Université de Caen Basse-Normandie
U.F.R. de Sciences
Département d'informatique

Bâtiment Sciences 3 - Campus Côte de Nacre
F-14032 Caen Cédex, FRANCE

Devoir 2016-2017

Niveau	**M1**
Parcours	**Informatique**
Unité d'enseignement	**UE3 - Programmation et Parallélisme**
Responsable	**Emmanuel Cagniot** **Emmanuel.Cagniot@ensicaen.fr**

1 Sujet

Nous désirons écrire un jeu doté d'une interface graphique dans lequel une grenouille doit partir d'une position pour en rallier une autre en sautant de nénuphar en nénuphar à l'intérieur d'une mare.

La mare est une surface carrée composée de $N \times N$ éléments de surface. Ces derniers sont respectivement l'eau, le nénuphar, le nénuphar nutritif, le nénuphar vénéneux, le nénuphar dopant, le nénuphar mortel et enfin, le nénuphar immortel. À l'exception de l'eau et du nénuphar immortel, chaque élément de surface est soumis à un processus de vieillissement. À son arrivée sur la mare (sa naissance), le nénuphar appartient à la catégorie des grands nénuphars. Au cours du temps, il passe progressivement par la catégorie moyen nénuphar puis petit nénuphar, avant de redevenir de l'eau (sa mort).

Les effets de chaque type d'élément de surface sur la grenouille sont les suivants :

- **eau** : une grenouille qui tombe à l'eau meurt dévorée par l'un des nombreux brochets qui infestent la mare ;
- **nénuphar** : de couleur verte, ce nénuphar n'a aucun effet sur la grenouille ;
- **nénuphar immortel** : de couleur verte, ce type de nénuphar est exclusivement implanté à la position de départ et d'arrivée de la grenouille. Il n'exerce aucun effet sur cette dernière et lui sert simplement de refuge ;
- **nénuphar vénéneux** : de couleur jaune, il rend la grenouille malade et divise par deux (division entière) le nombre de ses points de vie. Si la grenouille était déjà malade alors elle meurt de surinfection ;
- **nénuphar nutritif** : de couleur rose, il augmente le nombre de points de vie de la grenouille d'une unité et la guérit si elle était malade ;
- **nénuphar dopant** : de couleur rouge, il double les points de vie de la grenouille et la guérit si elle était malade ;
- **nénuphar mortel** : de même couleur que le nénuphar dopant (on ne peut pas les différencier), il provoque la mort instantanée de la grenouille.

Une partie se joue en temps limité. Avant chaque début de partie, la grenouille est posée sur le nénuphar immortel occupant la position de départ située dans le coin inférieur gauche de la mare. Un autre nénuphar immortel est implanté à la position d'arrivée située dans le coin opposé. La grenouille dispose alors d'un unique point de vie. Lorsque le joueur décide de lancer la partie, un chronomètre est activé pour égrainer le temps qui passe. La grenouille doit alors s'efforcer d'atteindre la position d'arrivée avant l'expiration

du delai tout en maximisant le nombre de ses points de vie. La partie dure 1 minute tandis que la période du chronomètre est réglée à 1 seconde.

À chaque top du chronomètre, tous les éléments de surface présents sur la mare (à l'exception de l'eau et des deux nénuphars immortels) vieillissent, risquant ainsi de provoquer la mort de la grenouille. Une fois la phase de vieillissement terminée, des nouveaux chemins pavés de nénuphars de tout type (autres qu'immortels) s'ouvrent entre la position actuelle de la grenouille et la position d'arrivée. Plus précisément :

— si la position de la grenouille et la position d'arrivée partagent la même ligne alors une ligne de nénuphars est ouverte entre ces deux positions ;
— si la position de la grenouille et la position d'arrivée partagent la même colonne alors une colonne de nénuphars est ouverte entre ces deux positions ;
— enfin, si la position de la grenouille et la position d'arrivée ne partagent aucune ligne ni colonne, alors deux chemins sont ouverts de telle manière que la position d'arrivée et celle de la grenouille définissent un rectangle.

Le type des nénuphars pavant ce ou ces chemins est choisi aléatoirement. Si un chemin traverse une position déjà occupée par un nénuphar alors ce dernier est maintenu dans son état actuel.

À chaque instant, le joueur peut choisir de déplacer sa grenouille sur l'une des quatre positions voisines (si elle existe) situées respectivement au nord, à l'est, au sud et à l'ouest de sa position actuelle. Le joueur remporte la partie s'il parvient à atteindre la position d'arrivée avant l'expiration du délai. Il la perd si le délai est expiré ou si sa grenouille meurt en chemin pour l'une des raisons suivantes :

— elle tombe à l'eau ou le nénuphar sur lequel elle se trouvait redevient de l'eau au terme de son processus de vieillissement ;
— son nombre de points de vie devient nul ;
— surinfection ;
— elle se déplace sur un nénuphar mortel.

La taille des nénuphars (à l'exception des immortels) diminue avec l'évolution du veillissement. La grenouille change de couleur lorsqu'elle tombe malade et revient à sa couleur d'origine lorsqu'elle guérit.

1.1 Le modèle

La première partie du travail consiste à écrire le modèle de l'application. Dans ce modèle, l'effet des différents types d'éléments de surface sur la grenouille est modélisé via le *design pattern* `Strategy` ;

Le paquetage maître de l'application est baptisé `grenouilloland` et les éléments du modèle sont rattachés au sous-paquetage `modele`. Un soin particulier sera apporté aux commentaires des fichiers sources écrits dans le format Doxygen.

1.2 La vue

La seconde partie du travail consiste à développer l'interface graphique de l'application. Vous pouvez vous inspirer de celle présentée en figure 1, voire la copier. Les icones de cette figure vous sont fournies avec l'énonce de ce devoir sur la page web dédiée à l'unité d'enseignement.

Cette interface comporte trois parties :

— les barres de menus et d'outils : à chaque entrée du menu correspond un outil. Ces barres proposent quatre commandes : présenter l'application (version, auteurs, etc.), quitter l'application, lancer la partie et préparer une nouvelle partie ;
— le panneau de contrôle : il est composé d'une zone d'affichage des points de vie de la grenouille, d'une barre de progression pour le chronomètre et d'une barre de défilement permettant de modifier la résolution de la mare. Une modification de ce dernière entraîne l'interruption de la partie en cours et la préparation d'une nouvelle partie avec la résolution demandée ;
— la zone de dessin : elle représente les éléments de surface composant la mare ainsi que la grenouille. Chaque élément de surface est représenté par une `EventBox`. Par conséquent, le joueur déplace sa grenouille en cliquant sur la case adjacente.

L'ensemble de l'application est basée sur une architecture Model-View-Presenter. Les éléments de la vue sont affectés au sous-paquetage `grenouilloland.vue` tandis que le présentateur est affecté au

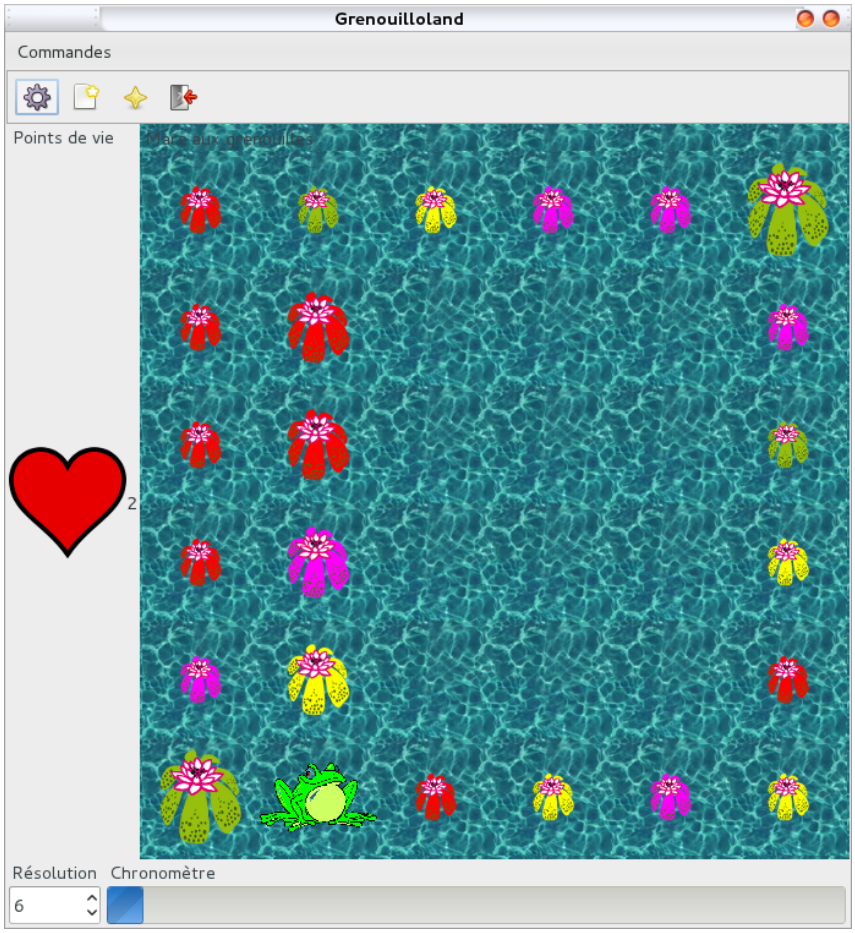

FIGURE 1 – Interface graphique de l'application.

sous-paquetage `grenouilloland.presentateur`. Un soin particulier sera apporté aux commentaires des fichiers sources écrits dans le format DOXYGEN.

Travail à rendre

L'objectif de ce devoir, à réaliser par groupes d'au maximum quatre étudiants (deux binômes), est la programmation de l'application décrite ci-dessus. Ce travail, qui représente le contrôle continu, compte pour un tiers de la note finale. Il consiste à créer une archive `devoir.tar.gz` constituée de :

- un répertoire `src` contenant les sources. Chaque paquetage et sous-paquetage demandé fait l'objet d'un sous-répértoire dans lequel les définitions (`.cpp`) sont séparées des déclarations (`.hpp`). Ces sources devront être totalement commentées au format DOXYGEN (paquetages, classes, attributs, méthodes (y compris leur code respectif)) ;
- un script de compilation `cmake` ;
- un fichier `doxyfile` permettant de générer automatiquement la documentation au format `html` ;
- un exemple de base de connaissances (attention, testez votre application sur plusieurs exemples et non pas sur un seul) ;
- un fichier texte `noms.txt` contenant les noms et groupes de TD et TP des auteurs du devoir ;
- un répertoire UML contenant le diagramme de classes de toute l'application au format `dia`. Vous pourrez partitionner en deux diagrammes : builder et grammaire d'un coté, modèle de l'autre.

Ce travail sera remis au plus tard le jour du dernier cours magistral (minuit dernier délai). Il sera déposé sur le système accessible à partir de `https://devoirs.info.unicaen.fr/`. Seul l'un des quatre étudiants déposera une archive `devoir.tar.gz` contenant tous les fichiers du devoir. Les autres déposeront une archive contenant simplement le fichier `noms.txt`. Ce système (qui vous paraît certainement étrange) permet aux enseignants de :

- ne pas saturer inutilement l'espace disque du serveur sous-jacent ;
- envoyer aux étudiants la correction détaillée par mail via le serveur ;
- disposer de la liste de tous les étudiants ayant rendu le devoir avec, pour chacun, la note correspondante. Cette liste est utilisée au moment de la collecte des notes de chaque unité de valeur en vue de la préparation des jurys.

La correction sera effectuée sur les machines de la salle S3 403-404. Tout manquement aux consignes données ci-dessus (absence ou quasi absence de commentaires, non respect de la conception objet demandée, absence de script de compilation ou script dépendant d'un environnement de programmation, format autre que dia pour les diagrammes de classes, etc.) se traduira par une réduction de la note attribuée. L'absence d'un fichier `nom.txt` se traduira par le retrait d'un point sur la note de l'étudiant fautif.