**Communication avec le serveur**

Le client nécessite les DLL SimpleChatClient.dll et SimpleChatFrameManager.dll

**Connection :**

1ère étape :

Il faut initialiser les piles d’émission et de réception et le socket TCP.

Pour ce faire il faut appeler la méthode statique :

**Client.Init(Uint16, Uint16) ;**

*Client.Init(600, 256) ;*

En paramètres, il faut passer la longueur des trames puis le nombre maximum de trames en attentes.

Si une nouvelle trame arrive alors que la pile de trames est pleine, cette dernière est rejetée.

2ème étape :

Il faut ensuite établir la connexion avec le serveur.

Il faut alors appeler la méthode statique :

**Client. EstablishConnection(IPAddress, int) ;**

*Client.EstablishConnection(ipServer, 9009) ;*

En paramètres, il faut passer l’adresse IP du serveur sous forme d’IPAddress et le port sous forme d’entier. La méthode retourne un booléen pour signifie l’état de la connexion.

3ème étape :

Pour terminer l’étape de connexion, il faut envoyer au serveur une requête d’identification.

Il faut pour cela créer une trame de type 100.

**byte[] frame = FrameManager.WrapFrame100(string, string) ;**

*byte[] frame = FrameManager.WrapFrame100(“monPseudo”, “monMotDePasse”) ;*

La méthode prend en paramètres un string pseudo, puis un string motDePasse.

Elle retourne un tableau de byte qui constitue la trame.

On peut ensuite l’envoyer au serveur.

**Client.SendFrame(byte[]) ;**

*Client.SendFrame(frame) ;*

La méthode renvoie un booléen signifiant la prise en considération ou non de la trame pour l’envoie. Si ce n’est pas le cas c’est que la pile des envois est pleine, il faut donc attendre un peu.

Après avoir reçu la trame d’identification, le serveur envoie une trame de réponse afin d’informer le client de l’acceptation ou non de sa demande.

**Communication :**

Pour envoyer un message au serveur, il faut créer une trame de type 110 :

**byte[] frame = FrameManager.WrapFrame110(string) ;**

*byte[] frame = FrameManager.WrapFrame110(“ceci est un message”);*

Puis l’envoyer :

**Client.SendFrame(byte[]) ;**

*Client.SendFrame(frame) ;*

Pour récupérer un message de la part du serveur, il faut décomposer une trame de type 10 :

D’abord il faut récupérer une trame dans la pile de réception (s’il y en a une)…

**Client.GetFrame(out byte[]) ;**

*Client.GetFrame(out frame) ;*

La méthode retourne true s’il y a une trame à traiter.

Ensuite pour connaitre le type de trame reçu :

**FrameManager.GetFrameType(byte[]) ;**

*FrameManager.GetFrameType(frame);*

La méthode retourne le type de la trame envoyé en paramètre (255 -> inconnu).

Si la trame est de type 10, alors on l’envoie dans la méthode UnwrapFrame10 :

**FrameManager.UnwrapFrame10(frame, out string, out string, out string, out string) ;**

*FrameManager.UnwrapFrame10(frame, out date, out time, out pseudo, out message) ;*

La méthode renvoie en out les composants de la trame.

**Type de trames :**

Il existe actuellement 7 types de trames…

Toutes font 600 octets

**Trames Serveur -> Client :**

La trame 0 qui est une réponse du serveur à une demande d’identification d’un client.

La trame 1 qui est une réponse du serveur à une demande d’enregistrement d’un client.

La trame 2 qui est un message système du serveur à destination des clients.

La trame 3 qui est la liste des utilisateurs.

La trame 10 qui est un message tchat du serveur à destination des clients.

**Trames Client->Serveur:**

La trame 100 qui est une demande d’identification d’un client vers le serveur.

La trame 101 qui est une demande d’enregistrement d’un client vers le serveur.

La trame 110 qui est un message tchat d’un client à destination du serveur pour que ce dernier la renvoie aux autres clients.

Le client reçoit des trames entre 0 et 99, et envoie celles entre 100 et 199.