Projet

Contexte du projet

Le but de ce projet consiste à implémenter le célèbre jeu de cartes ***Uno.***

Ce projet fera appel à des notions vues à travers différents cours que vous avez suivis cette année. En particulier, la programmation objet sera sans doute la compétence centrale du projet, mais vous devrez également mettre en œuvre des compétences réseaux, bases de données, et interface graphique.
La version du jeu utilisée dans ce projet est le Uno classique se jouant de 2 à N joueurs avec les cartes classiques comme celles illustrées dans l'image ci-dessus. Il faudra s'efforcer de respecter les règles même s'il est fréquent d'avoir des interprétations différentes d'un endroit à l'autre. La plus controversée est l'empilement des +2 ! Y a-t-on droit ou pas ? Peu importe, c'est vous qui déciderez !
L'un des points importants de ce projet est qu'il doit être possible d'y jouer en réseau. Un joueur lancera un serveur, et les autres joueurs s'y connecteront grâce à leur client respectif. Lorsque le joueur propriétaire du serveur le décidera, la partie démarrera. A la fin de la partie, le serveur enregistrera les scores des joueurs dans une base de données (pour pouvoir faire des statistiques plus tard, mais pas dans ce projet). L'interaction avec la base de données sera réalisée grâce aux paquets JDBC (vous aurez une initiation à ces paquets). Enfin, le client d'un joueur sera un client graphique développé avec JavaFX.

Le projet est assez vaste et si vous ne vous y prenez pas avec un peu de méthode, vous risquez de partir dans de mauvaises directions et ne jamais aboutir à quelque chose de fonctionnel. C'est pourquoi, nous vous imposons une architecture logicielle particulière.

1. Le jeu

Le jeu, c'est la partie « métier » de ce projet. C'est l'ensemble des classes Partie, Joueur, Carte, Pioche, etc. qui représentent les abstractions du Uno. Ces classes ne contiennent ni interface graphique, ni bases de données, ni programmation réseau. Ce sont des

classes Java bien construites, extensibles, et avec toutes les exceptions qu'il faut. Mais pas de saisie ni d'affichage dans la console !
Pour vous guider, dans cette partie, des tests unitaires vous sont proposés à la fin de cet énoncé.

RQ : toutes les cartes ne sont pas essentielles

2. Le serveur

Sur le modèle d'architecture que vous aurez vu lors de l'initiation à la programmation réseau, vous devez développer un serveur qui gérera les communications avec les différents joueurs. C'est-à-dire que lorsque le serveur reçoit un message réseau d'un joueur, il l'analyse et le transforme en appel de la méthode métier approprié. Il renvoie alors au joueur un message réseau contenant le résultat de cet appel. Le protocole de communication vous est imposé, ce qui permettra en principe à deux clients de deux équipes différentes de communiquer. Notez que pour tester ce serveur, il n'est pas nécessaire de développer un client. Vous pouvez en principe vous y connecter avec telnet ou puTTY en envoyer vos commandes manuellement (voir TP).

3. Le client

Évidemment, l'objectif est de pouvoir jouer en réseau sans avoir à saisir des lignes de commande complexes. Pour éviter cela, vous allez naturellement développer une interface graphique, où les clics souris, les boutons, etc. produiront automatiquement les commandes à envoyer au serveur

4. La base de données

La base de données est minimaliste. On souhaite juste enregistrer les joueurs, les parties, et les scores des joueurs qui ont participé à ces parties. Le MCD est illustré ci-dessous. Vous devez implémenter cette base de données dans MySQL.
C'est uniquement le serveur qui communique avec la base de données. Au début d'une partie, lorsqu'un joueur arrive (c'est-à-dire se connecte), le serveur vérifie si le joueur a déjà été enregistré (il regarde si le pseudo est présent). Et s'il ne l'est pas encore, alors il enregistre ce nouveau joueur. A la fin de la partie, il enregistre la partie (sa date uniquement), et les scores de chacun des joueurs.

Les tests de la partie métier

L'ensemble des tests ci-dessous, bien que n'utilisant que des cartes simples (numéro et couleur), permettent de valider l'essentiel de la « mécanique » du jeu. Par la suite, par extensions successives, nous ajouterons les autres types de cartes, avec les règles de vérification et les effets que ces cartes produisent. Pour autant, la « mécanique » du jeu ne devra pas être impactée par ces extensions. C'est tout le défi de bien organiser ses classes, de bien identifier où et comment mettre en place de l'héritage, bien identifier les exceptions, etc.

La réalisation de ces tests doit logiquement faire l'objet d'une fonction par test. Il sera sans doute important de bien « outiller » les classes pour pouvoir réaliser les tests facilement. Autrement dit, il sera utile d'ajouter aux classes des méthodes sans intérêt pour le jeu lui-même, mais très utiles (voire indispensables) pour l'écriture de ces tests.

Chaque test est décrit dans un environnement précis, c'est-à-dire avec une pile de cartes dans un ordre fixe et donné au début du test. Vous devez donc, au début de chaque test, faire en sorte que les joueurs possèdent bien les cartes prévues, que la pioche contienne bien dans l'ordre les cartes citées, et que la carte se trouvant au sommet du tas est bien celle indiquée.

Tests coups légaux avec des cartes simples

Alice	*Bob*	*Charles*
--------	*--------*	*--------*
Vert 2	*Bleu 2*	*Bleu 9*
Jaune 6	*Jaune 4*	*Bleu 7*
Rouge 1	*Rouge 9*	*Bleu 0*

Tas : CarteSimple [valeur=8, couleur=Vert]

Pioche

CarteSimple [valeur=6, couleur=Jaune]
CarteSimple [valeur=4, couleur=Rouge]
CarteSimple [valeur=2, couleur=Vert]
CarteSimple [valeur=5, couleur=Bleu]
CarteSimple [valeur=0, couleur=Vert]

Alice joue une carte de la bonne couleur

Vérifier que le joueur courant est bien Alice
Vérifier que Alice possède bien 3 cartes
Alice joue le « 2 Vert »
Vérifier que Alice possède bien 2 cartes
Vérifier que les cartes d'Alice sont le « 6 jaune » et le « 1 rouge »

Vérifier que la carte au sommet du tas est le « 2 Vert »
Vérifier que le nombre de cartes du tas est 2
Alice finit le tour
Vérifier que le joueur courant est Bob

Bob joue une carte de couleur différente mais de même valeur

Vérifier que Bob possède bien 3 cartes
Bob pose le « 2 bleu »
Vérifier que Bob possède bien 2 cartes
Vérifier que les cartes de Bob sont le « 4 jaune » et le « 9 rouge »
Vérifier que la carte au sommet du tas est le « 2 Bleu »
Vérifier que le nombre de cartes du tas est 3
Bob finit le tour
Vérifier que le joueur courant est Charles

Tests coups illégaux avec des cartes simples

Tous ces tests vont lancer des exceptions. Toutes ces anomalies sont (en principe !) sanctionnées par 2 cartes de punitions, et fin du tour automatique si c'est le joueur courant qui commet la faute. C'est dans le catch de l'exception que doivent être réalisées ou non ces deux actions. Mais pour l'instant, il n'est pas utile de gérer la « punition ». Nous le ferons dans d'autres tests par la suite (une chose après l'autre).
Pour chacun des tests de cette partie, il faut réinitialiser la partie pour se retrouver dans les conditions des tests précédents

Test d'une carte illégale

Alice pose le « 6 jaune »
Vérifier dans le catch approprié que Alice possède toujours 3 cartes dont le « 6 Jaune »

Test d'un joueur qui pose deux cartes légales de suite

Alice pose le « 2 Vert » et finit son tour
Bob pose le « 2 Bleu » et finit son tour

Charles pose le « 6 Bleu » (RAS, c'est correct mais Charles ne finit pas le tour)
Vérifier que Charles possède 2 cartes
Charles pose le « 7 Bleu » (Carte légale mais il a déjà posé…)
Vérifier dans le catch approprié que Charles possède toujours 2 cartes dont le « 2 Bleu »

Test d'un joueur qui finit son tour sans rien faire

Alice finit son tour
Vérifier dans le catch approprié que Alice possède toujours 3 cartes

Test d'un joueur qui joue puis pioche

Alice joue le « 2 Vert » (RAS, le coup est légal)
Alice pioche
Vérifier dans le catch approprié que Alice possède toujours 2 cartes
Vérifier que la carte de la pioche est toujours le « 6 jaune »

Tests de la punition

Pour chacun des tests de cette partie, il faut réinitialiser la partie pour se retrouver dans les conditions des tests précédents

Test de la punition pour un coup illégal d'Alice (joueur courant)

Vérifier que le joueur courant est bien Alice
Alice pose le « 6 jaune » (coup illégal)
Dans le catch approprié :

- Punir Alice
- Vérifier que Bob est le joueur courant
- Vérifier que Alice possède 5 cartes dont le « 6 jaune » et le « 4 rouge » (les 2 cartes de la pioche)
- Vérifier que la prochaine carte de la pioche est le « 2 vert »

Test d'une action de bob lorsque ce n'est pas son tour

Vérifier que le joueur courant est bien Alice
Bob pioche (ce n'est pas son tour)
Dans le catch approprié :

- Punir Bob
- Vérifier que Alice est toujours le joueur courant
- Vérifier que Bob possède 5 cartes dont le « 6 jaune » et le « 4 rouge » (les 2 cartes de la pioche)
- Vérifier que la prochaine carte de la pioche est le « 2 vert »

Test du « Uno ! »

Une règle de base du jeu consiste à dire « Uno ! » dès qu'il nous reste plus qu'une seule carte en main. Bien sûr, il faut le dire quand c'est encore son tour…

Alice	*Bob*	*Charles*
Vert 2	*Bleu 2*	*Bleu 9*
Jaune 6	*Jaune 4*	*Bleu 7*

Tas : CarteSimple [valeur=8, couleur=Vert]

Pioche

```
------
CarteSimple [valeur=6, couleur=Jaune]
CarteSimple [valeur=2, couleur=Vert]
CarteSimple [valeur=5, couleur=Bleu]
CarteSimple [valeur=0, couleur=Vert]
CarteSimple [valeur=3, couleur=Bleu]
```

Test lorsqu'Alice dit « Uno ! » au bon moment

Vérifier qu'Alice a bien 2 cartes
Alice pose le « 2 Vert »
Alice dit « Uno ! »
Alice finit son tour
Vérifier qu'Alice n'a plus qu'une seule carte
Vérifier que la carte au sommet du tas est le « 2 Vert »
Vérifier que le joueur courant est Bob

Test lorsqu'Alice oubli de dire « Uno ! »

Alice pose le « 2 Vert »
Alice finit son tour
Dans le catch approprié :
Punir Alice
Vérifier qu'Alice a maintenant 4 cartes
Vérifier que la carte au sommet du tas est le « 8 Vert »
Vérifier que le joueur courant est Bob

Test lorsque Bob dit « Uno ! » quand ce n'est pas son tour

Vérifier que Alice est le joueur courant
Bob dit « Uno ! »
Dans le catch approprié :
Punir Bob
Vérifier que Bob a maintenant 4 cartes
Vérifier qu'Alice est toujours le joueur courant
Vérifier que la carte au sommet du tas est le « 8 Vert »

Tous les tests ci-dessous permettent de vérifier que la mécanique du jeu est en place. Il ne manque plus qu'à ajouter les différents types de carte (passe ton tour, +2, +4, changement de couleur, et d'autres variantes encore). En principe, la chaine de responsabilités va permettre ces évolutions très facilement. Il ne sera plus nécessaire de retester tous les cas de figure ci-dessus.

Tests avec des cartes « Passe ton tour »

La partie contient 3 joueurs : Alice, Bob et Charles
Distribution de 3 cartes, et Alice commence

Alice	*Bob*	*Charles*
----------	*----------*	*----------*
Rouge Passe	*Jaune 6*	*Bleu 1*
Bleu 9	*Vert 6*	*Vert Passe*
Jaune 4	*Bleu 7*	*Rouge 1*

Tas : CarteSimple [valeur=9, couleur=Rouge]

Pioche

CarteSimple [valeur=0, couleur=Bleu]
CarteSimple [valeur=8, couleur=Vert]
CarteSimple [valeur=2, couleur=Vert]
CarteSimple [valeur=4, couleur=Rouge]
CarteSimple [valeur=2, couleur=Vert]

Test de coups légaux avec des cartes « Passe ton tour »

Vérifier que Alice est bien le joueur courant
Alice pose le « Passe ton tout rouge »
Alice finit son tour
Vérifier que Charles est le joueur courant
Vérifier que la carte du tas est bien le « Passe ton tour rouge »

Charles pose le « Passe ton tour vert »
Charles finit son tour
Vérifier que Bob est le joueur courant
Vérifier que la carte du tas est bien le « Passe ton tour vert »

Bob pose le « 6 Vert »
Bob finit son tour
Vérifier que Charles est le joueur courant
Vérifier que la carte du tas est bien le « 6 Vert »

Test d'une carte simple illégale sur un « Passe ton tour »

Vérifier que Alice est bien le joueur courant
Alice pose le « Passe ton tout rouge »

Alice finit son tour

Vérifier que Charles est le joueur courant
Vérifier que Charles possède bien 3 cartes
Charles pose le « 1 Bleu »
Charles finit son tour
Vérifier dans l'exception appropriée que Charles a toujours 3 cartes

Test d'un « Passe ton tour » illégal sur une carte simple

Vérifier que Alice est bien le joueur courant
Alice pose le « 9 bleu »
Alice finit son tour
Bob pose le « 7 Bleu »
Bob finit son tour

Vérifier que Charles est le joueur courant
Vérifier que Charles possède bien 3 cartes
Charles pose le « Passe ton tour vert »
Charles finit son tour

Vérifier dans l'exception appropriée que Charles a toujours 3 cartes

Tests avec des cartes « +2 »

Test d'un coup légal avec une carte « +2 »

Distribution de 3 cartes, et Alice commence

Alice	*Bob*	*Charles*
Vert Plus2	*Jaune 6*	*Bleu 1*
Bleu 9	*Vert 6*	*Vert Plus2*
Jaune 4	*Bleu 7*	*Vert 1*

Tas : CarteSimple [valeur=9, couleur=Vert]

Pioche

CarteSimple [valeur=0, couleur=Bleu]
CarteSimple [valeur=8, couleur=Vert]
CarteSimple [valeur=2, couleur=Vert]
CarteSimple [valeur=4, couleur=Rouge]
CarteSimple [valeur=2, couleur=Vert]

Vérifier qu'Alice est le joueur courant
Alice pose « +2 Vert »
Alice finit son tour
Vérifier que Bob est le joueur courant
Vérifier que Bob possède 3 cartes
Bob doit encaisser l'attaque (donc piocher 2 cartes et finir automatiquement son tour)
Vérifier que Bob possède 5 cartes
Vérifier que Charles est le joueur courant
Charles pose le « 1 Vert »
Charles finit son tour
Vérifier que Charles possède 2 cartes

Test d'un coup légal avec cumul de cartes « +2 »

Vérifier qu'Alice est le joueur courant
Alice pioche une carte
Alice finit son tour
Vérifier que Bob est le joueur courant
Bob pioche une carte
Bob finit son tour
Vérifier que Charles est le joueur courant
Charles pose le « +2 Vert »
Charles finit son tour
Vérifier que Alice est le joueur courant
Alice pose le « +2 Vert »
Alice finit son tour
Vérifier que Bob est le joueur courant
Vérifier que Bob possède 4 cartes
Bob encaisse l'attaque (donc pioche 4 cartes et finit son tour automatiquement)
Vérifier que Bob possède 8 cartes
Vérifier que Charles est le joueur courant