Documentation du Projet Scrabble en Réseau

Architecture Java

7 avril 2025

Table des matières

1 Introduction

Ce document présente l'architecture et le fonctionnement d'un jeu de Scrabble en réseau développé en Java. Le projet utilise une architecture client-serveur avec une interface graphique JavaFX et suit le patron de conception MVC (Modèle-Vue-Contrôleur).

2 Architecture Globale

Le projet est structuré selon le patron de conception MVC (Modèle-Vue-Contrôleur) :

1. **Modèle** (package `modele`) : contient les classes qui représentent les données et la logique métier du jeu
2. **Vue** (package `vue`) : contient les classes d'interface utilisateur qui affichent le jeu
3. **Contrôleur** (package `controleur`) : contient les classes qui font le lien entre le modèle et la vue
4. **Réseau** (package `reseau`) : gère la communication client-serveur
5. **Utilitaires** (package `utils`) : contient des classes utilitaires

3 Rôles Principaux des Classes

3.1 Classes de Lancement

— `ScrabbleClientApp` : Point d'entrée de l'application client

— `ScrabbleServerApp` : Point d'entrée du serveur

— `ScrabbleApp` : Classe principale JavaFX qui démarre l'interface graphique

3.2 Modèle

— `Partie` : Gère la logique d'une partie (règles, tours, fin de partie)

— `Plateau` : Représente le plateau de jeu 15×15

— `Joueur` : Représente un joueur (nom, score, lettres en main)

— `Lettre` : Représente une lettre (caractère, valeur en points)

— `Case` : Représente une case du plateau (position, lettre, bonus)

— `Sac` : Gère le sac de lettres et la distribution

— `Dictionnaire` : Vérifie si les mots existent
— `Direction` : Énumération (HORIZONTAL, VERTICAL)

3.3 Contrôleur

— `ClientControleur` : Contrôleur principal qui coordonne tout côté client
— `PlateauControleur` : Gère les interactions avec le plateau
— `LettresControleur` : Gère les interactions avec les lettres du joueur

3.4 Vue

— `PlateauVue` : Affiche le plateau de jeu
— `MainJoueurVue` : Affiche les lettres du joueur
— `ScoreVue` : Affiche les scores et informations de tour
— `SacInfoVue` : Affiche les informations sur le sac de lettres
— `ConnexionVue` : Écran de connexion au serveur
— `DirectionDialog` : Boîte de dialogue pour choisir la direction du mot

3.5 Réseau

— `ScrabbleServer` : Serveur qui gère les connexions et la partie côté serveur
— `ScrabbleClient` : Client qui communique avec le serveur
— `ClientHandler` : Gère un client connecté (thread côté serveur)

3.6 Utilitaires

— `ConfigurationJeu` : Gère les paramètres de configuration
— `ObservateurPartie` : Interface pour le pattern Observer

4 Flux de Fonctionnement

1. L'utilisateur lance `ScrabbleClientApp` qui démarre `ScrabbleApp`
2. L'écran de connexion apparaît (`ConnexionVue`)
3. Après connexion, l'interface de jeu est créée avec le plateau et les contrôles
4. `ClientControleur` communique avec le serveur via `ScrabbleClient`

5. Le serveur (`ScrabbleServer`) gère la partie et coordonne les clients
6. Quand un joueur place un mot :
   - Le drag-and-drop est géré par `PlateauVue`
   - La validation est faite par `PlateauControleur`
   - La mise à jour est envoyée au serveur via `ScrabbleClient`
   - Le serveur (`ScrabbleServer`) vérifie et met à jour tous les clients

L'utilisation du pattern Observer (`ObservateurPartie`) permet aux différentes parties de l'application de réagir aux changements d'état de la partie.

Les scripts de lancement (`lancer_scrabble.bat/.sh`) permettent de démarrer facilement le serveur ou le client, tandis que `recompiler_scrabble.bat` permet de recompiler le code source.

5 Ordre d'Exécution d'une Partie

5.1 Démarrage du Serveur

```
1. Ex cuter le script lancer_scrabble.bat (Windows) ou
   lancer_scrabble.sh (Linux/Mac)
2. S lectionner l'option 1 "Lancer le serveur"
3. Indiquer le nombre de joueurs (2-4)
```

Techniquement, cela lance `ScrabbleServerApp` qui crée une instance de `ScrabbleServer` sur le port 5000 par défaut.

5.2 Démarrage des Clients

Pour chaque joueur :

```
1. Ex cuter le script lancer_scrabble.bat (Windows) ou
   lancer_scrabble.sh (Linux/Mac)
2. S lectionner l'option 2 "Lancer un client"
3. Saisir l'adresse du serveur (localhost si sur la m me
    machine)
```

Techniquement, cela lance `ScrabbleClientApp` qui démarre l'application JavaFX `ScrabbleApp`.

5.3 Connexion des Clients

Pour chaque client :

1. La classe `ScrabbleApp` affiche l'écran de connexion (`ConnexionVue`)
2. Le joueur saisit son nom et se connecte
3. `ClientControleur` établit la connexion via `ScrabbleClient`
4. Le serveur (`ScrabbleServer`) associe le joueur à un `ClientHandler`

5.4 Démarrage de la Partie

Une fois que tous les joueurs sont connectés :

1. Le serveur (`ScrabbleServer`) initialise une instance de `Partie`
2. Le serveur distribue les lettres initiales aux joueurs depuis le `Sac`
3. Le serveur envoie un message "PARTIE_COMMENCE" à tous les clients
4. Les clients affichent l'interface de jeu

5.5 Déroulement du Jeu

Pour chaque tour :

1. Le serveur indique quel joueur doit jouer (`Partie.joueurActuel`)
2. Le joueur actif voit ses contrôles activés dans `MainJoueurVue`
3. Le joueur place des lettres sur le plateau via `PlateauVue` (drag & drop)
4. Quand le joueur clique sur "Valider mot" :
   - `PlateauControleur.validerMotPlace()` est appelé
   - Le contrôleur vérifie la validité du mot avec `Dictionnaire`
   - `ScrabbleClient` envoie la commande "PLACER_MOT" au serveur
5. Le serveur (`ScrabbleServer`) :
   - Vérifie la validité du mot
   - Calcule les points
   - Met à jour l'état du plateau
   - Passe au joueur suivant
   - Envoie les mises à jour à tous les clients
6. Les clients mettent à jour leurs vues

5.6 Fin de Partie

La partie se termine quand :

— Un joueur a placé toutes ses lettres et le sac est vide

— Un joueur clique sur "Terminer la partie"

Dans les deux cas :

1. Le serveur calcule les scores finaux
2. Le serveur détermine le vainqueur
3. Le serveur envoie un message "FIN_PARTIE" à tous les clients
4. Les clients affichent un message de fin de partie

5.7 Mode Local (pour Test)

Pour tester en local sur une seule machine :

```
1. Exécuter le script lancer_scrabble.bat (Windows) ou
   lancer_scrabble.sh (Linux/Mac)
2. Sélectionner l'option 3 "Lancer le tout (mode local)"
```

Cela lance automatiquement un serveur et deux clients sur la même machine.

5.8 En Cas de Modification du Code

Si vous modifiez le code source, utilisez `recompiler_scrabble.bat` pour le recompiler avant de lancer le jeu.

6 Schéma d'Architecture

7 Conclusion

Ce projet est bien structuré selon les principes de conception modernes, avec une séparation claire des responsabilités et une bonne architecture client-serveur. Le pattern MVC facilite la maintenance et l'évolution du code, tandis que le pattern Observer permet une communication fluide entre les différentes parties de l'application.

La communication réseau est gérée efficacement par les classes dédiées, permettant une expérience de jeu multi-joueurs fluide. Les scripts de lancement facilitent l'utilisation du jeu par les utilisateurs finaux.

Architecture MVC du Projet Scrabble		
Modèle	**Vue**	**Contrôleur**
Partie	PlateauVue	ClientControleur
Plateau	MainJoueurVue	PlateauControleur
Joueur	ScoreVue	LettresControleur
Lettre	SacInfoVue	
Case	ConnexionVue	
Sac	DirectionDialog	
Dictionnaire		
Direction		
Réseau		
ScrabbleServer, ScrabbleClient, ClientHandler		
Utilitaires		
ConfigurationJeu, ObservateurPartie		

FIGURE 1 – Architecture du projet Scrabble en réseau