Projet commun POO – COO : Hex Light Bot

Date de remise impérative du rapport : vendredi 7 avril 2017 à 16h

On peut « développer » via des jeux qui ne portent pas forcément sur du "code informatique" (affectations, calculs, etc.). Sur Internet on trouve facilement le jeu **LightBot**, qui consiste à construire une suite d'actions pour piloter un robot. Il existe de multiples déclinaisons, le terrain peut être plat ou avec relief, 2D, 3D. . . , etc.

2 Résumé du jeu

Pour résumer :

- on voit sur l'écran un labyrinthe avec des cases, et des trésors dans certaines cases. Il y a un robot sur une case.
- le robot doit passer au moins une fois sur les cases indiquées par une couleur spécifique.
- le jeu comporte deux phases : programmation et exécution.

2.1 Phase de programmation

Dans la première phase, le joueur compose un programme en construisant une séquence d'actions. Une interface intuitive "glisser déposer" sera a réalisé : le joueur prend une action (figurée par une icone) et la dépose à l'endroit voulu de son programme (contenu dans une boite).
Il y a deux sous-programmes : le programme "main" peut faire appel à l'action P1 et à l'action P2. P1 correspond à l'exécution des actions du sous programmes PROC1 ; et P2 à celles de PROC2.

Un bouton "exécution" fait passer à la phase suivante : exécution du programme main.

2.2 Phase d'exécution

Cette phase d'animation montre le déroulement du programme construit par l'utilisateur.
Le robot effectue les actions du programme principal. Il s'arrête quand il n'y a plus d'actions à effectuer, ou si une action est impossible (par exemple, avancer dans un mur).

Si l'objectif est atteint, l'utilisateur peut passer au niveau suivant, sinon il peut recommencer.

Les niveaux sont décrits dans un fichier de configuration.

3 Votre projet

3.1 Objectif

Vous travaillerez en binôme, au sein d'un même groupe de TD.
Vous réaliserez une adaptation de ce jeu sur terrain **à structure hexagonale** :
Votre exécutable comportera deux parties :

- un programme pour jouer,
- un éditeur de terrains.

Le programme sera réalisé en C++11, en utilisant la bibliothèque SFML 2.0.

Cahier des charges de la partie conception

Forme du document

- Le rapport est à réaliser en binôme (deux étudiants du même groupe de S2). Vous devez obtenir la permission de votre chargé de TD pour réaliser le projet seul ou en trinôme. **Votre chargé de TD peut refuser de corriger un projet non réalisé en binôme, si l'accord n'a pas été obtenu au préalable (il sera donc noté par un zéro).**
- Le rapport sous format papier non manuscrit doit être rendu impérativement pour le **vendredi 7 avril 2017.**
- Sur chaque page, vous devez indiquer vos noms et vos prénoms, le nom de votre groupe.
- Les pages doivent être numérotées.
- Le document ne doit pas être manuscrit, vous pouvez utiliser un AGL.
- Le document doit être agréable à lire (avec une taille de la police de 11 ou 12, le texte doit être justifié à droite et à gauche), chaque diagramme doit avoir un titre ….
- Chaque diagramme doit être sur une seule feuille A3 ou A4.
- La qualité de la rédaction sera évaluée.

Questions

1 Détaillez tous vos choix. Par exemple :

- **a.** listez les actions élémentaires que vous proposer au « joueur/programmateur ».
- **b.** indiquez la taille maximale du programme principal (c'est à dire le nombre d'actions que peut contenir le programme principal).
- **c.** indiquez la taille maximale de PROC1.
- **d.** indiquez la taille maximale de PROC2.
- **e.** comment le joueur identifie-t-il les hexagones sur lesquels il doit faire passer son robot ?
- **f.** comment le joueur identifie-t-il les hexagones sur lesquels son robot est déjà passé ?
- **g.** listez et expliquez clairement les fonctions de contrôle du jeu que vous implémentez : reboot, reset, quit, stop, clean ….
- **h.** Donnez l'algorithme qui calcule le score du joueur.
- **i.** Indiquer le nombre de niveau que vous implémenter. Pour chaque niveau, expliquez en quoi, il se distingue du niveau précédent (en quoi la difficulté augmente).

D'autres informations peuvent être ajoutées.

Les diagrammes UML réalisés doivent être cohérents avec la description de vos choix.

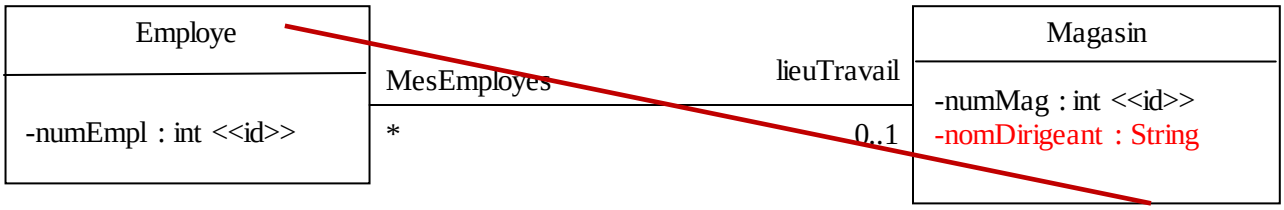
1. Réalisez le diagramme de contexte statique.
2. Listez les événements externes.
3. Listez les événements temporels.
4. Listez les événements résultats.
5. Etablissez le diagramme des cas d'utilisation.
6. Proposez 3 diagrammes de communication pertinents. Pour chaque diagramme de communication réalisé, écrivez le scénario représenté.
7. Réalisez le diagramme de classes (classes, attributs, méthodes, associations, nom des rôles, multiplicités, visibilités et navigabilités).
8. Réalisez le dictionnaire de données – définition des classes, des attributs et méthodes nécessaires à la compréhension de vos diagrammes.
9. Etablissez la signature des méthodes hors du diagramme de classes.
10. Donnez un état honnête d'avancement du projet, en précisant les lacunes de votre conception (si elles existent).

Consignes spécifiques pour le diagramme de classes

- Le diagramme de classes ne doit pas être généré automatiquement à partir du code **(il n'existe pas de générateur de diagramme de classes à partir de code qui réalise un diagramme correcte**).

- Chaque association doit avoir (1) ses deux multiplicités et (2) un nom de rôle ou une classe associative.

- Ne pas mettre un attribut ou des attributs à la place d'une association. Diagramme de classes faux (remplacement d'une association par un attribut).



Hexagone de rayon r centré en (x,y)