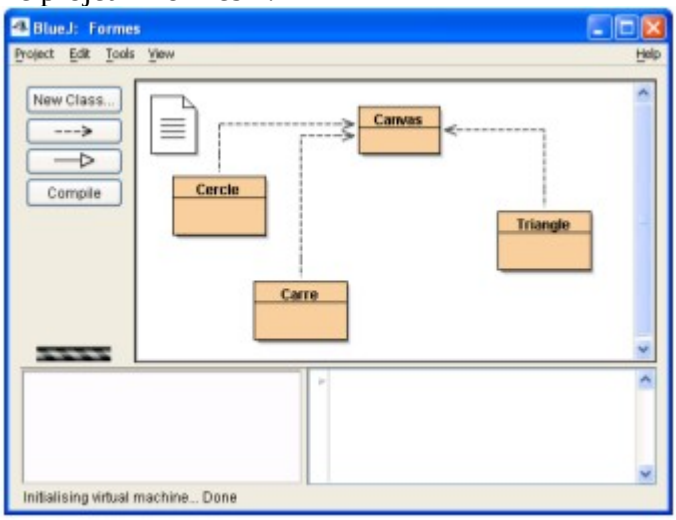
TP3 Programmation Java

Lancement de BlueJ et ouverture du projet « Formes »

Lancez le logiciel BlueJ. Une fenêtre apparaît, soit vide, soit contenant déjà un projet (le logiciel peut se souvenir d'un autre travail). Pour être bien sûr que vous travaillez sur un projet propre et non pollué par n'importe qui, vous allez demander « Close » (fermer le projet courant) dans le menu « Project » (si vous ne voyez pas le menu « Project », cliquez dans la fenêtre de BlueJ pour amener le logiciel au premier plan) ... Une fois que le projet courant s'est évaporé, demander « Open Project... » puis ouvrez le projet « Formes ».



Consultez rapidement la documentation du projet, par un double clic sur l'icône 
Il s'agit d'un fichier-texte, sans gras ni italique. Le nom du fichier est d'ailleurs README.TXT dont l'**extension**[1] TXT (ce qui se trouve après le point) indique du texte brut. Ne vous étonnez pas si vous voyez de temps en temps des caractères comme '‡' dans un fichier texte : cela peut signifie la présence d'un caractère accentué (ici 'à') mal codé. Pour écrire un texte contenant des caractères accentués ou mis en forme, on utilise un véritable traitement de texte comme « Microsoft Word » ou « Open Office » dont le même document sera normalement lu sur PC et Mac avec les mêmes accents, mots en italique, etc.

Bref, la documentation indique qu'il s'agit d'un **projet = ensemble de classes.** Ce projet contient quatre classes : Canvas , Cercle , Carre et Triangle . Les flèches en pointillés indiquent une dépendance de classe : les classes Cercle , Carre et Triangle utilisent la classe Canvas . Remarquez que vous pouvez déplacer les classes à la souris, les flèches en pointillés suivront. Le texte d'une classe est contenu dans un fichier-texte d'extension .java, ici Cercle.java. Ce texte, qui est la description de la classe et de la manière dont ses objets vont se comporter, se nomme le **« texte source » (ou code source)**.

Exercice 3.1

Effectuez un double clic sur la classe Cercle et vérifiez que son texte contient bien le mot Canvas. Pour rechercher un mot dans le texte-source de la classe, utilisez le bouton « Find ... » (l'option « Ignore case » permet d'ignorer la casse c'est-à-dire la distinction majuscule-minuscule). Fermez ensuite la fenêtre Cercle , il est bien trop tôt pour vous de comprendre le contenu de ce texte.

---

1 Les extensions de fichiers que vous rencontrerez dans ce cours seront surtout .txt (texte brut), .java (fichier source écrit en Java), .class (fichier binaire issu de la compilation d'un fichier Java), .html (destiné à être lu par un navigateur Web) et .pdf (format universel de document lisible sur tout ordinateur). Un nom de fichier comprend au généralement un point, l'extension étant à droite du point. Nous déconseillons l'utilisation d'espaces dans un nom de fichier, surtout pour la programmation !!!!!!

Un objet est une nouvelle (new) instance de classe. Retenez bien ce jargon technique !

Exercice 3.1

Créez un cercle, nouvel objet de type Cercle. Pour cela, cliquez sur la classe Cercle avec le bouton droit de la souris et demandez « new Cercle() »[2]. Si vous ne pouvez pas le faire, c'est que vous avez modifié le texte de l'une des classes du projet. Il faut dans ce cas recompiler le projet : le re-traduire en langage machine. Cliquez sur le bouton « Compile » dans le panneau gauche de la fenêtre. Par défaut, le cercle nouvellement créé se nomme cercle1 mais vous pouvez le nommer toto si cela vous fait plaisir. Attention, par convention :

Règle 1 : Le nom d'un objet commence par une lettre minuscule et ne contient que des lettres et des chiffres. S'il est formé de plusieurs mots, ils seront collés comme dans : unExempleDeNom.

N'oubliez pas qu'un objet est un individu alors que la classe représente en quelque sorte l'ensemble potentiel de tous ses individus : la classe Cercle est une description du concept de cercle, et cercle1 n'est qu'un cercle particulier, une instance de cette classe.

Règle 2 : Le nom d'une Classe commence toujours par une lettre Majuscule et ne contient que des lettres et des chiffres. Exemple : la classe Cercle.

Règle 3 : Le langage Java fait une importante distinction entre majuscules et minuscules. On parle de « case sensitive ». Les noms « boNjoUr » et « bonjour » font référence à des entités distinctes !

Règle 4 : Un nom de méthode en Java est toujours suivi d'une parenthèse ouvrante.

Exercice 3.2

a) Essayez de créer un autre cercle ayant le même nom que le premier ! Que se passe-t-il ?

b) Créez deux triangles.

c) En invoquant la méthode montre() , rendez ces trois objets (deux triangles et un cercle) visibles à l'écran. Voyez-vous les trois ? Pourquoi ?

d) Ouvrez le texte de la classe Cercle. Quelle est la signature de la méthode changeTaille(...) ? N'écrivez rien dans le texte de cette classe !

e) Changez la taille (le diamètre) du cercle cercle1 en 50 pixels. L'unité de taille graphique est le pixel : le plus petit point sur l'écran. Pouvez-vous entrer 25+25 au lieu de 50 comme valeur du paramètre nouveauDiametre ?

f) En invoquant les méthodes bougeHorizontal(...) et bougeVertical(...), tâchez d'amener le cercle vers le centre de la fenêtre graphique (le canvas).

g) Pilotez triangle1 de telle sorte que sa base devienne tangente au cercle.

h) Changez la couleur de triangle2 . Ouvrez le texte de la classe Triangle : quelles sont les couleurs valides ? Indication : cherchez le mot « changeCouleur »...

i) Que se passe-t-il si vous donnez une couleur inexistante ? Et si vous oubliez les guillemets ?...

j) Fermez le projet et relancez-le. Créez de nouveaux deux triangles et un cercle avec la souris. Continuez les mêmes actions depuis c) sans la souris, en tapant des instructions dans le Code Pad (fenêtre en bas à droite). Si le Codepad n'est pas visible cocher le sous-menu « show Code Pad » du menu « View ».

2 Rappel : si Cercle est bien le nom de la classe, Cercle() en est le constructeur. Il pourrait avoir des paramètres, ici il n'y en a pas, mais les parenthèses servent à différencier ce constructeur du simple nom de la classe. Ausi, si foo est un nom de méthode, nous parlerons de la méthode foo() pour indiquer qu'elle n'a aucun paramètre, et de la méthode foo(...) pour indiquer qu'elle a peut-être un ou plusieurs paramètres...

Rappel : Un objet possède un état. Cet état est défini à tout instant par les valeurs stockées dans ses champs.

Exercice 3.3

a) Avec un clic droit de la souris sur l'objet cercle1 dans le panneau des objets créés, demandez à inspecter cet objet. Quels sont ses champs et leurs types ?

b) L'inspecteur vous permet-il de modifier la valeur d'un champ, par exemple sa couleur ?

c) En gardant l'inspecteur ouvert, déplacez le cercle vers la droite avec la méthode bougeADroite() . Un champ a-t-il été mis à jour ? Lequel ?

d) Même question que la précédente après avoir rendu le cercle invisible.

e) Quel est le champ modifié par la méthode changeTaille(...) ? Essayez de comprendre intuitivement le texte de la méthode changeTaille() dans la classe Cercle ...

Exercice 3.4 (A FAIRE A LA MAISON!!!!)

Nous souhaitons intégrer au projet courant une nouvelle classe Dessin.java disponible à l'extérieur du projet (voir page web). Dans le menu « Edit » du projet « Formes », demandez « Add Class from File... » et optez pour cette classe Dessin.java . BlueJ en fait une copie dans le répertoire du projet et ne touchera plus à l'original ! Recompilez le projet (bouton « Compile »).

a) Quelles sont les classes utilisées par la classe Dessin ? Déplacez le cas échéant l'icône de la classe Dessin afin de bien voir toutes les flèches de dépendance.

b) Créez une instance de la classe Dessin et invoquez sa méthode dessine() . Testez les méthodes enNoirEtBlanc() et enCouleur().

c) En lisant le texte de la méthode dessine() , expliquez dans quel ordre sont tracés les éléments du dessin.

d) Modifiez cette méthode pour qu'elle dessine un soleil bleu et non jaune ! L'icône de la classe est zébrée, signe qu'elle a été modifiée et doit être recompilée. Après avoir opéré la modification, cliquez sur le bouton « Compile » pour recompiler le projet (la recompilation détruit les objets créés). Vérifiez que le soleil est bien maintenant en bleu !

e) Ajoutez un second soleil à l'image près du premier mais pas au même endroit et de couleur magenta. Indication : en haut de la classe sont définis les 4 champs d'une instance. Ajoutez un 5 ème champ soleil2 de type Cercle. Modifiez ensuite toutes les méthodes en rajoutant les lignes adéquates. En tout vous aurez rajouté 9 lignes à la classe Dessin.

Projet avec BlueJ

Pour créer un nouveau projet avec BlueJ :

1. Choisir « New Project ... » dans le menu Project ;
2. Sélectionner le répertoire adéquat. Nommer votre projet TP3 . Il contiendra toutes les classes programmées au cours de ce TP.
3. Une fenêtre de projet vide apparaît avec une note en haut d'écran. Cette note peut être utilisée pour remplir une courte description du projet ainsi que sur la procédure à utiliser pour lancer le projet. On y trouvera en particulier, le nom de classe principale pour laquelle il faut lancer la méthode main ou le nom des objets à instancier et les méthodes à invoquer. Lorsque vous créez vos projets, il est de votre responsabilité de remplir cette documentation.
4. Pour chaque TP « X », on créera une classe TestTPx contenant au moins autant de méthodes que d'exercices. Chaque méthode contiendra un bout de code qui illustre ce qui a été fait dans l'exercice. La classe TestTP3 est fournie.

Une fois que votre nouveau projet TP3 existe, un répertoire nommé TP3 a été créé dans votre espace de travail. Pour créer une classe dans ce projet (rappel, projet == ensemble de classes) :

1. Choisir « New class ... » dans le menu Edit ;
2. Entrer un nom de classe et valider. Le nom de la classe est le plus souvent indiqué dans l'énoncé. Rappel, un nom de classe ne contient pas d'espace et sa première lettre est une majuscule.
3. Éditer la classe ainsi créée par un double-clic sur son icône. BlueJ vous a préparé du code fictif avec commentaires, qu'il faut modifier. Ici, bous pouvez tout effacer.
4. Sur la feuille vierge ainsi obtenue, vous pouvez alors commencer à rédiger le texte de votre classe.…

Exercice 3.5

On souhaite modéliser un climatiseur par la classe Climatiseur. Un climatiseur est programmé avec une température exprimée en degré Celsius (un champ enCelsius). Cette consigne de température a une valeur initiale donnée en paramètre du constructeur. Il est muni de deux commandes (méthodes) pour augmenter() ou diminuer() la consigne de 1°C à la fois ainsi que de deux méthodes d'accès pour connaître la température programmée. La première méthode d'accès getTemperatureEnCelsius() renvoie un résultat en degré Celsius, la deuxième méthode d'accès getTemperatureEnFarenheit() renvoie un résultat en degré Farenheit. On rappelle les équivalences suivantes : C° = (F° - 32) * 9/5 et F° = C° * 5/9 + 32

a) Proposer un diagramme de classe qui représente les champs, les méthodes et le constructeur de la classe Climatiseur.

b) Proposer une implémentation en Java et commentée de cette classe.

c) Écrire une méthode toString qui imprime la température en °C et °F.

d) Utiliser la méthode ex1() de la classe TestTP3 fournie (« Document de Cours ») pour vérifier que votre programme fonctionne. Importer dans votre projet les fichiers TestTP3.java et TestTP3.class, ne pas modifier ou recompiler cette classe.

Exercice 3.6

On veut maintenant améliorer la classe Climatiseur et faire une nouvelle classe Climatiseur2 à partir de la cette précédente classe afin d'ajouter une valeur minimale (champ min), une valeur maximale (champ max) et une valeur d'incrémentation (champ increment). Les valeurs minimale et maximale sont spécifiées une fois pour toute lors de la construction. La valeur d'incrément peut être modifiée par une méthode de modification – setIncrement() – et retrouvée par une méthode d'accès – getIncrement() –, elle est initialisée à 1 par le constructeur. Seules les valeurs strictement positives d'incrément sont acceptées.

a) Proposer un nouveau diagramme de classe.

b) Réaliser les modifications en Java sur la classe Climatiseur2 .

c) Générer la javadoc (voir cours)pour votre classe et vérifiez que vous avez bien commenté.

Exercice 3.7

Avec les bonnes résolutions d'un début d'année, on veut se préparer aux résultats des examens qui viennent toujours trop vite. Pour cela, vous allez écrire en Java une classe permettant le calcul de la moyenne. On n'est pas encore en mesure de faire une belle classe graphique mais on peut d'ores et déjà faire une classe un tant soit peu utile.

a) De quoi a-t-on besoin pour calculer une moyenne ?

b) Proposer un diagramme pour une classe Moyenne . Cette classe aura au minimum une méthode saisieNote(double) qui permet d'entrer une nouvelle note et une méthode moyenne() : double pour calculer la moyenne de toutes les notes saisies

c) A-t-on besoin d'un constructeur ?

Exercice 3.8

Un ami a vous doit passer le bac à la fin de l'année et il veut de l'aide pour réviser la résolution des équations du deuxième degré. Réalisons un programme Java pour l'aider à ne plus faire toutes ces opérations de tête. Proposer une classe Poly2 qui modélise un polynôme du second degré de la forme ax 2 +bx+c (rappel : en maths, la lettre « x » représente une « indéterminée »). On souhaite disposer des 3 méthodes suivantes :

```
double eval(double x) // retourne la valeur au point x du polynôme this
double delta() // retourne le discriminant de ce polynôme
double uneRacine() // retourne UNE racine réelle de ce polynôme ou NaN
String toString() // retourne une représentation sous la forme "ax 2 +bx+c"
```

Pour calculer une racine carrée en Java il faut utiliser la méthode Maths.sqrt. Attention, malgré l'utilisation de la notation pointée, il n'y a pas d'objets mis en œuvre dans Math.sqrt(12). La méthode sqrt parle directement à la classe Math, elle est **statique.**

D’autres exercices

Les exercices sont à faire soit avec BlueJ soit avec JEdit.

Exercice 3.4

a) Comment se nomment les valeurs de vérité vrai et faux en Java ?

b) Ces deux valeurs sont-elles des objets ?

c) Quel est le nom Java de leur type de donnée ? Est-ce une classe ?

d) Le type int est-il une classe ?

e) Un entier est-il un objet ?

f) Une chaîne de caractères est-elle un objet ? Quel est le nom de son type ? Son type est-il une classe ?

Exercice 3.5

Dans le projet « Formes» vous manipulez des figures géométriques (cercles, triangles, carrés). Sachant qu’un cercle est de diamètre 30, de couleur bleue, et non visible par défaut, quelles seraient les instructions pour obtenir le dessin de deux cercles tangents, l’un rouge et l’autre bleu ?

Exercice 3.6

Une classe CompteBancaire décrit des objets (des comptes bancaires !). Un objet de cette classe comporte des champs et sait répondre à des méthodes (messages). Imaginez comment vous pourriez écrire en Java (quitte à programmer ce qu’il faut, ce qu’on ne fera pas) :

- « Soit c1 un nouveau compte bancaire »
- « Soit c2 un nouveau compte bancaire de ”Mr Gaston” »
- « Le solde du compte c2 » [deux solutions, accès à un champ, ou bien par une méthode]
- « Retirer 20€ du compte c1 »

Explicitez les déclarations des champs et les signatures des méthodes que vous utilisez ! Rédigez un début de classe …

Exercice 3.7

Déclarez deux variables entières a et b, et initialisez-les respectivement à 2 et 5. Montrez comment, en quelques lignes de Java, on peut échanger leurs valeurs (sans connaître précisément ces valeurs).

Exercice 3.8

On souhaite réaliser un jeu qui consiste à deviner le plus rapidement possible un nombre entré par quelqu’un d’autre.

a) Proposer une classe `Devinette` dont les objets permettent de mémoriser le nombre à découvrir. La classe Devinette est pourvue d’une méthode « `String guess(int essai)` » qui renvoie la chaîne « trop petit » si le nombre fourni en paramètre est inférieur au nombre à deviner, la chaîne « trop grand » s’il est supérieur et la chaîne « bravo ! » lorsqu’on a trouvé. Le nombre à deviner est initialisé par un paramètre du constructeur. Proposer une implémentation en Java.

b) Modifier la classe Devinette pour qu’elle compte le nombre d’essais infructueux. On ajoutera une méthode d’accès pour le nouveau champ.

c) Modifier la classe Devinette pour qu’elle mémorise si le nombre à deviner a déjà été trouvé. Dans ce cas, on rejettera les nouveaux essais par un message approprié.

d) Modifier une dernière fois la classe Devinette pour qu’elle choisisse le nombre à deviner au hasard. Le constructeur sera alors muni de deux paramètres `min` et `max` qui indiquent la fourchette à l’intérieur de laquelle le nombre à deviner doit être choisi. Pour sélectionner un nombre entier au hasard, on utilisera la classe `java.util.Random` et sa méthode `nextInt()`.