# Hello Socket

## ﻿1.

Microsoft Visual Studio Solution File, Format Version 12.00

# Visual Studio 14

VisualStudioVersion = 14.0.25420.1

MinimumVisualStudioVersion = 10.0.40219.1

Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "Hello Socket Programming", "Hello Socket Programming\Hello Socket Programming.vcxproj", "{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}"

EndProject

Global

GlobalSection(SolutionConfigurationPlatforms) = preSolution

Debug|x64 = Debug|x64

Debug|x86 = Debug|x86

Release|x64 = Release|x64

Release|x86 = Release|x86

EndGlobalSection

GlobalSection(ProjectConfigurationPlatforms) = postSolution

{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}.Debug|x64.ActiveCfg = Debug|x64

{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}.Debug|x64.Build.0 = Debug|x64

{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}.Debug|x86.ActiveCfg = Debug|Win32

{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}.Debug|x86.Build.0 = Debug|Win32

{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}.Release|x64.ActiveCfg = Release|x64

{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}.Release|x64.Build.0 = Release|x64

{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}.Release|x86.ActiveCfg = Release|Win32

{EAFF02C6-3176-4A27-86FD-48C37CFABAAB}.Release|x86.Build.0 = Release|Win32

EndGlobalSection

GlobalSection(SolutionProperties) = preSolution

HideSolutionNode = FALSE

EndGlobalSection

EndGlobal

## 2.

#include <stdio.h>

#include <WinSock2.h>

#include <WS2tcpip.h>

void main()

{

WSADATA DATA;

int error = 0;

int code = 0;

error = WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKADDR\_IN addr;

ADDRINFO\* result = NULL;

getaddrinfo("www.vnexpress.net", "http", NULL, &result);

ADDRINFO\* tmp = result;

while (tmp != NULL)

{

memset(&addr, 0, sizeof(addr));

memcpy(&addr, tmp->ai\_addr, tmp->ai\_addrlen);

printf("%s\n", inet\_ntoa(addr.sin\_addr));

if (addr.sin\_addr.S\_un.S\_addr != 0)

{

break;

}

tmp = tmp->ai\_next;

}

freeaddrinfo(result);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

error = connect(s, (sockaddr\*)&addr, sizeof(addr));

error = WSACleanup();

}

## 3.

﻿<?xml version="1.0" encoding="utf-8"?>

<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">

<ItemGroup>

<Filter Include="Source Files">

<UniqueIdentifier>{4FC737F1-C7A5-4376-A066-2A32D752A2FF}</UniqueIdentifier>

<Extensions>cpp;c;cc;cxx;def;odl;idl;hpj;bat;asm;asmx</Extensions>

</Filter>

<Filter Include="Header Files">

<UniqueIdentifier>{93995380-89BD-4b04-88EB-625FBE52EBFB}</UniqueIdentifier>

<Extensions>h;hh;hpp;hxx;hm;inl;inc;xsd</Extensions>

</Filter>

<Filter Include="Resource Files">

<UniqueIdentifier>{67DA6AB6-F800-4c08-8B7A-83BB121AAD01}</UniqueIdentifier>

<Extensions>rc;ico;cur;bmp;dlg;rc2;rct;bin;rgs;gif;jpg;jpeg;jpe;resx;tiff;tif;png;wav;mfcribbon-ms</Extensions>

</Filter>

</ItemGroup>

<ItemGroup>

<Text Include="ReadMe.txt" />

</ItemGroup>

<ItemGroup>

<ClInclude Include="stdafx.h">

<Filter>Header Files</Filter>

</ClInclude>

<ClInclude Include="targetver.h">

<Filter>Header Files</Filter>

</ClInclude>

</ItemGroup>

<ItemGroup>

<ClCompile Include="stdafx.cpp">

<Filter>Source Files</Filter>

</ClCompile>

<ClCompile Include="Hello Socket Programming.cpp">

<Filter>Source Files</Filter>

</ClCompile>

</ItemGroup>

</Project>

# Hello Threading

## 1.

Microsoft Visual Studio Solution File, Format Version 12.00

# Visual Studio 14

VisualStudioVersion = 14.0.25420.1

MinimumVisualStudioVersion = 10.0.40219.1

Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "HelloThreading", "HelloThreading\HelloThreading.vcxproj", "{2CA04D01-FBC3-423A-8949-202A3B668CBC}"

EndProject

Global

GlobalSection(SolutionConfigurationPlatforms) = preSolution

Debug|x64 = Debug|x64

Debug|x86 = Debug|x86

Release|x64 = Release|x64

Release|x86 = Release|x86

EndGlobalSection

GlobalSection(ProjectConfigurationPlatforms) = postSolution

{2CA04D01-FBC3-423A-8949-202A3B668CBC}.Debug|x64.ActiveCfg = Debug|x64

{2CA04D01-FBC3-423A-8949-202A3B668CBC}.Debug|x64.Build.0 = Debug|x64

{2CA04D01-FBC3-423A-8949-202A3B668CBC}.Debug|x86.ActiveCfg = Debug|Win32

{2CA04D01-FBC3-423A-8949-202A3B668CBC}.Debug|x86.Build.0 = Debug|Win32

{2CA04D01-FBC3-423A-8949-202A3B668CBC}.Release|x64.ActiveCfg = Release|x64

{2CA04D01-FBC3-423A-8949-202A3B668CBC}.Release|x64.Build.0 = Release|x64

{2CA04D01-FBC3-423A-8949-202A3B668CBC}.Release|x86.ActiveCfg = Release|Win32

{2CA04D01-FBC3-423A-8949-202A3B668CBC}.Release|x86.Build.0 = Release|Win32

EndGlobalSection

GlobalSection(SolutionProperties) = preSolution

HideSolutionNode = FALSE

EndGlobalSection

EndGlobal

## 2.

﻿<?xml version="1.0" encoding="utf-8"?>

<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">

<ItemGroup>

<Filter Include="Source Files">

<UniqueIdentifier>{4FC737F1-C7A5-4376-A066-2A32D752A2FF}</UniqueIdentifier>

<Extensions>cpp;c;cc;cxx;def;odl;idl;hpj;bat;asm;asmx</Extensions>

</Filter>

<Filter Include="Header Files">

<UniqueIdentifier>{93995380-89BD-4b04-88EB-625FBE52EBFB}</UniqueIdentifier>

<Extensions>h;hh;hpp;hxx;hm;inl;inc;xsd</Extensions>

</Filter>

<Filter Include="Resource Files">

<UniqueIdentifier>{67DA6AB6-F800-4c08-8B7A-83BB121AAD01}</UniqueIdentifier>

<Extensions>rc;ico;cur;bmp;dlg;rc2;rct;bin;rgs;gif;jpg;jpeg;jpe;resx;tiff;tif;png;wav;mfcribbon-ms</Extensions>

</Filter>

</ItemGroup>

<ItemGroup>

<Text Include="ReadMe.txt" />

</ItemGroup>

<ItemGroup>

<ClInclude Include="stdafx.h">

<Filter>Header Files</Filter>

</ClInclude>

<ClInclude Include="targetver.h">

<Filter>Header Files</Filter>

</ClInclude>

</ItemGroup>

<ItemGroup>

<ClCompile Include="stdafx.cpp">

<Filter>Source Files</Filter>

</ClCompile>

<ClCompile Include="HelloThreading.cpp">

<Filter>Source Files</Filter>

</ClCompile>

</ItemGroup>

</Project>

## 3.

#include <Windows.h>

#include <stdio.h>

#define NUMTHREADS 100

CRITICAL\_SECTION g\_cs;

int g\_sum = 0;

HANDLE hThread[NUMTHREADS];

HANDLE Lock[NUMTHREADS];

DWORD WINAPI HelloFunc(LPVOID arg)

{

int i = (int)arg;

Sleep(10);

EnterCriticalSection(&g\_cs);

g\_sum += i;

LeaveCriticalSection(&g\_cs);

SetEvent(Lock[i]); //Mo khoa

return 0;

}

void main()

{

DWORD ID = 0;

InitializeCriticalSection(&g\_cs);

printf("Main ThreadID = %d\n", GetCurrentThreadId());

for (int i = 0; i < NUMTHREADS; i++)

{

hThread[i] = CreateThread(NULL, 0, HelloFunc, (LPVOID)i, 0, &ID);

Lock[i] = CreateEvent(NULL, TRUE, FALSE, NULL);

ResetEvent(Lock[i]); //Dong khoa

}

//WaitForMultipleObjects(NUMTHREADS, Lock, TRUE, INFINITE);

for (int i = 0; i < NUMTHREADS; i++)

{

WaitForSingleObject(Lock[i], INFINITE);

}

DeleteCriticalSection(&g\_cs);

printf("Exit with sum: %ld\n", g\_sum);

}

# My Telnet

## 1.

Microsoft Visual Studio Solution File, Format Version 12.00

# Visual Studio 14

VisualStudioVersion = 14.0.25420.1

MinimumVisualStudioVersion = 10.0.40219.1

Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "MyTelnet", "MyTelnet\MyTelnet.vcxproj", "{944AF345-A54A-4E49-91FF-A4B18BC92393}"

EndProject

Global

GlobalSection(SolutionConfigurationPlatforms) = preSolution

Debug|x64 = Debug|x64

Debug|x86 = Debug|x86

Release|x64 = Release|x64

Release|x86 = Release|x86

EndGlobalSection

GlobalSection(ProjectConfigurationPlatforms) = postSolution

{944AF345-A54A-4E49-91FF-A4B18BC92393}.Debug|x64.ActiveCfg = Debug|x64

{944AF345-A54A-4E49-91FF-A4B18BC92393}.Debug|x64.Build.0 = Debug|x64

{944AF345-A54A-4E49-91FF-A4B18BC92393}.Debug|x86.ActiveCfg = Debug|Win32

{944AF345-A54A-4E49-91FF-A4B18BC92393}.Debug|x86.Build.0 = Debug|Win32

{944AF345-A54A-4E49-91FF-A4B18BC92393}.Release|x64.ActiveCfg = Release|x64

{944AF345-A54A-4E49-91FF-A4B18BC92393}.Release|x64.Build.0 = Release|x64

{944AF345-A54A-4E49-91FF-A4B18BC92393}.Release|x86.ActiveCfg = Release|Win32

{944AF345-A54A-4E49-91FF-A4B18BC92393}.Release|x86.Build.0 = Release|Win32

EndGlobalSection

GlobalSection(SolutionProperties) = preSolution

HideSolutionNode = FALSE

EndGlobalSection

EndGlobal

## 2.

﻿<?xml version="1.0" encoding="utf-8"?>

<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">

<ItemGroup>

<Filter Include="Source Files">

<UniqueIdentifier>{4FC737F1-C7A5-4376-A066-2A32D752A2FF}</UniqueIdentifier>

<Extensions>cpp;c;cc;cxx;def;odl;idl;hpj;bat;asm;asmx</Extensions>

</Filter>

<Filter Include="Header Files">

<UniqueIdentifier>{93995380-89BD-4b04-88EB-625FBE52EBFB}</UniqueIdentifier>

<Extensions>h;hh;hpp;hxx;hm;inl;inc;xsd</Extensions>

</Filter>

<Filter Include="Resource Files">

<UniqueIdentifier>{67DA6AB6-F800-4c08-8B7A-83BB121AAD01}</UniqueIdentifier>

<Extensions>rc;ico;cur;bmp;dlg;rc2;rct;bin;rgs;gif;jpg;jpeg;jpe;resx;tiff;tif;png;wav;mfcribbon-ms</Extensions>

</Filter>

</ItemGroup>

<ItemGroup>

<Text Include="ReadMe.txt" />

</ItemGroup>

<ItemGroup>

<ClInclude Include="stdafx.h">

<Filter>Header Files</Filter>

</ClInclude>

<ClInclude Include="targetver.h">

<Filter>Header Files</Filter>

</ClInclude>

</ItemGroup>

<ItemGroup>

<ClCompile Include="stdafx.cpp">

<Filter>Source Files</Filter>

</ClCompile>

<ClCompile Include="MyTelnet.cpp">

<Filter>Source Files</Filter>

</ClCompile>

</ItemGroup>

</Project>

## 3.

#include <stdio.h>

#include <WinSock2.h>

DWORD WINAPI ClientThread(LPVOID arg)

{

SOCKET c = (SOCKET)arg;

char\* welcome = "Gui user va pass theo cu phap: <username> <password>\n";

send(c, welcome, strlen(welcome), 0);

char buffer[1024];

char line[1024];

while (0 == 0)

{

memset(buffer, 0, sizeof(buffer));

recv(c, buffer, sizeof(buffer) - 1, 0);

FILE\* f = fopen("C:\\Temp\\UP.TXT", "rt");

bool valid = false;

while (!feof(f))

{

memset(line, 0, sizeof(line));

fgets(line, sizeof(line) - 1, f);

if (strcmp(line, buffer) == 0)

{

valid = true;

break;

}

}

fclose(f);

if (valid == false)

{

char\* again = "Invalid user/password, please try again\n";

send(c, again, strlen(again), 0);

}

else

{

char\* start = "Please send any command to execute\n";

send(c, start, strlen(start), 0);

while (0 == 0)

{

memset(buffer, 0, sizeof(buffer));

recv(c, buffer, sizeof(buffer) - 1, 0);

while (buffer[strlen(buffer) - 1] == '\n')

{

buffer[strlen(buffer) - 1] = 0;

}

char command[1024];

memset(command, 0, sizeof(command));

sprintf(command, "%s > c:\\temp\\%d.txt", buffer, c);

system(command);

memset(command, 0, sizeof(command));

sprintf(command, "c:\\temp\\%d.txt", c);

FILE\*f = fopen(command, "rt");

while (!feof(f))

{

memset(line, 0, sizeof(line));

fgets(line, sizeof(line) - 1, f);

send(c, line, strlen(line), 0);

}

fclose(f);

}

}

}

return 0;

}

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

{

if (argc < 2)

return;

short port = atoi(argv[1]);

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKADDR\_IN saddr;

saddr.sin\_family = AF\_INET;

saddr.sin\_addr.S\_un.S\_addr = INADDR\_ANY;

saddr.sin\_port = htons(port);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

bind(s, (sockaddr\*)&saddr, sizeof(saddr));

listen(s, 5);

while (0 == 0)

{

SOCKADDR\_IN caddr;

int clen = sizeof(caddr);

SOCKET c = accept(s, (sockaddr\*)&caddr, &clen);

DWORD ID = 0;

CreateThread(NULL, 0, ClientThread, (LPVOID)c, 0, &ID);

}

## 4.

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <WinSock2.h>

HANDLE\* g\_handleThread = NULL;

SOCKET\* g\_socket = NULL;

int\* g\_threadStatus = NULL;

int g\_clientCount = 0;

DWORD WINAPI ClientThread(LPVOID param)

{

    char\* welcome = (char\*)"Welcome to Telnet Server\n Please login by sending [user password]\n";

    SOCKET c = (SOCKET)param;

    send(c, welcome, strlen(welcome), 0);

    char buffer[1024];

    char cmd[1024];

    char line[1024];

    char name[1024];

    char psw[1024];

    char cname[1024];

    char cpsw[1024];

    while (0 == 0) //Login

    {

        memset(buffer, 0, sizeof(buffer));

        recv(c, buffer, sizeof(buffer), 0);

        memset(cname, 0, sizeof(cname));

        memset(cpsw, 0, sizeof(cpsw));

        sscanf(buffer, "%s%s", cname, cpsw);

        FILE\* f = fopen("C:\\Temp\\Users.txt", "rt");

        int found = 0;

        while (!feof(f))

        {

            memset(line, 0, sizeof(line));

            fgets(line, sizeof(line), f);

            memset(name, 0, sizeof(name));

            memset(psw, 0, sizeof(psw));

            sscanf(line, "%s%s", name, psw);

            if (strcmp(name, cname) == 0 && strcmp(psw, cpsw) == 0)

            {

                found = 1;

                break;

            }

        }

        fclose(f);

        if (found)

            break;

        else

        {

            char\* invalid = (char\*)"Invalid name or password. Please try again!\n";

            send(c, invalid, strlen(invalid), 0);

        }

    }

    char\* login = (char\*)"Now please send a command to be executed\n";

    send(c, login, strlen(login), 0);

    while (0 == 0)

    {

        memset(buffer, 0, sizeof(buffer));

        recv(c, buffer, sizeof(buffer), 0);

        while ( buffer[strlen(buffer) - 1] == '\n' ||

                buffer[strlen(buffer) - 1] == '\r')

        {

            buffer[strlen(buffer) - 1] = 0;

        }

        printf("SOCKET: %d - %s\n", c, buffer);

        memset(cmd, 0, sizeof(cmd));

        sprintf(cmd, "%s > out.txt", buffer);

        if (strcmp(buffer, "quit") == 0)

        {

            break;

        }

        system(cmd);

        FILE\* f = fopen("out.txt", "rt");

        while (!feof(f))

        {

            memset(buffer, 0, sizeof(buffer));

            fgets(buffer, sizeof(buffer), f);

            send(c, buffer, strlen(buffer), 0);

        }

        fclose(f);

        send(c, "\n", 1, 0);

    }

    closesocket(c);

    for (int i = 0; i < g\_clientCount; i++)

    {

        if (c == g\_socket[i])

        {

            g\_threadStatus[i] = 1; //Stop

            break;

        }

    }

    return 0;

}

void main()

{

    WSADATA DATA;

    WSAStartup(MAKEWORD(2, 2), &DATA);

    SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

    SOCKADDR\_IN saddr, caddr;

    int clen = sizeof(caddr); //BAT BUOC

    saddr.sin\_family = AF\_INET;

    saddr.sin\_port = htons(8888);

    saddr.sin\_addr.S\_un.S\_addr = 0; //Any address

    bind(s, (sockaddr\*)&saddr, sizeof(saddr));

    listen(s, 10);

    while (0 == 0)

    {

        DWORD ID = 0;

        SOCKET c = accept(s, (sockaddr\*)&caddr, &clen);

        g\_clientCount += 1;

        g\_handleThread = (HANDLE\*)realloc(g\_handleThread, g\_clientCount \* sizeof(HANDLE));

        g\_socket = (SOCKET\*)realloc(g\_socket, g\_clientCount \* sizeof(SOCKET));

        g\_threadStatus = (int\*)realloc(g\_threadStatus, g\_clientCount \* sizeof(int));

        g\_socket[g\_clientCount - 1] = c;

        g\_threadStatus[g\_clientCount - 1] = 0; //Running

        g\_handleThread[g\_clientCount - 1] = CreateThread(NULL, 0, ClientThread, (LPVOID)c, 0, &ID);

        //CreatThread moi de chay vong for duoi day

        for (int i = 0; i < g\_clientCount; i++)

        {

            if (g\_threadStatus[i] == 1)

            {

                CloseHandle(g\_handleThread[i]);

                g\_handleThread[i] = INVALID\_HANDLE\_VALUE;

                g\_socket[i] = INVALID\_SOCKET;

            }

        }

    }

    for (int i = 0; i < g\_clientCount; i++)

    {

        if (g\_handleThread[i] != INVALID\_HANDLE\_VALUE)

        {

            CloseHandle(g\_handleThread[i]);

        }

        if (g\_socket[i] != INVALID\_SOCKET)

        {

            closesocket(g\_socket[i]);

        }

    }

    free(g\_handleThread);

    free(g\_socket);

    free(g\_threadStatus);

}

# HTTP FILE SEVER

## 1.

﻿

Microsoft Visual Studio Solution File, Format Version 12.00

# Visual Studio 14

VisualStudioVersion = 14.0.25420.1

MinimumVisualStudioVersion = 10.0.40219.1

Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "HTTP FILE SERVER", "HTTP FILE SERVER\HTTP FILE SERVER.vcxproj", "{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}"

EndProject

Global

GlobalSection(SolutionConfigurationPlatforms) = preSolution

Debug|x64 = Debug|x64

Debug|x86 = Debug|x86

Release|x64 = Release|x64

Release|x86 = Release|x86

EndGlobalSection

GlobalSection(ProjectConfigurationPlatforms) = postSolution

{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}.Debug|x64.ActiveCfg = Debug|x64

{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}.Debug|x64.Build.0 = Debug|x64

{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}.Debug|x86.ActiveCfg = Debug|Win32

{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}.Debug|x86.Build.0 = Debug|Win32

{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}.Release|x64.ActiveCfg = Release|x64

{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}.Release|x64.Build.0 = Release|x64

{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}.Release|x86.ActiveCfg = Release|Win32

{CBA67BA7-45EC-450D-B5FB-C1B360BF1D41}.Release|x86.Build.0 = Release|Win32

EndGlobalSection

GlobalSection(SolutionProperties) = preSolution

HideSolutionNode = FALSE

EndGlobalSection

EndGlobal

## 2.

﻿<?xml version="1.0" encoding="utf-8"?>

<Project ToolsVersion="4.0" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">

<ItemGroup>

<Filter Include="Source Files">

<UniqueIdentifier>{4FC737F1-C7A5-4376-A066-2A32D752A2FF}</UniqueIdentifier>

<Extensions>cpp;c;cc;cxx;def;odl;idl;hpj;bat;asm;asmx</Extensions>

</Filter>

<Filter Include="Header Files">

<UniqueIdentifier>{93995380-89BD-4b04-88EB-625FBE52EBFB}</UniqueIdentifier>

<Extensions>h;hh;hpp;hxx;hm;inl;inc;xsd</Extensions>

</Filter>

<Filter Include="Resource Files">

<UniqueIdentifier>{67DA6AB6-F800-4c08-8B7A-83BB121AAD01}</UniqueIdentifier>

<Extensions>rc;ico;cur;bmp;dlg;rc2;rct;bin;rgs;gif;jpg;jpeg;jpe;resx;tiff;tif;png;wav;mfcribbon-ms</Extensions>

</Filter>

</ItemGroup>

<ItemGroup>

<Text Include="ReadMe.txt" />

</ItemGroup>

<ItemGroup>

<ClInclude Include="stdafx.h">

<Filter>Header Files</Filter>

</ClInclude>

<ClInclude Include="targetver.h">

<Filter>Header Files</Filter>

</ClInclude>

</ItemGroup>

<ItemGroup>

<ClCompile Include="stdafx.cpp">

<Filter>Source Files</Filter>

</ClCompile>

<ClCompile Include="HTTP FILE SERVER.cpp">

<Filter>Source Files</Filter>

</ClCompile>

</ItemGroup>

</Project>

## 3.

#include <stdio.h>

#include <WinSock2.h>

#include <Windows.h>

DWORD WINAPI ClientThread(LPVOID arg);

void main()

{

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

SOCKADDR\_IN saddr;

saddr.sin\_addr.S\_un.S\_addr = 0;

saddr.sin\_port = htons(8888);

saddr.sin\_family = AF\_INET;

bind(s, (sockaddr\*)&saddr, sizeof(saddr));

listen(s, 10);

while (0 == 0)

{

SOCKADDR\_IN caddr;

int clen = sizeof(caddr);

SOCKET c = accept(s, (sockaddr\*)&caddr, &clen);

DWORD ID = 0;

CreateThread(NULL, 0, ClientThread, (LPVOID)c, 0, &ID);

}

}

void SendFileToClient(char\* path, SOCKET c)

{

FILE\* f = fopen(path, "rb");

if (f != NULL)

{

fseek(f, 0, SEEK\_END);

long filesize = ftell(f);

char\* data = (char\*)calloc(filesize, 1);

fseek(f, 0, SEEK\_SET);

fread(data, 1, filesize, f);

fclose(f);

send(c, data, filesize, 0);

free(data);

data = NULL;

}

}

void ConstructFolderHTML(char\*path, char\* html)

{

strcpy(html, "<html><H>SUBDIRECTORY<BR></H>");

WIN32\_FIND\_DATAA FDATA;

char fullpath[1024];

memset(fullpath, 0, sizeof(fullpath));

sprintf(fullpath, "C:%s/\*.\*", path);

HANDLE hFind = FindFirstFileA(fullpath, &FDATA);

//First result

if (FDATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

sprintf(html + strlen(html), "<a href=\"%s/%s\">%s</a><br>", path, FDATA.cFileName, FDATA.cFileName);

else

sprintf(html + strlen(html), "<b><a href=\"FILE\_%s/%s\">%s</a></b><br>", path, FDATA.cFileName, FDATA.cFileName);

while (FindNextFileA(hFind, &FDATA) == TRUE)

{

//Next result

if (FDATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

sprintf(html + strlen(html), "<a href=\"%s/%s\">%s</a><br>", path, FDATA.cFileName, FDATA.cFileName);

else

sprintf(html + strlen(html), "<b><a href=\"FILE\_%s/%s\">%s</a></b><br>", path, FDATA.cFileName, FDATA.cFileName);

}

sprintf(html + strlen(html), "</html>");

}

void ConstructRootHTML(char\* html)

{

strcpy(html, "<html><H>ROOT DIRECTORY<BR></H>");

WIN32\_FIND\_DATAA FDATA;

HANDLE hFind = FindFirstFileA("C:\\\*.\*", &FDATA);

//First result

if (FDATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

sprintf(html + strlen(html), "<a href=\"%s\">%s</a><br>", FDATA.cFileName, FDATA.cFileName);

else

sprintf(html + strlen(html), "<b><a href=\"%s\">%s</a></b><br>", FDATA.cFileName, FDATA.cFileName);

while (FindNextFileA(hFind, &FDATA) == TRUE)

{

//Next result

if (FDATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

sprintf(html + strlen(html), "<a href=\"%s\">%s</a><br>", FDATA.cFileName, FDATA.cFileName);

else

sprintf(html + strlen(html), "<b><a href=\"%s\">%s</a></b><br>", FDATA.cFileName, FDATA.cFileName);

}

sprintf(html + strlen(html), "</html>");

}

DWORD WINAPI ClientThread(LPVOID arg)

{

SOCKET c = (SOCKET)arg;

char buffer[1024];

char verb[1024];

char path[1024];

char version[1024];

int html\_size = 1024 \* 1024;

char\* html = (char\*)calloc(html\_size, 1);

memset(buffer, 0, 1024);

recv(c, buffer, 1023, 0);

memset(verb, 0, 1024);

memset(path, 0, 1024);

memset(version, 0, 1024);

sscanf(buffer, "%s%s%s", verb, path, version);

if (strcmp(verb, "GET") == 0)

{

if (strcmp(path, "/") == 0) //Root Directory

{

memset(html, 0, html\_size);

ConstructRootHTML(html);

}

else if (strstr(path,"FILE\_") == NULL)

{

memset(html, 0, html\_size);

ConstructFolderHTML(path, html);

}

else

{

char\* filepath = strstr(path, "FILE\_") + strlen("FILE\_");

char fullpath[1024];

memset(fullpath, 0, 1024);

sprintf(fullpath, "C:/%s", filepath);

SendFileToClient(fullpath, c);

}

}

if (strcmp(verb, "POST") == 0) {};

if (strcmp(verb, "PUT") == 0) {};

if (strcmp(verb, "DELETE") == 0) {};

send(c, html, strlen(html), 0);

closesocket(c);

free(html);

return 0;

}

## 4.

#include <stdio.h>

#include <WinSock2.h>

#include <Windows.h>

DWORD WINAPI ClientThread(LPVOID arg);

void main()

{

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

SOCKADDR\_IN saddr;

saddr.sin\_addr.S\_un.S\_addr = 0;

saddr.sin\_port = htons(8888);

saddr.sin\_family = AF\_INET;

bind(s, (sockaddr\*)&saddr, sizeof(saddr));

listen(s, 10);

while (0 == 0)

{

SOCKADDR\_IN caddr;

int clen = sizeof(caddr);

SOCKET c = accept(s, (sockaddr\*)&caddr, &clen);

DWORD ID = 0;

CreateThread(NULL, 0, ClientThread, (LPVOID)c, 0, &ID);

}

}

void SendFileToClient(char\* path, SOCKET c)

{

//Gui HTTP Response Header

char response\_header[1024];

memset(response\_header, 0, sizeof(response\_header));

sprintf(response\_header, "HTTP/1.1 200 OK\r\n");

if (strcmp(path + strlen(path) - 3, "mp3") == 0)

{

sprintf(response\_header + strlen(response\_header), "Content-Type: audio/mp3\r\n");

}

else if (strcmp(path + strlen(path) - 3, "mp4") == 0)

{

sprintf(response\_header + strlen(response\_header), "Content-Type: video/mp4\r\n");

}

else if (strcmp(path + strlen(path) - 3, "doc") == 0)

{

sprintf(response\_header + strlen(response\_header), "Content-Type: application/msword\r\n");

}

else if (strcmp(path + strlen(path) - 4, "docx") == 0)

{

sprintf(response\_header + strlen(response\_header), "Content-Type: application/msword\r\n");

}

else if (strcmp(path + strlen(path) - 3, "txt") == 0 || strcmp(path + strlen(path) - 3, "cpp") == 0 || strcmp(path + strlen(path) - 3, ".cs") == 0)

{

sprintf(response\_header + strlen(response\_header), "Content-Type: text/html\r\n");

}else

sprintf(response\_header + strlen(response\_header), "Content-Type: application/octet-stream\r\n");

FILE\* f = fopen(path, "rb");

if (f != NULL)

{

fseek(f, 0, SEEK\_END);

long filesize = ftell(f);

sprintf(response\_header + strlen(response\_header), "Content-Length: %ld\r\n", filesize);

sprintf(response\_header + strlen(response\_header), "\r\n");

send(c, response\_header, strlen(response\_header), 0);

char\* data = (char\*)calloc(filesize, 1);

fseek(f, 0, SEEK\_SET);

fread(data, 1, filesize, f);

fclose(f);

send(c, data, filesize, 0);

free(data);

data = NULL;

}

}

void ConstructFolderHTML(char\*path, char\* html)

{

strcpy(html, "<html><H>SUBDIRECTORY<BR></H>");

WIN32\_FIND\_DATAA FDATA;

char fullpath[1024];

memset(fullpath, 0, sizeof(fullpath));

sprintf(fullpath, "C:%s/\*.\*", path);

HANDLE hFind = FindFirstFileA(fullpath, &FDATA);

//First result

if (FDATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

sprintf(html + strlen(html), "<a href=\"%s/%s\">%s</a><br>", path, FDATA.cFileName, FDATA.cFileName);

else

sprintf(html + strlen(html), "<b><a href=\"FILE\_%s/%s\">%s</a></b><br>", path, FDATA.cFileName, FDATA.cFileName);

while (FindNextFileA(hFind, &FDATA) == TRUE)

{

//Next result

if (FDATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

sprintf(html + strlen(html), "<a href=\"%s/%s\">%s</a><br>", path, FDATA.cFileName, FDATA.cFileName);

else

sprintf(html + strlen(html), "<b><a href=\"FILE\_%s/%s\">%s</a></b><br>", path, FDATA.cFileName, FDATA.cFileName);

}

sprintf(html + strlen(html), "</html>");

}

void ConstructRootHTML(char\* html)

{

strcpy(html, "<html><H>ROOT DIRECTORY<BR></H>");

WIN32\_FIND\_DATAA FDATA;

HANDLE hFind = FindFirstFileA("C:\\\*.\*", &FDATA);

//First result

if (FDATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

sprintf(html + strlen(html), "<a href=\"%s\">%s</a><br>", FDATA.cFileName, FDATA.cFileName);

else

sprintf(html + strlen(html), "<b><a href=\"FILE\_%s\">%s</a></b><br>", FDATA.cFileName, FDATA.cFileName);

while (FindNextFileA(hFind, &FDATA) == TRUE)

{

//Next result

if (FDATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

sprintf(html + strlen(html), "<a href=\"%s\">%s</a><br>", FDATA.cFileName, FDATA.cFileName);

else

sprintf(html + strlen(html), "<b><a href=\"FILE\_%s\">%s</a></b><br>", FDATA.cFileName, FDATA.cFileName);

}

sprintf(html + strlen(html), "</html>");

}

DWORD WINAPI ClientThread(LPVOID arg)

{

SOCKET c = (SOCKET)arg;

char buffer[1024];

char verb[1024];

char path[1024];

char version[1024];

int html\_size = 1024 \* 1024;

char\* html = (char\*)calloc(html\_size, 1);

memset(buffer, 0, 1024);

recv(c, buffer, 1023, 0);

memset(verb, 0, 1024);

memset(path, 0, 1024);

memset(version, 0, 1024);

sscanf(buffer, "%s%s%s", verb, path, version);

if (strcmp(verb, "GET") == 0)

{

//Thay the %20 bang SPACE

while (strstr(path, "%20") != NULL)

{

char\* space = strstr(path, "%20");

int offset = space - path;

path[offset] = ' ';

offset += 1;

for (int i = 3; i < strlen(space); i++, offset++)

{

path[offset] = space[i];

}

path[offset] = 0;

}

if (strcmp(path, "/") == 0) //Root Directory

{

memset(html, 0, html\_size);

ConstructRootHTML(html);

}

else if (strstr(path,"FILE\_") == NULL)

{

memset(html, 0, html\_size);

ConstructFolderHTML(path, html);

}

else

{

char\* filepath = strstr(path, "FILE\_") + strlen("FILE\_");

char fullpath[1024];

memset(fullpath, 0, 1024);

sprintf(fullpath, "C:/%s", filepath);

SendFileToClient(fullpath, c);

}

}

if (strcmp(verb, "POST") == 0) {};

if (strcmp(verb, "PUT") == 0) {};

if (strcmp(verb, "DELETE") == 0) {};

send(c, html, strlen(html), 0);

closesocket(c);

free(html);

return 0;

}

## 5.

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <WinSock2.h>

char\* g\_html = NULL;

void SendFile(SOCKET c, char\* path)

{

    char header[1024];

    memset(header, 0, 1024);

    strcpy(header, "HTTP/1.1 200 OK\n");

    char\* pattern = strstr(path, "VINHLT");

    pattern[0] = 0;

    if (\_stricmp(path + strlen(path) - 4, ".jpg") == 0)

    {

        sprintf(header + strlen(header), "Content-Type: image/jpeg\n");

    }

    else if (\_stricmp(path + strlen(path) - 4, ".mp4") == 0)

    {

        sprintf(header + strlen(header), "Content-Type: video/mp4\n");

    }

    else if (\_stricmp(path + strlen(path) - 4, ".mp3") == 0)

    {

        sprintf(header + strlen(header), "Content-Type: audio/mp3\n");

    }

    else

    {

        sprintf(header + strlen(header), "Content-Type: application/octet-stream\n");

    }

    char buffer[1024];

    FILE\* f = fopen(path, "rb");

    fseek(f, 0, SEEK\_END);

    int flen = ftell(f);

    fseek(f, 0, SEEK\_SET);

    sprintf(header + strlen(header), "Content-Length: %d\n\n", flen);

    send(c, header, strlen(header), 0);

    while (!feof(f))

    {

        int r = fread(buffer, 1, sizeof(buffer), f);

        send(c, buffer, r, 0);

    }

    fclose(f);

}

void ScanFolder(char\* path)

{

    WIN32\_FIND\_DATAA DATA;

    char fullpath[1024];

    memset(fullpath, 0, sizeof(fullpath));

    if (path[strlen(path) - 1] == '/')

        sprintf(fullpath, "C:%s\*.\*", path);

    else

        sprintf(fullpath, "C:%s/\*.\*", path);

    HANDLE hFind = FindFirstFileA(fullpath, &DATA);

    char tmpHtml[1024];

    memset(tmpHtml, 0, sizeof(tmpHtml));

    if (DATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

    {

        if (path[strlen(path) - 1] == '/')

        {

            sprintf(tmpHtml, "<a href=\"%s%s\"><b>%s</b></a><br>", path, DATA.cFileName, DATA.cFileName);

        }

        else

        {

            sprintf(tmpHtml, "<a href=\"%s/%s\"><b>%s</b></a><br>", path, DATA.cFileName, DATA.cFileName);

        }

    }

    else

    {

        if (path[strlen(path) - 1] == '/')

            sprintf(tmpHtml, "<a href=\"%s%sVINHLT\">%s</a><br>", path, DATA.cFileName, DATA.cFileName);

        else

            sprintf(tmpHtml, "<a href=\"%s/%sVINHLT\">%s</a><br>", path, DATA.cFileName, DATA.cFileName);

    }

    int oldlen = g\_html != NULL ? strlen(g\_html) : 0;

    g\_html = (char\*)realloc(g\_html, oldlen + strlen(tmpHtml) + 1);

    sprintf(g\_html + oldlen, "%s", tmpHtml);

    if (hFind != INVALID\_HANDLE\_VALUE)

    {

        while (FindNextFileA(hFind, &DATA) == TRUE)

        {

            memset(tmpHtml, 0, sizeof(tmpHtml));

            if (DATA.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

            {

                if (path[strlen(path) - 1] == '/')

                    sprintf(tmpHtml, "<a href=\"%s%s\"><b>%s</b></a><br>", path, DATA.cFileName, DATA.cFileName);

                else

                    sprintf(tmpHtml, "<a href=\"%s/%s\"><b>%s</b></a><br>", path, DATA.cFileName, DATA.cFileName);

            }

            else

            {

                if (path[strlen(path) - 1] == '/')

                    sprintf(tmpHtml, "<a href=\"%s%sVINHLT\">%s</a><br>", path, DATA.cFileName, DATA.cFileName);

                else

                    sprintf(tmpHtml, "<a href=\"%s/%sVINHLT\">%s</a><br>", path, DATA.cFileName, DATA.cFileName);

            }

            int oldlen = g\_html != NULL ? strlen(g\_html) : 0;

            g\_html = (char\*)realloc(g\_html, oldlen + strlen(tmpHtml) + 1);

            sprintf(g\_html + oldlen, "%s", tmpHtml);

        }

    }

}

int FindBoundaryPattern(char\* pattern, char\* data, int len, int from)

//Tim mau pattern trong du lieu data bat dau tu vi tri from, len la tong so byte cua data

{

    for (int i = from; i < len; i++)

    {

        if (pattern[0] == data[i])

        {

            int j = 0;

            for (j = 0; j < strlen(pattern); j++)

            {

                if (pattern[j] != data[j + i])

                {

                    break;

                }

            }

            if (j == strlen(pattern))

            {

                return i;

            }

        }

    }

    return -1;

}

DWORD WINAPI ClientThread(LPVOID param)

{

    SOCKET c = (SOCKET)param;

    char\* buffer = (char\*)calloc(1024, 1); //Cap phat dong 1024 byte

    memset(buffer, 0, sizeof(buffer));

    int r = recv(c, buffer, 1024, 0);

    char ACTION[1024];

    char PATH[1024];

    char VER[1024];

    char BOUNDARY[1024];

    char LENGTH[1024];

    memset(ACTION, 0, sizeof(ACTION));

    memset(PATH, 0, sizeof(PATH));

    memset(VER, 0, sizeof(VER));

    memset(BOUNDARY, 0, sizeof(BOUNDARY));

    memset(LENGTH, 0, sizeof(LENGTH));

    sscanf(buffer, "%s%s%s", ACTION, PATH, VER);

    if (g\_html != NULL)

    {

        free(g\_html);

        g\_html = NULL;

    }

    if (strcmp(ACTION, "POST") == 0)

    {

        FILE\* f = NULL;

        char\* tmp = strstr(buffer, "boundary=") + strlen("boundary=");

        sscanf(tmp, "%s", BOUNDARY);

        tmp = strstr(buffer, "Content-Length:") + strlen("Content-Length:");

        sscanf(tmp, "%s", LENGTH);

        int length = atoi(LENGTH);

        buffer = (char\*)realloc(buffer, 1024 + length); //Chac chan du boi vi 1024 byte thua de chua header

        int r1 = 0;

        int received = r;

        do

        {

            r1 = recv(c, buffer + received, 1024, 0);

            if (r1 > 0)

            {

                received += r1;

            }

            else

                break;

        } while (received < length);

        length = received; //Tong so byte nhan trong ca hai lan goi recv

        int found = FindBoundaryPattern(BOUNDARY, buffer, length, 0); //Boundary o tren phan HEADER cua HTTP

        while (found)

        {

            found = FindBoundaryPattern(BOUNDARY, buffer, length, found + 1); //Boundary bat dau phan du lieu

            if (found >= 0)

            {

                //Tach ten file tu metadata sau boundary

                char\* filename = strstr(buffer + found, "filename=\"");

                if (filename != NULL)

                {

                    char fnamevalue[1024];

                    memset(fnamevalue, 0, sizeof(fnamevalue));

                    strcpy(fnamevalue, "C:\\Temp\\");

                    sscanf(filename + strlen("filename=\""), "%s", fnamevalue + strlen(fnamevalue));

                    fnamevalue[strlen(fnamevalue) - 1] = 0;

                    f = fopen(fnamevalue, "wb");

                }

                char\* linebreak = strstr(buffer + found, "\r\n\r\n");

                if (linebreak != NULL)

                    found = linebreak - buffer + 4; //Bo qua doan du lieu bao gom 4 dau \r\n\r\n

                else

                    found += strlen(BOUNDARY);

                int next = FindBoundaryPattern(BOUNDARY, buffer, length, found + 1); //Vi tri tiep theo

                if (next >= 0)

                {

                    fwrite(buffer + found, 1, next - found, f);

                }

                else

                {

                    next = length - 1;

                    fwrite(buffer + found, 1, next - found, f);

                    break;

                }

            }

        }

        free(buffer); //Neu dung calloc/malloc/realloc thi phai free de thu hoi bo nho

        fclose(f);

        char\* response = (char\*)"HTTP/1.1 200 OK\n\n";

        send(c, response, strlen(response), 0);

    }

    if (strcmp(ACTION, "GET") == 0)

    {

        while (strstr(PATH, "%20") != NULL)

        {

            char\* found = strstr(PATH, "%20");

            found[0] = ' ';

            strcpy(found + 1, found + 3);

        }

        if (strstr(PATH, "VINHLT") != NULL)

        {

            SendFile(c, PATH);

        }

        else

        {

            char\* header = (char\*)"HTTP/1.1 200 OK\nServer: LTM\nContent-Type: text/html\n\n";

            send(c, header, strlen(header), 0);

            char\* input = (char\*)"<form action=\"/\" method=\"post\" enctype=\"multipart/form-data\"><input type=\"file\" id=\"txtmyid\" name=\"txtmyname\"><br><input type=\"submit\"></form><br>";

            send(c, input, strlen(input), 0);

            ScanFolder(PATH);

            g\_html = (char\*)realloc(g\_html, strlen(g\_html) + 8);

            sprintf(g\_html + strlen(g\_html), "</html>");

            send(c, g\_html, strlen(g\_html), 0);

        }

    }

    closesocket(c);

    return 0;

}

void main()

{

    WSADATA data;

    WSAStartup(MAKEWORD(2, 2), &data);

    SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

    struct sockaddr\_in saddr;

    saddr.sin\_family = AF\_INET;

    saddr.sin\_port = htons(8888);

    saddr.sin\_addr.S\_un.S\_addr = 0; //htons(INADDR\_ANY)

    bind(s, (sockaddr\*)&saddr, sizeof(saddr));

    listen(s, 10);

    while (0 == 0)

    {

        DWORD ID = 0;

        SOCKADDR\_IN caddr;

        int clen = sizeof(caddr);

        SOCKET c = accept(s, (sockaddr\*)&caddr, &clen);

        CreateThread(NULL, 0, ClientThread, (LPVOID)c, 0, &ID);

    }

}

# MUILTITHREADSELECT

## 1.

#define FD\_SETSIZE 2

#include <stdio.h>

#include <WinSock2.h>

SOCKET\* g\_client = NULL; //Danh sach socket da ket noi

int g\_count = 0; //So luong client da ket noi

DWORD WINAPI ClientThread(LPVOID arg)

{

int startIndex = (int)arg;

int endIndex = startIndex + FD\_SETSIZE - 1;

DWORD ID = GetCurrentThreadId();

printf("Thread: %ld, Start socket: %d, End socket: %d\n", ID, startIndex, endIndex);

fd\_set fdread;

while (0 == 0)

{

FD\_ZERO(&fdread);

for (int i = startIndex; i <= endIndex && i < g\_count; i++)

{

FD\_SET(g\_client[i], &fdread);

}

TIMEVAL to;

to.tv\_sec = 1;

to.tv\_usec = 0;

select(0, &fdread, NULL, NULL, &to);

for (int i = startIndex; i <= endIndex && i < g\_count; i++)

{

if (FD\_ISSET(g\_client[i], &fdread))

{

char buffer[1024];

memset(buffer, 0, sizeof(buffer));

recv(g\_client[i], buffer, sizeof(buffer), 0);

printf("%d: %s\n",g\_client[i], buffer);

}

}

}

return 0;

}

void main()

{

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

SOCKADDR\_IN saddr;

saddr.sin\_family = AF\_INET;

saddr.sin\_port = htons(5000);

saddr.sin\_addr.S\_un.S\_addr = ADDR\_ANY;

bind(s, (sockaddr\*)&saddr, sizeof(saddr));

listen(s,10);

while (0 == 0)

{

SOCKADDR\_IN caddr;

int clen = sizeof(caddr);

SOCKET tmpSocket = accept(s, (sockaddr\*)&caddr, &clen);

//Them tmpSocket vao danh sach va xem co phai tao them luong hay khong

g\_client = (SOCKET\*)realloc(g\_client, (g\_count + 1) \* sizeof(SOCKET));

g\_client[g\_count] = tmpSocket; //Luu vao vi tri cuoi cung

if (g\_count % FD\_SETSIZE == 0)

{

DWORD ID = 0;

CreateThread(NULL, 0, ClientThread, (LPVOID)g\_count, 0, &ID);

}

g\_count += 1; //Tang so luong len mot

}

}

# MUILTI THREAD EVENT SELECT

#include <stdio.h>

#include <WinSock2.h>

#define WSA\_MAXIMUM\_WAIT\_EVENTS 2

SOCKET\* g\_client = NULL; //Danh sach socket da ket noi

HANDLE\* g\_event = NULL; //Danh sach khoa di kem socket

int g\_count = 0; //So luong client da ket noi

DWORD WINAPI ClientThread(LPVOID arg)

{

int startIndex = (int)arg;

int endIndex = startIndex + WSA\_MAXIMUM\_WAIT\_EVENTS - 1;

DWORD ID = GetCurrentThreadId();

printf("Thread: %ld, Start socket: %d, End socket: %d\n", ID, startIndex, endIndex);

while (0 == 0)

{

endIndex = startIndex + WSA\_MAXIMUM\_WAIT\_EVENTS - 1;

if (endIndex >= g\_count)

{

endIndex = g\_count - 1;

}

int countIndex = endIndex - startIndex + 1;

int r = WSAWaitForMultipleEvents(countIndex, g\_event + startIndex, FALSE, 1000, TRUE);

if (r == WSA\_WAIT\_FAILED || r == WSA\_WAIT\_TIMEOUT)

{

continue;

}

r = r + startIndex - WSA\_WAIT\_EVENT\_0;

for (int i = r; i <= endIndex && i < g\_count; i++)

{

int check = WSAWaitForMultipleEvents(1, g\_event + i, FALSE, 100, TRUE);

if (check == WSA\_WAIT\_FAILED || check == WSA\_WAIT\_TIMEOUT)

{

//Khoa thu i trong mang g\_event chua mo

}

else

{

//Khoa thu i da bi mo boi su kien co du lieu den

char buffer[1024];

memset(buffer, 0, sizeof(buffer));

recv(g\_client[i], buffer, sizeof(buffer), 0);

printf("%d: %s\n", g\_client[i], buffer);

WSAResetEvent(g\_event[i]);

}

}

}

return 0;

}

void main()

{

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

SOCKADDR\_IN saddr;

saddr.sin\_family = AF\_INET;

saddr.sin\_port = htons(5000);

saddr.sin\_addr.S\_un.S\_addr = ADDR\_ANY;

bind(s, (sockaddr\*)&saddr, sizeof(saddr));

listen(s, 10);

while (0 == 0)

{

SOCKADDR\_IN caddr;

int clen = sizeof(caddr);

SOCKET tmpSocket = accept(s, (sockaddr\*)&caddr, &clen);

//Them tmpSocket vao danh sach va xem co phai tao them luong hay khong

g\_client = (SOCKET\*)realloc(g\_client, (g\_count + 1) \* sizeof(SOCKET));

g\_event = (HANDLE\*)realloc(g\_event, (g\_count + 1) \* sizeof(SOCKET));

g\_client[g\_count] = tmpSocket; //Luu vao vi tri cuoi cung

g\_event[g\_count] = WSACreateEvent();

WSAEventSelect(g\_client[g\_count], g\_event[g\_count], FD\_READ);

if (g\_count % WSA\_MAXIMUM\_WAIT\_EVENTS == 0)

{

DWORD ID = 0;

CreateThread(NULL, 0, ClientThread, (LPVOID)g\_count, 0, &ID);

}

g\_count += 1; //Tang so luong len mot

**2.**

#include <stdio.h>

#include <WinSock2.h>

#define MAX\_SOCKET\_PER\_THREAD 3

//Mang de chua socket cua cac client dang ket noi den server

SOCKET g\_client[1024];

int g\_count = 0;

DWORD WINAPI ClientThread(LPVOID param)

{

    fd\_set read;

    int startIndex = (int)param;

    int threadID = GetCurrentThreadId();

    while (true)

    {

        FD\_ZERO(&read);

        int number = 0;

        for (int i = startIndex; i < g\_count && number < MAX\_SOCKET\_PER\_THREAD; i++)

        {

            printf("Thread %d: Add socket %d(%d) to the read set\n", threadID, i, g\_client[i]);

            FD\_SET(g\_client[i], &read);

            number += 1;

        }

        timeval t;

        t.tv\_sec = 1;

        t.tv\_usec = 0;

        select(0, &read, NULL, NULL, &t);

        number = 0;

        for (int i = startIndex; i < g\_count && number < MAX\_SOCKET\_PER\_THREAD; i++, number++)

        {

            if (FD\_ISSET(g\_client[i], &read))

            {

                char buffer[1024];

                memset(buffer, 0, sizeof(buffer));

                recv(g\_client[i], buffer, sizeof(buffer), 0);

                printf("Socket %d: %s\n", g\_client[i], buffer);

                for (int j = 0; j < g\_count; j++)

                {

                    if (j != i)

                    {

                        send(g\_client[j], buffer, strlen(buffer), 0);

                    }

                }

            }

        }

    }

    return 0;

}

int main()

{

    WSADATA DATA;

    WSAStartup(MAKEWORD(2, 2), &DATA);

    SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

    struct sockaddr\_in saddr;

    saddr.sin\_family = AF\_INET;

    saddr.sin\_port = htons(8888);

    saddr.sin\_addr.S\_un.S\_addr = INADDR\_ANY;

    bind(s, (sockaddr\*)&saddr, sizeof(saddr));

    listen(s, 10);

    fd\_set read;

    while (true)

    {

        FD\_ZERO(&read);

        FD\_SET(s, &read); //Them cac socket da co vao trong tap tham do

        for (int i = 0; i < g\_count && i < MAX\_SOCKET\_PER\_THREAD; i++)

        {

            printf("Thread %d: Add socket %d (%d) to the read set\n", GetCurrentThreadId(), i, g\_client[i]);

            FD\_SET(g\_client[i], &read);

        }

        select(0, &read, NULL, NULL, NULL);

        if (FD\_ISSET(s, &read))

        {

            SOCKET c;

            struct sockaddr\_in caddr;

            int clen = sizeof(caddr);

            c = accept(s, (sockaddr\*)&caddr, &clen);

            printf("Socket %d connected\n", c);

            g\_client[g\_count] = c;

            g\_count += 1;

            if (g\_count > 1 && ((g\_count - 1) % MAX\_SOCKET\_PER\_THREAD == 0))

            {

                DWORD ID = 0;

                CreateThread(NULL, 0, ClientThread, (LPVOID)(g\_count - 1), 0, &ID);

            }

        }

        for (int i = 0; i < g\_count; i++)

        {

            if (FD\_ISSET(g\_client[i], &read))

            {

                char buffer[1024];

                memset(buffer, 0, sizeof(buffer));

                recv(g\_client[i], buffer, sizeof(buffer), 0);

                printf("Socket %d: %s\n", g\_client[i], buffer);

                for (int j = 0; j < g\_count; j++)

                {

                    if (j != i)

                    {

                        send(g\_client[j], buffer, strlen(buffer), 0);

                    }

                }

            }

        }

    }

    return 0;

}

}

# SELECT EXAMPLE

#include "stdafx.h"

#include <WinSock2.h>

SOCKET g\_client[1024];

int n\_client = 0;

int main()

{

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

SOCKADDR\_IN saddr;

saddr.sin\_addr.S\_un.S\_addr = 0;

saddr.sin\_port = htons(5000);

saddr.sin\_family = AF\_INET;

bind(s, (sockaddr\*)&saddr, sizeof(saddr));

listen(s, 10);

fd\_set fdread;

while (0 == 0)

{

FD\_ZERO(&fdread);

FD\_SET(s, &fdread);

for (int i = 0; i < n\_client; i++)

{

FD\_SET(g\_client[i], &fdread);

}

select(0, &fdread, NULL, NULL, NULL);

if (FD\_ISSET(s, &fdread))

{

SOCKADDR\_IN caddr;

int clen = sizeof(caddr);

SOCKET c = accept(s, (sockaddr\*)&caddr, &clen);

g\_client[n\_client++] = c;

}

else

{

for (int i = 0; i < n\_client; i++)

{

if (FD\_ISSET(g\_client[i], &fdread))

{

char buffer[1024];

memset(buffer, 0, 1024);

recv(g\_client[i], buffer, 1023, 0);

printf("%s\n", buffer);

}

}

}

}

return 0;

}

# SIMPLE SEVER

#include <stdio.h>

#include <WinSock2.h>

void main()

{

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

SOCKADDR\_IN saddr;

saddr.sin\_family = AF\_INET;

saddr.sin\_port = htons(8888);

saddr.sin\_addr.S\_un.S\_addr = 0;

bind(s, (sockaddr\*)&saddr, sizeof(SOCKADDR\_IN));

listen(s, 10);

SOCKADDR\_IN caddr;

int clen = sizeof(SOCKADDR\_IN);

SOCKET c = accept(s, (sockaddr\*)&caddr, &clen);

char\* hello = "Hello Network Programming";

int r = send(c, hello, strlen(hello), 0);

char\* buffer = NULL;

int blen = 0;

while (0 == 0)

{

char ch[3];

r = recv(c, ch, 3, 0);

if (r > 0)

{

buffer = (char\*)realloc(buffer, blen + r + 1);

memcpy(buffer + blen, ch, r);

buffer[blen + r] = 0; //Guard

blen += r + 1;

}

if (r < sizeof(ch))

break;

}

if (blen > 0)

{

printf("%s\n", buffer);

}

free(buffer);

buffer = NULL;

closesocket(c);

closesocket(s);

}

# SIMPLE CILENT

#include <stdio.h>

#include <WinSock2.h>

void main()

{

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKET c = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

SOCKADDR\_IN saddr;

saddr.sin\_family = AF\_INET;

saddr.sin\_port = htons(80);

saddr.sin\_addr.S\_un.S\_addr = inet\_addr("216.58.200.14");

int r = connect(c, (sockaddr\*)&saddr, sizeof(saddr));

char\* command = "GET / HTTP1.1\r\n\r\n";

r = send(c, command, strlen(command), 0);

char buffer[1024];

memset(buffer, 0, sizeof(buffer));

r = recv(c, buffer, sizeof(buffer) - 1, 0);

if (r > 0)

{

printf("%s", buffer);

}

closesocket(c);

}

# SIMPLE CHAT SILENT

#include <WinSock2.h>

#include <Windows.h>

#include <string.h>

DWORD WINAPI ClientThread(LPVOID arg);

SOCKET g\_clients[1024]; //Toi da 1024 client

int n = 0; //So luong client hien dang ket noi

int g\_status[1024]; //Trang thai client:0 = ngat ket noi, 1 = dang ket noi

void main()

{

WSADATA DATA;

WSAStartup(MAKEWORD(2, 2), &DATA);

SOCKET s = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);

SOCKADDR\_IN saddr;

saddr.sin\_family = AF\_INET;

saddr.sin\_addr.S\_un.S\_addr = 0;

saddr.sin\_port = htons(8888);

bind(s, (sockaddr\*)&saddr, sizeof(saddr));

listen(s, 10);

while (0 == 0) //Vong lap doi ket noi

{

SOCKADDR\_IN caddr;

int clen = sizeof(caddr);

SOCKET c = accept(s, (sockaddr\*)&caddr, &clen);

g\_clients[n] = c; //Luu vao danh sach clients

g\_status[n++] = 1; //Dang ket noi

DWORD ID = 0;

CreateThread(NULL, 0, ClientThread, (LPVOID)c, 0, &ID);

}

}

DWORD WINAPI ClientThread(LPVOID arg)

{

SOCKET c = (SOCKET)arg;

char name[1024];

char\* welcome = "Hay dang nhap bang cach gui lenh \"Name: <Ten>\"\n";

send(c, welcome, strlen(welcome), 0);

//Nhan du lieu va kiem tra cu phap dang nhap

memset(name, 0, 1024);

while (0 == 0)

{

char buffer[1024];

memset(buffer, 0, 1024);

recv(c, buffer, 1023, 0);

if (strncmp(buffer, "Name: ", 6) == 0)

{

strcpy(name, buffer + 6);

if (strlen(name) > 0)

{

while (name[strlen(name) - 1] == '\n' ||

name[strlen(name) - 1] == '\r' ||

name[strlen(name) - 1] == '\t' ||

name[strlen(name) - 1] == ' ')

{

name[strlen(name) - 1] = 0;

}

}

if (strlen(name) == 0)

strcpy(name, "Anonymous");

break;

}

else

{

send(c, welcome, strlen(welcome), 0);

}

}

//Nhan du lieu va forward den cac client khac

while (0 == 0)

{

char buffer[1024];

memset(buffer, 0, 1024);

recv(c, buffer, 1023, 0);

if (strlen(buffer) > 0)

{

while ( buffer[strlen(buffer) - 1] == '\r' ||

buffer[strlen(buffer) - 1] == '\n' ||

buffer[strlen(buffer) - 1] == '\t' ||

buffer[strlen(buffer) - 1] == ' ')

{

buffer[strlen(buffer) - 1] = 0;

}

}

if (strcmp(buffer, "quit") == 0)

{

int i = 0;

for (i = 0; i < n; i++)

{

if (g\_clients[i] == c)

break;

}

g\_status[i] = 0; //Offline

break;

}

else

{

for (int i = 0; i < n; i++)

{

if (g\_clients[i] != c && g\_status[i] == 1)

{

char data[1024];

memset(data, 0, 1024);

wsprintfA(data, "%s: %s\n", name, buffer);

send(g\_clients[i], data, strlen(data), 0);

}

}

}

}

return 0;

}