**NCU NWLab socket programming homework**

**史碩三**

-------------------------------------------------------------------------------------------------------------------------

這次的socket programming作業,我的主題是一個簡易的跨平台Socket API,主要提供三個模型,讓使用者能夠快速建立自己的Socket程式,此API的三個模型其一為簡易的多工Server,能夠透過簡單的操作方法來建立一個伺服器,以下為我使用這API寫的範例程式碼,此DEMO程式為一個簡易的TimeServer,能夠提供遠端用戶連線並取得伺服器的時間,透過此API撰寫的程式碼：

-------------------------------------------------------------------------------------------------------------------------

#include "EZSocketCore.h"

#include <time.h>

void Loop(struct Address\_and\_Port From , \

char \*ReceiveData , int ReceiveLength , \

char \* ResponseData , int \* ResponseLength)

{

if(ReceiveLength>0)

{

if(strstr(ReceiveData,"time?")!=NULL)

{

printf("Connect From %s:%d\n",From.ip,From.port);

time\_t now;

time(&now);

char timestr[100];

memset(timestr,0x0,100);

sprintf(timestr,"%s",ctime(&now));

printf("Time=%s\n",timestr);

strcpy(ResponseData,timestr);

\*ResponseLength = strlen(ResponseData);

}

}

};

int main(int argc,char \*\*argv)

{

int Port;

if(argc<2)

{

Port = 9999;

printf("Port=%d\n",Port);

}

else

{

Port = atoi(argv[1]);

printf("Port=%d\n",Port);

}

int Errorcode;l

char ErrorMsg[1024];

struct EZSocketCore \* ServerHandler = \

GetServerHandler(Port,ServerMainLoop\_EZUserdef,Loop,&Errorcode);

if(ServerHandler==NULL)

{

GetServerErrorMsg(Errorcode,ErrorMsg,1024);

printf("%s",ErrorMsg);

}

else

{

printf("Time Server Create!\n");

ServerHandler->StartServerForever(ServerHandler);

}

return 0;

}

-------------------------------------------------------------------------------------------------------------------------

此程式的架構是這樣：

使用者先建立一個這種格式的函式來服務用戶：

void Loop(struct Address\_and\_Port From , \

char \*ReceiveData , int ReceiveLength , \

char \* ResponseData , int \* ResponseLength)

{

/\* 要服務使用者的程式碼 \*/

}

並且在main中呼叫

GetServerHandler(Port,ServerMainLoop\_EZUserdef, Loop,&Errorcode);

來跟API要求一個伺服器端的控制柄,傳入要把伺服器開在哪個port,以及把API設定成哪個模式(在此為ServerMainLoop\_EZUserdef模式,代表需要此API提供如下功能：當Client端送訊息時,自動呼叫特定函式,並把收到的資料傳進來,並且將回應訊息的控制指標也傳入以利控制),並將函式Loop以函式指標傳入GetServerHandler,代表指定Loop函式為要自動要被API呼叫的特定函式.

接著看到Loop函式：

由於此函式已經被指定給ServerHandler,所以當Client端連入時API將會自動調用,在API調用時,第一個參數為連入Client的地址以及埠的資訊,第二個參數為Client發送過來的訊息,第三個參數為Client發送訊息的長度,第四個參數為將要自動回應給Client端的訊息,第五個參數為回應訊息的長度.

所以我們在Loop函式中先檢查ReceiveLength是否大於0,代表Client端有發來訊息,接著我們檢查訊息中是否包含”time?”字串,有的話代表Client端正在詢問伺服器時間,確認Client端送來的訊息是在詢問時間後,我們就調用ctime把伺服器時間複製到參數四去,代表等等讓伺服器回覆此時間訊息,並把第五個參數（回應訊息長度）設為此時間訊息的次串長度,設定完畢後, Loop函式返回,API自動把訊息回覆給Client端

void Loop(struct Address\_and\_Port From , \

char \*ReceiveData , int ReceiveLength , \

char \* ResponseData , int \* ResponseLength)

{

if(ReceiveLength>0)

{

if(strstr(ReceiveData,"time?")!=NULL)

{

printf("Connect From %s:%d\n",From.ip,From.port);

time\_t now;

time(&now);

char timestr[100];

memset(timestr,0x0,100);

sprintf(timestr,"%s",ctime(&now));

printf("Time=%s\n",timestr);

strcpy(ResponseData,timestr);

\*ResponseLength = strlen(ResponseData);

}

}

};

-----------------------------------------------------------------------------------------------------------------------

三個模型其二為簡易的Client端程式,能夠透過簡單的操作方法來建立一個到伺服器的連線,以下為我使用這API寫的範例程式碼,此DEMO程式為一個簡易的TimeClient,能夠連線到Timeserver並取得伺服器的時間並顯示出來：

-------------------------------------------------------------------------------------------------------------------------

#include "EZSocketCore.h"

int main(int argc,char \*\*argv)

{

int Errorcode;

char ErrorMsg[1024];

struct Address\_and\_Port server;

memset(&server,0x0,sizeof(struct Address\_and\_Port));

if(argc==3)

{

strcpy(server.ip,argv[1]);

server.port=atoi(argv[2]);

}

else

{

strcpy(server.ip,"127.0.0.1");

server.port=9999;

}

struct EZSocketCore \* ClientHandler = GetClientHandler(server,&Errorcode);

if(ClientHandler==NULL)

{

GetClientErrorMsg(Errorcode,ErrorMsg,1024);

printf("%s",ErrorMsg);

}

else

{

printf("Connect!\n");

char buffer[100];

memset(buffer,0x0,100);

strcpy(buffer,"time?\n");

ClientHandler->WriteToServer(ClientHandler,buffer,strlen(buffer));

ClientHandler->ReadFromServer(ClientHandler,buffer,sizeof(buffer));

ClientHandler->DisconnectToServer(ClientHandler);

printf("Response From Server:\n%s\n",buffer);

}

return 0;

}

-------------------------------------------------------------------------------------------------------------------------

程式的流程為先設定一個Address\_and\_Port結構,填入伺服器地址以及埠的資訊,並調用GetClientHandler取得客戶端控制柄,透過此控制柄來對伺服器寫入/接收資料.

-------------------------------------------------------------------------------------------------------------------------

三個模型其三為簡易的簡易的WebServer模型,能夠透過簡單的操作方法來建立一個網頁伺服器,以下為我使用這API寫的範例程式碼,此DEMO程式為一個簡易的WebServer,在程式執行後,能使用瀏覽器輸入 <http://127.0.0.1:9999/> 看到此Project的網頁畫面)

以下為程式碼：

-------------------------------------------------------------------------------------------------------------------------

void Loop(struct Address\_and\_Port From,struct EZWeb\_ResourceInfo ReqFile , \

struct EZWeb\_ResourceController \*ResourceController, \

struct EZWeb\_ResponseController \*RsponseController)

{

char errormsg[1024];

int errorcode;

printf("GetResourceByInfo\n");

struct EZWeb\_Resource \*Resource = \

ResourceController->GetResourceByInfo(ResourceController,ReqFile,&errorcode);

if(Resource==NULL)

{

GetEZWebResourceControllerErrorMsg(errorcode,errormsg,1024);

printf("Error : %s\n",errormsg);

}

else

{

RsponseController->SetResponseResource(RsponseController,Resource);

}

};

int main(void)

{

int Port = 9999;

int Errorcode;

char ErrorMsg[1024];

struct EZSocketCore \* ServerHandler = \

GetServerHandler(Port,ServerMainLoop\_EZWeb, Loop,&Errorcode);

if(ServerHandler==NULL)

{

GetServerErrorMsg(Errorcode,ErrorMsg,1024);

printf("%s",ErrorMsg);

}

else

{

printf("Web Server Create!\n");

ServerHandler->StartServerForever(ServerHandler);

}

return 0;

}

-------------------------------------------------------------------------------------------------------------------------

此程式的架構和Example1的Timeserver十分相似,只是在這裡我們自定義的服務函式參數不太一樣,在此的服務函式格式為這樣

void Loop(struct Address\_and\_Port From,struct EZWeb\_ResourceInfo ReqFile , \

struct EZWeb\_ResourceController \*ResourceController, \

struct EZWeb\_ResponseController \*RsponseController)

{

/\* 要服務使用者的程式碼 \*/

}

並且在main中呼叫

GetServerHandler(Port,ServerMainLoop\_EZWeb,Loop,&Errorcode);

來跟API要求一個伺服器端的控制柄,傳入要把伺服器開在哪個port,以及把API設定成哪個模式(在此為ServerMainLoop\_EZWeb模式,代表需要此API提供如下功能：作為一個網頁伺服器,把目前瀏覽器要求的資源資訊,以及網站資源的管理工具指標,以及伺服器該如何回應訊息的控制指標,一起送到特定自訂函式來處理),並將Loop以函式指標傳入GetServerHandler,代表指定Loop為上述要被API呼叫來處理的函式.

當自訂的Loop函式被調用後,會傳入下列參數：

struct Address\_and\_Port 型別的 From：瀏覽器的要求來自的地址與埠

EZWeb\_ResourceInfo 型別的 ReqFile ：瀏覽器要求的HTTP GET目標 （在EZWeb.h定義）

EZWeb\_ResourceController 型別的 ResourceController 指標：

伺服器資源的控制指標,透過此指標提供的方法,能夠讀取伺服器中的檔案,或是對檔案進行修改

EZWeb\_ResponseController 型別的 RsponseController 指標：

用來控制針對這次的GET要求伺服器該如何回應

void Loop(struct Address\_and\_Port From,struct EZWeb\_ResourceInfo ReqFile , \

struct EZWeb\_ResourceController \*ResourceController, \

struct EZWeb\_ResponseController \*RsponseController)

{

char errormsg[1024];

int errorcode;

printf("GetResourceByInfo\n");

struct EZWeb\_Resource \*Resource = \

ResourceController->GetResourceByInfo(ResourceController,ReqFile,&errorcode);

if(Resource==NULL)

{

GetEZWebResourceControllerErrorMsg(errorcode,errormsg,1024);

printf("Error : %s\n",errormsg);

}

else

{

RsponseController->SetResponseResource(RsponseController,Resource);

}

};

在此段程式碼中,我們的服務使用者的程式碼只是單純的使用

ResourceController 指標 的方法,依照傳入的ReqFile所記載的資訊,調用

EZWeb\_ResourceController. GetResourceByInfo

去伺服器內部取出瀏覽器要GET的東西,並使用RsponseController指標 的方法

EZWeb\_ResponseController. SetResponseResource

將撈到的東西塞到預計回應的列表去,由此一來Web伺服器就會將瀏覽器要GET的東西送出,讓瀏覽器能顯示,當然,我們可以在執行

EZWeb\_ResponseController. SetResponseResource之前,就即時修改資源,舉例來說,我們能夠把經此伺服器處理的HTML全部加上某特定資訊,讓瀏覽器看到一些不存在於HTML中的資訊,如加上時戳標記等等.

-------------------------------------------------------------------------------------------------------------------------

編譯指令:

make 編譯所有的Example

make all 編譯所有的Example

make ex1 只編譯Example1\_Timeserver

make ex2 只編譯Example2\_Timeclient

make ex3 只編譯Example3\_Webserver

make clean 清除所有編譯結果

執行:

Example1\_Timeserver:

/\* 在本機的port 9999執行 \*/

./Timeserver

/\* 在本機的port X執行 \*/

./Timeserver X

Example2\_Timeclient:

1.先啟動Timeserver

2.啟動Timeclient

/\* 連接到本機port 9999的Timeserver \*/

./Timeserver

/\* 連接到ip為xxx.xxx.xxx.xxx,port為Y的Timeserver \*/

./Timeclient xxx.xxx.xxx.xxx Y

Example: ./Timeclient 192.168.1.100 9999

Example3\_Webserver:(佔用port 9999,可自行修改source code)

1.先啟動Webserver於本機(假設ip為xxx.xxx.xxx.xxx)

./Webserver

2.開啟任意瀏覽器,於網址列輸入http://xxx.xxx.xxx.xxx:9999/

注意:目前在Windows上執行WebServer的話瀏覽器看到會有些問題,建議在linux或Mac上執行WebServer

-------------------------------------------------------------------------------------------------------------------------

詳細可參考我的Github：<https://github.com/SAM33/EZSocketCore>

或是把Example3編譯起來(Example3是一個WebServer socket程式,在程式啟動後,只要在瀏覽器輸入 <http://127.0.0.1:9999/> 就能看到網頁畫面)

-------------------------------------------------------------------------------------------------------------------------