# « Phlacheux »
# Projet du cours de
# Programmation d'Interfaces Graphiques

Master Informatique

Université Paris Diderot

V. Padovani, J.-B. Yunès, W. Zielonka

2014

## Introduction

Le but de ce projet est de programmer un logiciel de création de film d'animation.

L'idée de base est d'animer un objet géométrique bi-dimensionnel à l'écran en contrôlant sa trajectoire et des transformations géométriques diverses et variées.

Un film d'animation sera alors obtenu par agglomérat d'animations de base et une visionneuse de film sera disponible pour regarder le résultat.

## Description détaillée

### Objets et animation

Cette partie est indépendante de toute visualisation, c'est-à-dire peut-être conçue indépendamment du fait qu'au final les objets auront une représentation à l'écran...

#### Objets

Le logiciel devra permettre de manipuler divers objets géométriques bi-dimensionnels. La liste n'est pas fixée exhaustivement, sa complétion est laissée à l'appréciation de chacun. Il est obligatoire de permettre d'animer les objets suivants : segments de droite, triangles équilatéraux, carrés et cercles. L'application finale devra permettre d'animer **au minimum** 8 types d'objets géométriques différents (cherchez lesquels).

Il pourra être envisagé de construire des objets par agglomération d'autres objets...

Un point important est de noter que les objets et animations sont décrits dans un monde idéal « continu » (c'est-à-dire en coordonnées réelles).

### Animations

Une animation est l'application temporalisée d'une transformation géométrique : à partir d'un objet dans l'espace on cherche à arriver à sa version transformée **en un certain temps**. Toute transformation est *a priori* bonne à considérer. Comme pour les objets, la liste des transformations n'est pas fixée exhaustivement, chacun complètera comme bon lui semble. Il est obligatoire d'implémenter les transformations suivantes : translation, rotation autour du centre de gravité de l'objet, modification de l'épaisseur du trait du contour de l'objet, modification de la couleur du trait de l'objet. L'application devra permettre de sélectionner au moins 5 types d'animations différentes (l'idéal serait que les possibilités soient dépendantes des caractéristiques des objets eux-mêmes). Bien entendu, une animation pourra être la composition d'animations plus simples, soit qui se recouvrent l'une l'autre intégralement dans le temps, soit qui se succèdent strictement dans le temps. On peut envisager des animations qui se recouvrent mais il faut définir une notion de compatibilité temporelle (vous êtes autorisés à y réfléchir).

Une animation aura un point de départ dans le temps, une durée et une fonction d'accélération afin de contrôler finement la temporalité de la transformation. La fonction de base doit être linéaire, mais on pourra envisager choisir des fonctions du type *easing* (rechercher *easing function* sur Internet pour obtenir des exemples).

Un exemple d'animation peut donc être : déplacer de $(x_0, y_0)$ l'objet $O$ à la position $(x_1, y_1)$ en commençant au temps $t_d$ et en terminant au temps $t_f$...

## Éditeur d'animation

La partie édition devra permettre d'éditer graphiquement, soit par des « gestes », soit par des saisies manuelles les caractéristiques des animations et des objets. Pour aider l'utilisateur à s'y retrouver dans les nombreux objets/animations qu'il manipulera, chacun d'eux possèdera un nom (soit automatiquement déterminé à sa création, soit modifié ultérieurement). Ainsi l'utilisateur pourra sélectionner un objet soit directement à l'écran, soit dans une liste de tous les objets présents (des listes d'objets rangés par catégorie peuvent être envisagées). Il est conseillé de trouver pour chaque édition l'interaction ergonomique la plus appropriée : par exemple les paramètres type position de départ et position d'arrivée devraient être sélectionnées à la souris. Certains autre paramètres (nom par exemple) ne peuvent l'être que par l'intermédiaire du clavier. Une édition fine des paramètres utilisant le clavier doit toujours être possible.

Cette partie n'est pas nécessairement découplée de la partie visionneuse décrite à la suite. En effet, il doit être possible de visionner durant l'édition : soit la seule animation qui est actuellement en cours de modification, soit l'ensemble de toutes les animations créées, soit toutes celles associées à un objet donné, etc. À vous de décider ce qui est adéquat et nécessaire.

## Visionneuse

La partie visionneuse devra permettre de représenter à l'écran un film (collection d'animations d'objets). La visionneuse devra posséder les fonctions de base attendues pour toute visionneuse, c'est-à-dire, (re)-démarrer, arrêter/mettre en pause. On pourra aussi y trouver une avance rapide ainsi qu'un visionnage arrière et un recul rapide.

Avant tout visionnage, le projectionniste devra choisir différents paramètres pour celui-ci, comme le taux d'images générées par seconde (*framerate*), la taille de l'écran de projection, la définition de la partie « utile » de l'espace continu des objets qui aura une représentation à l'écran, ainsi que tout autre paramètre qui sera considéré comme utile.

Il pourra être envisagé de fournir en sortie de la visionneuse, l'ensemble des images du film, et de fournir avec un petit programme java permettant de visionner ce film de façon autonome.

### Fonctionnalités autres

Il est impératif de fournir un moyen de lire/sauvegarder l'édition d'une animation. Le logiciel devra donc le permettre. Le format de stockage est à définir, mais il vaudrait mieux rester dans le raisonnable (un format texte éditable manuellement, XML ou similaire, par exemple). On pourra aussi envisager pouvoir sauvegarder des objets compliqués de façon autonomes afin de pouvoir les intégrer dans plusieurs projets par exemple, un format est donc aussi à définir. Même remarque pour les animations composées qui pourraient avantageusement être sauvegardées.

## Réalisation

La réalisation se fera nécessairement en Java en utilisant la couche Swing (mais pas seulement). Un groupe **d'au moins deux étudiants et d'au plus trois** (ni plus ni moins) devra réaliser une application conforme à la description ci-dessus. Cette dernière est relativement floue permettant à chaque groupe d'en faire une interprétation qui lui est propre ; attention toutefois à respecter l'esprit et **surtout** à obtenir une application utilisable avec une ergonomie raisonnable. Il faut donc éviter la livraison d'un prototype trop primitif !

Il est impératif que chaque groupe utilise le système de dépôt `git` offrant des capacités de versioning que l'UFR possède : `moule.informatique.univ-paris-diderot.fr:8080`. Vous pouvez vous y connecter en utilisant votre compte de l'UFR (ne pas utiliser de compte « extérieur »). Apprenez à utiliser `git`, ce point sera particulièrement apprécié ! Pensez aussi à inviter les enseignants (obligatoirement le responsable du cours !) dans le projet `git` que vous aurez créé et leur donner suffisamment les droits *master*.

Attention si pour obtenir ce qui est demandé ci-dessus, la communication verbale entre groupes est non seulement autorisée mais encouragée, il est **strictement interdit** d'échanger du code ; ceci serait considéré comme plagiat et par conséquent jugé sévèrement (ne comptez surtout pas sur une quelconque incompétence des examinateurs à détecter du plagiat). Les discussions doivent donc seulement porter sur l'interprétation du sujet ou des considérations générales sur la façon dont quelque chose peut fonctionner ; il vaut donc mieux éviter de donner trop d'indications sur la façon de coder les fonctionnalités (le cours, internet, etc vous donneront toutes les indications nécessaires, **à condition d'éviter comme la peste** les sites comme `codeparcopiercoller.fr`, `programmersansrienfaire.com`, `laprogrammationproen5minut.es`).

La **notation finale** prendra en compte le fait que des groupes auront réussi à faire obtenir une application réellement utilisable. Il faudra donc penser à en faire la démonstration. D'une certaine manière, ceux qui collaborent dans les limites indiquées ci-dessus seront récompensés ; pour ceux qui décident de faire quelque chose dans leur coin et qui ressemblerait vaguement à ce qui est décrit ce sera l'inverse, la notation sera revue à la baisse. Un **programme qui s'exécute n'est pas suffisant** ! Vous **devez** écrire un programme qui réalise des choses raisonnables.

Nous mettrons en place un forum sur didel pour que vous puissiez communiquer entre vous et avec nous, par exemple pour faire part d'imprécisions dans le sujet, etc.. **Dans tous les cas**, vous pouvez également contacter vos enseignants par mail ou lorsque vous les croisez à l'occasion.

Prenez garde à ne pas vous fier à la rumeur, les seules sources d'information fiables sont vos enseignants ! Au moindre doute, demandez-leur !