計算機網路報告書

Term Project(Part 1)

資工三甲　B0929034　林芊妤

中華民國 111 年 12 月 11 日

目錄

目錄 2

1. 文件目的 3

1.1. 程式需求 3

1.2. 程式範圍 4

2. 操作說明 4

3. 程式說明 7

3.1. 程式流程 7

3.2. 程式架構 7

4. 參考資料 8

1. 文件目的

此程式提供可下載並且離線瀏覽網站的功能，參閱此文件可以理解如何使用，以及可以達到什麼樣的需求。

1.1 程式需求

基本功能:

I. 提供兩種終端機模式供使用者輸入參數(URL 與輸出目錄)

i. 命令列模式: 直接將必要的參數透過命令列輸入。

ii. 互動模式: 程式執行後會輸出提示訊息要求輸入參數。

II. 基本功能#1 中檢查使用者輸入的參數是否有誤，並且輸出適當的說明訊息。

III. 僅下載使用者所輸入之URL所代表之物件以及其包含的其他物件，並且視需要修改物件的位址以供離線瀏覽。

IV. 下載結束後顯示狀態與統計資訊。

進階功能:

I. 下載過程顯示各物件的下載狀況

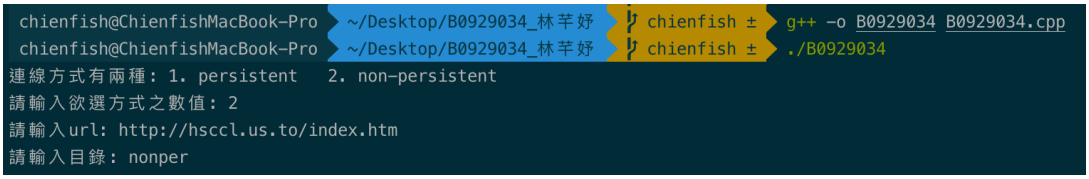
II. 可支援 persistent 與 non-persistent(預設方式)connections。

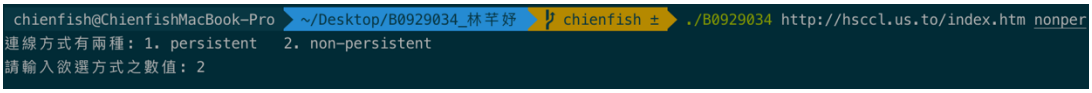
1.2 程式範圍

僅支援 http，以及 html 中有圖片的網頁。

2. 操作說明

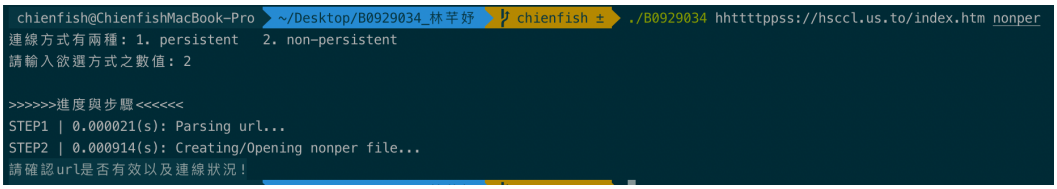
首先，編譯完成後有兩種模式可以輸入參數，第一種是互動模式，第二種是命令列模式。前者是在程式執行後輸出提示訊息要求輸入參數，後者是在執行時直接參數透過命令列輸入。





接著，會列出有支援的兩種 connection 供使用者選擇。選擇完畢後，請使用者輸入欲下載之網頁的 URL，以及要輸出的資料夾目錄名稱。

若輸入之 URL 有誤，也會跳出錯誤訊息，並直接停止執行。如下圖，我使用第二種模式輸入參數，而 URL 的網址是錯誤的，因此會跳出請確認 URL 是否有效。



以下圖為例，使用者所選擇的連線方式為 2，也就是 non-persistent connection，而欲下載的網址為 http://hsccl.us.to/index.htm，最後要輸出的資料夾目錄為 nonper。

```
連線方式有兩種：1. persistent   2. non-persistent
請輸入欲選方式之數值：2
請輸入url: http://hsccl.us.to/index.htm
請輸入目錄：nonper
```

接著，在路徑下會找到建立的資料夾，打開後會看到網址上的 html 檔，以及其所包含的圖片。

下載結束後會顯示狀態以及統計資訊，如檔案數量、下載總容量、下載花費時長。另外，也會顯示連線的資訊，如 domain, IP address 等。

```
>>>>>>狀態與統計資訊<<<<<<
http://hsccl.us.to/index.htm為有效的url
檔案數量：4
下載總容量：738 KB
下載花費時間：0.013829 (s)

>>>>>>連線資訊<<<<<<
URL: http://hsccl.us.to/index.htm
DOMAIN: hsccl.us.to
IP: 163.25.101.225
PORT: 80
```

下載的過程中，會顯示目前執行的步驟在做的事，同時列出所花費的時間。此外，在步驟和步驟間也會列出是在做哪個物件的下載。

```
>>>>>>進度與步驟<<<<<<
STEP1 | 0.000036(s): Parsing url...
STEP2 | 0.001090(s): Creating/Opening nonper file...
STEP3 | 0.002233(s): Connecting to server...
-- 物件1 --
STEP4 | 0.000218(s): Downloading the content of index.htm...
-- 物件2 --
STEP5 | 0.000525(s): Connecting to server...
STEP6 | 0.002772(s): Dowdloading http://hsccl.us.to/images/animals/cats/image.jpg...
-- 物件3 --
STEP7 | 0.003441(s): Connecting to server...
STEP8 | 0.003508(s): Dowdloading http://hsccl.us.to/image.jpg...
-- 物件4 --
STEP9 | 0.004178(s): Connecting to server...
STEP10 | 0.002871(s): Dowdloading http://hsccl.us.to/images/image.jpg...
```

最後，提供選擇 persistent connection，跟上面圖示比較可以發現，connecting to server 只做了一次，並且統計資訊中的下載花費時長也較少。

```
連線方式有兩種: 1. persistent   2. non-persistent
請輸入欲選方式之數值: 1
請輸入url: http://hsccl.us.to/index.htm
請輸入目錄: per

>>>>>>進度與步驟<<<<<<
STEP1 | 0.000038(s): Parsing url...
\STEP2 | 0.001028(s): Creating/Opening per file...
STEP3 | 0.001981(s): Connecting to server...
-- 物件1 --
STEP4 | 0.001040(s): Downloading the content of index.htm...
-- 物件2 --
STEP5 | 0.002612(s): Dowdloading http://hsccl.us.to/images/animals/cats/image.jpg...
-- 物件3 --
STEP6 | 0.003122(s): Dowdloading http://hsccl.us.to/image.jpg...
-- 物件4 --
STEP7 | 0.002294(s): Dowdloading http://hsccl.us.to/images/image.jpg...

>>>>>>狀態與統計資訊<<<<<<
http://hsccl.us.to/index.htm為有效的url
檔案數量: 4
下載總容量: 767 KB
下載花費時間: 0.012750 (s)

>>>>>>連線資訊<<<<<<
URL: http://hsccl.us.to/index.htm
DOMAIN: hsccl.us.to
IP: 163.25.101.225
PORT: 80
```

3. 程式說明

說明程式執行之流程，以及其架構和功能。

3.1 程式流程

1.判斷連線方式(persistent, non-persistent) -> 2.判斷執行檔後是否有加入參數 -> 3.切割 URL(得到 domain, path) -> 4.創建目錄資料夾與切換路徑 -> 5.得到 request 訊息(包含 start-line 和 header) -> 6.建立 socket -> 7.建立 TCP 連線 -> 8.取得 response -> 9.把抓到的 response 解析(拿最後的 message-body) -> 10.檢查並下載 message-body 的照片 -> 11.建立 html 檔以及創建圖片檔

3.2 程式架構

根據上述流程順序，依序介紹較為複雜之程式碼。

2. 判斷執行檔後是否有加入參數

在 main 函式後有兩個參數，第一個是紀錄執行檔後的參數個數(包含自己，以空白分隔)，第二個是記錄輸入的值。

```
// 判斷在執行檔後是否有加入參數
string url, dir;
if (argc == 1) { // 在終端機沒有輸入任何字串
    cout << "請輸入url: ";
    cin >> url;
    cout << "請輸入目錄: ";
    cin >> dir;
} else if (argc == 3) { // 在終端機輸入url, dir
    url = argv[1];
    dir = argv[2];
```

3. 切割 URL(得到 domain, path)

先將 http://拿掉，可以得到 domain 和 path。

```
void ParseUrl(string url) {  // 分解url -> 產生domain, path
    int init;
    if (url.substr(0, 7) == "http://") init = 7;
    else if (url.substr(0, 8) == "https://") init = 8;

    bool dash = false;
    for (int i = init; i < url.length(); i++) {
        if (url[i] == '/') dash = true;
        if (url[i] == '/' && dash == true) {
            Domain = url.substr(init, i-init);
            Path = url.substr(i, url.length());
```

4. 創建目錄資料夾與切換路徑

利用終端機指令創建目錄資料夾，getcwd 可以得到目前所在的路徑，只要將此路徑加上新的資料夾名稱，再利用 chdir 就可以將路經切換。

```
// 建立資料夾
char pastPath[1024], nowPath[1024];
string createDir = "mkdir -p " + dir;
START = clock();
system(createDir.c_str());
getcwd(pastPath, 1024);
sprintf(nowPath, "%s/%s",pastPath, dir.c_str());
chdir(nowPath);
```

5. 得到 request 訊息(包含 start-line 和 header)

根據選擇的連線方式，有不同的 request(下面會說明兩者相異處)。send 會把 request 送出去，之後 read 將所拿到的資料先暫存在 buffer 裡面。

```
// 得到request訊息(start-line & header) -> 建立TCP連線 -> 取得response
string request = "";
if (method == 1) request = PerRequest();
else if (method == 2) request = NonPerRequest();
int sock = ConnectHttp();
char buffer[1024] = {};
printf("-- 物件%d --\n", fileNum+1);
send(sock, request.c_str(), strlen(request.c_str()), 0);
read(sock, buffer,1024);
```

承 5.

兩種連線方式的 request 要分開撰寫，因為 non-persistent 在發送完請求就會關閉連線；而 persistent 的連線則必須一直開著。

```
string PerRequest() {    // persistent使用的request
    string file = Path;
    char message[2048] = {};
    sprintf(message,
        "GET %s HTTP/1.1\r\n"
        "Host: %s\r\n"
        "Connection: keep-alive\r\n"    // 連線會一直開著
        "\r\n" ,Path.c_str(), Domain.c_str());

    return message;
}
```

```
string NonPerRequest() {     // non-persistent使用的request
    string file = Path;
    char message[2048] = {};
    sprintf(message,
        "GET %s HTTP/1.1\r\n"
        "Host: %s\r\n"
        "Connection: Close\r\n"     // 發送完請求就會關閉連線
        "\r\n" ,Path.c_str(), Domain.c_str());

    return message;
}
```

6. 建立 socket

利用 socket 提供的 syscall，以及函式庫提供的 struct 來建立 socket。

```
// 建立socket
int sock = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
if (sock == -1){
    cout << "無效的socket!" << endl;
    exit(-1);
}

// socket的結構
struct sockaddr_in info;
string ip = HostToIp(Domain);
bzero(&info,sizeof(info));  // 初始化 將struct涵蓋的bits設為0
info.sin_family = AF_INET; // 通訊協定(Internet)
info.sin_port = htons(80); // 連接的埠口位址
if (inet_pton(info.sin_family, ip.c_str(), &info.sin_addr) == 0) {
    cout << "請確認url是否有效以及連線狀況!" << endl;
    exit(0);
}
```

承 6.

利用 gethostbyname 函式，將分割出來的 domain 轉換成 IP address

```
string HostToIp(const string& host) {   // 找出IP address
    hostent* hostname = gethostbyname(host.c_str());
    if (hostname) return string(inet_ntoa(**(in_addr**)hostname->h_addr_list));
    else return "-1";
}
```

7. 建立 TCP 連線

利用 syscall 提供的 connect 來進行連線(需要用到第 6 點創建的 socket)

```
// 連線
if (connect(sock, (struct sockaddr *)&info, sizeof(info)) == -1){
    cout << "連線失敗!" << endl;
    exit(-1);
}
```

10. 檢查並下載 message-body 的照片

allIMG 用來存取所有<img src=””>中雙引號裡的字串，left 和 right 會傳

至 renameSRC 的函式，將其重新命名。

```
// 抓所有照片的src
vector<string> allIMG;
int left = 0, right = 0;
string substr = "src=";
while ((left = content.find(substr, left)) != string::npos) {
    for (int j = left+5; j < content.length(); j++){
        if (content[j] == '"') {
            right = j;
            break;
        }
    }
    string imgSRC = content.substr(left+5, right-left-5);
    renameSRC(left+5, right, allIMG.size());
    allIMG.push_back(imgSRC);
    left += substr.length();
}
```

承 10.

照片會重新被命名成 pic(數字).jpg，其中數字為現在總共存取幾張照片。之後便將 message-body 中 img tag 的 src 都改成上面這些名稱。

```
void renameSRC(int left, int right, int imgNUM) {
    string name = "pic"+ to_string(imgNUM) + ".jpg";
    content.replace(left, right-left, name);
}
```

11. 建立 html 檔

將改過 src 的 message-body 寫入檔案中，檔案已存在的話直接覆寫，若不存在則創建一個。後面 S_IWUSR, S_IRUSR 是給使用者此檔案的權限。

```
//建立html檔案
START = clock();
int fd = open(Path.substr(1, BUFFER.length()).c_str(), O_WRONLY | O_CREAT, S_IWUSR | S_IRUSR);
write(fd, content.c_str(), strlen(content.c_str()));
close(fd);
```

承 11. 創建照片檔

如果存下來的照片有 http，則可以直接去 download 函式，如果沒有 http，則要將其補上，同時還要加上 domain，再去 download 函式下載。

```
// 下載圖片
for (int i = 0; i < allIMG.size(); i++) {
    string src = "";
    if (allIMG[i][0] == '/') allIMG[i].erase(0, 1);

    if (allIMG[i].find("http") == -1)   src = "http://"+ Domain + "/"+ allIMG[i];
    else   src = allIMG[i];
    fileNum++;
    download(src, i, sock);
}
```

承 11.

一樣要先切割 URL，接著要看是哪種連線方式，決定要用哪種 request。

接著 send 會將 request 傳出去，之後用 recv 將資料讀到 buf，這邊先讀一次，因為要知道 content-length 是多少。

```
void download(string imgSRC, int imgNUM, int sd) {
    // 切割imgSRC -> 得到domain&path
    ParseUrl(imgSRC);

    // 發出請求並連線
    printf("-- 物件%d --\n", fileNum);
    string req = "";
    if (method == 1) req = PerRequest();
    else if (method == 2) {
        req = NonPerRequest();
        sd = ConnectHttp();
    }
    send(sd, req.c_str(), strlen(req.c_str()), 0);

    // 建立檔案
    START = clock();
    char buf[10000]={};
    string name = "pic" + to_string(imgNUM) + ".jpg";
    int pic = open(name.c_str(), O_WRONLY | O_CREAT, S_IWUSR | S_IRUSR);
    int recvSize = recv(sd, buf, sizeof(buf)-1, 0);
```

承 11.

從暫存在 buf 的字串中找到 content-length，用 strstr 可以把後面的字串抓出來，while 迴圈是為了只抓出數值，不要其他的文字內容。

```
// 取得img的content-length
string contentLen = "Content-Length: ";
char *sub = strstr(buf, contentLen.c_str()) + strlen(contentLen.c_str());
string targetLen;
while (isdigit(*sub)) {
    targetLen += *sub;
    sub++;
}
```

承 11.

imgContent 是指 message-body 的內容，而 imgContentLen 是將收到的 size 減掉 imgContent-buf(此為 header 的長度)。因為讀取的長度小於緩衝區中的長度，沒辦法一次讀完，需執行 recv 函數多次，直到讀到的字串長度等於前面得到的 targetLen。

```
// 取得圖片內容並寫入檔案
char *imgContent = strstr(buf, "\r\n\r\n") + 4; /
int imgContentLen = recvSize-(imgContent-buf);
write(pic, imgContent, imgContentLen);
while (true) {
    recvSize = recv(sd, buf, sizeof(buf)-1, 0);
    write(pic, buf, recvSize);
    imgContentLen += recvSize;

    if (imgContentLen >= stoi(targetLen)) break;
}
fileSize += imgContentLen;
close(pic);
if (method == 2) shutdown(sd, SHUT_WR);
```

4. 參考資料

- 建立 socket

  https://snsd0805.github.io/jekyll/update/2019/05/27/筆記-Linux 環境用 c++建立 Socket 連線.html

  http://www.tsnien.idv.tw/Internet_WebBook/chap8/8-4%20Socket%20 傳輸位址.html

  https://stackoverflow.com/questions/9400756/ip-address-from-host-name-in-windows-socket-programming

- http client request

  https://notfalse.net/47/c-socket-http-client

  https://www.796t.com/content/1550270006.html

- 設定命令列模式參數

  https://edisonx.pixnet.net/blog/post/57060736-%5Bt%5D-vs-設定命令參數列

- 設定 Persistent Connection 的 header

  https://byvoid.com/zht/blog/http-keep-alive-header/

  https://blog.insightdatascience.com/learning-about-the-http-connection-keep-alive-header-7ebe0efa209d