Projet Java III - 2022-2023

HELBAquarium

Seconde session

Introduction :

Dans le cadre du cours de Java III, il vous est demander d'implémenter un jeu de simulation d'aquarium.

L'aquariophilie est le loisir qui consiste à s'occuper d'animaux et de plantes aquatiques dans un aquarium, une mare ou un bassin d'ornement en mettant en valeur l'aspect esthétique d'un milieu aquatique. L'aquarium est un bac vitré, rempli d'eau, de petite dimension. Il est une reproduction de l'habitat naturel des espèces qui habitent dans cet aquarium. Il peut aussi contenir des décorations diverses à l'apparence plus ou moins naturelles.

Description des éléments du jeu :

L'espace de jeu :

L'aquarium est un espace en deux dimension, rectangulaire, dont les délimitations sont des obstacles pour les poissons qu'il contient.

Les poissons :

L'aquarium contient un certain nombre des poissons. Chaque poisson peut se déplacer dans l'aquarium, soit verticalement, soit horizontalement, soit en diagonal.

Les décorations :

L'aquarium contient quatre décorations. Ces décorations sont des obstacles rectangulaires pour les poissons. Les poissons ne peuvent pas les traverser. Les décorations sont largement plus grandes que les poissons mais chacune est largement plus petite que la taille de l'aquarium.
Les décorations se déplacent de haut en bas dans l'aquarium.

Les espèces de poissons :

Parmi les poissons, il existe différentes espèces avec des comportements qui leur sont propres.

Les poissons orange sont des poissons hasardeux qui se déplacent constamment, dans une direction aléatoire dans l'espace de l'aquarium, jusqu'à rencontrer un obstacle ou un bord de l'aquarium. Dès qu'un obstacle ou un bord est rencontré, ils prennent une nouvelle direction aléatoire.

Les poissons mauves sont des poissons prudents, ils se déplacent toujours dans la direction opposée au poisson rouge le plus proche. Les poissons mauves sont également stimulés par les poissons orange de l'aquarium, ce qui a un effet sur leur vitesse de déplacement.

Les poissons bleus sont des poissons sociaux, ils se déplacent toujours dans la direction du poisson bleu ou mauve le plus proche.

Les poissons rouges sont des poissons prédateurs, ils se déplacent toujours dans la direction du poisson, non rouge, le plus proche. Quand un poisson rouge rencontre un poisson d'une autre couleur, celui-ci le mange, ce qui a pour effet de le faire disparaitre de l'aquarium. Les poissons rouges sont stimulés par la température de l'aquarium.

Déplacements et vitesses :

Sauf exception, les poissons n'arrêtent jamais leur déplacement dans l'aquarium.

Les poissons se déplacent avec une certaine vitesse de base.

Les poissons ont tous la même vitesse de base.

Même si la vitesse de base des poissons mauve est la même que celle des autres poissons, celle-ci augmente avec le nombre de poissons orange présents dans l'aquarium.

Insectes et Pastilles comestibles :

Au lancement de la simulation, l'aquarium peut contenir un certain nombre de pastilles comestibles et d'insectes. Ces éléments sont fixes et placés de façon aléatoire dans l'aquarium.

Quand un poisson rencontre un insecte, il le mange, ce qui a pour effet d'augmenter temporairement la vitesse du poisson. Il existe trois types d'insectes.

La seule différence entre ces trois types d'insectes est la durée du bonus de vitesse qu'ils procurent.

L'aquarium peut aussi contenir un certain nombre de pastilles comestibles. Quand un poisson mange une pastille comestible, il ralentit sa vitesse (en la divisant par 3) pendant x secondes où x est le nombre de décorations présentes dans l'aquarium.

Température de l'aquarium :

L'aquarium possède une température de base, qui par défaut est tiède. Il est possible de changer la température de l'aquarium sur trois configurations différentes : Froid, tiède et chaud.

Dans la configuration froide, les poissons rouges ont un malus de vitesse par rapport à leur vitesse de base, ce qui les rends plus lents que tous les autres types de poissons.

Dans la configuration chaude, les poissons rouges ont un bonus de vitesse, ce qui les rends plus rapides que tous les autres types de poissons.

Le changement de température est visible grâce à un changement de la couleur de background de l'aquarium.

Vitesse maximale :

Il n'y a pas de vitesse maximale déterminée pour les poissons.

Reproduction :

Enfin, il existe un système de reproduction pour les poissons. Quand deux poissons de la même couleur entrent en contact, ils peuvent s'accoupler.

L'accouplement se déroule de la façon suivante :

Les deux poissons qui sont entrés en contact disparaissent, et trois nouveaux poissons de la même couleur que les deux précédents sont placés, chacun, à une nouvelle position aléatoire dans l'aquarium.

Il vous est demandé de considérer que "plus il y a de poissons dans l'aquarium, plus la reproduction est difficile". Concrètement, lorsque deux poissons se rencontrent, ils ne vont plus systématiquement générer un troisième poisson. Les chances qu'ils génèrent un nouveau poisson dépendent du nombre de poissons déjà présents dans l'aquarium.

Commandes de l’aquarium :

Afin de permettre des interactions entre l'utilisateur et l'aquarium, il existe un système de commandes permettant d'influencer la simulation. Quand l'utilisateur appuie sur une touche, cela déclenche un évènement précis :

0 : Reset l'aquarium.
1 : Définir la température de l'aquarium sur froid.
2 : Définir la température de l'aquarium sur tiède.
3 : Définir la température de l'aquarium sur chaud.
4 : Ajouter à une position aléatoire un insecte.
5 : jouter à une position aléatoire une pastille comestible.
6 : Passer en mode insectivore. Dans ce mode, tous les poissons qui ne sont pas stoppés, se dirigent vers l'insecte le plus proche, s’il y en a un dans l'aquarium.
7 : Passer en mode pastille. Dans ce mode, tous les poissons qui ne sont pas stoppés, se dirigent vers la pastille la plus proche, s’il y en a une dans l'aquarium.
8 : Passer en mode reproduction. Dans ce mode, tous les poissons qui ne sont pas stoppés, cherchent à rencontrer, le poisson de la même couleur, le plus proche.
9 : Rajoute un poisson, de type aléatoire, à une position aléatoire dans l'aquarium.
r : Cela permet de stopper tous les poissons sauf les rouges.
b : Cela permet de stopper tous les poissons sauf les bleu.
m : Cela permet de stopper tous les poissons sauf les mauves.
o : Cela permet de stopper tous les poissons sauf les oranges.

Fonctionnalité supplémentaire :

Additionnement, il vous est demandé d’implémenter au minimum une fonctionnalité supplémentaire originale – par exemple :

- Nouveau type de comestible.
- Nouveau type de poisson avec comportement propre.
- Nouvel environnement.
- Etc…

Notes Importantes :

Les contraintes et fonctionnalités du projet sont susceptibles d’évoluer au cours du temps. Pensez donc à adapter une stratégie de développement adéquate.

Certains points de la description ne sont pas précisés ou sont laissés volontairement vagues. Il revient à vous de faire certains choix d’interprétations. Veillez toutefois à ce que votre approche soit logique et justifiée.

Contrainte de développement :

- Votre programme devra être compilable et exécutable dans un environnement Linux Ubuntu tel qu'une des machines virtuelles utilisées en cours et disponible sur le SharePoint d'eCampus. Un script bash nommé « run.sh » devra permettre la compilation et l'exécution de l'application en utilisant seulement la commande suivante «bash run.sh » en terminale. Si le fichier n'est pas fourni, ou si la compilation et l'exécution ne fonctionnent pas dans l'environnement spécifié, le projet ne sera pas corrigé et sanctionné d'un zéro. **Testez donc avant que tout fonctionne !**

- Votre code devra respecter les principes de designs orientés objet comme vu au cours. Pensez donc à faire des choix logiques de design de classes afin de produire un code propre et maintenable.

- Votre code devra présenter une structure correcte et maintenable. Notamment :
  - Evitez la duplication de code.
  - Evitez les constantes magiques.
  - Evitez le code mort.
  - Commentez intelligemment et suffisamment votre code
  - Commentez correctement vos variables et méthodes (ex : pas de majuscules comme première lettre).

  Un code dont la qualité sera jugée insuffisante **sera sanctionné d'un zéro**.

- Votre jeu devra être développé en utilisant **exclusivement la librairie graphique Java Swing** comme vu en cours **et à partir du jeu du Snake** qui devra être utilisé comme base fonctionnelle.

- A l'exception des commentaires, l'entièreté de votre programme devra être codé en anglais (nom de variables/fonction/classes/etc…).

Rapport :

Il vous est demandé de rédiger un rapport décrivant votre projet. Votre rapport devra contenir au minimum les sections suivantes :

Introduction : Cette section devra introduire votre projet. Décrire ce qui a été réalisé et présenter brièvement la structure de votre rapport.
Fonctionnalités de base : Cette section devra expliquer les fonctionnalités offertes par votre application d'un point de vue à la fois fonctionnel et technique.
Fonctionnalités supplémentaires : Cette section devra expliquer les fonctionnalités supplémentaires offertes par votre application d'un point de vue à la fois fonctionnel et technique.
Analyse : Cette section devra expliquer la structure de votre implémentation en utilisant les outils d'analyses déjà vus durant votre parcours. Si aucun outil n'a été vu pour l'instant, considérez qu'il vous est demander d'élaborer des schémas explicatifs sur la structure de votre code. Attention : Tous les diagrammes doivent être commentés !
Limitations : Les limites de votre application, par exemple : dans quels cas d'utilisation votre application pourrait ne pas fonctionner comme prévu ? Y a-t-il des aspects techniques qui n'ont pas été traité ? Si vous aviez plus de temps pour le projet, qu'auriez-vous amélioré ? Plusieurs points de vue sont possibles, il revient au groupe d'étudiant de choisir les points qu'il considère les plus pertinents pour réaliser son autocritique.
Conclusion : Votre conclusion sur le projet. Ce que vous avez réussi à faire ou non durant le projet et les apprentissages que vous en tirez.

Le rapport sera notamment évalué sur la qualité écrite et l'effort de présentation ainsi que la pertinence et la complétude des points abordés.

Deadline et remise :

La date limite pour la remise du projet est le **mercredi 16 août à 23h59**. Le projet devra être déposé sur eCampus à l'intérieur d'un fichier **.zip** contenant toutes les sources de votre projet ainsi que le rapport au format **PDF**.

Développement et Triche :

- Tout acte de triche sera sanctionné par **une note de fraude au bulletin et sera notifié à la direction qui pourra possiblement décider de sanctions supplémentaires**. Des parties de code réutilisés d'un projet existant (d'un autre étudiant ou disponible sur le net) sans références dans votre rapport et sans mention de l'utilité du code utilisé est considéré comme une fraude.

- Pour ce projet, **vous ne pouvez pas reprendre des parties du code d'un autre étudiant**.

- Pour ce projet **vous ne pouvez pas vous inspirer/servir d'un jeu/code disponible sur internet.** Vous pouvez toutefois réutiliser les ressources vues en cours tel que le jeu du Snake (http://zetcode.com/javagames/snake/)

- Pour ce projet **vous ne pouvez pas vous inspirer/servir d'un code généré par des outils de génération de code tels que, par exemple, ChatGPT.**

- Si vous avez un doute, contactez l'enseignant le plus tôt possible afin d'éviter du refactoring inutile, ou pire, **une note de zéro/fraude**.

Conseil pratique :

Voici quelques conseils qui j'espère pourront vous aider.

- Veillez à ce que votre code ne contienne pas de constantes magique et/ou de duplication qui serait facilement évitable avec l'utilisation de méthodes.
- Veillez à effectivement implémenter les différents comportements demandés.
- Réfléchissez en terme d'« attribution de responsabilités » en accord avec les principes vu au cours (segmentation logique en classes).
- Ne négligez pas la théorie du cours (vous serez interrogés dessus).
- Ne négligez pas votre rapport. Tachez d'y expliquer/justifier explicitement vos choix d'implémentation. (Exemple : pourquoi un héritage ici ? pourquoi l'implémentation comme ceci ? quel avantage en termes de structure ? …)
- Prenez le temps de bien comprendre tout l'énoncé avant de vous lancer (et lisez la FAQ).
- Vérifiez que votre code compile et run effectivement via le script bash prévu sur la machine Ubuntu présente dans le SharePoint du cours.

FAQ :

- **Puis je ajouter d'autres sections ou sous-sections dans le rapport ?**

Oui. La partie rapport de ce document donne seulement la structure minimum.

- **Puis je coder ou rendre mon rapport en anglais ?**

Oui. Pour ce qui est du code vous devez toutefois respecter les usages corrects des conventions de nommage. Codez en anglais ou en français, pas les deux. L'anglais étant recommandé.

- **Puis je programmer sur Windows avec Eclipse ?**

Oui vous pouvez programmer comme vous le désirez mais vous devez respecter les contraintes de ce document, notamment : votre code doit être exécutable sur un environnement linux Ubuntu via un script run.sh que vous devez fournir en même temps que vos sources (voir les contraintes de développement). Une machine préconfigurée sera disponible sur le SharePoint.

- **Le rapport est-il important ?**

Oui. Le rapport est une **pièce centrale de votre projet** et c'est le premier outil de communication qui me servira à juger de la bonne réalisation du projet, pas seulement du point de vue du code mais également de la méthodologie utilisée.

- **Quel est le niveau de complexité que vous attendez pour les fonctionnalités supplémentaires ?**

La complexité de ce qui été développé sera prise en compte dans l'évaluation. Toutefois, entamer le dev. d'une fonctionnalité trop ambitieux risque de vous pénaliser si cela implique que vous deviez bâcler votre rapport pour compenser ou omettre des fonctionnalités demandées.

- **Que voulez-vous dire par « tous les diagrammes doivent être commentés ».**

Les diagrammes doivent servir à illustrer et appuyer vos explications sur la structure de votre implémentation. Ils ne remplacent aucunement un texte explicatif revenant sur les points d'attention.

- **Je n'ai pas réussi à tout réaliser. Est-ce que ça vaut la peine de vous rendre le projet ?**

Oui veillez toutefois à être claire sur les parties non implémentées. Il est très déconseiller de dissimuler ou d'« oublier » de mentionner qu'une partie n'a pas été réalisée. Veillez toutefois à bien respecter les consignes. Par exemple, votre code doit pouvoir compiler avec le script bash demandé, la qualité du code doit être suffisante, etc…

- **Puis je réaliser le projet en groupe ?**

Non le projet doit être réalisé individuellement.

- **Que voulez-vous dire par « Votre code devra respecter les principes de designs orientés objet comme vu au cours. »**

L'orienté objet fait intervenir certains principes comme l'héritage ou les méthodes statiques. Il revient à vous de décider quand les mettre en œuvre ou non. Votre approche devra toutefois être logique et justifiée. Cela implique notamment, de faire apparaitre de l'héritage quand cela a du sens, de rendre une classe abstraite quand cela a du sens, d'utiliser intelligemment l'encapsulation etc…

- **Dois-je vraiment faire de l'orienté objet ? Mon programme peut fonctionner sans.**

Vous devez absolument mettre en œuvre l'orienté objet pour ce projet. Cela fait partie des contraintes du projet. Un non-respect de ces contraintes sera pénalisé. L'utilisation de l'orienté objet n'est pas une contrainte faible. Veillez donc à la respecter.

- **Je ne peux vraiment pas utiliser de code venant d'internet ?**

Non appart pour ce qui peut être considéré comme des briques fonctionnelles. (Par exemple le code permettant de lire/écrire un fichier, le code relatif à l'utilisation des listes ou autres structures de données). Dans tous les cas ne prenez aucun risque et contactez l'enseignant le plus tôt possible si vous avez un doute !

- **L'aspect graphique du jeu et la jouabilité sont-ils des critères importants ?**

Ces critères seront pris en compte dans l'évaluation mais sont nettement moins importants que l'implémentation des fonctionnalités et le respect des contraintes. Dis grossièrement : Mieux vaut un jeu moche mais avec toutes les fonctionnalités implémentées qu'un jeu magnifique mais avec des fonctionnalités manquantes.

- **Quels sprites puis je utiliser pour le projet ? Dois-je vraiment trouver une image de poisson ?**

Vous êtes libre d'utiliser toutes les sprites/images que vous désirez. Pensez toutefois à spécifier leur source dans le rapport. Vous pouvez également utiliser des images abstraites. Vous êtes vraiment libre en ce qui concerne l'aspect purement graphique.