ancien article sur l'édition d'un état jasper à partir d'une application JSF, j'apporterai les mises à jour nécessaires à cet article.

## VI - Remerciements

Mes remerciements à :

- [nico-pyright\(c\)](#) pour sa lecture attentive de cet article
- [vbrabant](#) pour ses remarques.
- [diogene](#) pour ses corrections orthographiques.