Université Claude Bernard Lyon 1
Département Informatique

Projet APOO

Jeu de stratégie et de plateau inspiré de *SmallWord*

Travail à réaliser

Développer une application graphique java objet et modulaire la plus aboutie possible (utilisant Swing) inspirée du jeu de stratégie et de plateau *SmallWorld*.

1. Implémenter les fonctionnalités de base en respectant le modèle MVC et les consignes de modélisation orientée objet.
2. Une fois l'application fonctionnelle, développer des extensions, et réalisez l'application la plus aboutie possible.

Travail en binômes

— Travail personnel entre les séances
— Démonstration par binômes, évaluations individuelles
— Rapport par binôme : 6 pages au maximum, listes de fonctionnalités et extensions (indiquer la proportion de temps associée à chacune d'elles), copies d'écran, 1 diagramme UML de votre choix.
— Démonstration lors de la dernière séance encadrée (présence des deux binômes obligatoire).
— Test anti-plagiat sur les code soumis

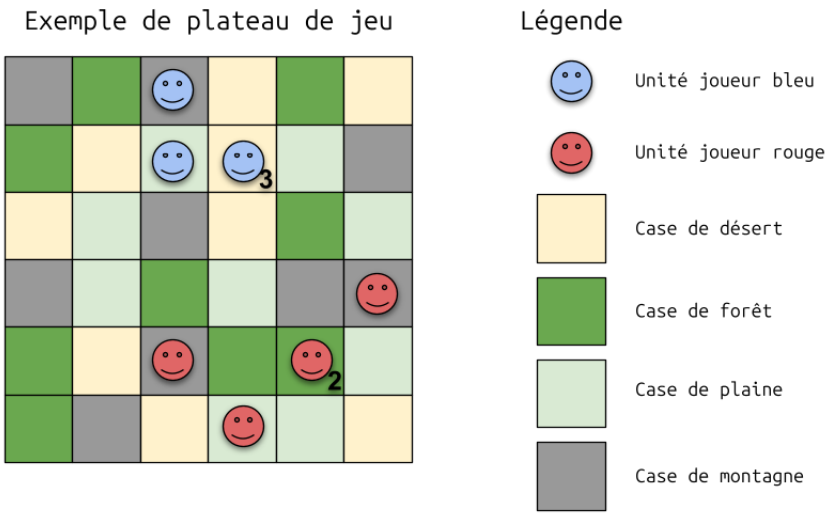
1 Fonctionnalités de base à modéliser et développer

Le jeu à réaliser, inspiré se *SmallWorld*, est un jeu de stratégie se jouant de 2 à 4 joueurs. Une partie se joue au tour par tour sur un nombre de tours prédéfinit : **6 tours** par défaut.

1.1 Le plateau

Le plateau de jeu est définit par une grille carrée de **côté 6x6** (à 2 joueurs) ou **7x7** (à 3 ou 4 joueurs). Chaque case de la grille est d'un type de terrain parmi les suivants : **Plaine**, **Forêt**, **Montagne** ou **Désert**.

Au lancement d'une partie, la grille est générée aléatoirement avec une répartition aussi équitable que possible de chaque type de terrain.



1.2 Les joueurs

Chaque joueur et joueuse joue un peuple (attribué aléatoirement ou sélectionnée au lancement) parmi les suivantes : **Humain**, **Elfe**, **Nain** ou **Gobelin** (et d'autres si vous le souhaitez). Chaque peuple à un terrain favori qui lui rapporte des points à la fin d'un tour de jeu.

Peuple	**Terrain**
Humain	Plaine
Elfe	Forêt
Nain	Montagne
Gobelin	Désert

Au lancement de la partie, toutes les unités d'un joueur sont toutes positionnées dans un coin de la grille (les coins opposés pour un jeu à 2 joueurs). Le nombre d'unité au lancement est de 8 par joueur, mais vous pouvez choisir de le faire varier selon les caractéristiques de chaque peuple.

1.3 Les unités

Une unité est positionnée sur une case du plateau. Plusieurs unités peuvent se trouver sur la même case si elles sont du même type.

Lors de son tour de jeu, chaque unité peut être jouée jusqu'à une fois. Quand on joue une unité, deux action facultatives sont disponibles dans cet ordre :

- Déplacer l'unité d'autant de cases que sa caractéristique 'mouvement'. Un déplacement se fait sur une des 4 cases adjacente si elle ne contient pas d'unité du même type (pas de déplacement en diagonale).
- Attaquer une unité adjacente appartenant à un autre joueur.

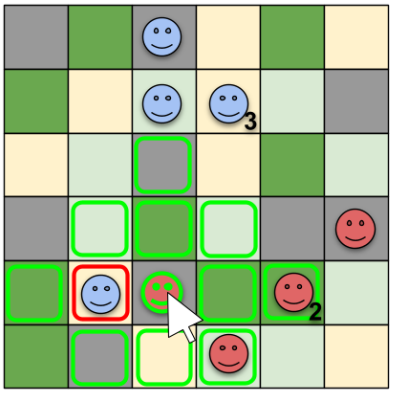
Remarque : une unité peut donc en attaquer une autre sans se déplacer, mais elle ne peux alors pas bouger après.

1.4 Tour de jeu

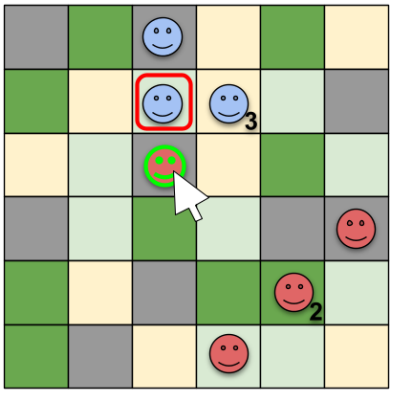
Un joueur peut jouer autant d'unité qu'il le souhaite (chaque unité ne peut être jouée qu'une seule fois).

La sélection se fait en cliquant sur une de ses unités. On affiche alors les cases accessibles et attaquables. Si on clique sur une case, l'unité est déplacée dessus. On continue d'afficher les cases attaquables. Si on clique sur une case attaquable, l'unité sélectionnée attaque une unité adverse présente sur la case.

1. Après sélection : affichage des cases accessibles/attaquables



2. Après déplacement : affichages uniquement des cases attaquables



Quand on clique sur une case contenant plusieurs de nos unités, une seule d'entre elles effectivement sélectionnée.

À tout moment, il est possible de mettre fin à son tour et passer au joueur suivant.

Quand une unité en attaque une autre, un calcul aléatoire détermine le vainqueur. Le vaincu est éliminé et l'unité est retiré du jeu. Si la case de l'unité vaincu contient plusieurs unités, une seule d'entre elles est retirée par attaque.

Pour calculer le résultat d'une attaque on pourra par exemple donner un avantage selon le terrain, le peuple, le nombre d'unités sur la case, ...

1.5 Points et fin de partie

Un joueur gagne des points :

— un point à chaque fois qu'il élimine une unité adverse.

— à la fin de son tour, un point pour chaque case occupée par au moins une de ses unité dont le type correspond au préféré du peuple joué.

La partie s'arrête quand tous les joueurs ont joué le nombre de tour prédéfini. Celui ou celle ayant le plus de points gagne.

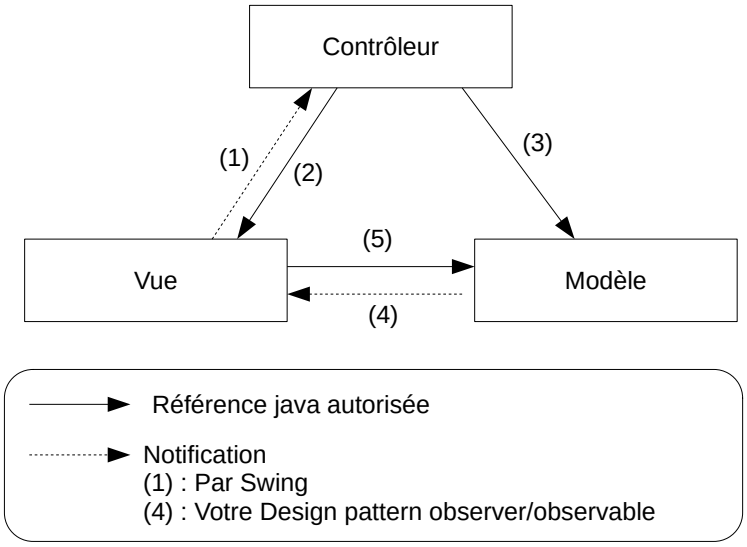
2 Suggestion d'étapes de développement

— Se familiariser avec le code fourni. Identifier le modèle, la vue, les contrôleurs. Comprendre le passage de coups à jouer entre le processus graphique et le processus de jeu.
— Ajouter les biomes à la classe Case, générer les environnements, mettre à jour de rafraîchissement.
— Modifier la classe unité pour matérialiser le nombre d'unités sur la Case, mettre à jour l'affichage afin de visualiser le nombre d'unités.
— Développer les fonctionnalités de base décrites dans le sujet. S'appuyer sur l'analyse étudiée en cours.
— Ajouter une ou plusieurs extensions.

3 Extensions

— Développer une vue console (vérifier qui le changement de vue peut se faire sans recompiler le modèle)
— Ajouter la notion d'obstacle à l'environnement (rivière, Arbres denses / broussailles, Champignons toxiques, etc.) qui influence les condition de déplacement
— ajouter des évènements aléatoires à chaque tour de jeu par case (Tempêtes de sable, chaleur extrême, etc.) qui influencent les conditions de déplacement et de combat suivant les peuples jouées
— Sauvegarder et charger une partie
— Version réseau
— Joueur IA
— Une extension que vous proposez

Rappel MVC



MVC Strict :
(1) Récupération de l'événement Swing par le contrôleur
(2) Répercutions locale directe sur la vue sans exploitation du modèle
(3) Déclenchement d'un traitement pour le modèle
(4) Notification du modèle pour déclencher une mise à jour graphique
(5) Consultation de la vue pour réaliser la mise à jour Graphique

Exemple : Une application Calculette
(1) récupération clic sur bouton de calculette
(2) construction de l'expression dans la vue (1,2...9, (,))
(3) déclenchement calcul (=)
(4) Calcul terminé, notification de la vue
(5) La vue consulte le résultat et l'affiche