Mini-projet n° 1 :

Développement de l'architecture de : « Jeu de Bingo »

A. Description

Vous êtes en charge de développer un jeu interactif appelé « **Bingo** ». Le jeu repose sur le concept de prédiction des boules sortant d'une urne, similaire à une urne contenant 10 boules numérotées de 0 à 9. Chaque réponse correcte permet d'augmenter le score du joueur, avec un maximum de 10 tentatives par partie. De plus, les joueurs peuvent consulter les meilleurs scores.

B. Règles du jeu :

1. Le joueur doit prédire les numéros des boules sortant d'une urne, numérotées de 0 à 9, lors d'une série de tentatives.
2. Chaque partie consiste de 10 tentatives pour chaque joueur.
3. Le score du joueur augmente d'un point à chaque prédiction correcte.
4. Le score maximal atteignable est de 10 points pour une partie et le joueur reçoit un score x/10, où x est le nombre de prédictions correctes.
5. Les joueurs ont la possibilité de consulter les meilleurs scores précédemment enregistrés.
6. Plusieurs joueurs peuvent jouer simultanément.
7. Le tirage de l'urne se fait de manière sans remise, c'est-à-dire que si je sors une boule, je ne la remets pas.
8. Le joueur peut accéder à un menu avec les options suivantes :
    - **Option 1 :** Jouer BINGO.
    - **Option 2 :** Connaître le meilleur score.
    - **Option 3 :** Quitter.

C. Objectif de joueur

Développer une architecture Java en trois couches (Serveur d'Application, Gateway, Client) pour implémenter les règles du jeu Bingo tout en respectant les règles du jeu.

D. Exigences Techniques

I. Couche Serveur d'Application

1. La première couche du serveur d'application doit gérer la logique métier du jeu « Bingo ».
2. Dans l'implémentation du serveur d'application, vous serez à appliquer le « RMI » puis le « RPC ».
3. Elle doit inclure la logique du tirage du nombre dans l'urne, ainsi que la vérification des réponses des clients.
4. Pour le suivi du meilleur score, vous ajoutez l'enregistrement et la mise à jour de celui-ci.
5. La couche métier doit communiquer efficacement avec la couche Gateway pour transmettre les résultats et les informations du jeu.

II. Couche Gateway (Middleware)

- La couche **Gateway** doit être conçue pour gérer plusieurs clients simultanément à l'aide du multithreading.

III. Couche Client

1. La couche client doit contenir la logique de communication avec la couche Gateway du serveur comme l'indique la figure 1 ci-dessous.
2. Gestion des entrées/sorties utilisateur, conversion des données, et envoi/réception des informations.
3. Affichage des informations du jeu, des résultats de manière claire et compréhensible.

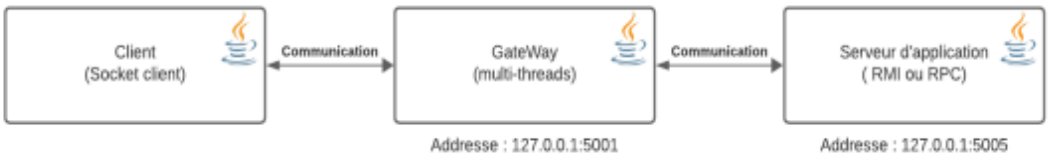

Figure 1 : La logique de communication avec la couche Gateway du serveur.

Annexe :

Génération du 4 chiffres aléatoires :

- **int** randomNumber = **new** Rando ().nextInt (10000);
- **String** number = **String**.format ("%04d", randomNumber);

HashMap : Fonctions

- map.containsKey (clientId)
- map.put (clientId, object)
- map.get (clientId)
- HashMap <String, Object> map = new HashMap<> ();

Random: Fonctions

- map.containsKey (clientId)
- map.put (clientId, object)

String: Fonctions

- String [] strElements = messages.split (“#”);

Figure 2 : Fonctions pour le développement Java de l’architecture Client / Serveur.

Chargés de TP :

- Mohamed Amine GUESMI
- Amen BIDANI

Chargé de cours et Responsable pédagogique :

- Hatem BEN STA