Programmation distribuée 2010-2011
DUT Informatique Nancy 2A

Projet RMI – Blaguemule – part 2
Architecture pair à pair

1 Préalable

Créez tout d'abord un nouveau projet dans lequel vous pourrez faire une nouvelle application (inspirée de l'application de la partie 1).

On souhaite conserver la structuration en répertoires bin/src/lib(evetuellement) et en packages.

2 Fonctionnement général

L'objectif de cette partie est de faire communiquer les **BlagueProvider** selon une architecture pair à pair. Chaque **BlagueProvider** est serveur et client à la fois. Il peut récupérer les blagues des autres serveurs et met à disposition toutes les blagues qu'il connaît (contenant éventuellement celles qu'il a déjà récupérées).

Désormais, il n'y aura plus de **BlagueProviderClient** et **BlagueProviderServeur.** Il n'existera qu'une application **BlagueProvider** qui fera office de client et de serveur.

Au lancement du **BlagueProvider**, on passera comme paramètre le nom du **BlagueProvider** et les noms sous lesquels sont référencés les autres **BlagueProvider** déjà enregistrés.

3 Interface BlagueProviderPairApair

L'interface **BlagueProviderPairAPair** fournit l'interface sur les parties client et serveur et dispose des méthodes

- **String getNom()** retournant le nom du **BlagueProvider**
- **String[] getAllNAmes()** retournant la liste des noms des blagues connues par le BlagueProviderPairAPair
- **Blague getBlague(String nom)** retournant la blague ayant pour nom le String passé en parametre.

4 Classe BlagueProvider

La classe **BlagueProvider** implémente l'interface **BlagueProviderPairAPair**

Cette Classe dispose en outre

- des attributs

    - **nom** le nom du **BlagueProvider**
    - une **hashmap** contenant les blagues
    - une hashmap **listeRef** associant le nom des autres **BlagueProvider** connus à leurs références distantes (ces références sont à récupérer au lancement de l'application)
- des méthodes
    - **ajoutBlague(Blague b)** permettant d'ajouter une blague en local
    - **getAllNames** permettant de récupérer la liste des noms des blagues d'un serveur donné à partir de sa référence distante
    - **telechargeBlague** qui permet de télécharger une blague en l'ajoutant aux blagues référencées à partir d'une référence distante et du nom de la blague

Question a :
Écrire la classe **BlagueProvider.**

Question b
Quel est le codebase nécessaire ? Générez le fichier .jar correspondant.

5 Main

Le main de la classe **BlagueProvider** a pour objectif de remplir la liste **listeRef**, de créer une interface graphique et de s'enregistrer auprès du rmiregistry.

Le main doit fonctionner de la manière suivante

- création du **BlagueProvider** ayant pour nom le premier argument passé dans le main.
- ajout de la blague en fonction du nom du **BlagueProvider** (pour avoir des serveurs avec des contenus différents)
- récupération du registry
- récupération des références distantes (les autres cases du tableau args)
- création de la référence distance et enregistrement au registry (référencement au nom du **BlagueProvider**)

Il est nécessaire que

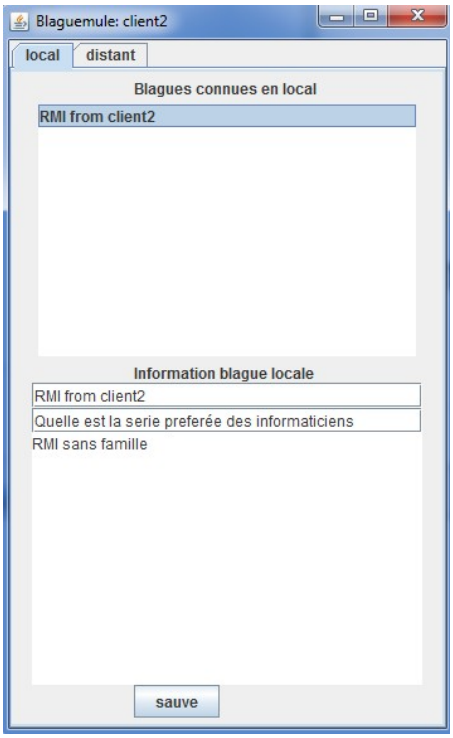
- tous les clients soient lancés sur la même machine (on ne peut pas faire de rebind sur un rmiregsitry non local pour des raisons de sécurité)
- les clients reçoivent en paramètre les noms de clients déjà lancés
- les clients soient lancés dans le bon ordre (conformément aux paramètres passés)

Question e :
Écrire le main sans vous préoccupez de l'interface graphique. Vérifier que tout fonctionne correctement à l'aide de tests adaptés.

6 Interface graphique

On vous fournit une interface graphique constituée de deux onglets.

6.1 Onglet local



Le premier onglet est lié à l'aspect local. Il contient

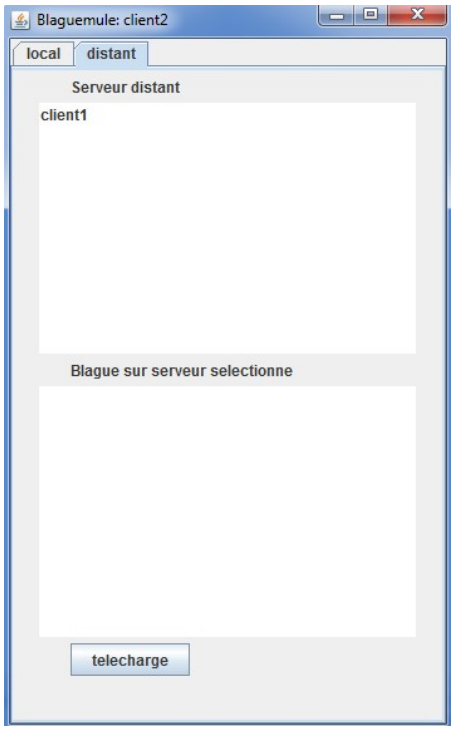
- la liste des blagues en local
- des Jtext permettant d'afficher la blague locale sélectionnée et de la modifier
- un bouton 'sauve' permettant de sauver ces modifications (éventuellement en ajoutant une nouvelle blague)

Cet onglet est créé dans la méthode ongletLocal() de la classe générant l'interface graphique. Il repose sur les composants suivants déclarés comme attributs de l'interface

- une Jlist blaguesLocales contenant les blagues locales
- différents JtextField et JtextArea pour la partie visualisation/modification

A vous d'ajouter les listeners pour avoir un application qui fonctionne correctement (mise à jour de l'affichage de la blague selectionnée, click sur le bouton de sauvegarde, …)

6.2 Onglet distant



Le second onglet est lié à l'aspect distant. Il contient

- la liste des autres blagueprovider référencés
- la liste des blagues disponible sur le serveur distant sélectionné
- un bouton permettant de télécharger la blague sélectionnée et de l'ajouter à ses blagues locales

Cet onglet est crée dans la méthode **ongletDistant()**. Il repose sur les composants suivants:

- une Jlist serveurs contenant la liste des noms des serveurs connus.
- Une Jlist blaguesDistantes contenant la liste des blagues du serveur sélectionné dans la liste précédente.

De la même manière, c'est à vous d'ajouter les listener qui vont bien (mise à jour des blagues distantes en fonction de la sélection du serveur, …)

6.3 Comportement général

Concernant la partie distante

- quand on clique sur un serveur, la liste des blagues disponibles sur le serveur se met à jour
- quand on sélectionne une blague et qu'on clique sur le bouton télécharger, la blague est ajoutée à la liste des blagues locales

Concernant la partie locale

- quand on sélectionne une blague, celle-ci s'affiche dans la partie information (des JtextArea modifiables)
- quand on clique sur le bouton sauvegarder, on ajoute une nouvelle blague correspondant aux informations des JTextArea, cette blague est ajoutée à la liste au dessus et est instantanément disponible pour les autres serveurs.

Question d :
Créez l'interface graphique

On utilisera la clase **JList** pour gérer les listes. Pour modifier le contenu de la **JList**, il faut utiliser la méthode **setListData** et pour accéder au contenu sélectionné, utilisez la méthode **getSelectedValue**.

7 Exécution

Question e :
Testez que tout fonctionne en lançant plusieurs clients.