IN101 – TP6 2009/2010

Sujet (à n'imprimer qu'une seule fois par poste de travail)

Durée : 3 h

1 OBJECTIFS

- Maîtriser les classes et méthodes abstraites
- Maîtriser les interfaces
- Maîtriser les exceptions
- Savoir se passer de BlueJ

2 TRAVAIL A REALISER

Nota : le travail demandé doit être terminé, en séance ou, à défaut, hors séance.

2.1 Créer un répertoire de travail

Si Bluej n'est pas ouvert, le lancer. **Visualiser le sujet dans un navigateur** pour bénéficier des liens.

Créer un répertoire `tp6` dans `In101` sur votre compte. *C'est dans ce répertoire que devront être stockés tous les programmes Java et exercices relevant de ce tp.*

2.2 Exercice 1 : Projet "DBShapes" (classe/méthode abstraite, interface)

Cet exercice va consister à transformer le projet `DBShapes` du tp4.

2.2.1 Ouvrir le projet

Télécharger le fichier `DBShapesP.jar` lié à cet énoncé, et l'enregistrer dans le répertoire `tp6` précédemment créé.

Lancer *BlueJ* et ouvrir, le fichier `.jar` sauvegardé ci-dessus. [*Ouvrir non-BlueJ* ...].

2.2.2 Découvrir et essayer le projet `DBShapesP`

Ce projet correspond à peu près à ce à quoi vous deviez aboutir à la fin du TP4.

2.2.3 Restructurer ce projet

1. Comme cela a été expliqué en cours, il n'est pas souhaitable d'autoriser l'instanciation de la classe `DBShape`. Faites ce qu'il faut pour cela et vérifiez que cela n'a rien changé ni à la compilation ni à l'exécution (c'est normal, il n'y avait pas d'instanciation de `DBShape`).
2. Ajoutez une sous-classe `Hexagone` (recopiée sur `Square` par exemple) mais en supprimant `drawSpec()` car vous ne savez pas comment dessiner un hexagone pour le moment. Ajoutez un `Hexagone` dans `Picture` et rendez-le visible. Compilez. Créez une `Picture` : le message est cohérent, mais est-il souhaitable ?
3. Comme cela a été expliqué en cours, il n'est pas souhaitable d'une part que les instructions de certaines méthodes soient inutiles, et d'autre part qu'une sous-classe puisse être ajoutée sans qu'elle possède les caractéristiques minimales de ses « sœurs » telles que les méthodes `drawSpec()` et `changeSizeSpec()`. Faites ce qu'il faut à ces méthodes pour corriger cela dans `DBShape` et recompilez `Hexagone`; il devrait y avoir une erreur.

4. Pour résoudre ce problème de compilation, il faut maintenant savoir dessiner un hexagone. En s'inspirant de `drawSpec()` dans `Triangle`, de la javadoc du constructeur de `Polygon`, et de l'aide ci-dessous, écrivez la méthode `drawSpec()` dans `Hexagone`.
   Aide : Une façon de dessiner un hexagone est d'utiliser un `Polygon` avec les 6 sommets suivants : (x, y+s), (x+g, y+p), (x+g, y-p), (x, y-s), (x-g, y-p), et (x-g, y+p), avec (x, y) les coordonnées du centre, s la taille du côté, p=s/2, et g=(√3)s/2. *(voir dessin en bas de page)*
   Vérifiez que l'hexagone dessiné est « régulier » et non pas « allongé ».

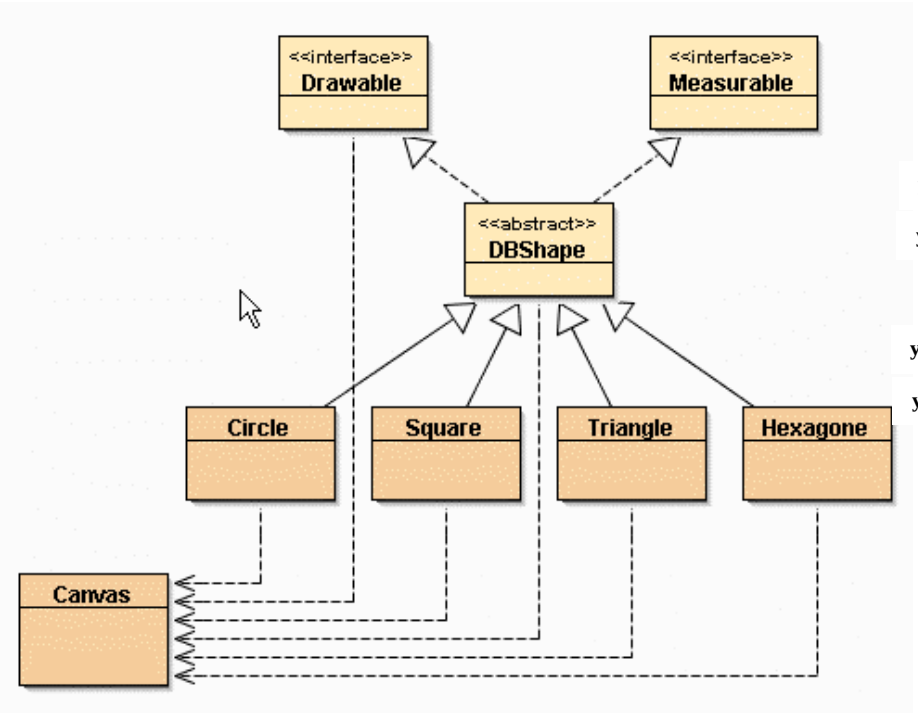
5. On veut maintenant pouvoir « hériter » de plusieurs caractéristiques telles que `Drawable`, `Measurable`, … L'héritage multiple étant interdit en java, nous allons donc implanter des interfaces. Créez l'interface `Drawable` ne comportant que `draw()`, *(pourquoi pas `drawSpec()` ?)*, `erase()`, `isVisible()`, `makeVisible()`, et `makeInvisible()`. `DBShape` doit maintenant respecter cette interface, mais peut-elle implémenter `drawSpec()` ? Modifiez ce qui doit l'être pour permettre la compilation et vérifiez que tout fonctionne comme avant.

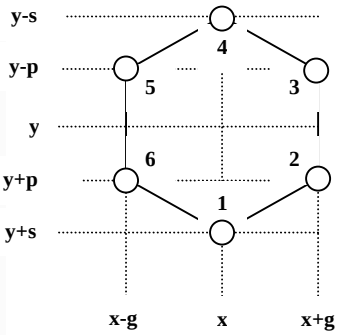
6. Créez une nouvelle interface `Measurable` imposant les 2 méthodes réelles `surface()` et `perimetre()`. `DBShape` doit maintenant respecter aussi cette interface, mais peut-elle implémenter `surface()`et `perimetre()` ? Ajouter ce qu'il faut dans les sous-classes.
   Aide : la surface d'un hexagone vaut 1.5(√3)s² et tous nos triangles sont forcément isocèles. `math.hypot()` peut certainement vous servir.

7. Ajoutez dans `Picture` avant le deuxième `System.out.println` (resp. après le dernier `System.out.println`) la création d'un tableau de `Measurable` contenant les 6 (resp. 3) formes existant à ce moment du programme.

8. Ajoutez juste après un appel à la procédure (qu'il vous faudra écrire) `printStats()` qui devra calculer et afficher la somme des surfaces puis la somme des périmètres des formes du tableau qu'on lui passe en paramètre.
   Testez ; ça devrait donner environ 27139.08 et 1454.43, puis 6531.36 et 513.36.
   Au fait, le nombre d'instances de DBShape est-il correct ?

9. A la fin de l'exercice, on devrait se retrouver dans cette situation :





2.2.4 Développer sans BlueJ

1. Fermez *BlueJ* et ouvrez un terminal dans le répertoire `DBShapesP`.
2. Utilisez un éditeur de texte (*nedit*, *emacs*, *KWrite*, …) pour modifier le fichier `pkg_application/Picture.java`.
3. Ajoutez l'affichage d'une ligne d' `*` au début et à la fin de `printStats()`.
4. Sauvegardez, et quittez l'éditeur.
5. Compilez par : `javac pkg_application/Picture.java`
   (on indique ici le chemin d'accès complet au fichier `.java`)
6. Exécutez par : `java pkg_application.Picture`
   (on indique ici le nom complet de la classe, avec son paquetage)
   une erreur se produit ? Corrigez-la en créant juste un objet `Picture` là où il faut.
   (attention à la signature O-BLI-GA-TOIRE de la méthode à créer – **voir résumé du cours 5**)
7. Recompilez, retestez : ça marche mais c'est bloqué ? `System.exit(0);` permet de terminer le programme (on peut remplacer le 0 par un code d'erreur destiné au système).
8. Recompilez, retestez : ça marche mais on n'a plus le temps de voir ? **On attendra le 2.3.1 pour corriger.**
9. Relancez l'éditeur (en n'oubliant pas le `&` à la fin de la ligne pour ne pas avoir à quitter l'éditeur de texte) pour modifier les 4 sous-classes de `DBShape`. Changez la couleur par défaut du cercle en vert, du carré en jaune, du triangle en bleu, et de l'hexagone en rouge.
10. Sauvegardez chaque fichier sans sortir de l'éditeur.
11. Compilez par : `javac pkg_graphique/*.java`
12. Exécutez par : `java pkg_application.Picture` (ou déboguez avec `jdb`)
13. Générez la documentation par : `javadoc –author –version –d userdoc pkg_*`
    En cas de problème d'accents, ajoutez l'option `-encoding ISO8859-1`
14. Ouvrez le fichier `userdoc/index.html` dans un navigateur et parcourez la documentation.
15. Générez maintenant une documentation « programmeur » en ajoutant les options :
    `–private –linksource –d progdoc`
16. Ouvrez le fichier `progdoc/index.html` dans un navigateur et parcourez la documentation et les sources !

T. SVP →

2.3 Exercice 2 : Les exceptions

2.3.1 Exemple simple de traitement d'exception « silencieux »

1. Pour pouvoir observer le dessin depuis le 2.2.4.8, il faut ajouter une pause grâce à `Thread.sleep( 5000 );` Essayez. Une erreur de compilation ?
2. Consultez la documentation de cette méthode pour comprendre ce qui se passe, et ajoutez le try/catch nécessaire, sans rien faire en cas d'exception *(à éviter en général)*.
3. Recompilez et retestez.

2.3.2 Programmation défensive

Reprenez votre projet Eratosthène du tp5. Ajoutez des déclenchements d'exception (par exemple `IllegalArgumentException`) lorsque le paramètre de certaines méthodes est négatif ; veillez à ce que les messages soient informatifs et en particulier contiennent la valeur considérée comme « mauvaise » et l'intervalle souhaité.

2.3.3 Traitement des exceptions plausibles

Reprenez votre projet Moyenne du tp5. Traitez l'exception `NumberFormatException` qui peut se produire si autre chose qu'un nombre se trouve sur la ligne de commande ; veillez à ce que les messages soient informatifs, mais le calcul de la moyenne doit continuer comme si de rien n'était.

2.3.4 Compréhension des mécanismes

Ajoutez dans un des projets BlueJ du jour une nouvelle classe `FinallyDemo` que vous devez copier/coller à partir de cette page.
Compilez. Exécutez `main()`. Comparez le code java et les affichages dans le terminal.

2.4 Terminer la séance

Si pas fait antérieurement, **générer** (après l'avoir complétée !) la documentation de ce dernier projet et de DBShapesP, puis sauvegarder les projets ouverts, puis fermer BlueJ. Si besoin, envoyer par mél à votre binôme, en fichiers attachés, tous les projets de ce tp (exportés sous forme de fichiers `.jar`). Se déloger.

Ce sujet a été élaboré par Denis Bureau.