Generateur de formulaire GUI

Documentation Utilisateur

Team Infotik

Projet Tutoré S3 – 2017/2018

# Introduction

L’objectif de cette documentation est de vous introduire à l’utilisation ainsi qu’à l’utilité de notre générateur de formulaire graphique.

Table des matières

[Introduction 2](#_Toc503534440)

[I. Installation 4](#_Toc503534441)

[II. Création du document XML 5](#_Toc503534442)

[III. Prévisualisation du formulaire dans un navigateur 7](#_Toc503534443)

[IV. Intégration et récupération des données du formulaire dans un code JAVA 8](#_Toc503534444)

[V. Pour aller plus loin… 10](#_Toc503534445)

[VI. Exemples 11](#_Toc503534446)

# Installation

Tout d’abord afin de vous servir de nos outils, vous devez avoir déjà téléchargé le package iut.jar ainsi que le fichier form.xsl et form.dtd disponibles aux adresses suivantes :

[iut.jar](https://193.52.166.6/pedago/info1/M1102_algo/ressource/fichiersource/iut.jar)

form.xsl

form.dtd

Si vous n’avez pas procéder à l’installation de JAVA sur votre machine, veuillez la suivre à l’adresse suivante : [installation de java](https://193.52.166.6/pedago/info1/M1102_algo/ressource/java_installation.xml)

Vous êtes désormais prêt à vous servir de nos outils.

# Création du document XML

La première étape est de créer votre formulaire, pour ce faire vous devez tout d’abord ouvrir votre éditeur de texte préféré et créer un fichier « nomdefichier.xml ».

Copier-Coller cette entête dans votre fichier (vous n’avez nul besoin de savoir à quoi celui-ci sert, vos professeurs vous l’expliqueront très bien en 2ème année en cours de XML) :

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8" ?>  <!DOCTYPE form SYSTEM "form.dtd">  <?xml-stylesheet type="text/xsl" href="form.xsl"?> |

Désormais vous pouvez commencer à déclarer votre formulaire :

Le XML devra se présenter ainsi :

|  |
| --- |
| <form>  <fenetre longueur="650" titre="test" largeur="1340" x="100" y="100">  <type option1="500" option2="a01" />  <menu option1="option">  <choix option1= "2"/>  <choix option1="1"/>  </menu>  …  </fenetre>  </form> |

Les paramètres de la balise « fenetre » permettent de la positionner(x, y) et de lui donner une taille (longueur, largeur). Le « titre » définit le titre du formulaire. L’ensemble de ces paramètres sont obligatoires.

Maintenant voici les types gérés par notre application et comment sont-ils déclarés :

case (checkbox) :

|  |
| --- |
| <case x="36" y="30" label="texte" id="a01"/> |

La balise « case » permet de créer une case à cocher.

texte (text) :

|  |
| --- |
| <texte x="36" y="60" label="groupeBoutton" id="a02" type ="entier"/> |

La balise « texte » permet de créer une zone de texte renvoyant son contenu du type défini dans sa déclaration.

menu (dropdown) :

|  |
| --- |
| <menu x="36" y="120" label="deroulant" id="a03">  <choix label="ah"/>  <choix label="oh"/>  <choix label="uh"/>  </menu> |

La balise « menu » permet de déclarer un menu déroulant avec différent choix déclarés avec les balises « choix ».

tableau (array) :

|  |
| --- |
| <tableau x="500" y="120" label="box" id="a04" type="booleen" nb\_lig="7" nb\_col="8"/> |

nb\_lig : nombre de lignes dans le tableau

nb\_col : nombre de colonne dans le tableau

Attention : ces deux paramètres sont obligatoires.

La balise  « tableau » permet de déclarer un tableau renvoyant un tableau du type défini.

boutons (buttons) :

|  |
| --- |
| <boutons x="36" y="150" id="a05" label="bouton">  <bouton>25</bouton>  </boutons> |

La balise « boutons » crée une liste de « bouton » : attention sur cette liste, un seul bouton pourra être coché.

label (label) :

|  |
| --- |
| <label x="36" y="0" label="Mon label seul" id="a06"/> |

La balise « label » permet de créer un texte affiché à l’écran que l’utilisateur ne pourra modifier.

calendrier (calendar) :

|  |
| --- |
| <calendrier x="300" y="300" id="a07" label="date"/> |

La balise « calendrier » permet de créer un sélecteur de date.

Maintenant nous allons voir à quoi servent chacune des options :

x et y : permettent de donner une position à l’élément. Attention si un seul élément n’a pas de x ou de y défini, les éléments s’afficheront tous par défaut les uns en dessous des autres.

id : cette option est obligatoire pour chaque élément, il permettra dans le futur de récupérer les informations rentrées dans le formulaire et doit toujours être de la forme "axx" sachant qu’il ne peut y avoir 2 id identiques dans le même formulaire.

label : cela permet de donner un nom devant chaque élément et est aussi obligatoire.

type : permet de définir le type de retour, il est obligatoire pour les éléments texte et tableau. Les valeurs possibles sont : chaine, entier, double, booleen, caractere.

# Prévisualisation du formulaire dans un navigateur

# Intégration et récupération des données du formulaire dans un code JAVA

La troisième étape est d’intégrer le formulaire à votre code JAVA.

Avant toute chose, vous devez import notre outil par la commande :

|  |
| --- |
| import iut.algo.form.FormController; |

Cette commande se fait en haut de votre fichier.

Cela se fait donc par cette commande :

|  |
| --- |
| FormController fm = FormController.createAndGetForm("exemple.xml"); |

Cette méthode vous retournant un FormController vous devrez désormais utiliser le retour de cette méthode pour utiliser les autres méthodes. (Vous avez d’autres possibilités de créations de formulaire dans le chapitre Pour aller plus loin)

Ensuite vous avez la possibilité de modifier les valeurs des éléments comme ceci :

|  |
| --- |
| fm.setValue("a05", "test"); |

Enfin l’affichage de la frame se fait par cette commande :

|  |
| --- |
| fm.showForm(); |

Une fois affiché votre programme est mis en pause jusqu’à validation ou fermeture du formulaire.

Après que l’utilisateur a validé ou quitté, la main vous est rendue. Vous pouvez donc procéder à la récupération de vos données.

Attention si l’utilisateur a quitté sans remplir le formulaire, vous pouvez obtenir des erreurs lors de la récupération des données. Vous avez donc à votre disposition une méthode permettant de savoir si l’utilisateur a quitté ou validé :

|  |
| --- |
| boolean b = fm.isValid(); |

Cela se fait donc par cette commande par exemple :

|  |
| --- |
| int i = fm.getInt("02"); |

Attention les id ne sont que les chiffres présents dans l’id du fichier XML :

Exemple : a02 🡺 02

Il existe donc autant de commandes qu’il y a de types de retour :

getInt(id);

getString(id);

getDouble(id);

getChar(id);

getBoolean(id);

getArray(id);

Attention : pour récupérer le bouton qui a été sélectionné dans la balise « boutons », vous devez faire getInt(id) car ceux-ci renvoient le numéro, passé en paramètre lors de leur création, de celui sélectionné.

De plus, afin de récupérer les valeurs du tableau, il vous faut effectuer ceci :

|  |
| --- |
| Object[][] tableau = fm.getArray("04") |

# Pour aller plus loin…

Notre outil vous offre plusieurs options supplémentaires vous permettant d’aller plus loin.

Premièrement vous avez possibilité de changer la langue de votre formulaire, en l’occurrence en anglais.

Dans ce cas il vous suffira juste de changer les mots :

* fenetre 🡺 window
* longueur 🡺 length
* largeur 🡺 width
* titre 🡺 title
* texte 🡺 text
* chaine 🡺 string
* entier 🡺 int
* booleen 🡺 boolean
* caractere 🡺 char
* menu 🡺 dropdown
* choix 🡺 choice
* case 🡺 checkbox
* tableau 🡺 array
* nb\_lig 🡺 nb\_row
* bouton 🡺 button
* boutons 🡺 buttons
* calendrier 🡺 calendar

Le reste ne change pas.

Si vous ne souhaitez pas utiliser la méthode : FormController.createAndGetForm("exemple.xml"); Vous pouvez toujours utiliser ces deux méthodes :

|  |
| --- |
| String id = FormController.createForm("exemple.xml"); |

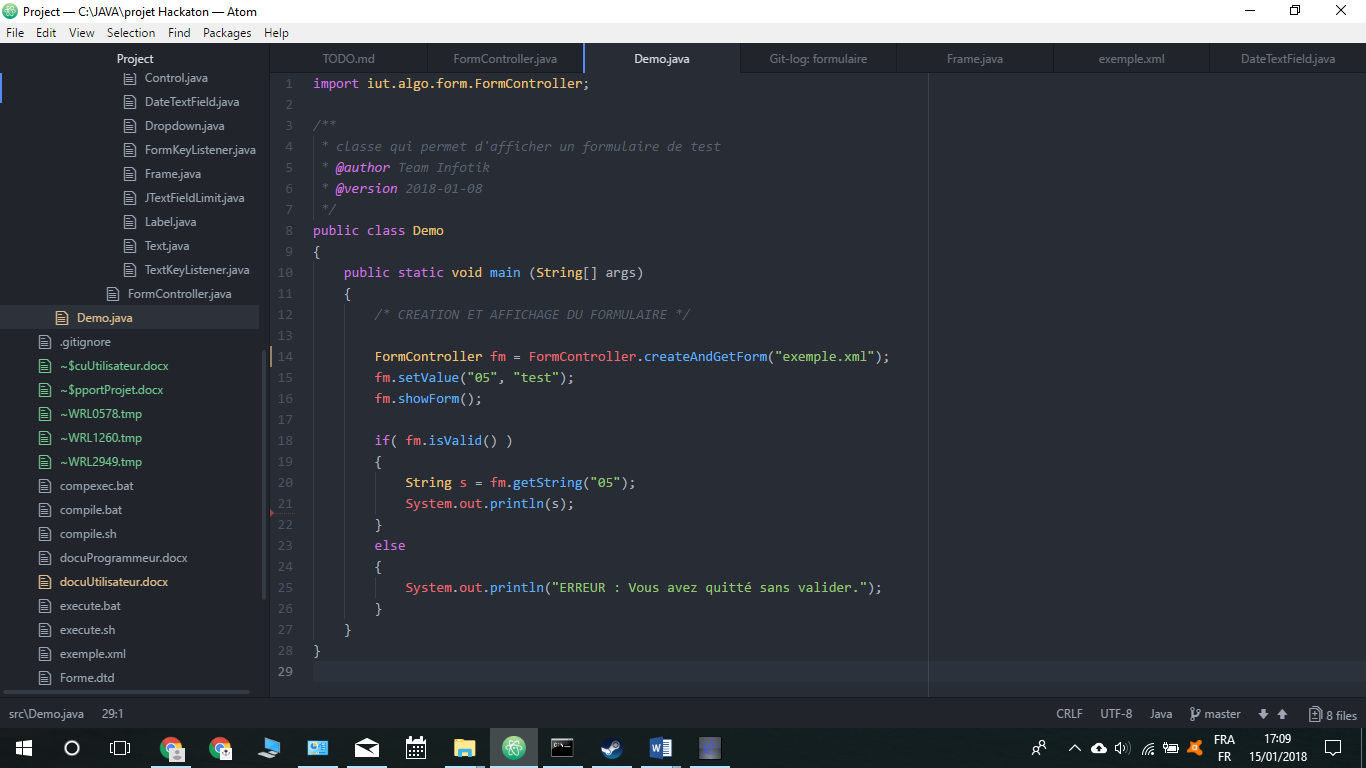
Cette méthode vous retourne un id de FormController à utiliser dans la prochaine méthode.

|  |
| --- |
| FormController fm = FormController.getForm(id); |

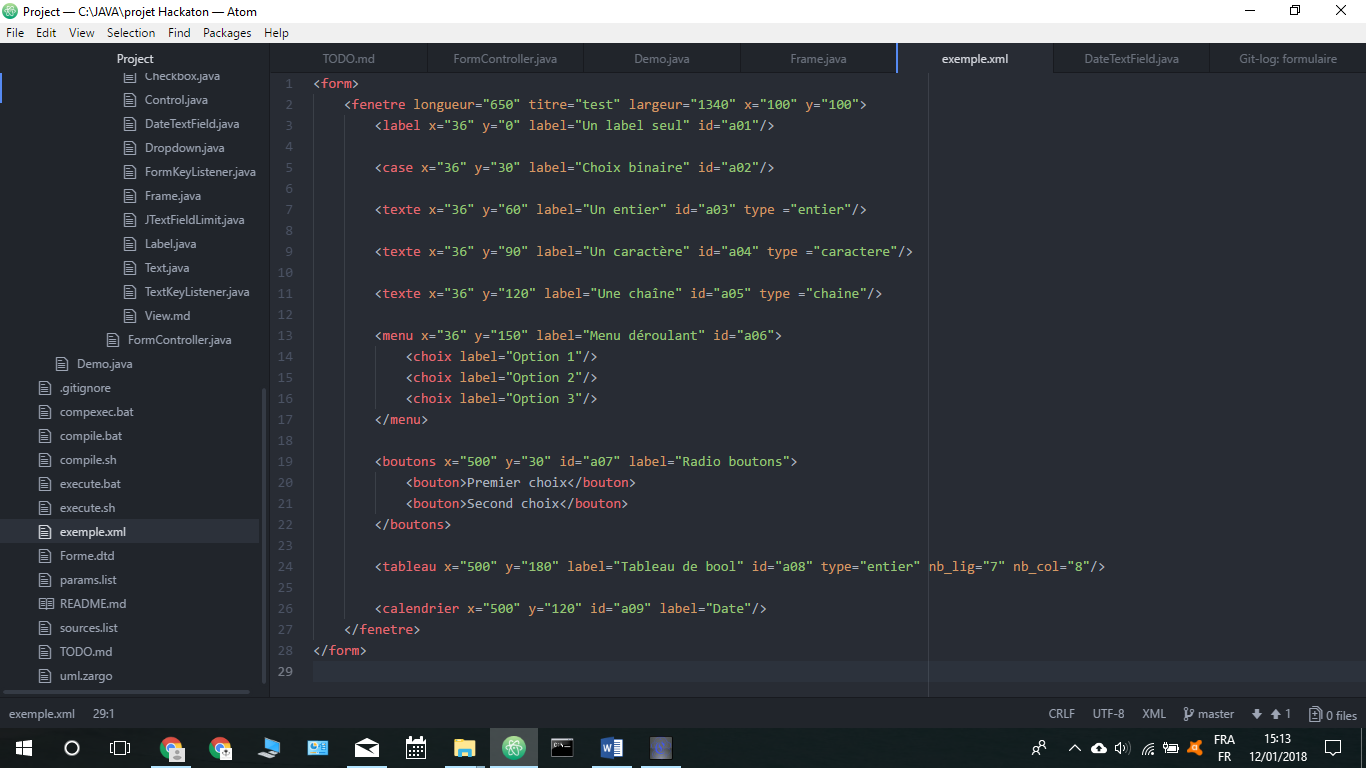
Cette méthode vous retourne alors le FormController à utiliser pour les autres méthodes.

# Exemples

Demo.java :



exemple.xml :



Résultat :

