

BRUIT Gabriel gbruit@etud.u-pem.fr ou gbruit@hotmail.fr
NEANG Thomas tneang@etud.u-pem.fr ou thneang@gmail.com

Protocole pour Matou

Sommaire

1. Introduction.....	1
2. Fonctionnement général.....	1
3. Connexion au serveur.....	2
4. Connexion privé entre deux clients.....	2
5. Connexion pour envoie de fichier.....	4
6. Envoie de fichier.....	4
7. Envoie de message.....	5

1. Introduction

Nous présentons dans ce document comment le serveur et les clients échangent leurs informations pour réaliser le service de chat Matou.

Toutes les adresses sont en représentation textuelle, les formats sont IPv4 et IPv6.

2. Fonctionnement générale

Les échanges client/serveur dans ce protocole fonctionnent grâce à la structure générale suivante :

int int type de la donnée
id paquet taille des données(octet) données

L'id correspond à l'identifiant du paquet.

L'id à mettre et le type de donnée nécessaire seront détaillé plus bas dans la client et la partie serveur.

Il peut bien sûr y avoir plusieurs type de donnée lors d'un échange.

L'id permet de définir le traitement à adopter que ce soit pour le client ou pour le serveur. On peut ainsi savoir s'il s'agit d'une demande de connexion, de la réception d'un message, etc...

Un ByteBuffer de la bonne taille est alloué pour la réception des données.

Côté client :

- une connecté au serveur,
- une connecté en privé à un client.
- une connecté en privé au même client pour les transferts de fichier.

Côté serveur :

- Une socket personnelle (sa boîte au lettre)
- Autant de socket qu'il y a de client connecté pour le chat.

Nous abordons maintenant les différentes demandes que les clients/serveurs peuvent demander et recevoir.

3. Connexion au serveur

id 0 : Demande de connexion au serveur

```
-----  
int | int          | String  
0   |taille pseudo  | pseudo en utf-8  
-----
```

Le client peut demander une connexion au serveur pour rejoindre le chat grâce à l'id 0. Il doit envoyer la taille de son pseudo encodé en UTF-8 puis son pseudo qui permettra au serveur de vérifier sa disponibilité.

id 1 : Acceptation du serveur

```
-----  
|int| long  
|1  | clientID  
-----
```

id 2 : Refus du serveur

```
----  
|int|  
|2  |  
----
```

Le serveur envoie 1 au client avec un long généré aléatoirement coté serveur pour lui permettre de s'identifier si le pseudo choisi est disponible, il envoie 2 sinon.

4. Connexion privé entre deux clients (c1 veut se connecter à c2)

id 3 : Demande connexion du client c1 au serveur

int		int			String			long			int			String	
3		taille pseudo c1		pseudo c1 en utf-8		clientID		taille pseudo c2		pseudo c2 en utf-8					

Le client doit envoyer son pseudo et celui de la personne à qui il veut se connecter.
Le pseudo c1 permet à c2 de savoir qui veut se connecter et le pseudo c2 permet au serveur d'envoyer la demande à la bonne personne.

id 4 : Demande de connexion au client c2 par le serveur.

int		int			string		
4		taille pseudo c1		pseudo c1			

Le serveur demande ensuite à c2 s'il accepte la connexion de c1
id 5 : Acceptation par c2 de la demande du client c1.

int		int			String			long		int			String		int			String		
5		taille pseudo c1		pseudo c1		ClientID		taille addr		address		port		taille pseudo c2		pseudo c2				

Le client c2 accepte la demande et envoie son adresse avec le numéro de son port pour permettre au client c1 de se connecter. On redonne le pseudo de c1 pour permettre au serveur de transmettre la réponse.

id 6 : Refus demande du client c1.

int	int			String			long			int			String		
6		taille pseudo c1		pseudo c1		ClientID		taille pseudo c2		pseudo c2					

On refuse la demande de connexion du client c1, même remarque pour le pseudo.

id 7 : Notification d'acceptation au client

int	int			String			int		
7		taille address		address			port		

Si le client c2 accepte la demande de c1, le serveur transmet l'adresse et le port de c2 à c1.

id 8 : Notification de refus au client

```
|int|
| 8 |
```

Si le client c2 refuse la demande de c1, le serveur envoie l'entier 8 au client c1.
Attention lors d'une acceptation, il faut éviter le cas suivant :

```

      3      4
client c1 --->serveur --->client c2
      7      5
client c1 <---serveur <---client c2
      5      7
client c1---->serveur --->client c2 On doit s'arrêter ici, c'est au client de le gérer
      7      5
client c1 <---serveur <---client c2
      5      7
client c1---->serveur --->client c2
```

5. Connexion pour envoi de fichier

La connexion pour envoyer des fichiers doit se faire uniquement après la connexion des deux clients, la demande se fait donc sans passer par le serveur.

id 9 : Demande de connexion pour envoi de fichier

```
|int|int          | String    | int    |
| 9 |taille address c1| Address c1 | port c1|
```

id 10 : retour connexion pour de envoie fichier

```
|int |int          | String    | int    |
| 10 |taille address c2| Address c2 | port c2 |
```

Nous avons donc le schéma suivant :

```

      3      4
client c1 ->serveur ->client c2
      7      5
client c1 <-serveur <-client c2
      5      7
client c1 ->serveur ->client c2
      9
client c1 ----->client c2
      10
client c1 <-----client c2

```

6. Envoie de fichier

id 11 : Demande envoie de fichier. (Si c1 envoie a c2)

```

-----
|int | long      |int          | string  |
|11 | taille fichier |taille nom_fichier | Nom fichier|
-----

```

Le client c1 doit demander l'autorisation au client c2 avant d'envoyer un fichier.
Le nom et la taille du fichier est donné pour informer c2.

id 12 : Acceptation demande envoie de fichier

```

-----
|int|
|12|
-----

```

id 13 : Refus demande envoie de fichier

```

-----
|int |
|13 |
-----

```

Le client c2 peut accepter ou refuser avec l'id 12 ou 13.

id 14 : Envoie d'un fichier.

int	int	byte	
14	taille_fichier	donnée	

L'envoi du fichier se fait en précisant la taille des données.
Le choix de la taille max est laissé au client.

7. Envoie de message

id 15 : Envoie d'un message sur le chat.

Les messages vides ne sont pas tolérés afin d'éviter le flood sur le chat.

int	int	String	long	int	String	
15	taille pseudo	pseudo	ClientID	taille message	Message	

Chaque client envoie son message avec son propre pseudo pour être identifié sur le chat.
Le message est envoyé au serveur qui doit envoyer le message à tous les autres clients.
L'ID est envoyé pour vérifier l'identité du client auprès du serveur et éviter l'usurpation d'identité.

L'ID est retiré par le serveur après vérification puis envoyer aux autres clients.

id 15 : Envoie d'un message privé ou message transmis par le serveur.

int	int	String	int	String	
15	taille pseudo	pseudo	taille message	Message	

Ceci est le format des messages transmis et reçu aux clients