Licence MASS – INF F3 – TP n° 2

2013–2014

Le but de ce TP est de se familiariser avec l'agencement des éléments à l'intérieur d'une fenêtre graphique. L'intérêt est de connaître les possibilités offertes pour disposer les composants de façon logique et ordonnée.

1 Quelques layouts de base

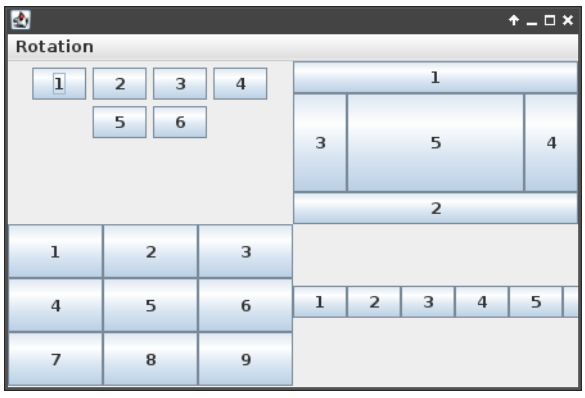
1.1 Préambule

Le but de ce premier exercice est de se familiariser avec les `JPanel`s et les différents layouts pour comprendre comment les différents placements possibles peuvent répondre à divers besoins en terme d'affichage. Plusieurs variantes de `JPanel` sont disponibles dans l'éditeur visuel mais seul le `JPanel` de base sera vu dans ce TP.

En bref, un `JPanel` est un conteneur. C'est à dire que son rôle est de contenir et disposer des éléments graphiques. C'est aussi un élément graphique qui peut être contenu dans un autre `JPanel`. L'agencement des éléments à l'intérieur d'un `JPanel` est réalisé à l'aide d'un layout qui est associé au `JPanel`. Chaque type de layout a ses propres règles pour placer les composants les uns à côté des autres.

Ce qu'il faut retenir, c'est que chaque « zone » d'une fenêtre graphique doit être contenue dans son propre `JPanel` avec un layout adapté. L'agencement des zones les unes par rapport aux autres se fait à l'aide d'un `JPanel` qui va contenir et agencer les différentes catégories et ainsi de suite jusqu'a ce que tous les `JPanel`s restant soient contenus dans le contentPane de la fenêtre (ce contentPane étant le seul `JPanel` que la fenêtre peut contenir directement).

1.2 Exercice



Utilisez des `JPanel`s et les layouts `BoxLayout`, `GridLayout`, `FlowLayout` et `BorderLayout` pour réaliser l'interface précédente. Prenez le temps de comprendre le comportement de chacun des layouts en redimensionnant la fenêtre horizontalement et verticalement.

Utilisez les éléments de menu créés par défaut par l'éditeur visuel pour créer le menu `Rotation` qui contient deux éléments (`JMenuItem`) : `Sens horaire` et `Sens anti-horaire`. Les éléments de menu sont semblable à des boutons et un clic sur l'un d'entre eux doit déclencher un déplacement des différentes zones de la fenêtre dans le sens sélectionné. Pour ce faire, il faut d'abord vider le contentPane des `JPanel`s qui le composent et les remettre dans le bon ordre et, ensuite, appeler la fonction `validate` pour redessiner la fenêtre avec les éléments déplacés. La figure 1 illustre une rotation en sens horaire.

Et pour finir, ajoutez un menu `BoxLayout` contenant une case à cocher (`JCheckBox` au lieu de `JMenuItem`). Cocher ou décocher la case est similaire à un clic sur un bouton et il est possible de savoir si la case est cochée ou non grâce à un appel de la forme `nomCase.isSelected()` où `nomCase` est le nom de la variable contenant la case à cocher. Lorsque cette case est cochée, le layout du `JPanel` paramétré avec un `BoxLayout` doit devenir un `BoxLayout` vertical. Et quand la case est décoché, ce layout doit redevenir un `BoxLayout` horizontal.

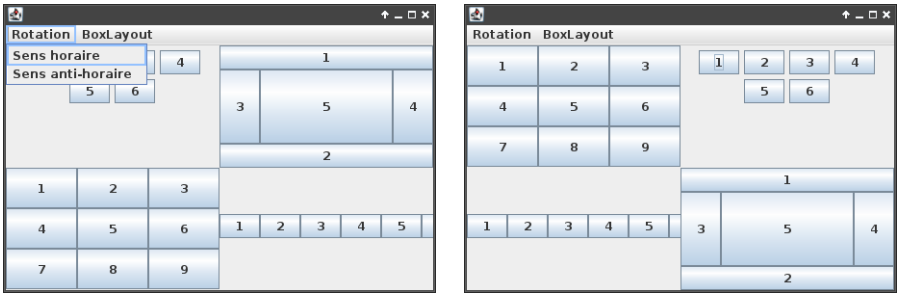


FIGURE 1 – Illustration de la rotation

2 Layouts inadaptés

Créez une fenêtre contenant deux zones distinctes. La zone de gauche devra comprendre 20 boutons répartis dans une grille 4×5 et la zone de droite servira pour la sélection. Utilisez la technique que vous souhaitez pour proposer 4 choix différents (4 boutons, 4 boutons radio, une liste de choix, un sélecteur...). Chacun de ces choix changera le layout de la zone de gauche pour permettre d'alterner entre les 4 layouts proposés dans l'exercice précédent.

Que se passe-t-il avec le `BorderLayout` ?

Faites ensuite en sorte de vider la zone de gauche de ses boutons avant de changer le layout et de remettre les boutons ensuite.

Même question, que se passe-t-il avec le `BorderLayout` ?