# Programmation EJB3 JMS sous Widfly 8.2

## Introduction

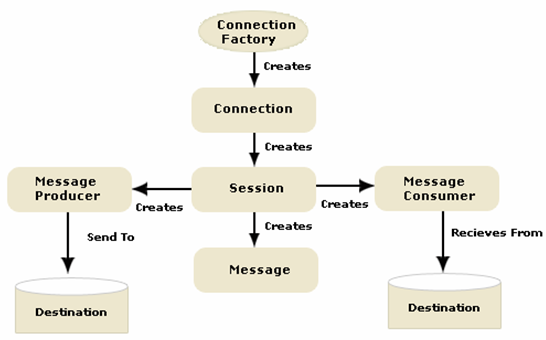
Dans ce sujet, vous allez découvrir la création d'un EJB3 JMS sous le serveur d’application Widfly 8.2l

Java Message Service (JMS) API est un système de messages qui permet à une application, en utilisant les composants Java 2 Platform Enterprise Edition (J2EE), de créer, d’envoyer , de recevoir et de lire des messages. Cette API offre **une communication distribuée fiable et asynchrone**. JMS permet de consommer un message en **synchrone (receive) ou en asynchrone**.

JMS définit plusieurs entités :

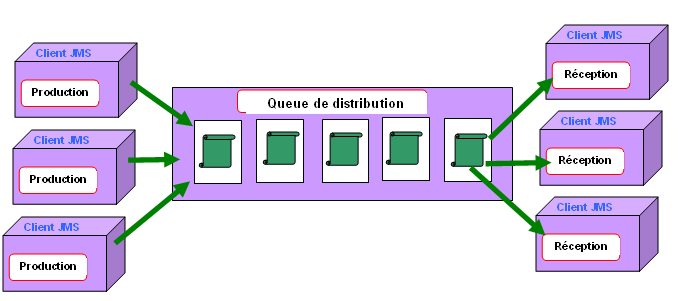
* un **provider JMS** : outil qui implémente l'API JMS pour échanger les messages : ce sont les « brokers » de messages
* un **client JMS** : composant écrit en java qui utilise JMS pour émettre et/ou recevoir des messages.
* un **message** : données échangées entre les composants

Le processus d’échange de messages peut être montré dans la figure suivante :



### Exemple d’application : Mode Publish/Souscribe

Le mode publication/abonnement repose sur le concept de sujets (Topic). Plusieurs producteurs peuvent placer les messages pour divers destinataires dans une queue. L'émetteur du message ne connaît pas les destinataires qui se sont abonnés



* + **Inscription à une manifestation** (mode topic)

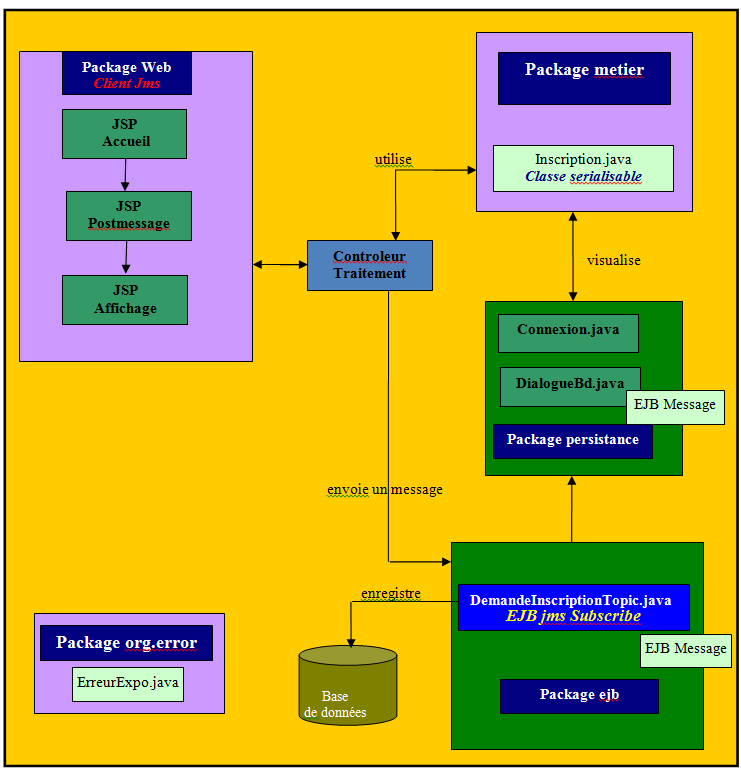
Des clients s’inscrivent, leurs renseignements sont enregistrés dans une base Mysql

#### Composants de l’application :

* + Architecture de l’application

Elle est composée :

* + - Serveur d’applications ***Widfly 8.2***
    - IDE Eclipse Luna
    - Pages JSP
    - Java 1.8
    - Driver Mysql *mysql-connector-java-5.1.29-bin.jar*
    - EJB de type Message Driven Bean
    - Mysql 6.0 (base de données) pour le stockage des messages

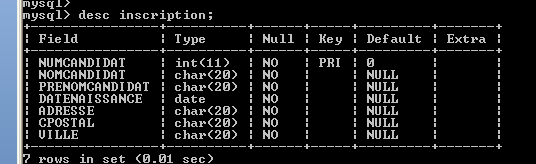


# Réalisation de l’application

## Base de données sous Mysql : lesinscriptions ;

Sous mysql, créez une nouvelle base de données nommée lesinscriptions avec une table nommée inscription.

Structure de la table inscription.



## Installation d’une queue sous le serveur Widfly 8.2

La création d’une queue sous JBOSS demande l’écriture d’un fichier XML qui définit la queue de publication.

|  |
| --- |
| On lance un client jboss-cli On se connecte au serveur    On exécute la commande suivante :  [standalone@localhost:9990 /]  **jms-topic add --topic-address= DemandeInscriptionJmsTopic** **--entries=** **java:jboss/exported/topic/InscriptionJmsTopic**    **On contrôle l’entré de notre file topic**  **[standalone@localhost:9990 /] /subsystem=messaging/hornetq-server=default/jms-topic=DemandeInscriptionJmsTopic:read-resource**    On vérifie que la queue s’est bien installée dans le fichier standlone.xml    Sous le serveur |

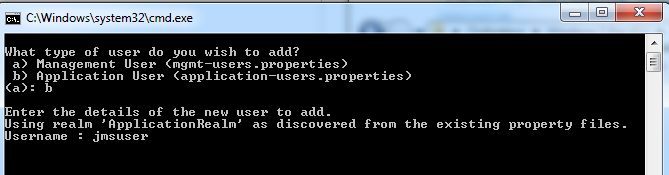
Notre file est à présent opérationnelle pour envoyer ou consommer des messages.

#### Suppression d’une file

#### 

**jms-topic remove --topic-address= DemandeInscriptionJmsTopic** **--entries=** **java:jboss/exported/topic/InscriptionJmsTopic**

### Création d’un utilisateur : jmsuser/ jmsepul98! add-user



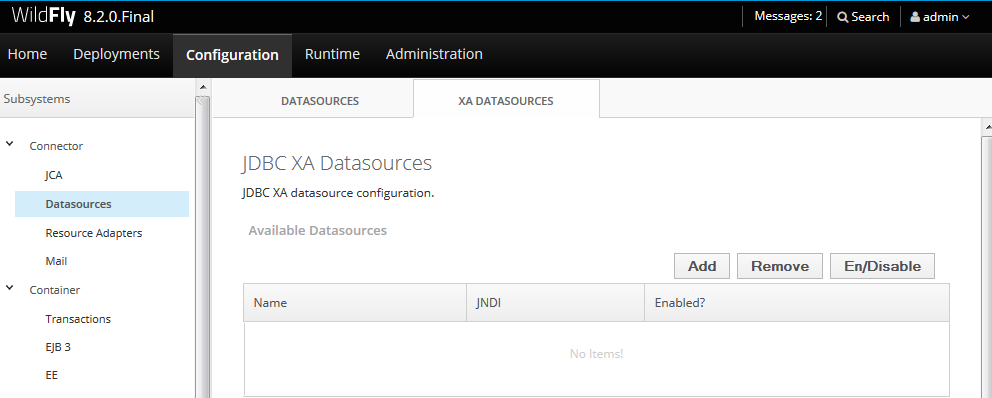
Ajoutez-le dans le groupe guest

Le fichier application-roles sera mis à jour.

Si cet utilisateur a été créé dans le tp précédent : message et image en asynchrone, vous n’avez pas à le refaire.

DataSource

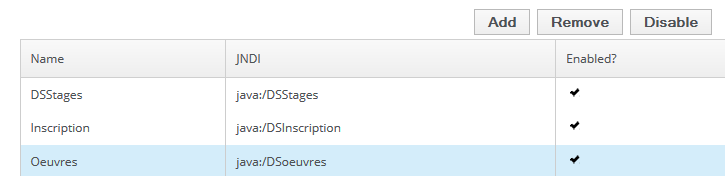
On configure la datasource sous le serveur WidFly 8.2



On ajoute le driver

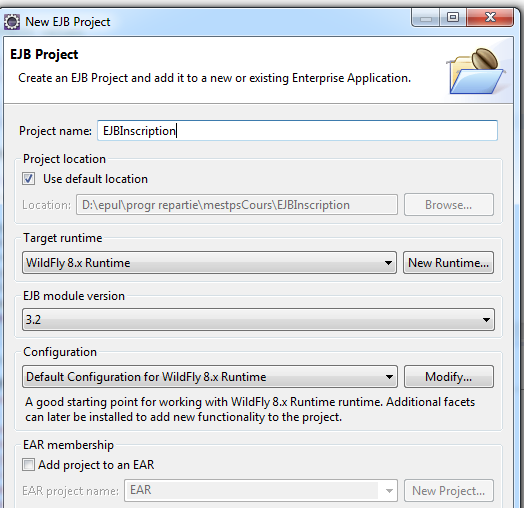
|  |  |
| --- | --- |
|  |  |

On la passe à l’état Enable



## Réalisation de l’application sous Eclipse

Démarrez un nouveau projet  de type  EJBProject

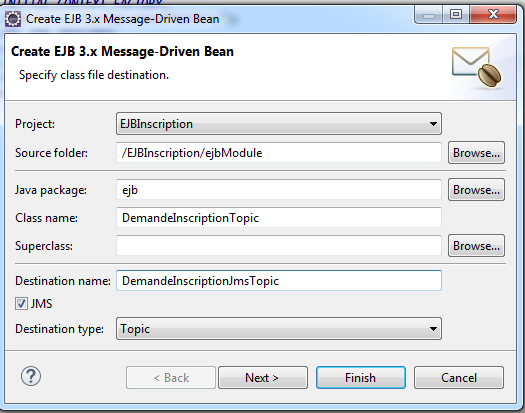


Sous ejbModule créez les packages suivants :

|  |  |
| --- | --- |
|  | Dans le package metier  réalisez la classe Inscription, cette classe doit être sérialisable. |

Classe DemandeInscriptionTopic

On crée cette classe en ajoutant un EJB de type Message Driven Bean



Le système crée

**package** ejb;

**import** javax.ejb.ActivationConfigProperty;

**import** javax.ejb.MessageDriven;

**import** javax.jms.Message;

**import** javax.jms.MessageListener;

/\*\*

\* Message-Driven Bean implementation class for: DemandeInscriptionTopic

\*/

@MessageDriven(activationConfig = {

@ActivationConfigProperty(propertyName = "destination", propertyValue = "java:jboss/exported/topic/DemandeInscriptionJmsTopic"),

@ActivationConfigProperty(propertyName = "destinationType", propertyValue = "javax.jms.Topic") }, mappedName = "DemandeInscriptionJmsTopic")

**public** **class** DemandeInscriptionTopic **implements** MessageListener {

/\*\*

\* Default constructor.

\*/

**public** DemandeInscriptionTopic() {

// **TODO** Auto-generated constructor stub

}

/\*\*

\* **@see** MessageListener#onMessage(Message)

\*/

**public** **void** onMessage(Message message) {

// **TODO** Auto-generated method stub

}

}

### Classe Inscription dans le package metier

### EJB de type JMS

**package** ejb;

**import** javax.annotation.Resource;

**import** javax.ejb.ActivationConfigProperty;

**import** javax.ejb.MessageDriven;

**import** javax.jms.Message;

**import** javax.jms.MessageListener;

**import** javax.jms.ObjectMessage;

**import** java.text.SimpleDateFormat;

**import** javax.ejb.MessageDrivenContext;

**import** javax.jms.JMSException;

**import** meserreurs.MonException;

**import** metier.Inscription;

**import** java.io.\*;

**import** persistance.\*;

/\*\*

\* Message-Driven Bean implementation class for: DemandeInscriptionTopic

\*/

@MessageDriven(activationConfig = {

@ActivationConfigProperty(propertyName = "destination", propertyValue = "java:jboss/exported/topic/DemandeInscriptionJmsTopic"),

@ActivationConfigProperty(propertyName = "destinationType", propertyValue = "javax.jms.Topic") }, mappedName = "DemandeInscriptionJmsTopic")

**public** **class** DemandeInscriptionTopic **implements** MessageListener {

@Resource

**private** MessageDrivenContext context;

/\*\*

\* Default constructor.

\*/

**public** DemandeInscriptionTopic() {

// **TODO** Auto-generated constructor stub

}

/\*\*

\* **@see** MessageListener#onMessage(Message)

\*/

**public** **void** onMessage(Message message) {

// **TODO** Auto-generated method stub

**boolean** ok = **false**;

// On gère le message récupéré dans le topic

**try** {

// On transforme le message en demande d'inscription

**if** (message != **null**)

{

ObjectMessage objectMessage = (ObjectMessage) message;

Inscription uneInscription = (Inscription) objectMessage

.getObject();

// On insère cette demande d'inscription dans la base de données

**try** {

ok = InsertionDemandeInscription(uneInscription);

} **catch** (MonException e) {

EcritureErreur(e.getMessage());

}

}

} **catch** (JMSException jmse) {

EcritureErreur(jmse.getMessage());

context.setRollbackOnly();

}

}

/\*\*

\* Permet d'enregistrer une erreur dans un fichier log

\*

\* **@param** message

\* Le message d'erreur

\*/

**public** **void** EcritureErreur(String message) {

BufferedWriter wr;

String nomf = "erreurs.log";

java.util.Date madate = **new** java.util.Date();

SimpleDateFormat sdf = **new** SimpleDateFormat("dd/MM/yy hh:mm");

**try** {

// On écrit à la fin du fichier

wr = **new** BufferedWriter(**new** FileWriter(nomf, **true**));

wr.newLine();

wr.write(sdf.format(madate));

wr.newLine();

wr.write(message);

wr.close();

} **catch** (FileNotFoundException ef) {

;

} **catch** (IOException eio) {

;

}

}

/\*\*

\* Permet d'insérer une demande d'inscription La table est auto incrémentée

\*

\* **@param** d

\* La demande d'inscription à insérer

\* **@return** Vrai si l'insertion s'est bien passée, Faux sinon

\* **@throws** MonException

\*/

**public** **boolean** InsertionDemandeInscription(Inscription d)

**throws** MonException {

String mysql = "";

**boolean** ok = **true**;

**try** {

SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd");

String myDate = sdf.format(d.getDatenaissance()).toString();

// On crée la requête d'insertion

mysql = "INSERT INTO INSCRIPTION (NOMCANDIDAT , PRENOMCANDIDAT, DATENAISSANCE, ADRESSE, CPOSTAL,VILLE ) ";

mysql = mysql + " VALUES ( \'" + d.getNomcandidat() + "\', ";

mysql = mysql + "\' " + d.getPrenoncandidat() + "\', " + "\' "

+ myDate + "\', ";

mysql = mysql + "\' " + d.getAdresse() + "\', " + "\' "

+ d.getCpostal() + "\',\' " + d.getVille() + "\' )";

// On exécute la requête d'insertion et on ferme la connexion

DialogueBd.*insertionBD*(mysql);

} **catch** (MonException e) {

ok = **false**;

**throw** e;

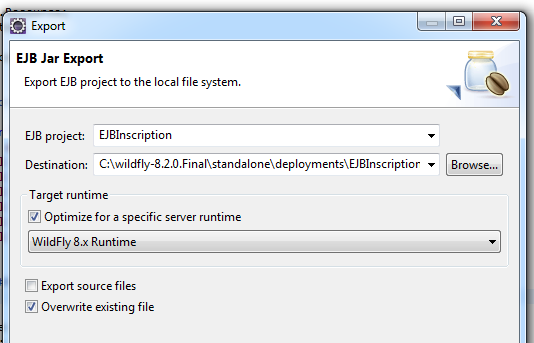
}

**return** ok;

}

}

La partie EJB de notre application est à présent terminée. Vous pouvez le déployer sous Jboss



#### Déploiement de l’EJB DemandeInscriptions

Vous devez rajouter dans le fichier standlone.xml la configuration pour un message MDB

<subsystem xmlns="urn:jboss:domain:ejb3:2.0">

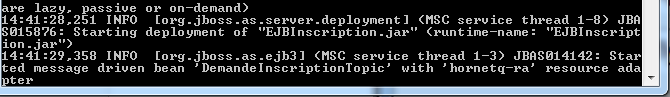
<session-bean>

<stateful default-access-timeout="5000" cache-ref="simple" passivation-disabled-cache-ref="simple"/>

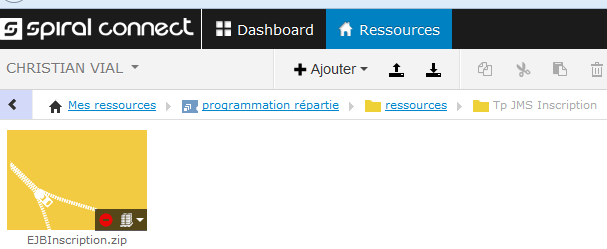
<singleton default-access-timeout="5000"/>

</session-bean>

<mdb>



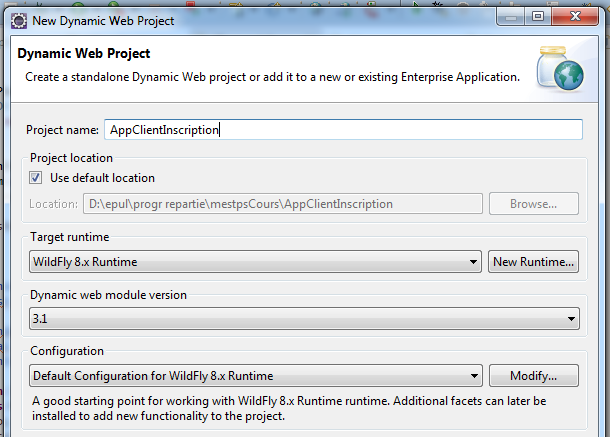
Vous pouvez récupérer le code de l’EJB sous SPIRAL



# Partie Cliente : production de nos messages

La partie cliente se compose de pages JSP et dune servlet nommée Controleur.

Démarrez une nouvelle application de type Dynamic Web Project



|  |  |
| --- | --- |
|  |  |

**Remarque** :

Les packages metier et meserreurs sont dupliqués entre l’EJB et l’application war. Ce problème doit être solutionné par une configuration du serveur WidFly…

Les pages Accueil.jsp et PostMessage.jsp sont simples à écrire. La page PostMessage affiche un formulaire et appelle la servlet Controleur pour produire le message.

## Accueil.jsp

<%--

Document : Accueil

Created on : 25 nov. 2009, 15:49:23

Author : TECRA\_A10

--%>

<%@page contentType=*"text/html"* pageEncoding=*"UTF-8"*%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"*>

<title>Accueil </title>

</head>

<body>

<h1>Page d'accueil <%= request.getContextPath () %> </h1>

<a href=*'Controleur?action=ajouteInscription’*>Ajout d'une nouvelle inscription </a>

</body>

</html>

## PostMessage.jsp

<%--

Document : PostMessage

Created on : 25 nov. 2009, 15:56:49

Author : Christian

--%>

<%@page contentType=*"text/html"* pageEncoding=*"UTF-8"*%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"

"http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"*>

<title>JSP Page</title>

<style type=*"text/css"*>

label {

display:*block*;

width:*150px*;

float:*left*;

}

</style>

</head>

<body>

<h1>Servlet PostMessage at <%= request.getContextPath () %> </h1>

<form action=*'Controleur?action=envoiInscription'* method=*'post'*>

<label for=*"nom"*>Nom:</label> <input type=*'text'* name=*'nom'* id=*"nom"*> <br/><br/>

<label for=*"prenom"*> Prénom :</label> <input type=*'text'*name=*'prenom'* id=*"prenom"*> <br/><br/>

<label for=*"datenaissance"*>Date de naissance : </label> <input type=*'text'* name=*'datenaissance'* id=*"datenaissance"*><br/><br/>

<label for=*"adresse"*> Adresse :</label> <input type=*'text'* name=*'adresse'* id=*"adresse"*><br/><br/>

<label for=*"cpostal "*> Code Postal :</label> <input type=*'text'* name=*'cpostal'*id =*"cpostal"*><br/><br/>

<label for=*"ville"*> Ville :</label> <input type=*'text'* name=*'ville'*id=*"ville"*><br/><br/>

<input type=*'submit'*><br/>

</form>

</body>

</html>

## Code la servlet Controleur

**package** ejb;

**import** javax.annotation.Resource;

**import** javax.ejb.ActivationConfigProperty;

**import** javax.ejb.MessageDriven;

**import** javax.jms.Message;

**import** javax.jms.MessageListener;

**import** javax.jms.ObjectMessage;

**import** java.text.SimpleDateFormat;

**import** javax.ejb.MessageDrivenContext;

**import** javax.jms.JMSException;

**import** meserreurs.MonException;

**import** metier.Inscription;

**import** java.io.\*;

**import** persistance.\*;

/\*\*

\* Message-Driven Bean implementation class for: DemandeInscriptionTopic

\*/

@MessageDriven(activationConfig = {

@ActivationConfigProperty(propertyName = "destination", propertyValue = "java:jboss/exported/topic/DemandeInscriptionJmsTopic"),

@ActivationConfigProperty(propertyName = "destinationType", propertyValue = "javax.jms.Topic") }, mappedName = "DemandeInscriptionJmsTopic")

**public** **class** DemandeInscriptionTopic **implements** MessageListener {

@Resource

**private** MessageDrivenContext context;

/\*\*

\* Default constructor.

\*/

**public** DemandeInscriptionTopic() {

// **TODO** Auto-generated constructor stub

}

/\*\*

\* **@see** MessageListener#onMessage(Message)

\*/

**public** **void** onMessage(Message message) {

// **TODO** Auto-generated method stub

**boolean** ok = **false**;

// On gère le message récupéré dans le topic

**try** {

// On transforme le message en demande d'inscription

**if** (message != **null**)

{

ObjectMessage objectMessage = (ObjectMessage) message;

Inscription uneInscription = (Inscription) objectMessage

.getObject();

// On insère cette demande d'inscription dans la base de données

**try** {

ok = InsertionDemandeInscription(uneInscription);

} **catch** (MonException e) {

EcritureErreur(e.getMessage());

}

}

} **catch** (JMSException jmse) {

EcritureErreur(jmse.getMessage());

context.setRollbackOnly();

}

}

/\*\*

\* Permet d'enregistrer une erreur dans un fichier log

\*

\* **@param** message

\* Le message d'erreur

\*/

**public** **void** EcritureErreur(String message) {

BufferedWriter wr;

String nomf = "erreurs.log";

java.util.Date madate = **new** java.util.Date();

SimpleDateFormat sdf = **new** SimpleDateFormat("dd/MM/yy hh:mm");

**try** {

// On écrit à la fin du fichier

wr = **new** BufferedWriter(**new** FileWriter(nomf, **true**));

wr.newLine();

wr.write(sdf.format(madate));

wr.newLine();

wr.write(message);

wr.close();

} **catch** (FileNotFoundException ef) {

;

} **catch** (IOException eio) {

;

}

}

/\*\*

\* Permet d'insérer une demande d'inscription La table est auto incrémentée

\*

\* **@param** d

\* La demande d'inscription à insérer

\* **@return** Vrai si l'insertion s'est bien passée, Faux sinon

\* **@throws** MonException

\*/

**public** **boolean** InsertionDemandeInscription(Inscription d)

**throws** MonException {

String mysql = "";

**boolean** ok = **true**;

**try** {

SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd");

String myDate = sdf.format(d.getDatenaissance()).toString();

// On crée la requête d'insertion

mysql = "INSERT INTO INSCRIPTION (NOMCANDIDAT , PRENOMCANDIDAT, DATENAISSANCE, ADRESSE, CPOSTAL,VILLE ) ";

mysql = mysql + " VALUES ( \'" + d.getNomcandidat() + "\', ";

mysql = mysql + "\' " + d.getPrenoncandidat() + "\', " + "\' "

+ myDate + "\', ";

mysql = mysql + "\' " + d.getAdresse() + "\', " + "\' "

+ d.getCpostal() + "\',\' " + d.getVille() + "\' )";

// On exécute la requête d'insertion et on ferme la connexion

DialogueBd.*insertionBD*(mysql);

} **catch** (MonException e) {

ok = **false**;

**throw** e;

}

**return** ok;

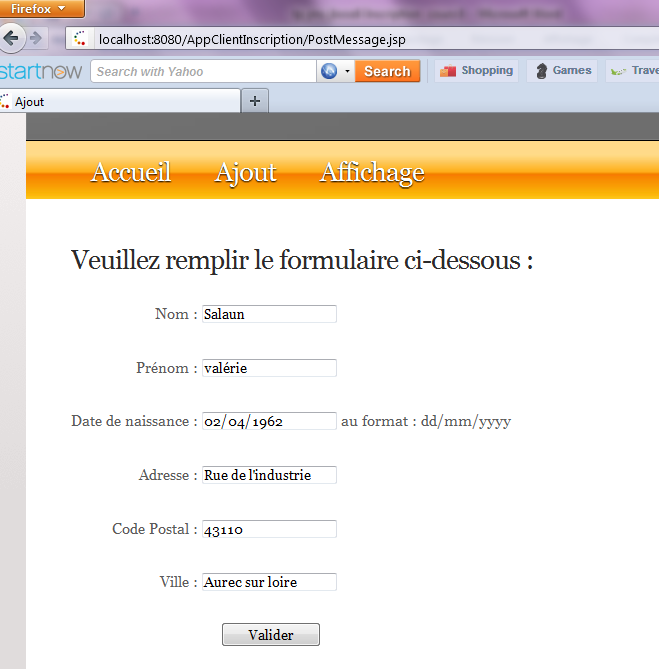
}

}

### Fonctionnement de l’application

Cette application est écrite en utilisant les formats de présentation trouvés sur le NET. Vous pouvez personnaliser votre application.

|  |
| --- |
|  |



|  |
| --- |
|  |

On peut vérifier l’ajout dans la base de données Mysql



# Gestion des Erreurs

On doit intercepter les erreurs dans :

* le chargement du driver
* la connexion à la base et sa gestion (lecture, modification, ajout..)

Les erreurs sont interceptées et stockées dans la classe Monexception. Cette classe est instanciée si une erreur survient.

Il n’est pas possible d’afficher les erreurs générées par l’EJB JMS avec les pages de l’application Cliente. Il s’agit en fait de deux processus différents qui ne peuvent pas communiquer. La solution proposée consiste à écrire les erreurs dans un fichier **erreurs.log**.

Pour les erreurs qui surviennent dans la page JSP, il est possible de faire afficher les erreurs avec une fonction javascript.

## Affichage des erreurs générées dans les pages de l’application cliente

Il faut définir un champ caché qui recevra l’éventuelle erreur en provenance de la servlet Controleur

**public** **void** service(ServletRequest request, ServletResponse response)

**throws** ServletException, IOException {

………

**catch** (MonException e) {

// On passe l'erreur à la page JSP

System.*out*.println("Erreur client :"+ e.getMessage()) ;

request.setAttribute("MesErreurs", e.getMessage());

request.getRequestDispatcher("Resultat.jsp").forward(request, response);

}

**catch** (Exception e) {

System.*out*.println("Erreur client Exception :"+ e.getMessage()) ;

request.setAttribute("MesErreurs", e.getMessage());

request.getRequestDispatcher("Resultat.jsp").forward(request, response);

}Cette erreur arrive dans le champ caché de la page

<input type =*"hidden"* name=*"uneErreur"* value=*"*${MesErreurs}*"* id =*"id\_erreur"* >

Il faut à present l’afficher par une fonction javascript :

**function** Chargement()

{

**var** obj = document.getElementById("id\_erreur");

**if** (obj.value!='')

alert('Erreur signalée : "'+obj.value+"'");

}

</script>

Cette fonction est activée au lancement de la page

<BODY onLoad=*"Chargement();"* >

### Gestion des erreurs générées dans l’EJB JMS

On les enregistre dans un fichier erreurs.log du répertoire jboss../bin

**private** **void** EcritureErreur ( String message)

{

BufferedWriter wr ;

String rep=System.*getProperty*("user.dir");

String nomf ="erreurs.log";

java.util.Date madate = **new** java.util.Date();

SimpleDateFormat sdf = **new** SimpleDateFormat("dd/MM/yy hh:mm");

**try**

{

wr = **new** BufferedWriter(**new** FileWriter(nomf));

wr.write( sdf.format(madate));

wr.newLine();

wr.write(message);

wr.close();

}

**catch** (FileNotFoundException ef) {

;

} **catch** (IOException eio) {

;

}

}

**public** **void** onMessage(Message message) {

…..

**catch** (MonException e)

{

EcritureErreur (e.getMessage());

}

}

**catch (JMSException jmse) {**

**EcritureErreur (jmse.getMessage());**

**context.setRollbackOnly();**

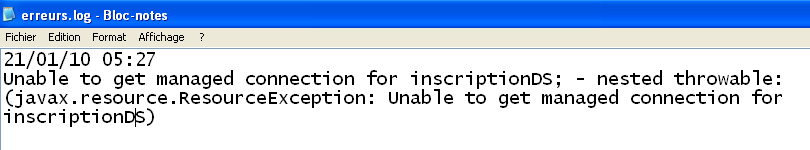
**}**

}

### Simulation d’une erreur  et enregistrement dans le fichier

On modifie le mode passe pour se connecter à la base de données.

On obtient :



### Base de données

-- phpMyAdmin SQL Dump

-- version 3.5.2.2

-- http://www.phpmyadmin.net

--

-- Client: 127.0.0.1

-- GÃ©nÃ©rÃ© le: Mar 15 Avril 2014 Ã  10:47

-- Version du serveur: 5.5.27

-- Version de PHP: 5.4.7

SET SQL\_MODE="NO\_AUTO\_VALUE\_ON\_ZERO";

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8 \*/;

--

-- Base de données: `lesinscriptions`

--

-- --------------------------------------------------------

--

-- Structure de la table `inscription`

--

CREATE DATABASE `lesinscriptions` ;

USE `lesinscriptions` ;

CREATE TABLE IF NOT EXISTS `inscription` (

`numcandidat` int(11) NOT NULL AUTO\_INCREMENT,

`nomcandidat` varchar(20) DEFAULT NULL,

`prenomcandidat` varchar(20) DEFAULT NULL,

`datenaissance` date DEFAULT NULL,

`adresse` varchar(150) DEFAULT NULL,

`cpostal` varchar(20) DEFAULT NULL,

`ville` varchar(150) DEFAULT NULL,

PRIMARY KEY (`numcandidat`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO\_INCREMENT=9 ;

--

-- Contenu de la table `inscription`

--

INSERT INTO `inscription` (`nomcandidat`, `prenomcandidat`, `datenaissance`, `adresse`, `cpostal`, `ville`) VALUES

( 'SEGUIN', 'Lionel', '1990-07-21', '74, rue Bellecombe', '74370', 'VILLAZ'),

( 'VERSET', 'Stephane', '1986-04-13', '768, route de Chazal', '74370', 'VILLAZ'),

( 'DEPRES', ' Adrien', '1990-12-11', ' 23, rue Notre-Dame', ' 69006', ' LYON'),

( 'PATRICE', ' Keenan', '1990-03-23', '1, rue Jean-R Lacoste', ' 69780', ' MIONS'),

( 'VIAL', 'CHRISTIAN', '0006-07-18', '6 Rue de luniversité', '69100', 'Villeurbanne'),

( 'Jourdin', ' CHRISTIAN', '1978-05-12', ' 2 Rue de l''université', ' 69100', ' Villeurbanne'),

( 'Dumez', ' CHRISTIAN', '1978-05-12', ' 43 Cours Emile Zola', ' 69100', ' Villeurbanne'),

( 'Fayolle', 'Alain', '2014-04-08', '5 Cours Emile Zola', '69100', 'Villeurbanne'),

( 'Brun', 'Bernard', '2014-04-13', '5 place du peuple', '42000', 'Saint Etienne');