Kévin MAUGE
kmauge@etud.u-pem.fr
Pape NDIAYE
pndiaye@etud.u-pem.fr

Master Informatique
M1-S2
Groupe : 2

Projet
Programmation réseau

Matou - Service de Chat

Manuel utilisateur
- 01/05/2016 -

Compilation

Pour compiler le projet, il est nécessaire d'utiliser l'outil Apache ANT. Le projet dispose d'un fichier "build.xml" qui possède plusieurs cibles exécutables :

- **Cible par défaut :**
  - Compile les sources, génère les archives JAR exécutables et génère la javadoc.
- **Cible "init" :**
  - Crée les répertoires nécessaires à l'exécution des autres tâches.
- **Cible "clean" :**
  - Nettoie les fichiers générés par les exécutions précédentes de Ant.
- **Cible "compile" :**
  - Compile les sources.
- **Cible "jar"** :
  - Génère les archives JAR exécutables dans le répertoire courant.
- **Cible "javadoc" :**
  - Génère la javadoc.

Utilisation

Le programme nécessite la version 8 de Java. Il peut être lancé à partir de l'archive JAR exécutable et de la commande :

Exécuter le serveur

java -jar server.jar [options] **port**

- port : numéro du port d'écoute du serveur

Exécuter le client

java -jar client.jar [options] ***hostname port*** [*username*]

- *hostname* : adresse ou nom d'hôte du serveur
- *port* : numéro du port d'écoute du serveur
- *username* : pseudo à utiliser (facultatif)

Options de la ligne de commande

Ces options sont disponibles à la fois pour le serveur et pour le client :

- -logger *path* : rediriger la sortie normale du logger dans un fichier (au lieu du terminal).
- -exception *path* : rediriger la sortie des exception du logger dans un fichier (au lieu du terminal).

Commandes de chat

Une fois connecté et authentifié par le serveur, l'utilisateur peut interagir avec le programme en utilisant les commandes suivantes :

- *<message>* : envoyer un message
- **/open** *<pseudo>* : demander à ouvrir une connexion privé avec quelqu'un (il faut qu'il soit connecté actuellement)
- **/accept** *<pseudo>* : accepter d'ouvrir une connexion privée avec quelqu'un (il faut qu'il ait demandé la connexion au préalable)
- **/pv** *<pseudo>* *<message>* : envoyer un message privé à quelqu'un (il faut être connecté en privé avec cette personne au préalable)
- **/file** *<pseudo>* *<fichier>* : envoyer un fichier privé à quelqu'un (il faut être connecté en privé avec cette personne au préalable)
- **/close** *<pseudo>* : fermer la connexion privée avec quelqu'un (il faut être connecté en privé avec cette personne actuellement)
- **/exit** : quitter le chat (toutes les connexions actives seront fermées)

Fichier de configuration

Les fichiers de configurations sont chargés au lancement de l'application.
Si aucun fichier de configuration ne peut être chargé, alors c'est la configuration par défaut du code source qui sera appliquée à l'exécution.

Le caractère '#' est utilisé pour les commentaires dans le fichier de configuration. Tout ce qui suit ce caractère est alors ignoré lors du chargement du fichier.
Le caractère '=' est utilisé comme opérateur d'affectation dans le fichier de configuration.

Chaque champ doit être suivi du caractère d'affectation ainsi que d'un booléen "true" ou "false".

Champ(s) disponible(s) dans les fichier de configuration du client et du serveur :
- COLORATOR
- HEADER
- ERROR
- WARNING
- INFO
- DEBUG

Champ(s) disponible(s) dans les fichier de configuration du serveur uniquement :
- SELECT

Comportement par défaut

Si aucune option n'est fournie en ligne de commande : le logger utilise la sortie d'erreur standard (stderr) pour tous les événements.
Si aucun fichier de configuration n'est chargé : le logger n'affiche que les exceptions.

Le projet est livré avec des fichiers de configuration déjà préréglés aux valeurs ci-dessous. Ces valeurs peuvent être différentes des valeurs par défaut lorsqu'aucun fichier de configuration n'a pu être chargé.

Valeurs du fichier de configuration pour le client :

- COLORATOR=false
- HEADER=false
- ERROR=true
- WARNING=true
- INFO=false
- DEBUG=false

Valeurs du fichier de configuration pour le serveur :

- COLORATOR=false
- HEADER=false
- ERROR=true
- WARNING=true
- INFO=false
- DEBUG=false
- SELECT=true

Colorateur

Le colorateur utilise les séquences d'échappement ANSI du terminal.
Il fonctionne correctement sur la plupart des terminaux GNU/Linux et améliore la lisibilité du logger.

Le code couleur utilisé est le suivant :

- Rouge : erreur (problème critique)
- Jaune : avertissement (problème non critique)
- Vert : information sur les données échangées sur le réseau
- Violet : message de debug
- Bleu (serveur) : liste des clés surveillées par le sélecteur
- Cyan (serveur) : liste des clés sélectionnées par le sélecteur

Dysfonctionnements possibles :

Nous avons effectué plusieurs tests sur notre programme. Voici les fonctionnements anormaux que nous avons remarqué et que nous n'avons pas pu pleinement corriger :

- Si un client A est connecté sur la même machine X que le serveur et qu'un autre client B connecté sur une autre machine Y essaye d'établir une connexion privée avec A, la connexion privée ne pourra pas être toujours être établie.
    - Ce dysfonctionnement intervient uniquement lorsque le client A se connecte au serveur en utilisant l'adresse de rebouclage du serveur (localhost / 127.0.0.1 par exemple). Le serveur va identifier ce client A avec cette même adresse et l'enverra au client B. Mais comme cette adresse est spécifique à la machine X du client A, le client B ne pourra pas s'y connecter depuis sa machine Y.
- Un client A est connecté depuis une machine X et un client B est connecté depuis une machine Y. Le client A veut ouvrir une connexion privée avec le client B. Si un autre client C connecté depuis la machine Y connait les ports de connexion privée de A, alors il pourra se connecter à A en se faisant passer pour B.
    - En effet, les clients B et C sont sur la même machine donc ils seront potentiellement identifiés par la même adresse IP. C'est une problématique de sécurité, le mécanisme de vérification d'identité se limite aux adresses IP et à la transmission des numéros de port.
- Lorsqu'une instance de chat est interrompue, le programme ne s'arrête pas toujours immédiatement. Il faut parfois que l'utilisateur tape quelque chose sur le terminal pour que l'exécution s'arrête.
    - Ce dysfonctionnement est dû au fait que ShellInterface utilise un objet Scanner pour lire l'entrée standard. Lorsque le thread responsable d'envoyer les événements est bloqué sur l'attente d'un nouvel événement, le Scanner attend de lire une ligne sur l'entrée standard. Malheureusement cette méthode ne peut pas être interrompue, c'est pourquoi le thread continue son exécution.
- Lorsqu'un ferme le flux d'entrée standard du programme, toutes les instances de chat d'un ClientCore sont interrompues.
    - Bien que ça ressemble à un dysfonctionnement, c'est une situation normale. Si plusieurs ClientInstance sont lancées sur un même objet ClientCore, chaque session de chat utilisera la même interface utilisateur et donc le même terminal. Si le flux du terminal a été fermé brusquement, il est normal que toutes les sessions associées ne puissent poursuivre leur tâche.
- Les pseudos ne sont pas toujours insensibles à la casse.
    - Ce problème n'apparait qu'avec des lettres où les majuscules et les minuscules ne sont pas équivalentes (par exemple, l'alphabet géorgien). Il est dû au fait que les pseudos sont comparés entre eux par leur représentation en minuscule.