# Réseau 2012 - 2013

## Identification du groupe

Remplir le tableau avec le nom des membres, l'application qu'il réalise (serveur/client) et le langage utilisé (C, CPP, C#, java, php, ruby, ada, lisp, etc).

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Membre | Application | Langage |
| #1 | FRUTEAU Nicolas | Serveur | Java |
| #2 | RAUSSIN Clément | Client | Java |
| #3 | KHOULE Selle | Serveur | C++ |
| #4 | GABUTTO Grégory | Client | C++ |

## Protocole de transport

TCP multi-connexion synchrone

## Port utilisé pour communiquer

5678 TCP

## Organisation physique des types de base

Discriminant : char 8 bits

Booléen : 32bits entier bits, si = 0 -> false. Si c’est <> 0 -> true

Caractère : 8 bits

Entier court : entier 32bits

Entier long : entier long 64bits (int64 en cpp)

Flottant : float 32bits

Chaîne : longueur (4octet) +TEXTE

Tableau : longueur (4octet) + cellule de taille fixe

## Structure générale des messages

Les messages sont les suivants

### Messages clients vers serveurs et structure de ces messages

JOIN : enregistre client sur le serveur

Discriminent j

OK -> connexion réussi

KO -> - déjà connecté

- échec de la connexion

LEAVE : supprime le client du serveur

Discriminent q

OK -> client déconnecté du serveur

KO -> client non connecté

WIN : demande à la BRUTE Me de gagner

Discriminent+Int

OK -> Me gagne

KO -> - ME perds

- client non connecté

- Me ou OTHER non défini

LOOSE : demande à la BRUTE Me de perdre

Discriminent+Int

OK -> ME perds

KO -> - ME gagne

- non connecté

- ME ou OTHER non défini

GET\_ME : demande toute les propriétés de sa brute

Discriminent

OK -> propriété envoyé

KO -> - brute inexistante

- échec d’envoi

- non connecté

GET\_STATE : demande l’état du client

Discriminent

OK -> client est connecté

KO -> client non connecté

Pour ME, on choisi une BRUTE associé au login. Pour OTHER on choisi une BRUTE aléatoirement, on vérifie que ce ne soit pas la même que Me.

Discriminent+String

Me

OK -> la brute du client est trouvée

KO -> - la brute est inexistant

- si client non connecté

OTHER

OK -> brute adverse trouvé

KO -> - client non logué

- absence d’adversaire

- brute du client non choisit

### Message login

Le message login est composé …

OK -> login réussi

KO –> - login non reconnu

- déjà logué

- non connecté avec le serveur

## Messages serveurs vers client et structure de ces messages

### Message ok

Un message OK est composé d’un code de retour, et si ce dernier est 0, envoi de la réponse.

SEND\_ME : envoi de toutes les propriétés de la brute ME

SEND\_OTHER : envoie de toutes les propriétés de la brute OTHER

SEND\_WIN : envoie du message « You win »

SEND\_LOOSE : envoie du message du « You loose » au client

SEND\_CONNECT : accepte le client sur le serveur

SEND\_DISCONNECT : ferme la connexion du client et du serveur

## Sessions possibles

### Session login

Un message login est composé d’un nom d’utilisateur, et d’un mot de passe.

Discriminent+Tableau (String, String)

La session login débute par un client qui envoie le message login et reçoit le message ok si c'est bon ou le message ko si il y a une erreur.

GET\_LOGIN -> OK : envoi le login

KO : -login inconnu

-échec d’envoi

-utilisateur non logué