

# 計算機網路



Final Project (1)

系級:資工三

學號:B0827213

姓名:陳昱慈

操作說明:

- 解壓縮後，檔案內附有:

	檔案	相應之功能:
1.	httpserver.cpp	server 程式檔
2.	MakeFile	用以 build 之 Makefile 檔
3.	Index.html	網頁標頭檔(此為 server 連線時固定的連線檔案，即只可覆寫)
4.	Root.html	此為設置在連線後進行抓取測試檔案。
5.	server	編譯後可執行檔案
6.	MIMETYPE.txt	Mime type 文件，用以程式抓取 content-type

注意事項:

- 此程式只能運行在 linux 環境上。
- 本程式具有偵錯輸出，不影響結果。

步驟:

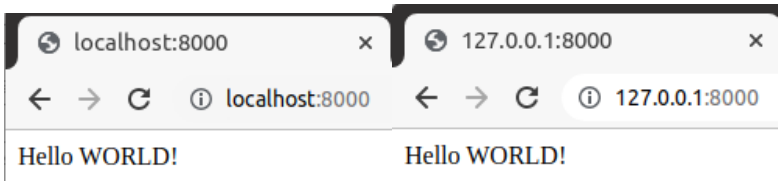
1. 移動至其資料夾路徑內
2. \*利用 Makefile 進行 bulid
3. 選擇輸入以下兩種格式開啟 server  
甲、./server  
乙、./server {Port Number}
4. 根據 3.甲與 3.乙分別開啟瀏覽器連線  
甲、{localhost or 127.0.0.1}:8000  
乙、{localhost or 127.0.0.1}:{Port Number}

偵錯輸出中須注意事項:

Client Disconnect	請求方關閉連線
Bind:Address already in use	重新換 ports 即可

截圖與 telnet 測試:

```
yuztu@yuztu-VirtualBox:~/http$ ./server
```



```
yuztu@yuztu-VirtualBox:~/Desktop$ telnet localhost 8000
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET/ /HTTP1.1
HTTP/1.0 400 Bad Request
```

```

yuztu@yuztu-VirtualBox:~/Desktop$ telnet 127.0.0.1 8000
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
GET / HTTP/1.1
HTTP/1.0 200 OK
Content-Type:text/html

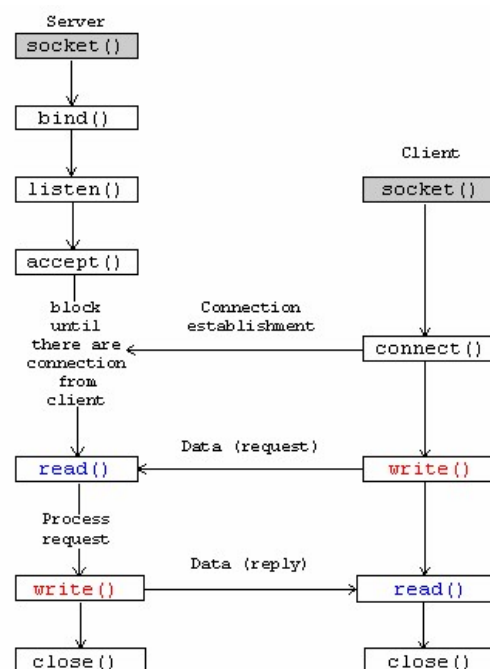
<html>
<body>
    <p>Hello WORLD!</p>
</body>
</html>
Connection closed by foreign host.

```

完整程式與說明

程式主要利用 TCP Socket Programing 進行實作。

- 程式流程圖



(圖片來源: <http://zake7749.github.io/2015/03/17/SocketProgramming/>)

標頭檔使用:

C/C++ 資料流	Socket/net and Unix system call	
#include<iostream>	#include<sys/socket.h>	#include<fcntl.h>
#include<fstream>	#include<sys/stat.h>	
#include<stdio.h>	#include<sys/types.h>	
#include<stdlib.h>	#include<netinet/in.h>	
#include<string.h>	#include<unistd.h>	

宣告說明:

<pre> struct sockaddr_in server_address 1.  server_address.sin_family = AF_INET; 2.  server_address.sin_port = htons(ports); </pre>	<p>此為宣告:</p> <ol style="list-style-type: none"> <li>1. 宣告 IPv4/IPv6</li> <li>2. Port</li> </ol>
---	---

3. server_address.sin_addr.s_addr = INADDR_ANY;	3. Address(此為任意)
---	------------------

函式使用:

函式	說明
socket()	開啟 socket；SOCK_STREAM 為 TCP 連線
bind()	定名且連接到傳輸提供者之位址上
listen()	監聽是否超過 5 人使用(此程式僅定義)
accept()	接受 client 連線
send()	傳輸回覆到 client 端
recv()	接收請求
fstream::seekg()	瀏覽檔案，可移動指針
fstream::tellg()	回傳讀取指針位置
fstream::read()	讀取檔案

\*程式 Debug 方法:

主要利用簡易 client 與 server 做 response header 傳遞上的 debug 後  
再利用 localhost 進行要求(request)，並確認 header 與回傳是否成功。

● 程式解說:

主要利用上述程式流程圖所示，一步步進行:

Server socket 連線	其中: ■ 判斷格式內會有 HTTP 400 的回覆狀況 ■ 讀檔失敗或無此檔會有 HTTP 404 的回覆
→bind 定名	
→listen 監聽	
→while(accept 接受	
→判斷格式	
→讀檔	
→傳輸	
→關閉接受之連線)	
→關閉 server 連線。	

■ Header 宣告

//建立一個 struct 型態的 httpheader 標頭檔內文	
struct httpHeader{	
char request[2048];	//Request Header
char request_line[60];	//第一行:Get /...
char file_name[30];	//請求物件名稱
char method[10];	//請求形式 ex:GET

```
char  type[20];                //請求物件資料結構

char  response_header[2048];   //Response Header
char  content_type[50];        //回傳 MIME 型態
char* response_data;           //回覆資料
};
```

■ Socket

```
int main(int argc,char * argv[]){
    //初始化宣告 socket 變數
    int server_socket=0;
    int client_socket=0;
    int ports =0;

    //宣告 client_address 長度；與 int 無異 POSIX
    socklen_t client_addr_size;
    //宣告 sockaddr_in 型態；區分 port 和 ip address
    struct sockaddr_in server_address,client_address;

    //開啟 socket；SOCK_STREAM 為 TCP
    if((server_socket = socket(AF_INET, SOCK_STREAM,0))== -1)
    {
        cout<<"Error: Socket "<<strerror(errno)<<endl;
        exit(1);
    }
    //判斷是否輸入 port 數，無則設定為 8000
    if(argv[1]==NULL)
    {
        ports=8000;
    }
    else
    {
        sscanf(argv[1],"%d",&ports);
    }
    //宣告 server 所屬家族、port 和 address
    server_address.sin_family = AF_INET;
    server_address.sin_port =  htons(ports);
    server_address.sin_addr.s_addr = INADDR_ANY;    //任意主機
```

```

//定名且連接到傳輸提供者之位址上，偵錯輸出關閉
if(bind(server_socket, (struct sockaddr *)&server_address, sizeof(server_address))!=0)
{
    cout<<"Bind:"<<strerror(errno)<<endl;
    close(server_socket);
    exit(1);
}
//監聽是否超過 5 人使用(僅定義)，偵錯輸出關閉
if(listen(server_socket, 5)!=0)
{
    cout<<"Listen:"<<strerror(errno)<<endl;
    close(server_socket);
    exit(1);
}

```

#### ■ 接受連線

```

client_addr_size = sizeof(client_address);
//接受連線，偵錯輸出
if((client_socket = accept(server_socket, (struct sockaddr *)&client_address,&client_addr_size))== -1)
{
    cout<<"Accept:"<<strerror(errno)<<endl;
    break;
}

```

#### ■ 接收請求訊息

```

while(1){
    //呼叫剛剛宣告的 httpHeader 格式
    httpHeader data;

    //清空陣列
    memset(data.request,0,sizeof(data.request));
    memset(data.response_header,0,sizeof(data.response_header));
    client_addr_size = sizeof(client_address);

    //接受連線
    if((client_socket = accept(server_socket, (struct sockaddr *)&client_address,&client_addr_size))== -1)
    {
        cout<<"Accept:"<<strerror(errno)<<endl;
        break;
    }
}

```

```

}
//接收 request
int receive=recv(client_socket,data.request,sizeof(data.request),0);

//判斷是否有誤 or client 關閉連線
if(receive==-1)
{
    cout<<"Receive:"<<strerror(errno)<<endl;
    break;
}
if(receive==0)
{
    cout<<"Client Disconnect"<<endl;
    break;
}

//將 request 的首行:GET /file HTTP/1.1 拆解
strcpy(data.request_line,strtok(data.request,"\r\n"));
strcpy(data.method,strtok(data.request," "));
strcpy(data.file_name, strtok(NULL, " "));

//是否為 GET 請求
if(strcmp(data.method,"GET")!=0){
    strcpy(data.response_header,"HTTP/1.0 400 Bad Request\r\n\r\n");
    send(client_socket, data.response_header,      sizeof(data.response_header),0);

}

//是否為 HTTP 請求
else if(strstr(data.request_line,"HTTP")== NULL)
{
    strcpy(data.response_header,"HTTP/1.0 400 Bad Request\r\n\r\n");
    send(client_socket, data.response_header, sizeof(data.response_header),0);
}
}

```

## ■ 傳遞回覆訊息

```

fstream http_file;
// GET / HTTP/1.1 則直接抓取 index;
if(strcmp(data.file_name,"/") ==0)
{
    http_file.open("index.html",ios::binary|ios::in);
}

```

```

strcpy(data.type,get_content("index.html"));
}
//GET /file HTTP/1.1 抓取 file 物件
else
{
http_file.open(data.file_name+1,ios::binary|ios::in);
strcpy(data.type,get_content(data.file_name+1));

}
//無法開啟或找不到，則回傳 404
if(!http_file)
{
strcpy(data.response_header,"HTTP/1.0 404 Not Found\r\n\r\n");
send(client_socket, data.response_header, sizeof(data.response_header),0);

}
else{
//回傳 200 與 MIME 型態給 client
strcpy(data.response_header,"HTTP/1.0 200 OK\r\n");
sprintf(data.content_type,"Content-Type:%s\r\n\r\n",data.type);
strcat(data.response_header,data.content_type);

send(client_socket, data.response_header, sizeof(data.response_header),0);

//將檔案從頭看到尾，找出 length
http_file.seekg(0, ios::end);
long fileSize = http_file.tellg();
http_file.seekg(0, ios::beg);

//動態新增一個 fileSize 長度陣列
data.response_data=new char[fileSize];

//讀入陣列並傳送回覆
http_file.read(data.response_data,fileSize);
send(client_socket, data.response_data, fileSize,0);

memset(data.response_data,0,sizeof(data.response_data));}

```

## ■ 抓取 MIME type



```

char* get_content(const char * file){
//從需求物件名稱裡找尋回傳 MIME 型態
    fstream mime;
    //回傳宣告
    char* file_type =new char [30];
    char  gets [128];
    char* token;

    //抓取型態 ex:1.png → png
    strcpy(file_type,file);
    strtok(file_type,".");
    strcpy(file_type,strtok(NULL," "));

    //讀取 MIME 型態表
    mime.open("MIMETYPE.txt",ios::in);
    //一行一行抓取
    while(mime.getline(gets,sizeof(gets)))
    {
        //抓取檔案型態與 MIME 型態間的空格 — 格式:html text/html
        if((token = strtok(gets," "))!=NULL)
        {
            //比較檔案型態與檔案內型態是否一致
            if(strcmp(gets,file_type)==0)
            {
                strcpy(file_type,strtok(NULL," ")); //複製到回傳陣列
                break;
            }
        }
    }
    return file_type;
}

```

封面圖片來源: [https://www.flaticon.com/free-icon/http\\_1674969](https://www.flaticon.com/free-icon/http_1674969)

參考資料:

1. Socket Programming Tutorial in C For Beginners Part1.2 — Youtube:  
 甲、 <https://www.youtube.com/watch?v=LtXEMwSG5-8>  
 乙、 <https://www.youtube.com/watch?v=mStnzIEprH8&feature=youtu.be>
2. Beej's Guide to Network Programming

- 甲、 <https://beej-zhtw-gitbook.netdpi.net/>
3. c++ 网络编程（十一） LINUX 下 初步制作基于 HTTP 的 WEB 服务器  
甲、 <https://www.cnblogs.com/DOMLX/p/9663028.html>
4. Socket 庫存函數:  
甲、 [http://www.tsnien.idv.tw/Internet\\_WebBook/chap8/8-5%20Socket%20%E5%BA%AB%E5%AD%98%E5%87%BD%E6%95%B8.html](http://www.tsnien.idv.tw/Internet_WebBook/chap8/8-5%20Socket%20%E5%BA%AB%E5%AD%98%E5%87%BD%E6%95%B8.html)
5. TCP Socket Programming 學習筆記:  
甲、 <http://zake7749.github.io/2015/03/17/SocketProgramming/>
6. [筆記]Linux 環境用 c++建立 Socket 連線  
甲、 <https://snsd0805.github.io/jekyll/update/2019/05/27/%E7%AD%86%E8%A8%98-Linux%E7%92%B0%E5%A2%83%E7%94%A8c++%E5%BB%BA%E7%AB%8B%E9%80%A3%E7%B7%9A.html>