ÉCOLE D'INGÉNIEUR-E-S
Sceaux I Troyes I Montpellier

Projet : Super Puissance 4

Cahier des charges et documentation technique

Glossaire

Partie : Durée de jeu pendant laquelle les joueurs sont actifs, et doivent effectuer des choix. La partie débute après que chaque joueur a eu une couleur désignée, et se termine à l'annonce d'un vainqueur.

Match nul : Situation irréversible laquelle aucun joueur ne peut espérer remporter la victoire. Le match nul est possible dans la version du jeu utilisant les règles de base. Il n'est plus possible dans la version du jeu utilisant les règles étendues

Tour de jeu d'un joueur : Durée de jeu pendant laquelle le joueur effectue une ou plusieurs actions immédiatement consécutives qui ne peuvent être interrompues par l'action d'un autre joueur.

Tour de jeu : Durée de jeu pendant laquelle chaque joueur aura effectué son tour de jeu, en commençant par le premier joueur

1. Présentation du projet

1.1. Résumé

Le puissance 4 est un célèbre jeu dans lequel deux joueurs s'affrontent en déposant à tour de rôle des pions sur une grille de 6 lignes et 7 colonnes, et doivent réaliser un alignement de 4 pions de leur couleur. L'objectif de ce projet est le développement d'une version améliorée de ce jeu qui incluse de nouvelles règles incluant trous noirs et désintégrateurs . Le développement de ce projet s'appuie sur le présent document, reprenant un cahier des charges détaillés ainsi les spécifications techniques retenues pour le développement de ce projet. Le projet devra répondre au différentes exigences et verrous levés par ces deux documents. Le suivi du développement de ce projet suivra une organisation présentée en Section Page 6 du présent document.

2. Cahier des charges du projet

2.1. Règles du jeu

La présente sous-section présente les règles du jeu, découpées en deux sous-parties : les règles de base, présentées en sous-sous section, rappelle les règles usuelles de la version classique du jeu. Les règles étendues, présentées en sous-sous section présente les nouvelles règles à considérer dans ce projet.

2.1.1 Règles de base

Le jeu met en scène deux joueurs qui s'affrontent autours d'une grille de jeu de 6 lignes par 7 colonnes pour un total de 42 cellules.

Préparation de la partie : Lors de la préparation de la partie, un des joueurs se voit assigner une couleur aléatoire parmi Rouge ou Jaune. L'autre joueur se voit assigner l'autre couleur. Chaque joueur se voit remettre 21 jetons de la couleur qui lui a été assignée. On tire une couleur aléatoire parmi jaune et rouge ; le joueur associé à cette couleur sera le premier joueur.

Déroulement de la partie : A tour de rôle, chaque joueur choisit une colonne de la grille et y glisse un jeton de sa couleur, qui descend jusqu'à atteindre la ligne la plus basse non occupée. Si l'ajout de ce nouveau jeton ne produit pas un alignement de quatre jetons ou plus de la même couleur, on passe au joueur suivant et ainsi de suite.

Fin de partie : Le premier joueur qui arrive à aligner au moins 4 jetons de sa couleur gagne la partie et remporte une victoire. Si aucun joueur n'arrive à aligner 4 jetons lorsque la grille de jeu est remplie, la partie se termine sur un match nul.

2.2. Règles étendues

Les règles suivantes modifient le jeu en y ajoutant de nouveaux mécanismes :

- Présence de trou noirs
- Récupération de ses jetons
- Désintégration de jetons adverses

Présence de trous noirs

En début de partie, 5 trous noirs sont positionnés aléatoirement sur des cases de la grille de jeu et sont visibles. Lorsqu'un jeton est joué et s'arrête sur une case contenant un trou noir, alors le Jeton est absorbé par le trou noir et disparait. Le trou noir disparaît également mais le joueur courant ne peut plus jouer pour ce tour.

Récupération de ses jetons

Durant son tour de jeu, un joueur peut choisir de ne pas jouer de jeton, mais de récupérer à la place un jeton de sa couleur. Cette action entrainera un décalage de tous les jetons de la même colonne situés au-dessus du jeton récupéré d'une ligne vers le bas. Cette action peut apporter immédiatement la victoire ou la défaite : la nouvelle configuration de la grille est analysée, et si un alignement de quatre jetons ou plus se produit pour un seul joueur uniquement, ce dernier remporte la partie. Si les deux joueurs se retrouvent chacun avec quatre jetons alignés ou plus, alors le joueur étant à l'origine de la récupération de jeton perd immédiatement pour avoir provoqué une faute de jeu.

Désintégration de jetons adverses

En début de partie sont positionnés aléatoirement 5 désintégrateurs, dont 2 sur des cases occupées par des trous noirs. Les 2 désintégrateurs positionnés sur les cases à trou noir ne sont pas visibles, car cachés derrière les trous noirs. Les autres désintégrateurs sont visibles. Le premier joueur qui positionne un de ses jetons sur une case occupée par un désintégrateur le remporte. Chaque joueur a un nombre de désintégrateurs, initialisé à zéro en début de partie, qui se voit augmenter d'une unité lorsqu'il remporte un désintégrateur. Lorsqu'un joueur dispose d'un désintégrateur, il peut choisir à tout moment de l'utiliser pendant son tour de jeu au lieu de jouer un Jeton. Un désintégrateur se joue uniquement sur un jeton adverse déjà positionné, et désintègre celui-ci. Le jeton désintégré explose en particules subatomiques et est définitivement perdu. Cette action peut apporter immédiatement la victoire ou la défaite : la nouvelle configuration de la grille est analysée, et si un alignement de quatre jetons ou plus se produit pour un seul joueur uniquement, ce dernier remporte la partie. Si les deux joueurs se retrouvent chacun avec quatre jetons alignés ou plus, alors le joueur étant à l'origine de la désintégration de jeton perd immédiatement pour avoir provoqué une faute de jeu.

A noter que si un désintégrateur était caché derrière un trou noir, le désintégrateur est remporté lorsque le joueur positionne son jeton sur le trou noir, en même temps que le trou noir est détruit.

2.3. Organisation des versions de développement

Compte tenu de la dimension du projet, il a été convenu de découper le projet en versions incrémentales, auxquelles pourront s'ajouter des fonctionnalités indépendantes mais complémentaires.

Version 1.0 : Cette version offre un mode de jeu utilisant seulement les règles de base. L'affichage est présenté en mode console. A chaque tour, chaque joueur entre simplement un numéro de ligne compris entre 1 et 7, et le jeton est positionné en conséquence. La partie est unique et le jeu se termine une fois une manche gagnée.

Version 1.1 : Cette version ajoute la fonctionnalité « Trous noirs » des règles étendues. Les conditions de jeu restent inchangées.

Version 1.2 : En plus des fonctionnalités de la version 1.1, cette version ajoute la règle « récupération de ses jetons». L'interface console de l'utilisateur est légèrement modifiée : lors de son tour de jeu, le joueur doit saisir s'il souhaite jouer un jeton ou récupérer un jeton. Selon le choix opéré, il saisira simplement le numéro de ligne dans laquelle ajouter un jeton, ou les coordonnées du jeton à récupérer.

Version 1.3 : La version 1.3 offre toutes les fonctionnalités nécessaires pour l'application des règles avancées. L'interface console demandera au joueur courant s'il souhaite jouer un jeton, en récupérer un, ou en désintégrer un. Ce dernier choix ne sera possible que si le joueur possède un désintégrateur en sa possession .

Version 1.4 : La version 1.4 propose une interface graphique proposant une meilleure expérience utilisateur. L'interface a été imaginée pour être la plus simple et minimale possible : chaque colonne sera associée à un bouton, et le clic sur ce bouton signifie que le joueur joue dans la colonne indiquée. Un clic sur un Jeton de sa couleur indique l'action « récupérer son jeton », tandis qu'un clic sur un jeton d'une couleur adverse indique que le joueur souhaite désintégrer le jeton ciblé. Ces actions doivent bien entendu être réalisables, autrement le clic sur le bouton ou le jeton correspondant devra rester sans effet.

2.4. Identification des verrous et contraintes

3.Spécifications techniques

3.1. Outils utilisés,

- Le projet sera entièrement développé en langage JAVA, et reposera sur le paradigme de programmation objet.
- Le développement de ce jeu s'appuiera, sauf sur justification du développeur , sur l'environnement de développement intégré Netbeans, version 8.2.

3.2. Classes composant le modèle

3.2.1 Liste des classes retenues

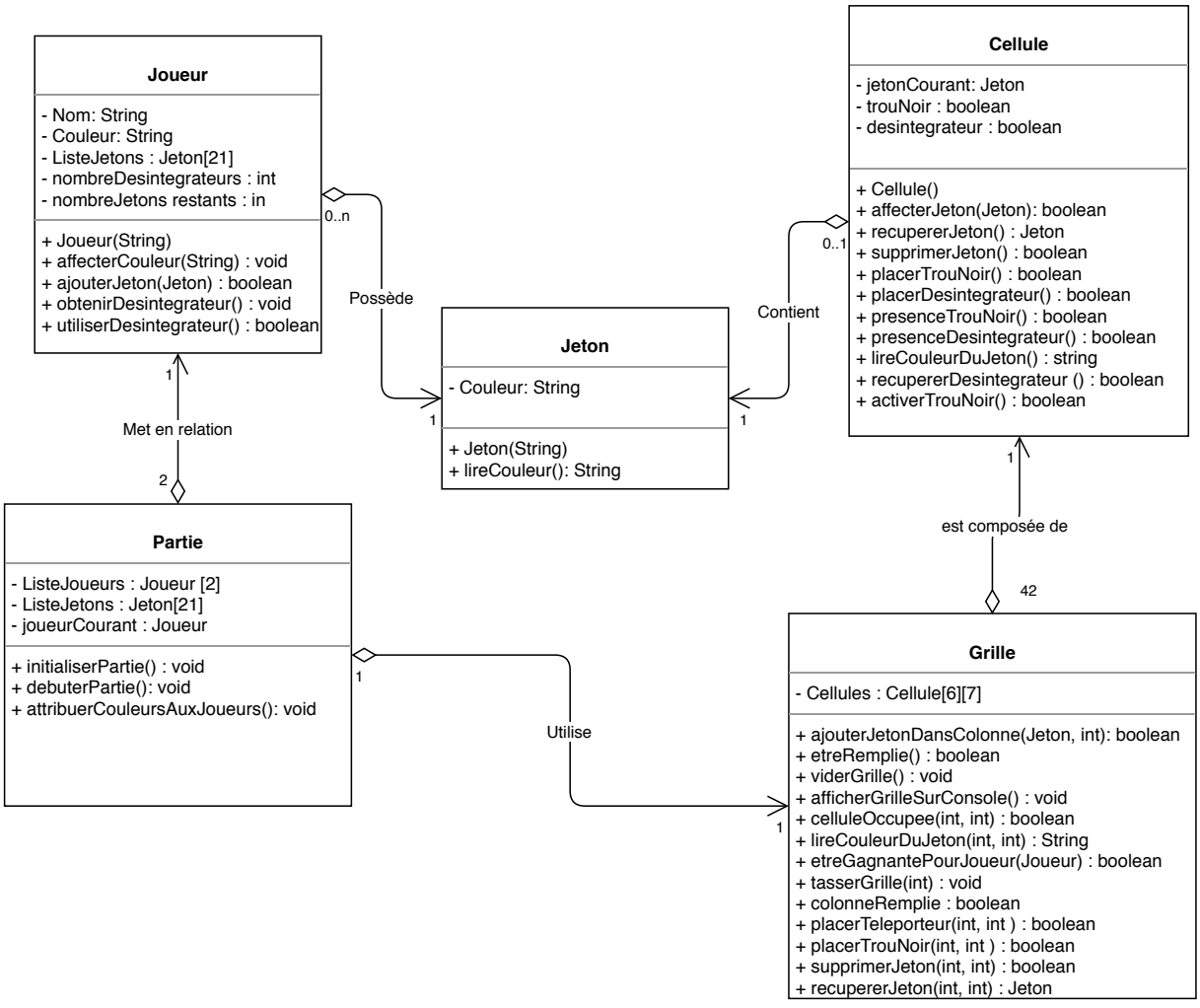
1. Une classe **Joueur**, qui décrivant un joueur. Un joueur possède notamment un nom, est associé à une couleur, et dispose au début de la manche d'un ensemble de jetons de sa couleur.
2. Une classe **Jeton**, décrivant un jeton, essentiellement représenté par un attribut couleur.
3. Une classe **Grille**, modélisant la grille de jeu de 6 lignes par 7 colonnes. Cette grille est composée d'un tableau de cellules, chaque cellule étant un objet de type Cellule. La classe Grille possède l'ensemble des méthodes permettant de vérifier un coup par rapport à l'état de la grille, et également des méthodes permettant d'initialiser la grille en plaçant des téléporteurs et trous noirs.
4. Une classe **Cellule** qui décrit une des 42 cellules de la grille. C'est dans cette cellule qu'il est possible de placer un Jeton, un téléporteur et/ou un trou noir. Cette classe possède donc les méthodes permettant la gestion de ces entités.
5. Une classe **Partie**, liant les objets entre eux : Elle contient notamment les deux joueurs qui s'affrontent, et une grille de jeu. Elle possède l'ensemble des méthodes permettant de vérifier si le coup d'un joueur est valide, et s'appuie notamment sur les méthodes proposées

par la classe Grille. Elle permet aussi de lancer la partie après avoir initialisé la grille et autres éléments.

Une liste exhaustive des attributs et des méthodes **nécessaires** de chaque classe est présentée dans le diagramme des classes. Par soucis de clarté, les noms utilisés sont explicites et la définition donnée pour lever toute ambiguïté.

Il est laissé au développeur le choix d'ajouter ses propres méthodes et attributs, sous réserve que ces derniers aient une cohérence avec le modèle et apportent une facilité d'implémentation.

3.2.2 Diagramme des classes



3.2.3 Description succincte des attributs et méthodes

Classe joueur :

Attributs :

- Nom : nom du joueur
- Couleur : couleur affectée au joueur
- ListeJetons : tableau décrivant les jetons encore en possession du joueur
- nombreDesintegrateurs : nombre de désintégrateurs actuellement en possession
- nombreJetonsRestant : nombre de jetons restant en possession du joueur, correspondant à la taille effective de ListeJetons

Méthodes :

- Joueur (String) : constructeur initialisant le nom du joueur avec son paramètre
- affecterCouleur(String): affecte la couleur en paramètre au joueur
- ajouterJeton(Jeton) : ajoute le jeton passé en paramètre à la liste des jetons
- obtenirDesintegrateur() : incrémente le nombre de désintégrateurs du joueur
- utiliserDesintegrateur() : décrémente le nombre de désintégrateurs et confirme l'utilisation de ce dernier, ou renvoie faux s'il ne restait plus de désintégrateurs.

Classe Jeton :

Attributs :

- Couleur : couleur affectée au joueur

Méthodes :

- Jeton (String) : constructeur initialisant la couleur du jeton avec le paramètre
- lireCouleur():renvoie la couleur du jeton

Classe Cellule:

Attributs :

- jetonCourant :référence vers le jeton occupant la cellule, ou null
- trouNoir : indique ou non la présence d'un trou noir
- desintegrateur : indique ou non la présence d'un désintégrateur

Méthodes :

- + Cellule() : construteur initialisant les attributs avec des valeurs par défaut
- + affecterJeton(Jeton): ajoute le jeton en paramètre à la cellule, et retourne vrai si l'ajout s'est bien passé, ou faux sinon (ex : jeton déjà présent)
- + recupererJeton() : renvoie une référence vers le jeton de la cellule
- + supprimerJeton() : supprime le jeton et renvoie vrai si la suppression s'est bien passée, ou faux autrement (ex : pas de jeton présent)
- + placerTrouNoir() : ajoute un trou noir à l'endroit indiqué et retourne vrai si l'ajout s'est bien passé, ou faux sinon (exemple : trou noir déjà présent)
- + placerDesintegrateur() : ajoute un désintégrateur à l'endroit indiqué et retourne vrai si l'ajout s'est bien passé, ou faux sinon (exemple : désintégrateur déjà présent)
- + presenceTrouNoir() : renvoie vrai si un trou noir est présent sur la cellule
- + presenceDesintegrateur() : renvoie vrai si un desintégrateur est présent sur la cellule
- + lireCouleurDuJeton() : renvoie la couleur du jeton occupant la cellule
- + recupererDesintegrateur () : supprime le désintégrateur présent de la cellule, et renvoie vrai, ou faux sinon (exemple : pas de désintégrateur présent)
- + activerTrouNoir() : active le trou noir : le trou noir engloutit le jeton et disparait. Retourne vrai si tout s'est correctement déroulé, ou faux sinon (pas de trou noir)

Classe Grille:

Attributs :

- Cellules: grille de 42 cellules

Méthodes :

- + ajouterJetonDansColonne(Jeton, int): ajoute le jeton dans la colonne ciblée, sur la cellule vide la plus basse. Renvoie faux si la colonne était pleine.
- +etreRemplie() : renvoie vrai si la grille est pleine
- + viderGrille() : vide la grille
- + afficherGrilleSurConsole() : fonction d'affichage de la grille sur la console. Doit faire apparaitre les couleurs, et les trous noirs.

- + celluleOccupee(int, int) : renvoie vrai si la cellule de coordonnées données est occupée par un jeton.
- + lireCouleurDuJeton(int, int) : renvoie la couleur du jeton de la cellule ciblée.
- + etreGagnantePourJoueur(Joueur) : renvoie vrai si la grille est gagnante pour le joueur passé en paramètre, c'est-à-dire que 4 pions de sa couleur sont alignés en ligne, en colonne ou en diagonale.
- + tasserGrille(int) : lorsqu'un jeton est capturé ou détruit, tasse la grille en décalant de une ligne les jetons situés au dessus de la cellule libérée.
- + colonneRemplie : renvoie vrai si la colonne est remplie (on ne peut y jouer un Jeton)
- + placerTrouNoir(int, int) : ajoute un trou noir à l'endroit indiqué et retourne vrai si l'ajout s'est bien passé, ou faux sinon (exemple : trou noir déjà présent)
- + placerDesintegrateur(int,int) : ajoute un désintégrateur à l'endroit indiqué et retourne vrai si l'ajout s'est bien passé, ou faux sinon (exemple : désintégrateur déjà présent)
- + supprimerJeton(int, int) : supprime le jeton de la cellule visée. Renvoie vrai si la suppression s'est bien déroulée, ou faux autrement (jeton absent)
- + recupererJeton(int, int) : enlève le jeton de la cellule visée et renvoie une référence vers ce jeton.

Classe Partie:

A ttributs :

- - ListeJoueurs : tableau des deux joueurs de la partie
- - joueurCourant : désigne le joueur courant à tout moment de la partie

Méthodes :

- + attribuerCouleursAuxJoueurs(): attribue des couleurs aux joueurs
- + initialiserPartie() : crée la grille, la vide si elle existait déjà, place les trous noirs et le téléporteurs, crée les jetons et les attribue aux aux joueurs correspondants.
- + debuterPartie(): lance la partie