Licence MASS – INF F3 – Examen de TP

24 janvier 2014 – Durée 2h

Le but de ce sujet de TP est de réaliser un mini jeu de mastermind en séparant le cœur du jeu de l'interface graphique. Pour rappel, le mastermind est un jeu de déduction par élimination. Il faut deviner une séquence (chiffres ou couleurs) par propositions successives. À chaque proposition, les éléments bien et mal placés sont indiqués, jusqu'à qu'il n'y ait plus que des éléments bien placés ou que le joueur ait épuisé son quota de propositions.

1 Déroulement de la séance

Commencez par créer un nouveau projet avec votre nom. Toutes les classes que vous créerez au cours de cette séance le seront dans le package mastermind (que vous devrez créer) de ce projet. Une fois votre travail terminé, exportez le projet (menu fichier) sous forme d'archive et suivez les consignes du professeur pour rendre votre travail.

2 Travail demandé

2.1 Première partie (9 points)

2.1.1 Mise en place

Commencez par créer une classe `Mastermind` dont le constructeur va initialiser un nombre à deviner entre 1000 et 9999 ainsi qu'un nombre de tentatives qui sera passé par paramètres.

Ajoutez y un *getter* pour connaître le nombre de tentatives restantes ainsi qu'une méthode dont la signature est `boolean essai(int)`. Cette méthode renvoie une valeur indiquant si le nombre passé en paramètre est le nombre à deviner ou non. Cette méthode devra aussi décrémenter le nombre de tentatives et gérer le cas où il n'y a plus de tentatives restantes.

2.1.2 Interface graphique

Créez une classe visuelle comprenant 3 zones (exemple plus bas) : l'une pour afficher les anciennes tentatives pour deviner le nombre, l'une pour afficher les retours les résultats qu'ont donné ces tentatives et la troisième contenant une zone de saisie de texte, un bouton de validation et un label indiquant le nombre de possibilités restantes.

Cette classe visuelle va contenir un attribut de type `Mastermind` initialisé par défaut à 20 tentatives. Le bouton est responsable de la lecture du nombre dans la zone de saisie de texte et de l'appel de la méthode correspondant à la divination de l'objet `Mastermind`. Le bouton est aussi responsable de la mise à jour de l'interface en fonction des informations renvoyées par cette méthode.

Ajoutez deux menus pour recommencer des parties avec des niveaux de difficulté variables. Ces menus vont réinitialiser l'attribut de type `Mastermind` avec des valeurs de tentatives variables (par exemple 20 et 10 pour normal et difficile). Ces menus sont aussi responsables d'enlever toutes les informations relatives à la partie précédente des différents panels. N'oublier pas de faire appel à la méthode `repaint` des panels *puis* à la méthode `validate` pour valider les changements graphiques.

2.2 Seconde partie (5 points)

Pour deviner plus facilement le nombre, modifiez et adaptez la méthode `essai` de la classe `Mastermind`.

Cette méthode doit dorénavant renvoyer un tableau de 4 éléments. Le type des éléments à l'intérieur du tableau est laissé à votre convenance. Les valeurs de ces éléments devront représenter si un chiffre est bien placé dans le nombre à deviner ou si ce n'est pas le bon chiffre à cette place.

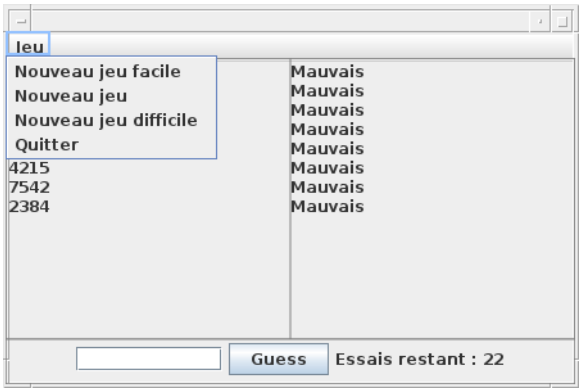

FIGURE 1 – Exemple d'interface graphique

Modifiez l'interface graphique en conséquence pour représenter ces éléments dans la zone d'information.

2.3 Troisième partie (6 points)

On désire désormais gérer la difficulté à l'intérieur de la classe `Mastermind`.

Modifier le constructeur pour que le paramètre représente le niveau de difficulté (1 : difficile, n'importe quoi d'autre : normal). Vous pouvez utiliser une énumération à la place d'un entier si vous voulez. Ce niveau de difficulté va servir à initialiser le nombre de tentatives avec des valeurs adéquates. Modifiez la classe visuelle pour prendre en compte ce changement.

Modifiez ensuite la méthode de `essai`. En mode difficile, le tableau de réponse est trié (toutes les réponses « chiffre bien placé » en tête). En difficulté normale, les chiffres bien placés restent à leur position et vous devez rajouter une réponse pour permettre de différencier les chiffres mal placés de ceux qui ne sont pas dans le nombre à deviner.

Par exemple, si le nombre à deviner est 4526 et que le nombre proposé est 3428, alors le 2 est bien placé, le 4 est mal placé et les chiffres 3 et 8 ne sont pas présent. Une représentation visuelle de la réponse pourrait être —XO— en difficulté normale et O——— en difficulté élevée.

3 Bonus

Si les questions précédentes fonctionnent correctement, vous pouvez rajouter un niveau de difficulté supplémentaire : facile.

Commencez par ajouter le menu correspondant à l'interface graphique ainsi que l'`actionPerformed` qui lui est associé.

Ajoutez la gestion du mode facile au constructeur de la classe `Mastermind`.

La gestion du mode facile dans la méthode `essai` est la suivante : si un chiffre n'existe pas dans le nombre à deviner, on regarde s'il est proche (+1 ou -1) du chiffre situé à la même position dans le nombre à deviner. Une nouvelle valeur de réponse est alors affecté à ce chiffre.

En reprenant l'exemple précédent : 3428 proposé pour deviner 4526, alors 3 est proche, 4 est mal placé, 2 est bien placé et 8 n'existe pas. Ce qui pourrait se traduire visuellement par ~XO—.