



# Développement d'extensions

## Le Formulaire d'Inscription

# A propos de ce document

---

## Historique

0.1	11-08-04	Jean-Philippe Encausse	Création du document
0.2	-	-	-

## Contacts

Jalios SA 58 rue Pottier 78150 Le Chesnay
---

Si vous avez des questions ou souhaitez des éclaircissements sur ce document, vous pouvez nous contacter :

### Service commercial

Alain Risbourg  
Tél. : 01 39 23 92 88  
Mobile : 06 88 24 42 07  
E-mail : [alain.risbourg@jalios.com](mailto:alain.risbourg@jalios.com)

### Service technique

Equipe Technique  
Tél. : 01 39 23 92 89  
Mobile :  
E-mail : [support@jalios.com](mailto:support@jalios.com)

# Sommaire

---

<b>A propos de ce document</b> .....	<b>2</b>
<b>Historique</b> .....	<b>2</b>
<b>Contacts</b> .....	<b>2</b>
Service commercial .....	2
Service technique .....	2
<b>Sommaire</b> .....	<b>3</b>
<b>1. Création d'un Type de Formulaire</b> .....	<b>4</b>
1.1. Création des champs .....	4
1.2. Propriétés sur les champs.....	4
1.3. Propriétés sur le type .....	5
1.4. Redémarrage du site.....	5
<b>2. Mises en place du formulaire</b> .....	<b>6</b>
2.1. Accès direct par l'URL .....	6
2.2. Accès par une portlet WYSIWYG .....	7
2.3. Modification de la portlet SignUp.....	8
<b>3. Validation du formulaire</b> .....	<b>9</b>
3.1. Création d'un Workflow de validation .....	9
3.2. Association Workflow / Formulaire .....	10
3.3. Automatisation de la validation .....	11
3.4. Quelques Réflexions.....	12
3.4.1. Robustesse .....	12
3.4.2. Evolutivité.....	12
3.4.3. Ergonomie .....	12

# 1. Création d'un Type de Formulaire

L'objectif de cet article est de montrer comment mettre en place un formulaire d'inscription pour les membres désirant s'inscrire sur un site JCMS. Ce formulaire devra remplacer la demande d'inscription automatique déjà présente en standard. Le développement se fera sous la forme d'un plugin de manière à ce que la migration de la fonctionnalité soit la plus simple possible entre les différentes versions et instance de JCMS.

## 1.1. Création des champs

Dans un premier temps il faut créer tous les champs nécessaires au formulaire. Il faut tenir compte de l'ordre des champs ainsi que des libellés qui doivent être simple et compréhensibles. Les gabarits d'édition de types ne sont pas multi langue, par conséquent le formulaire devra être développé dans toutes les langues.

The screenshot shows the 'Edition du Type de Formulaire "Demande Inscription"' interface. It features a navigation menu at the top with options like 'Gestion du site', 'Groupes et Membres', 'Gestion des types', 'Mail', 'Documentations', and 'Avancé'. Below the menu is a list of fields with columns for 'Libellé', 'Onglet', 'Oblig.', 'Rech.', and 'Editeur'. The fields listed are: 2. Nom (Champ Texte), 3. Prénom (Champ Texte), 4. Adresse Email (Email), 5. Civilité (Enuméré), 6. Fonction (Champ Texte), 7. Identifiant Souhaité (Champ Texte), and 8. Question (Zone de Texte). To the right is a panel 'Ajouter un champ' with input fields for 'Libellé' and 'Editeur', and radio buttons for 'Multi-valué' and 'Obligatoire'. At the bottom, there are buttons for 'Abandonner tous les changements...', 'Enregistrer', 'Enregistrer et appliquer...', 'Enregistrer et redémarrer...', and 'Forcer la mise-à-jour du type'. The footer shows the date '9 août 2004' and the user 'Administrateur'.

## 1.2. Propriétés sur les champs

Une fois les champs créés, il faut ajuster les propriétés de chacun. Il faut déterminer si le champ est obligatoire et ajouter une aide contextuelle. Si le champ est trop technique ou obscur il est préférable de faire apparaître cette aide. Il faut garder en tête que l'utilisateur doit passer un minimum de temps sur cette interface. Donc se poser un minimum de question.

**Edition du Type de Formulaire "Demande Inscription"**

Liste des champs | Propriétés | Gabarits | Edition du champ "Adresse Email"

◀ ▶

\* Libellé

\* Nom Java

Description   
 Bulle d'aide  Texte sous le nom du champ

Onglet

Obligatoire  Oui  Non

Multi-valué  Oui  Non ⚠

Multilingue  Oui  Non

Résumé  Oui  Non

Recherchable  Oui  Non

Editeur

Taille

Lg. Max.

Syntaxe  ⚠

Valeur par défaut

### 1.3. Propriétés sur le type

Enfin il faut déterminer comment le site va se comporter après soumission du formulaire. Qui sera l'auteur de la soumission ? Est-ce qu'un espace de travail sera dédié aux demandes d'inscription ? Faut-il notifier le responsable des soumissions ou plutôt associer un workflow aux soumissions ?

### 1.4. Redémarrage du site

Une fois toutes les étapes réalisées il ne reste plus qu'à sauvegarder le type et redémarrer JCMS pour que le formulaire soit réellement généré.

## 2. Mises en place du formulaire

---

Cette deuxième étape consiste à donner un accès au formulaire depuis l'espace public du site. Pour le moment seulement une JSP `types/Type/editFormType.jsp` a été généré. Il faut maintenant créer un lien vers cette JSP.

### 2.1. Accès direct par l'URL

Pour vérifier que tout fonctionne correctement, dans un premier temps il faut essayer d'accéder, à la JSP générée, par l'URL du navigateur. Pour cela deux solutions sont possibles:

- Accès direct:

`http://www.monsite.com:8080/mawebapp/types/Type/editFormType.jsp`

- Accès par le portail:

`http://www.monsite.com:8080/mawebapp/display.jsp?id=c_5&jsp=types/Type/editFormType.jsp`

The screenshot shows a web form titled "Demande Inscription". It contains several input fields and a dropdown menu. The fields are: "Nom", "Prénom", "Adresse Email" (with a note: "Veillez entrer votre adresse email. Celle ci ne sera pas divulguée. Une fois inscrit, vous pourrez modifier votre profil pour ne pas recevoir de lettre de diffusion."), "Civilité" (dropdown menu with "Mme" selected), "Fonction" (with a note: "Veillez entrer votre fonction et/ou le nom de votre société."), "Identifiant Souhaité" (with a note: "Veillez entrer l'identifiant que vous souhaitez utiliser. Un mot de passe vous sera envoyé par email. Vous pourrez le modifier en éditant votre profil."), and "Qu'attendez vous de ce site ?" (with a note: "Veillez préciser ce que vous attendez de ce site, éventuellement les informations auxquelles vous souhaiteriez avoir accès."). At the bottom, there are "Envoyer" and "Régler" buttons. Below the form, a note states: "Les champs précédés de \* sont obligatoires."

Le portail doit bien évidemment être composé d'une portlet Sélection pour pouvoir afficher la JSP dans le portail. Le portail peut être spécifier par le paramètre `portal=`

## 2.2. Accès par une portlet WYSIWYG

Pour permettre un accès plus simple à ce formulaire, il faut créer une portlet WYSIWYG dans laquelle il y aura un lien vers le formulaire.

### Edition d'une publication Portlet WYSIWYG

Contenu | Droits de consultation | Droits de modification | Workflow | Avancé

\* Titre  
FR GB  
WYSIWYG SignUp

\* Wysiwyg  
FR GB

**Incrivez vous** sur ce site pour accéder à plus d'information et contribuer en toute liberté.  
Pour cela il vous suffit de cliquer [ici](#).

Comportement lors de la copie  
 Copie  Référence

Groupes capable de personnaliser

1.
2.
3.

Condition  
Masquer si le membre est identifié

Enregistrer | Rafraîchir | Annuler | Afficher tous les onglets

Les champs précédés de \* sont obligatoires.

Cette portlet ne devra bien évidemment être visible que si le membre n'est pas authentifié.

JCMS Portail Principal

Rechercher  OK

Accueil | Rechercher | Carte du Site

Vous êtes ici: Plan de Site

**Incrivez vous** sur ce site pour accéder à plus d'information et contribuer en toute liberté. Pour cela il vous suffit de cliquer [ici](#).

Interview

Brèves  
• Test

Pages Web

Forums

FAQs

Glossaires

Administrateur

Veuillez vous authentifier

Identifiant

Mot de Passe

Mémoriser

S'identifier

AvantGo

Accueil | Rechercher | Carte du Site

POWERED BY JALIOS

## 2.3. Modification de la portlet SignUp

La portlet « SignUp » redirige son formulaire vers `signup.jsp`. Cette JSP a pour rôle de créer un compte utilisateur sous réserve que la configuration du site l'autorise. Il faut donc modifier le gabarit d'affichage de cette portlet pour rediriger vers le nouveau formulaire.

```
1 <%@ include file='/doInitPage.jsp' %>
2 <%@ include file='/portal/doPortletParams.jsp' %>
3 <% PortletSignUp box = (PortletSignUp) portlet; %>
4
5 <%
6 if (!channel.getBooleanProperty("channel.sign-up",false) || !channel.isDataWriteEnabled()){
7     request.setAttribute("ShowPortalElement", Boolean.FALSE);
8 }
9 Portal signupPortal = box.getDisplayPortal() != null ? box.getDisplayPortal() : portal;
10 %>
11
12 <table width="100%" cellspacing="0" cellpadding="5" border="0">
13     <form method="get"
14     action="<%=contextPath%>/types/DemandeInscription/editFormDemandeInscription.jsp">
15         <tr>
16             <td>
17                 <jalios:if predicate='<%= Util.notEmpty(box.getIntro(userLang)) %>'>
18                     <span class="Intro"><%= box.getIntro(userLang) %></span>
19                 </jalios:if><br>
20                 <input type="text" name="email" size="18" value="" class="Form">
21                 <input type="image" border="0" name="imageField2" src="images/jalios/icons/ok.gif"
22                 align="middle">
23                 <input name="redirect" type="hidden" value="<%= ServletUtil.getUrl(request) %>" >
24                 <input name="portal" type="hidden" value="<%= signupPortal.getId() %>" >
25             </td>
26     </tr>
27 </form>
28 </table>
```

## 3. Validation du formulaire

Maintenant que le formulaire est mis en place, il serait assez intéressant d'automatiser le processus de validation. Il faut pouvoir envoyer un mail au valideur et pré créer un membre à partir du formulaire... Pour cela nous allons donc nous appuyer sur un workflow de validation.

### 3.1. Création d'un Workflow de validation

Pour que la demande d'inscription soit prise en compte, il faut définir un workflow de validation. Ce workflow permettra de simplifier et d'automatiser partiellement la création de compte utilisateur.

#### Édition du modèle de workflow WFInscription

Etats Rôles Transitions Propriétés

Définition des états dans lesquels peut se trouver une publication gérée par ce modèle de workflow.

	Nom	PStatus	Actions Entrantes	Actions Sortantes	
+	Demande Effectuée	-100			
	Demande Rejetée	-80			
	Demande Approuvée	-60			
	Planifié	-10			
	Publié	0			
	Expiré	10			
	Archivé	20			

+

Etat initial (toute nouvelle publication utilisant ce modèle de workflow "WFInscription" démarrera dans cet état)

Ajouter un état

Info

Nom

PStatus

Description

Etat initial  Oui

Actions

Actions Entrantes  Prévenir l'auteur  
 Prévenir les valideurs

Actions Sortantes  Prévenir l'auteur  
 Prévenir les valideurs

Workflow Express

Durée  minute

Destination

Ajouter

Abandonner tous les changements Enregistrer Supprimer...

#### Vue du modèle de workflow WFInscription



Ce workflow sera composé des états suivants:

- Demande effectuée
- Demande rejetée
- Demande approuvée (pstatus = -60)

## 3.2. Association Workflow / Formulaire

L'association d'un Type avec un Workflow se fait dans un espace de travail donné :

- Il faut déclarer dans la zone d'administration de l'espace de travail choisi que les Formulaire d'Inscription seront soumis à ce Workflow.
- Il faut déclarer dans l'éditeur de workflow (pour plus de sécurité) que ce Workflow sera créé dans l'espace de travail choisi.

**JALIOS espace de travail** Mon Site - Espace de Travail par Defaut

Contenus | Workflow | Catégories | Portlets | Formulaire | Profil | **Administration** | Rechercher

Configuration | Groupes | Membres | **Gestion des types** | Gestion des rôles | Envoyer un mail | Envoyer le bulletin

**Édition des Types de l'espace de travail: Espace de Travail par Defaut**

Contenu | PortalElement | **Form**

Nom du type	Workflow
Demande Inscription	WFInscription
Vote de Sondage	

Les champs précédés de \* sont obligatoires.

11 août 2004 - Bonjour Administrateur - Profil - Déconnexion Powered by Jalios

- Il faut affecter un group ou un membre au rôle de valideur dans cet espace de travail.

Contenus | Workflow | Catégories | Portlets | Formulaire | Profil | **Administration** | Rechercher

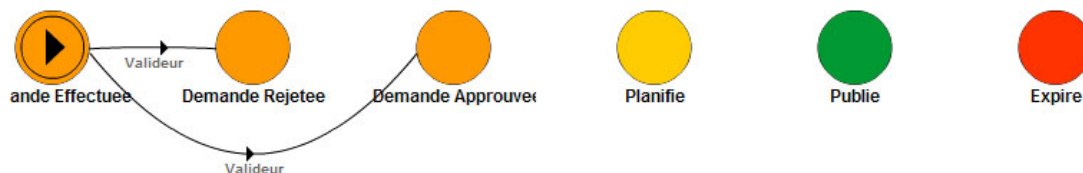
Configuration | Groupes | Membres | Gestion des types | **Gestion des rôles** | Envoyer un mail | Envoyer le bulletin

**Édition des Rôles de l'espace de travail: Espace de Travail par Defaut**

Définition des rôles du workflow. Un rôle est un ensemble de membres et de groupes pondérés. Les rôles sont affectés aux transitions. Lorsque tous les membres pondérés d'un rôle ont voté pour une transition, celle-ci est réalisée. Vous pouvez définir des rôles simples (1 membre d'un groupe donné) ou des rôles complexes (3 membres du groupe A ET 50% des membres du groupe B ET le membre Admin).

Workflow: WFInscription

Workflow	Rôle	Groupes	Membres	
WFInscription	Valideur	• Webmestres: 1 membre		<input type="button" value="Editer"/>



Les demandes seront donc créées dans l'état « Demande effectuée » dans l'espace de travail choisi. Les valideurs auront pour objectif d'approuver ou de rejeter la demande.

### 3.3. Automatisation de la validation

Lorsque la demande est approuvée, il serait intéressant de créer immédiatement un membre dans JCMS. Le valideur n'ayant plus qu'à finir la mise à jour et envoyer le mot de passes par mail au membre. Pour cela il faut de créer un « listener » qui se chargera de la création du membre lorsque le Formulaire passera dans l'état « pstatus = -60 »

```
1 package plugin.inscription;
2 import com.jalios.jcms.*;
3 import com.jalios.jstore.*;
4 import com.jalios.util.Util;
5
6 import java.util.*;
7 import generated.*;
8
9 public class DemandeInscriptionListener implements StoreListener{
10
11     // -----
12     // ||| Listener Code |||
13     // -----
14
15     public void handleCreate(Storable storable, boolean firstTime) {
16         if ((storable == null) || (!(storable instanceof DemandeInscription))){
17             return;
18         }
19         createnewMemberFromDemandeInscription((DemandeInscription) storable);
20     }
21
22     public void handleCommitUpdate(Storable storable, Storable oldStorable, boolean
firstTime){
23         if ((storable == null) || (!(storable instanceof DemandeInscription))){
24             return;
25         }
26         createnewMemberFromDemandeInscription((DemandeInscription) storable);
27     }
28
29     public void handleDelete(Storable storable, boolean firstTime) { }
30     public void handlePrepareUpdate(Storable storable, Map attributes, boolean firstTime){}
31
32
33
34     // -----
35     // ||| Specific Code |||
36     // -----
37
38     protected void createnewMemberFromDemandeInscription(DemandeInscription pub){
39
40         if (pub.getPstatus() != -60){
41             return;
42         }
43         Member newMember = new Member();
44
45         // TODO: Login must be unique
46
47         newMember.setLogin(Util.getString(pub.getIdentifiantSouhaite(),
Util.buildID(pub.getName())));
```

```

48     newMember.setName(pub.getName());
49     newMember.setFirstName(Util.getString(pub.getPrenom(),""));
50     newMember.setEmail(pub.getEmail());
51     newMember.setSalutation(pub.getCivilite());
52     newMember.setJobTitle(Util.getString(pub.getFonction(),""));
53     newMember.setInfo(pub.getQuestion());
54     newMember.setGroups(new Group[] {Channel.getChannel().getDefaultGroup()});
55     newMember.disable();
56
57     Channel.getChannel().createData(newMember,Channel.getChannel().getDefaultAdmin());
58 }
59 }

```

Ce listener a été développé dans le package `plugin.inscription`. Pour le moment il n'y a aucune norme pour les noms de package de plugins. Les plugins doivent être déclaré manuellement dans le fichier `classes/custom/JcmsInit.java`

```

1     public static void initAfterStoreLoad(Channel channel) {
2
3         channel.addStoreListener(
4             new plugin.inscription.DemandeInscriptionListener(),
5             generated.DemandeInscription.class, false);
6     }

```

## 3.4. Quelques Réflexions

### 3.4.1. Robustesse

Cet exemple soulève quelques questions intéressantes à se poser. Que se passe t il si un même login est soumis plusieurs fois ? Comment gérer le changement d'état du formulaire ? Quand le formulaire quitte l'état approuvé ou bien est supprimé, faut il désactiver le membre ?

### 3.4.2. Evolutivité

Il est très simple de faire évoluer ce type vers des formulaires plus complexes. On peut imaginer des formulaires beaucoup plus précis permettant une création automatique du compte utilisateur.

De la même manière le workflow peut très simplement être modifié pour ajouter des états notifiant périodiquement le valideur qu'il n'a pas fini de valider la demande.

### 3.4.3. Ergonomie

Pour améliorer l'ergonomie, après soumission du formulaire, il faut rediriger l'utilisateur vers une publication qui lui explique ce qui va se passer et quand son compte sera activé.