M2 Informatique - HMIN336 - Réutilisation et Composants
TD-TP No 3

Composants JavaBeans - Environnement NetBeans

Le but du TD-TP est la compréhension par la pratique du concept de composant dans sa version *JavaBeans* et l'apprentissage de l'environnement SUN(ORACLE) *NetBeans*. Ce TP a été écrit pour la version 7.2 et la version installée est la 8.1 (commande *netbeans*, ou *netbeans8*). C'est presque pareil mais le descriptif du TP n'est donc pas en exacte conformité avec la version installée. Il se peut en conséquense que quelques éléments d'interface soient différents (présence de tel ou tel item dans tel menu). J'ai testé que l'exercice fonctionne en v8.1.
Le concept de composant Bean est aujourd'hui intégré dans *JavaFX*, https://openjfx.io/javadoc/11/javafx.base/module-summary.html, sous-partie de *Oracle OpenJDK*. Je recherche une personne intéressée par le portage de ce TP en JavaFX (m'en parler si c'est le cas).

1 Présentation générale

Soit à réaliser une application graphique de contrôle de l'entrée des spectateurs dans une salle de concert dotée de trois portes. L'exemple prend son sens lors des évènements gratuits, la salle de concert compte les entrées et empêche l'accès (ferme les portes) lorsque le nombre de spectateurs maximum est atteint. Chaque porte d'accès à la salle est représentée par un composant contenant un compteur associé à deux boutons (*entrée* et *sortie*) et à un afficheur indiquant le cumul des entrées-sorties via cette porte. La salle est représentée par un composant contenant un compteur associé aux trois portes et à un afficheur indiquantdonc le nombre de personnes présentes dans la salle. Quand la salle est pleine les portes doivent être prévenues que toute nouvelle entrée est impossible tant qu'une sortie n'a pas eu lieu.
Ceci est une spécification de base que vous pouvez modifier à votre guise, l'essentiel étant dans tout cela de comprendre les concepts cachés sous les clics.

Seront donc à réaliser différentes sortes de composants.

- Un composant **invisible** `compteur` doté des méthodes classiques `incr`, `decr`, `raz` ainsi que des méthodes `start()` et `stop()`. Après réception du message `stop()` ou avant réception du message `start()`, un compteur ne peut ni incrémenter ni décrémenter sa valeur.
- Un composant **visible** `afficheur` capable d'afficher des chaînes ou des nombres.
- un composant visible `porte` intégrant un compteur, des boutons et un afficheur d'entiers.
- un composant visible `porte` encapsulant un compteur, un afficheur et deux boutons, capable de recevoir les messages `open()` et `close`, et d'indiquer la valeur de son compteur.
- Un composant `salleConcert` composé de différentes portes, capable d'afficher le nombre toral de spectateurs présents et mémorisant d'un spectacle à l'autre, en vue d'optimisation, les entrées les plus utilisées.

2 Prise en main de NetBeans

Cette section vous indique une solution pour créer un projet avec des composants. Dans cette prise en main vous créerez des fichiers d'exemple que vous détruirez ensuite.
Il existe diverses façons de réaliser la même fonctionnalité, je donne celle qui m'a semblé la plus efficace mais n'hésitez pas à faire des essais, à détruire et à recommencer. Soyez aussi patients que je l'ai été pour découvrir ce nouvel environnement.
C'est une bonne idée de consulter des aides en ligne :
http://docs.oracle.com/javase/tutorial/javabeans/index.html et
http://wiki.netbeans.org/NetBeansJavaBeansTutorial

- Créer une nouveau projet (de nom "Appli1" par exemle) de type "général" ou "Java Application". Ne pas cocher (ou décocher) l'option de création d'une méthode main.

— Dans ce projet, créer un nouveau package (“appli1”). Ne pas utiliser le package par défaut, source de divers problèmes. D’une façon générale toujours utiliser le menu contextuel associé à une entité (control-click sur l’entité) pour la modifier.
— Dans ce nouveau package vous pouvez créer diverses choses dont des classes ou des composants. Pour créer un composant “écouteur”, une classe standard fait l’affaire (donc “new-Java-class”), par contre pour créer un composant “écouté” on gagne un peu de temps à créer un “javabean component” (voir menu “new-others-javaBeansObjects”)
— Dans le package précédent, créer une classe `Test1`, définissant des composants visibles de type Jlabel doté d’un attribut permettant de stocker un texte. `Test1` est une sous classe de `Jlabel`.
Dotez cette classe d’une méthode publique `switchText` qui change le texte du label. Faites en sorte que cette méthode switche entre deux textes.
— Il y a diverses façons de compiler vos classes. Je vous suggère l’option “build project” dans le menu contextuel du projet “Appli1”.
— Pour placer le composant que vous venez de créer dans la palette d’édition, utiliser l’item “Palette” - option SWING/AWT components - dans le menu “Tools”. Choisissez “Add from JAR” et sélectionner l’archive “Appli1.jar” dans le répertoire “dist” du répertoire “Appli1” qui se trouve dans votre répertoire principal si vous avez suivi les options par défaut.
Ajouter le bean `Test1` que vous venez de créer à la palette dans la catégorie “Beans”.
Une autre solution est de faire le `build` puis d’utiliser le menu contextuel “tools - add to palette”.
Attention à chaque fois que vous faites une modification dans le code d’un composant, il faut enlever l’ancien de la palette, refaire le build et remettre le composant dans la palette (à moins que vous ne trouviez mieux ... car c’est pénible ... peut-être la version 8.1 évite-t-elle cela).
— La palette apparaît quand on développe la partie graphique de l’application. Pour cela, au niveau du projet “Appli1”, sélectionnez “new Jframe Form” et donnez un nom (par exemple “Graph1”) à cette partie graphique de l’application. Vérifier alors que la palette contient bien dans la section “Beans” le composant `Test1` que vous avez ajouté.
— Sélectionnez le composant `Test1` dans la palette et faites un glisser-déposer vers la zone d’édition centrale.
— L’environnement à ce stade présente 5 fenêtres importantes. A gauche la fenêtre des projets, en dessous un *navigateur-inspecteur* dont nous reparlerons. Au milieu la fenêtre d’édition avec deux options : “source” ou “design”. En haut à droite la palette des composants, en bas à droite, l’éditeur de propriétés des composants sélectionnés.
— La fenêtre d’édition vous permet en mode “design” de créer graphiquement votre application en **glissant/déposant** des composants (AWT, SWING ou BEANS) et en établissant des connexions entre eux. Le mode “source” permet de consulter et compléter (si nécessaire) le code généré.
— Les composants visibles (AWT, Swing, ou VisibleBeans) sont visibles dans la fenêtre d’édition. Les composants invisibles (NonVisibleBeans) ne sont visibles que dans le “navigateur-inspecteur” sous la rubrique (“other components”).
— La connexion entre composants s’effectue en mode “Connection Mode” (4ième icone de la fenêtre d’édition). Sélectionnez le mode, cliquez sur le composant source d’évènement puis sur le composant cible puis laissez vous guider par l’interface.
Déposer un composant AWT-Button sur la fenêtre d’édition puis connectez le à votre composant Test1, pour que, à chaque fois que le bouton est cliqué, la méthode `switchText()` soit appelée.
— Une fois la connexion réalisée, vous pouvez examiner le code résultant grace à l’onglet “source”.
— Pour tester l’application, utiliser “Run Project” dans le menu contextuel du projet, après avoir fait le “build”.

3 Etape 2 : créer une application Compteur

Réalisez une seconde application semblable à celle du cours dans laquelle on réalisera un compteur non visible, composant de type “écouté”. Pour cela une petite aide vous est donné si vous choisissez “new JavaBeans Component” au lieu de “new Class”, le code d’une propriété “SampleProperty” est écrit, que vous pouvez modifier.

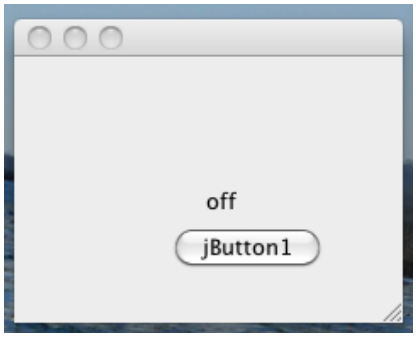

FIGURE 1 – Execution de l'application exemple No 1

Une fois la classe définissant les composants "compteur" créée et mise dans la palette, vous connecterez graphiquement un compteur à deux boutons *incr* et *decr* et à un afficheur d'entier.
Attention pour connecter un composant invisible à un autre, il faut faire les connections dans le "navigateur-inspecteur", pas dans la fenêtre des composants visibles.
Pour tester l'application, utiliser "Run Project" dans le menu contextuel du projet, après avoir fait le "build".
Points à constater/comprendre :

— A noter une implantation nouvelle du schéma de conception "Adapteur" dans laquelle la classe de l'adapteur est remplacée par une simple méthode de la classe. Visualisez ainsi les deux méthodes générées pour connecter les boutons au compteur.
Comprendre l'utilisation de l'évènement `PropertyChange` et l'utilisation qui en est faite pour connecter le compteur à l'afficheur. Si vous ne comprenez pas les options qui vous sont proposées pour le passage de paramètres, choisissez celle qui vous permet de donner vous-même le code.

4 Etape 3

Réalisez le composant `Porte` de l'application exemple. Pour cela il faut globalement reprendre ce qui a été fait à l'étape 2 mais de remplacer `JFrame` par `JPanel`. Ceci vous permettra de créer un composant que vous pourrez placer sur la palette plutôt qu'une application finale. Il faut également trouver une solution pour réaliser les méthodes `start` et `stop`.

5 Etape 4

Réalisez l'application de gestion de la salle de spectacle. L'idée est qu'il soit possible d'y intégrer des portes liées entre elles (idem il faudra y remplacer dans Porte, un `JFrame` par un `JPanel`. Cela montre une limite de ce système, il faut choisir entre une appli et un composant.

6 A vous de jouer

Faites des liens et testez. Les propriétés liées se retrouvent dans divers environnement ou frameworks d'assemblage par observation, dont `Spring` (recherche web).
Testez les propriétés contraintes. Testez la sérialisation. Testez les *beanDescriptors*. Inventez d'autres composants. Essayer par exemple de réaliser un composant *Timer* qui emet un évènement toutes les x secondes. Il pourrait servir dans l'application "salle de concert" pour mémoriser un état des entrées toutes les minutes en vue de connaître la répartition temporelle des arrivées.