Compte Rendu

TP5-1

Nour Bouguerra

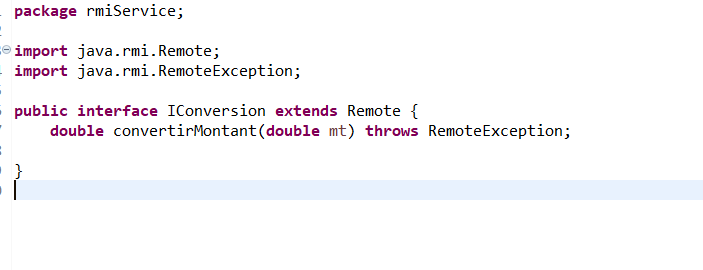
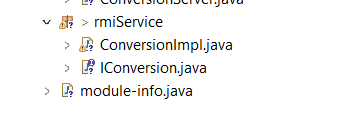
LSI3 1TP2

Description de la 2ème phase : Préparation du Serveur

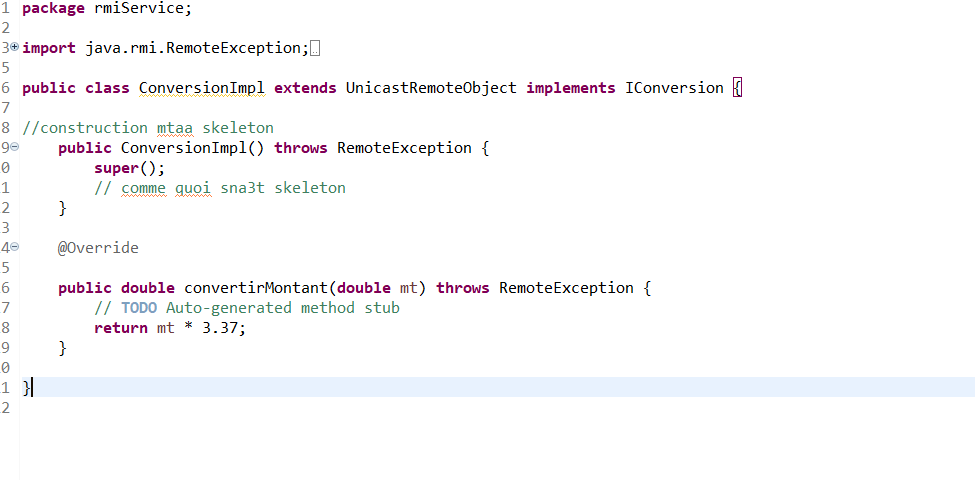
1. Créez un nouveau Projet Java en acceptant les réglages proposés par défaut sous le nom RMIConversion\_Server



2. Dans le répertoire src, le package : "**rmiService**", dans lequel on a créé une Interface qui porte le nom : **IConversion**". Cette interface définit la méthode **double convertirMontant(double mt) .**

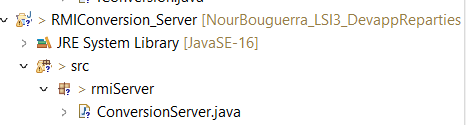


* Cette interface étend la classe **Remote**, qui est une interface spéciale permettant le fonctionnement des appels de méthode à distance, typiquement utilisée dans la technologie RMI (Remote Method Invocation), elle a une méthode unique **convertirMontant(double mt)** qui prend un montant en euros (**mt**) en paramètre et renvoie un montant converti en dinars. La déclaration **throws RemoteException** indique que cette méthode peut générer des exceptions liées à la communication à distance.
* Dans le même package on a créé la classe **ConversionImpl** et qui implémente l’interface **IConversion**.



* La classe **ConversionImpl** hérite de **UnicastRemoteObject**, qui est une classe fournie par Java pour faciliter la création d'objets distribués. Cette classe est utilisée pour créer un "skeleton" (structure de base) pour l'objet distant.
* Le constructeur **ConversionImpl()** déclare qu'il peut lancer une **RemoteException** (liée aux opérations à distance). Dans le constructeur, **super()** est appelé pour initialiser la partie du mécanisme de communication côté serveur.
* La méthode **convertirMontant(double mt)** est implémentée pour effectuer la conversion d'euros en dinars. Actuellement, elle multiplie simplement le montant en euros par 3.37 pour obtenir le montant équivalent en dinars.
* La classe **ConversionImpl** est la partie serveur de l'application, qui sera utilisée pour traiter les requêtes de conversion des clients.

4. Dans le répertoire src, créez le package : "rmiServer", dans lequel vous êtes demandés de créer une classe qui porte le nom : ConversionServer" et cochez « public static void main() ».



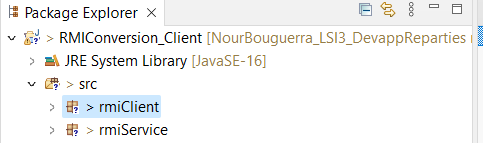


1. **Registre RMI :**
   * **LocateRegistry.createRegistry(1099);** : Crée un registre RMI sur le port 1099, offrant un emplacement où les objets distants peuvent être enregistrés pour que les clients puissent y accéder.
2. **Création de l'objet distant :**
   * **ConversionImpl od = new ConversionImpl();** : Instancie un objet **ConversionImpl**, agissant comme l'objet distant, fournissant la logique de conversion d'euros en dinars aux clients.
3. **Enregistrement de l'objet distant :**
   * **Naming.rebind("rmi//localhost:1099/OD", od);** : Enregistre l'objet distant dans le registre RMI sous le nom "OD" sur le port 1099, permettant aux clients de le trouver et d'y accéder pour effectuer des conversions.

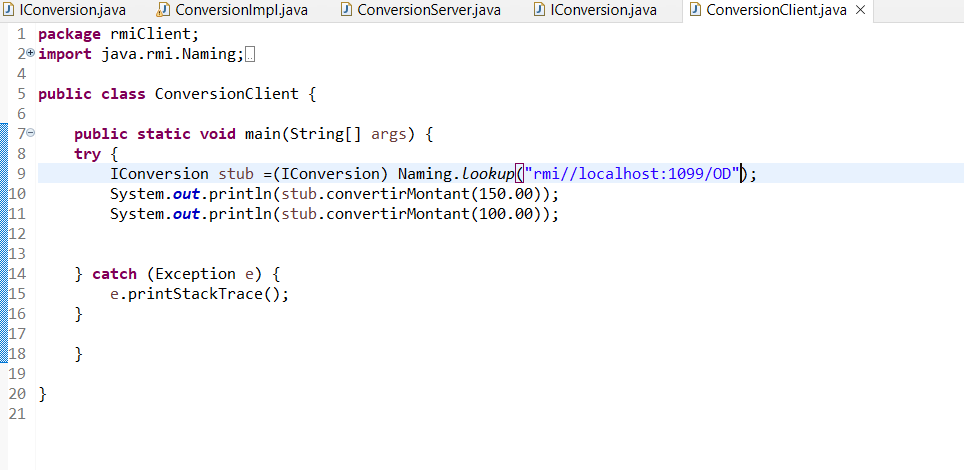
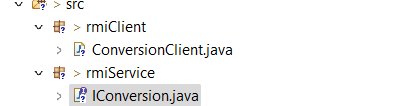
=>Le serveur est configuré pour répondre aux demandes de conversion d'euros en dinars. Une fois actif, il attendra les requêtes des clients et leur fournira le service de conversion à distance.

Description de la 3ème phase : Préparation du Client

un nouveau Projet Java en acceptant les réglages proposés par défaut sous le nom RMIConversion\_Client.



Dans le répertoire src, créez le package : "rmiClient", dans lequel on crée une classe qui porte le nom : "ConversionClient".



Le client RMI (**ConversionClient**) qui se connecte au serveur pour effectuer des conversions d'euros en dinars. Voici ce qu'il accomplit :

* **Recherche de l'objet distant :**
  + **IConversion stub =(IConversion) Naming.lookup("rmi//localhost:1099/OD");** : Utilise **Naming.lookup** pour rechercher l'objet distant enregistré sous le nom "OD" sur le port 1099 du serveur RMI situé sur la machine locale (localhost). Une fois trouvé, il retourne une référence à cet objet distant.
* **Appel de la méthode distante :**
  + **System.out.println(stub.convertirMontant(150.00));** : Appelle la méthode **convertirMontant** sur l'objet distant avec un montant de 150 euros, affichant le montant équivalent en dinars retourné par le serveur.
  + **System.out.println(stub.convertirMontant(100.00));** : De même, effectue une autre conversion avec un montant de 100 euros et affiche le résultat.

=>Ce client se connecte au serveur, recherche l'objet distant enregistré sous le nom "OD" et utilise ses méthodes pour effectuer des conversions. Les résultats sont affichés sur la console.

=>Une fois le serveur RMI en cours d'exécution, ce client peut être lancé pour effectuer des conversions en utilisant les méthodes fournies par l'objet distant sur le serveur.

EXECUTION



Je sais pas ou est le problème même au séance de TP madame hajer a voir le code et elle m’a dit qu’il est correct aussi de rouvert l eclipse mais la même chose se répète

**Synthèse**

1. **Phase 1 :** Création de la structure de travail
   * Création d'un répertoire pour le projet.
   * Mise en place d'un document Word pour documenter le processus avec des captures d'écran.
2. **Phase 2 :** Préparation du Serveur
   * Création d'une interface **IConversion** avec une méthode de conversion d'euros en dinars.
   * Implémentation de cette interface dans une classe **ConversionImpl**, définissant la logique de conversion.
   * Création d'un serveur (**ConversionServer**) qui enregistre et héberge l'objet distant pour les conversions.
3. **Phase 3 :** Préparation du Client
   * Mise en place d'un client (**ConversionClient**) qui se connecte au serveur pour effectuer des conversions.
   * Recherche de l'objet distant via RMI et appel de la méthode de conversion pour obtenir les résultats

Lien git :

https://github.com/Nour-23/TPREPARTIE