IN201 : 11 – GUI 1

Author : Christophe Garion <garion@isae.fr>
Public : SUPAERO 2A
Date :

Résumé

Le but de ce TP est de construire une application en utilisant le patron de conception MVC et d'utiliser quelques composants de Swing.

1 Objectifs

Les objectifs du TP sont les suivants :
– comprendre le MVC en construisant une application ;
– manipuler quelques composants simples de Swing.

2 Présentation du problème

On souhaite réaliser une application qui stocke des chaines de caractères permettant de tester la faisabilité d'un *chat*. Cette application devra posséder deux vues graphiques qui permettront d'envoyer une chaîne de caractères et qui afficheront sur une zone de texte les messages reçus par l'application.

3 Définition du modèle

Le modèle sera donc une application `Chat` qui stockera un ensemble de chaînes de caractères sous la forme d'un objet de type `java.util.ArrayList`. Les services rendus par cette classe seront :
– ajouter une chaîne de caractères dans la liste ;
– récupérer le dernier élément de la liste.

4 Définition des vues

Les vues seront des fenêtres graphiques composées des éléments suivants :
– un bouton permettant d'envoyer le texte à l'application ;
– une zone de texte `JTextField` pour écrire le texte à envoyer ;
– une zone de texte pour écrire les messages provenant de l'application. Ce sera un objet de type `JTextArea`. `JTextArea` possède une méthode `append(String s)` permettant de rajouter une chaîne de caractères dans la zone.

5 Conception de l'application

Proposer un diagramme de classes simples modélisant l'application. On s'appliquera à bien choisir la façon dont la vue et le modèle vont communiquer.

6 Implantation

Il faut maintenant implanter la classe applicative, les vues et les contrôleurs associés. Vous pourrez réutiliser les classes et interfaces développées dans le TP 10 sur le *pattern* Observateur. Vous pourrez également utiliser les classes et interfaces fournies par le JDK implantant ce *pattern*, i.e. `java.util.Observable` et `java.util.Observer`. Cette classe et cette interface fonctionnent à peu près comme celles que nous avons développées, sauf que l'on peut passer des paramètres aux méthodes correspondant à `miseAJour`, `avertir` etc. De plus, la classe `java.util.Observable` utilise un booléen pour représenter que l'état de l'objet a réellement changé (pour éviter de prévenir les observateurs si ce n'est pas nécessaire). Référez-vous à la javadoc de ces classes pour plus de détails ou au corrigé du TP10.

1. écrire la classe `Chat` et la tester rapidement.
2. écrire la classe `VueChat` et vérifier qu'elle s'affiche correctement. On pourra utiliser d'autres *layout managers* que celui vu en cours si nécessaire.

3. tester l'application.