Documentation : Class Net

Développé par : Innocent1

Version : 1.0

Langage : C++

Table des matières :

- Page 1 : Table des matières

- Page 2 : Valeurs spéciales, Constructeur, Getters, Setters

- Page 3 : Exemple Des Méthodes Basiques, Connexion et Déconnexion, Envoyer et Recevoir

- Page 4 : Exemples Des Autres Méthodes, Compilation

Important : Ne surtout pas oublier le cast des chaînes de caractères

→ `setName("Username")` devient `setName((char *)"Username")`

Les valeurs spéciales

→ NET_CONNECTION_ERROR_NO : Aucune erreur de connexion au serveur (= 4585)

→ NET_CONNECTION_ERROR_YES : Erreur de connexion (= 4586)

→ NET_CONNECTED : Connecté au serveur (= 4587)

→ NET_DISCONNECTED : Non connecté au serveur (= 4588)

Constructeur

L'objet **Net** peut être construit avec un constructeur sans arguments, ou avec le constructeur : Net(char *name, char *ip, int port, int id)

Paramètres :

→ char *name : Nom de l'utilisateur (Défaut : Vide)

→ char *ip : IP du serveur (Défaut : "127.0.0.1")

→ int port : Port à utiliser (Défaut : 69)

→ int id : ID de l'utilisateur (Défaut : 0)

Le constructeur initialise l'état de connexion à NET_DISCONNECTED dans les deux cas.

```
1  #include "Net.h"
2  using namespace std
3  int main(){
4    Net a;
5    Net b((char *)"Username", "127.0.0.1", 69, 1);
6    return 0 ;
7  }
```

Les Getters

Les getters renvois les valeurs des attributs protégés du même nom.

Liste des getters : char *getName(), char *getIp(), int getPort(), int getId(), int getState()

Les Setters

Les setters initialises les valeurs des attributs protégés du même nom.

Liste des setters : void setName(char *name), void setIp(char *ip), void setPort(int port), void setId(int id), void setState(int state)

Exemple Des Méthodes Basique

```
1  #include "Net.h"
2  using namespace std
3  int main(){
4    Net net;
5    net.setName((char *)"Innocent1");
6    cout << net.getName << " will connect on " << net.getIp() << ":" << net.getPort() << endl;
7    return 0;
8  }
```

Connexion et Déconnexion

Les méthodes int chatConnect() et void chatDisconnect() permettent respectivement la connexion et la déconnexion au serveur.

int chatConnect() renvoie NET_CONNECTION_ERROR_NO ou NET_CONNECTION_ERROR_YES. Cette méthode permet la connexion au serveur, à l'IP et sur le port renseigner lors de la création de l'objet, ou après les avoir initialisés à l'aide des setters.

void chatDisconnect ne renvoie aucune valeur. Cette méthode vas clean les pointeurs utilisés, ainsi que remettre le statue de la connexion à NET_DISCONNECTED.

(Le destructeur de la class fonctionne de la même manière)

Envoyer et Recevoir

Les méthodes void chatSend(char *message, int len) et void chatReceive(char *message, int len) permettent respectivement d'envoyer et de recevoir des données depuis le serveur.

void chatSend(char *message, int len) :

→ char *message : Message à envoyer (Ne pas oublier le cast)

→ int len : Longueur du message

void chatReceive(char *message, int len) :

→ char *message : Variable réceptrice du message

→ int len : Longueur du message (1024 de préférence)

La variable message intervenant dans les deux méthodes est un char array[1024]; de taille 1024. 1024 étant la valeur maximale du message à transmettre. Si ce tableau est utiliser dans une boucle, il doit être vider à chaque boucle : fill_n(array, 1024, 0);

Exemple Des Autres Méthodes

```
1  #include "Net.h"
2  using namespace std
3  int main()[
4   Net net((char *)"Innocent1", (char *)"127.0.0.1", 69, 1);
5   int resultat = net.chatConnect();
6   if(resuiltat == NET_CONNECTION_ERROR_YES){
7    cout << "Erreur de connexion au serveur" << endl;
8    return 1;
9   }
10  cout << net.getName << " connected on " << net.getIp() << ":" << net.getPort() << endl;
11  char message[] = "This is a test message";
12  if(net.getState() == NET_CONNECTED){
13   net.chatSend(message, sizeof(message));
14   char receive[1024];
15   net.chatReceive(receive, sizeof(receive));
16   cout << receive << end;
17   fill_n(receive, 1024, 0);
18  }
19 }
```

Compilation

La class Net utilise les librairies string.h, winsock2.h et ws2tcpip.h. Les deux dernières doivent donc êtres inclues durant la compilation.

Si le compilateur utiliser est **GCC (G++)**, il suffit de rajouter -lwsock32 à la commande de compilation. Normalement, GCC intègre déjà ses librairies.

→ C:\Users\Desktop\Path> g++ -I. main.cpp Net.cpp -lwsock32

Si ce n'est pas le cas, ou si le compilateur utiliser n'est pas GCC, les libraires se trouvent à l'emplacement : C:\Wndows\System32

Il suffit dans ce cas de rajouter le chemin des fichiers .h au compilateur.

Avec GCC, il suffit de préciser l'option -I :

→ C:\Users\Desktop\Path> g++ -IC:\Windows\System32 main.cpp Net.cpp -lwsock32

Pour les autres compilateur, il suffit de chercher comment ajouter un chemin d'accès aux fichiers .h.