

Server Asincrono

V.1.3.5

Generato da Doxygen 1.8.12

Indice

1	Pagina Principale	1
2	Indice delle strutture dati	3
2.1	Strutture dati	3
3	Indice dei file	5
3.1	Elenco dei file	5
4	Documentazione delle classi	7
4.1	Riferimenti per la struct Str	7
4.1.1	Descrizione dettagliata	7
4.1.2	Documentazione dei campi	7
4.1.2.1	msg_recv	7
4.1.2.2	msg_send	7
4.1.2.3	Server_Socket	7
5	Documentazione dei file	9
5.1	Riferimenti per il file Origine_server_as.cpp	9
5.1.1	Documentazione delle definizioni	10
5.1.1.1	MAX_BUFFER	10
5.1.2	Documentazione delle funzioni	10
5.1.2.1	main()	10
5.1.2.2	Recv()	10
5.1.2.3	Send()	11
Indice		13

Capitolo 1

Pagina Principale

Il programma 'Server_asincrono' tramite un Socket creato nel main, con la funzione 'WSAStartup' e 'socket', poi abbiamo assegnato un porta al canale della Socket tramite la funzione 'bind', adesso "creato" il nostro Socket, lo si mette in "ascolto" tramite la funzione 'listen', se finora non vi sono stati problemi possiamo aspettare che il Client si connetta' infine il Server tramite la funzione 'accept' completa la connessione. Da li' si avra' una connessione bidirezionale asincrona tra Server e Client, terminabile tramite una sequenza di escape. In caso di errore nella creazione o connessione del Socket il programma terminera' rilasciando un determinato codice d'errore. Inoltre nel progetto utlizzeremo una libreria non citata esplicitamente nel codice, ma individuabile nelle proprieta' del progetto, la libreria "Ws2_32.lib".

Capitolo 2

Indice delle strutture dati

2.1 Strutture dati

Queste sono le strutture dati con una loro breve descrizione:

Str	7
----------------------	----------

Capitolo 3

Indice dei file

3.1 Elenco dei file

Questo è un elenco di tutti i file con una loro breve descrizione:

Origine_server_as.cpp	9
------------------------------	---

Capitolo 4

Documentazione delle classi

4.1 Riferimenti per la struct Str

Campi

- SOCKET **Server_Socket**
- char **msg_recv** [MAX_BUFFER]
- char **msg_send** [MAX_BUFFER]

4.1.1 Descrizione dettagliata

La struct **Str** (pag. 7) contiene tutte le variabili utili all'utilizzo della comunicazione

4.1.2 Documentazione dei campi

4.1.2.1 msg_recv

```
char Str::msg_recv[MAX_BUFFER]
```

La variabile dedicata alla ricezione del messaggio

4.1.2.2 msg_send

```
char Str::msg_send[MAX_BUFFER]
```

La variabile dedicata all'invio del messaggio

4.1.2.3 Server_Socket

```
SOCKET Str::Server_Socket
```

Il canale logico di comunicazione

La documentazione per questa struct è stata generata a partire dal seguente file:

- **Origine_server_as.cpp**

Capitolo 5

Documentazione dei file

5.1 Riferimenti per il file Origine_server_as.cpp

```
#include <winsock.h>
#include <process.h>
#include <string.h>
#include <iostream>
```

Strutture dati

- struct **Str**

Definizioni

- #define **MAX_BUFFER** 100

Funzioni

- unsigned __stdcall **Recv** (void *param)
riceve il messaggio dal Client e lo mostra in output.
- unsigned __stdcall **Send** (void *param)
Invia il messaggio al Client.
- int __cdecl **main** (void)

Inizializziamo il Socket e li diamo un MAKEWORD(2.0, 2.0) per indicare la versione di Windows che il Socket andra' ad utilizzare, tutt'ora la versione piu' recente e' la "2.2". Il nostro Socket di "ascolto" 'listenSocket' e' un IPv4, con connessione sicura, e protocollo TCP. Il main dopo aver fatto partire i relativi Thread si blocca alla linea '121' tramite un 'WaitForMultipleObject' dove attende la fine o da parte del Server o del Client tramite una sequenza di escape uguale alla lettera 'q'. In caso di errore nella creazione o connessione del Socket il programma terminera' non prima di lasciare, in output, un codice relativo all'errore tramite la funzione 'WSAGetLastError()'.

5.1.1 Documentazione delle definizioni

5.1.1.1 MAX_BUFFER

```
#define MAX_BUFFER 100
```

Definisce il Massimo dei caratteri che puo' ricevere e inviare il Server

5.1.2 Documentazione delle funzioni

5.1.2.1 main()

```
int __cdecl main (
    void )
```

Inizializziamo il Socket e li diamo un MAKEWORD(2.0, 2.0) per indicare la versione di Windows che il Socket andra' ad utilizzare, tutt'ora la versione piu' recente e' la "2.2". Il nostro Socket di "ascolto" 'listenSocket' e' un IPv4, con connessione sicura, e protocollo TCP. Il main dopo aver fatto partire i relativi Thread si blocca alla linea '121' tramite un 'WaitForMultipleObject' dove attende la fine o da parte del Server o del Client tramite una sequenza di escape uguale alla lettera 'q'. In caso di errore nella creazione o connessione del Socket il programma terminera' non prima di lasciare, in output, un codice relativo all'errore tramite la funzione 'WSAGetLastError()'.

```
...
printf("Accettata Connessione con Client: %s\n", inet_ntoa(Client_addr.sin_addr));

TH[0] = (HANDLE)_beginthreadex(NULL, 0, &Recv, &S, 0, 0);
TH[1] = (HANDLE)_beginthreadex(NULL, 0, &Send, &S, 0, 0);

printf("Server\t\t\tClient\n");

WaitForMultipleObjects(2, TH, TRUE, INFINITE);
...
```

Autore

tarchi.giacomo

5.1.2.2 Recv()

```
unsigned __stdcall Recv (
    void * param )
```

riceve il messaggio dal Client e lo mostra in output.

Parametri

void*	data: Contiene la struttura Str (pag. 7)
-------	---

Restituisce

unsigned: zero.

```

{
    Str* S = (Str*)param;

    while (1)
    {
        if (recv(S->Server_Socket, S->msg_recv, sizeof(S->msg_recv), 0) > 0)
        {
            printf("\t \t \t%s\n", S->msg_recv);
            if (strcmp(S->msg_recv, "q") == 0 || strcmp(S->msg_send, "q") == 0) break; //sequenza di escape
        }
        _endthreadex(0);
        return 0;
    }
}

```

Autore

tarchi.giacomo

5.1.2.3 Send()

```

unsigned __stdcall Send (
    void * param )

```

Invia il messaggio al Client.

Parametri

<i>void*</i>	data: Contiene la struttura Str (pag. 7)
--------------	---

Restituisce

unsigned: zero.

```

{
    Str* S = (Str*)param;
    while (1)
    {
        std::cin.getline(S->msg_send, MAX_BUFFER);
        send(S->Server_Socket, S->msg_send, sizeof(S->msg_send), 0);
        if (strcmp(S->msg_recv, "q") == 0 || strcmp(S->msg_send, "q") == 0) break; //sequenza di escape
    }
    _endthreadex(0);
    return 0;
}

```

Autore

tarchi.giacomo

Indice analitico

MAX_BUFFER

 Origine_server_as.cpp, 10

main

 Origine_server_as.cpp, 10

msg_recv

 Str, 7

msg_send

 Str, 7

Origine_server_as.cpp, 9

 MAX_BUFFER, 10

 main, 10

 Recv, 10

 Send, 11

Recv

 Origine_server_as.cpp, 10

Send

 Origine_server_as.cpp, 11

Server_Socket

 Str, 7

Str, 7

 msg_recv, 7

 msg_send, 7

 Server_Socket, 7