

## Projet

### Dessin vectoriel

Le but de ce projet est de développer un logiciel de dessin vectoriel comme Geogebra, Inkscape, Karbon, Illustrator... dont le principe est de décomposer les dessins en figures géométriques simples dont on ne stocke que les paramètres. Cela permet d'avoir des images dont la précision est virtuellement infinie (on peut zoomer autant que l'on veut sans avoir de défauts) et qui prennent très peu de place en mémoire.

### Modalités

Le projet est à faire au plus en binôme. Les soutenances se feront à deux, mais la note pourra être individualisée si le travail a été trop inégalement réparti. Naturellement chacun doit être capable de répondre à toutes les questions. Si vous faites le projet seul, vous serez noté sur le même barème que si vous étiez en binôme.

Ce projet est intrinsèquement plus difficile que l'autre qui vous est proposé, et demande de travailler un peu plus l'interface graphique. Pour que vous puissiez fournir un travail adapté à votre aisance en Java, nous vous proposons plusieurs "niveaux" : un projet minimal, qui équivaut en difficulté à l'intégrité de l'autre projet, et de possibles extensions, plus ou moins difficiles. De plus ce projet impose de devoir utiliser quelques formules mathématiques simples (que l'on peut connaître ou retrouver sur internet). Ces difficultés supplémentaires seront bien évidemment prises en compte dans la note.

Notez bien qu'on s'attend à ce qu'il soit fait usage de concepts vus en cours (héritage, interfaces...) : la conception objet représente une part importante de la note.

En guise de rapport, vous fournirez une liste des fonctionnalités que vous avez effectivement implémentées, et si elles fonctionnent ou non, ainsi qu'un diagramme des classes utilisées. Un rapport détaillé dans lequel on expliquerait *comment* les fonctionnalités sont implémentées *n'est pas* demandé.

Vous devez fournir une javadoc et votre code devra être soigneusement commenté et indenté.

La date pour rendre le projet sera fixée ultérieurement.

Il nous faudra pouvoir tester le projet durant les soutenances, il est donc préférable qu'il fonctionne sur les machines du script si vous êtes en L2 ou sur celles de l'ufr si vous êtes en L3. Ceci même si vous avez prévu d'apporter un portable pour l'occasion car il peut tomber en panne au mauvais moment.

## Conseils pour réaliser le projet

1. Lisez bien l'intégralité du sujet. Prenez le temps de réfléchir à ce que vous allez implémenter et à la conception du projet avant de coder.
2. Réfléchissez aussi à la manière de répartir le travail entre les deux personnes du binôme. Faites des points réguliers entre vous.
3. Enfin, pensez à faire des sauvegardes fréquentes, sur au moins deux supports différents et en particulier sauvez les versions testables de votre projet même s'il reste des corrections à y apporter.

## 1 Interface graphique : consignes générales

L'exécution du projet devra ouvrir une interface dans laquelle on pourra dessiner en utilisant les outils implémentés.

Chacun des types de figure implémentés devra être créé à partir de votre interface (via des boutons, raccourcis clavier ou autre) avec une couleur. Néanmoins, une composante implémentée mais pas ajoutée à l'interface graphique sera prise en compte dans les points de conception.

On doit pouvoir sélectionner les figures séparément, et les déplacer lorsque c'est possible.

Lorsqu'une figure est modifiée, toutes ses dépendances doivent être modifiées dynamiquement.

Il doit être possible de zoomer ou dézoomer sur l'origine autant que l'on veut.

Les efforts sur la fluidité de l'animation seront pris en compte. Évitez par exemple (et autant que possible) de calculer des positions inutiles (hors de l'écran ou trop précises), surtout pour les options avancées. Évitez aussi de recalculer des positions s'il est évident qu'elles n'ont pas été changées.

Il n'est pas demandé dans le projet minimal d'offrir une option pour sauvegarder le dessin, car cela va au-delà du cadre du cours (il s'agit d'apprendre à utiliser l'API adéquate). C'est néanmoins une extension possible (niveau de difficulté : \*\*\*), et un petit bonus sera accordé aux groupes l'implémentant.

## 2 Projet minimal

On doit pouvoir dessiner à main levée des points, des droites et des segments qui seront stockés avec des coordonnées absolues.

Ces figures, stockées en coordonnées absolues, seront contrôlables, c'est à dire qu'on doit pouvoir les attraper et les bouger depuis l'interface.

Il faut pouvoir construire un point comme intersection de deux droites.<sup>1</sup> Atten-

---

1. L'intersection de deux droites  $a_1x_1 + b_1y_1 = 1$  et  $a_2x_2 + b_2y_2 = 1$  est

$$\left( \frac{b_2 - b_1}{a_1b_2 - a_2b_1}, \frac{a_2 - a_1}{b_1a_2 - b_2a_1} \right).$$

tion, ce point ne doit pas être représenté en interne avec des coordonnées absolues, mais restera la donnée d'une intersection : en particulier, si on déplace l'une des droites avec lesquelles il est construit, lui aussi doit bouger. Remarquez que cela veut dire qu'il ne pourra pas être bougé directement.

Une option doit être disponible pour sélectionner un certain nombre de points et créer un nouveau point isobarycentre (centre de gravité).<sup>2</sup> Comme pour l'intersection de deux droites, le barycentre doit rester lié aux objets (ici des points) à partir desquels il a été créé.

Enfin, un segment (resp. une droite) doit pouvoir être créé(e) entre deux points. Encore une fois celui-ci (resp. celle-ci) doit rester lié(e) aux points à partir desquels on l'a défini(e).

Le programme ne doit pas crasher et une situation impossible (par exemple point défini à l'intersection de droites parallèles) devra annuler l'opération en cours.

### 3 Extensions suggérées

Libre à vous de choisir dans cette liste les extensions au projet minimal que vous souhaitez implémenter.

#### 3.1 Au-delà des points, des droites, et des segments

- **Triangles (\*)**. On peut implémenter la possibilité de faire des triangles (remplis). Ceux-ci pourront être construits avec des coordonnées absolues (et donc être contrôlables), mais aussi à partir de trois points ou de trois droites.
- **Cercles (\*)**. De même on peut vouloir construire des cercles, définis en absolu, mais aussi avec le centre et un point sur le bord, ou encore avec 3 points sur le bord.
- **Couleurs (\*\*)**. On peut implémenter un moyen de changer la couleur des objets. Pour les cercles et les triangles, on pourra différencier bordure et intérieur, l'intérieur pouvant être transparent, d'une couleur pleine, ou avoir un dégradé axial (autour d'un point) ou central (autour d'une droite). Vous pourrez aussi implémenter un moyen de choisir les superpositions (droite qui passe devant ou derrière un triangle...).
- **Groupes (\*\*\*)**. Il pourra être possible de grouper des objets contrôlables pour tous les déplacer simultanément, tourner ou encore rétrécir/agrandir cette partie du dessin.

---

2. Les coordonnées  $(x, y)$  de l'isobarycentre de  $n$  points  $(x_1, y_1) \dots (x_n, y_n)$  sont

$$\left( \frac{x_1 + \dots + x_n}{n}, \frac{y_1 + \dots + y_n}{n} \right).$$

En particulier l'isobarycentre de deux points est leur milieu.

- **Suppression (\*\*\*)**. Il pourra aussi être possible de supprimer une figure tout en gérant les dépendances.

### 3.2 Objets semi-contrôlables (\*\*\*\*)

Jusqu'ici on n'a défini que des figures ou bien liées (définies à partir d'autres figures) ou bien contrôlables (définies de façon absolue). Il peut être intéressant d'avoir des figures semi-contrôlables (par exemple un point qui glisse sur une droite), avec un paramètre lié que l'on ne gère pas directement (être sur la droite) et un autre absolu qui est modifiable (la position sur la droite).

La difficulté arrive lorsque le paramètre lié change. En effet, dans le cas du point sur la droite on se rend compte que la droite ayant bougé on ne sait plus où est notre point. Le point doit rester sur la droite, libre à vous d'interpréter comment sa position (le paramètre absolu) est redéfinie en fonction des paramètres de la droite (et ce pour chaque type de droite...).

Vous pourrez définir ainsi :

- un point sur une droite,
- une parallèle à une droite,
- une perpendiculaire à une droite,
- un point sur un cercle,
- une tangente à un cercle,
- une intersection entre deux cercles/triangles.

### 3.3 Interface

- **Amélioration du zoom/dézoom (\*)**. Dans le projet minimal, le (dé)zoom était centré sur l'origine. Il est préférable, pour l'utilisateur du logiciel de dessin vectoriel, que le (dé)zoom soit centré sur la souris. Vous pouvez implémenter cela.
- **Sauvegarde (\*\*\*)**. Comme dit précédemment, on peut implémenter une façon de sauvegarder le dessin en cours (il faut alors utiliser l'API adéquate).

## 4 Pour aller plus loin

Certains TD/TP distribués au long du semestre (ordinaires ou optionnels) proposeront probablement des implémentations d'objets plus avancés pouvant s'intégrer à votre projet (fractales, courbes de Bézier, intersection de courbes...). Cela dépendra évidemment du nombre d'élèves choisissant ce projet.