

数据通信作业-1

姓名: 费扬 学号: 519021910917 日期: 2022.04.13

一、实验名称及内容

Assignment1: GetHostInfo

使用 winsock 编程，主要目标任务为：

Task 1: Write a sockets program to get the host name for a given IP address.

Project name: gethostname

Command: gethostname (printing out the host name of your local computer)

Command: gethostname xxx.xxx.xxx.x (printing out the host name of a remote host)

Task 2: Write a sockets program to get the IP address of a given host name.

Project name: gethostaddress

Command: gethostaddress www.sjtu.edu.cn

二、实验过程和结果

Task1:

基本操作步骤：

1. 首先要 Creating a Basic Winsock Application
2. Initializing Winsock
3. Get the host information of a local or remote computer
4. 打印信息关闭窗口

```
int gethostname(
    _Out_ char* name, //A pointer to a buffer that receives the local host name.
    _In_ int namelen //The length, in bytes, of the buffer pointed to by the name parameter
);

int WINAPI getnameinfo(
    _In_ const struct sockaddr FAR* sa,
    _In_ socklen_t salen,
    _Out_ char FAR* host,
    _In_ DWORD hostlen,
    _Out_ char FAR* serv,
    _In_ DWORD servlen,
    _In_ int flags
);

int main(int argc, char* argv[]) {
    WORD wVersion = MAKEWORD(2, 2); // Used to request version 2.2 of Windows sockets
    WSADATA wsaData;                // Data loaded by WSStartup
    int iResult;                     // Error check if WSStartup successful

    // Initialize Winsock
    iResult = WSStartup(wVersion, &wsaData);
    if (iResult != 0) {
        cout << "WSStartup failed: " << iResult << endl;
        return 1;
    }
}
```

前两步操作的代码

代码同实验手册中，不予展示：
在命令行操作输出：

```
E:\Desktop\秃头文件\●数据通信\assignment\assignment1\gethostname\x64\Debug>gethostname.exe
LAPTOP-FEIJANG
usage: gethostname.exe IPv4 address
       to return local hostname
       gethostname.exe 127.0.0.1

E:\Desktop\秃头文件\●数据通信\assignment\assignment1\gethostname\x64\Debug>gethostname.exe 127.0.0.1
LAPTOP-FEIJANG
getnameinfo returned hostname = localhost.sangfor.com.cn

E:\Desktop\秃头文件\●数据通信\assignment\assignment1\gethostname\x64\Debug>gethostname.exe 8.8.8.8
LAPTOP-FEIJANG
getnameinfo returned hostname = dns.google

E:\Desktop\秃头文件\●数据通信\assignment\assignment1\gethostname\x64\Debug>gethostname.exe 8.8.4.4
LAPTOP-FEIJANG
getnameinfo returned hostname = dns.google

E:\Desktop\秃头文件\●数据通信\assignment\assignment1\gethostname\x64\Debug>gethostname.exe 192.168.1.111
LAPTOP-FEIJANG
getnameinfo returned hostname = DESKTOP-04AC58P

E:\Desktop\秃头文件\●数据通信\assignment\assignment1\gethostname\x64\Debug>gethostname.exe 192.168.1.102
LAPTOP-FEIJANG
getnameinfo returned hostname = SERGIO

E:\Desktop\秃头文件\●数据通信\assignment\assignment1\gethostname\x64\Debug>
```

这里可以看到自己主机的 name，google，和同一局域网下的其它设备（一台是我的另一台 ubuntu，一台是室友的 mac）。

Task2:

基本操作步骤：

1. 首先要 Creating a Basic Winsock Application
2. Initializing Winsock
3. Get the address information of a local or remote host
4. 打印信息关闭窗口

```
/*int WINAPI getaddrinfo(
    _In_opt_ PCSTR pNodeName, //host (node) name or a numeric host address string
    _In_opt_ PCSTR pServiceName, //a service name or port number represented as a string
    _In_opt_ const ADDRINFOA* pHints, // provides hints about the type of socket the caller supports
    _Out_ PADDRINFOA* ppResult // A pointer to a linked list of one or more addrinfo structures
);

typedef struct addrinfo {
    int ai_flags;
    int ai_family;
    int ai_socktype;
    int ai_protocol;
    size_t ai_addrlen;
    char* ai_canonname;
    struct sockaddr* ai_addr;
    struct addrinfo* ai_next;
}; ADDRINFOA * PADDRINFOA;*/

int main(int argc, char* argv[]) {
    WORD wVersion = MAKEWORD(2, 2); // Used to request version 2.2 of Windows sockets
    WSADATA wsaData; // Data loaded by WSStartup
    int iResult; // Error check if WSStartup successful

    DWORD dwRetVal;

    // Initialize Winsock
    iResult = WSStartup(wVersion, &wsaData);
    if (iResult != 0) {
        cout << "WSStartup failed: " << iResult << endl;
        return 1;
    }
}
```

注意上面注释部分是封装好的不需要写，直接调用 api 就行

```

E:\Desktop\秃头文件\数据通信\assignment\assignment1\gethostaddr\x64\Debug>gethostaddr www.sjtu.edu.cn
usage: gethostaddr <hostname> <servicename>
        provides protocol-independent translation
        from an ANSI host name to an IP address
gethostaddr example usage
gethostaddr www.contoso.com 0

E:\Desktop\秃头文件\数据通信\assignment\assignment1\gethostaddr\x64\Debug>gethostaddr www.sjtu.edu.cn 80
getaddrinfo returned success
getaddrinfo response 1
    Flags: 0x0
    Family: AF_INET (IPv4)
    IPv4 address 202.120.2.119
    port = 80
    Socket type: SOCK_STREAM (stream)
    Protocol: IPPROTO_TCP (TCP)
    Length of this sockaddr: 16
    Canonical name: (null)

E:\Desktop\秃头文件\数据通信\assignment\assignment1\gethostaddr\x64\Debug>gethostaddr www.baidu.com 80
getaddrinfo returned success

E:\Desktop\秃头文件\数据通信\assignment\assignment1\gethostaddr\x64\Debug>gethostaddr www.baidu.com 80
getaddrinfo returned success
getaddrinfo response 1
    Flags: 0x0
    Family: AF_INET (IPv4)
    IPv4 address 112.80.243.75
    port = 80
    Socket type: SOCK_STREAM (stream)
    Protocol: IPPROTO_TCP (TCP)
    Length of this sockaddr: 16
    Canonical name: (null)
getaddrinfo response 2
    Flags: 0x0
    Family: AF_INET (IPv4)
    IPv4 address 112.80.243.76
    port = 80
    Socket type: SOCK_STREAM (stream)
    Protocol: IPPROTO_TCP (TCP)
    Length of this sockaddr: 16
    Canonical name: (null)

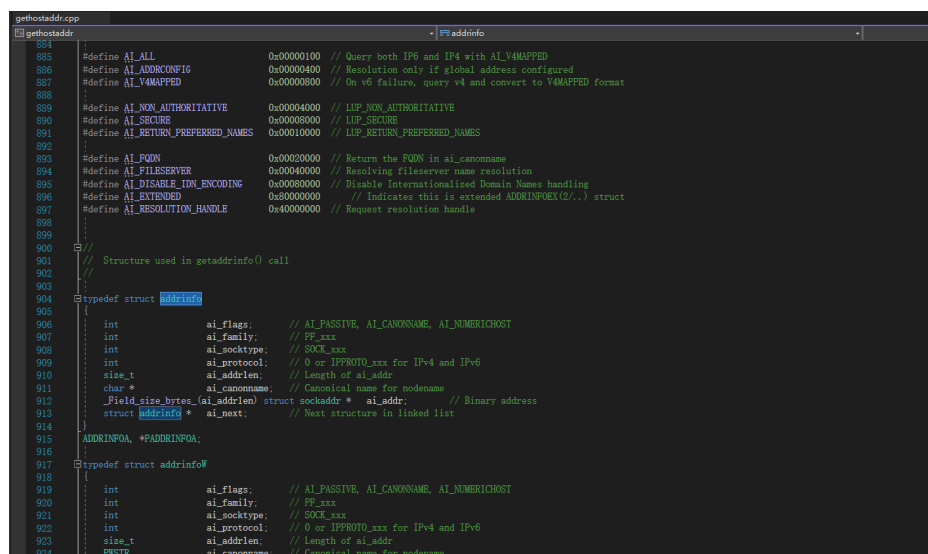
E:\Desktop\秃头文件\数据通信\assignment\assignment1\gethostaddr\x64\Debug>

```

这里可以看到 IP address of given host names (www.sjtu.edu.cn), 以及其他网站如 www.baidu.com 的 addrinfo, 包括 Flags, 端口, 协议, 网络规范名等。

三、问题与思考

1. Gethostaddr 中的 struct 不需要再定义, 在 winsock 中已经有定义。



```

gethostaddr.cpp
1884
1885 #define AI_ALL 0x00000100 // Query both IPv6 and IPv4 with AI_NUMERICSERV
1886 #define AI_ADDRCONFIG 0x00000400 // Resolution only if global address configured
1887 #define AI_V4MAPPED 0x00000800 // On v6 failure, query v4 and convert to V4MAPPED format
1888
1889 #define AI_NUM_AUTHORITATIVE 0x00004000 // LIP_NUM_AUTHORITATIVE
1890 #define AI_SECURE 0x00008000 // LIP_SECURE
1891 #define AI_RETURN_PREFERRED_NAMES 0x00010000 // LIP_RETURN_PREFERRED_NAMES
1892
1893 #define AI_FQDN 0x00020000 // Return the FQDN in ai_canonname
1894 #define AI_FILESERVER 0x00040000 // Resolving fileserver name resolution
1895 #define AI_DISABLE_IDN_ENCODING 0x00080000 // Disable Internationalized Domain Names handling
1896 #define AI_EXTENDED 0x80000000 // Indicates this is extended ADDRINFOEX(2/...) struct
1897 #define AI_RESOLUTION_HANDLE 0x40000000 // Request resolution handle
1898
1899
1900 //
1901 // Structure used in getaddrinfo() call
1902 //
1903
1904 #typedef struct ADDRINFOA
1905 {
1906     int ai_flags; // AI_PASSIVE, AI_CANONNAME, AI_NUMERICSERV
1907     int ai_family; // PF_XXX
1908     int ai_socktype; // SOCK_XXX
1909     int ai_protocol; // 0 or IPPROTO_XXX for IPv4 and IPv6
1910     size_t ai_addrlen; // Length of ai_addr
1911     char * ai_canonname; // Canonical name for nodename
1912     _Field_size_bytes_(ai_addrlen) struct sockaddr * ai_addr; // Binary address
1913     struct ADDRINFOA * ai_next; // Next structure in linked list
1914 }
1915 ADDRINFOA, *PADDRINFOA;
1916
1917 #typedef struct addrinfoW
1918 {
1919     int ai_flags; // AI_PASSIVE, AI_CANONNAME, AI_NUMERICSERV
1920     int ai_family; // PF_XXX
1921     int ai_socktype; // SOCK_XXX
1922     int ai_protocol; // 0 or IPPROTO_XXX for IPv4 and IPv6
1923     size_t ai_addrlen; // Length of ai_addr
1924     WCHAR * ai_canonname; // Canonical name for nodename

```

2. #include "pch.h" 是 visual studio 的预编译文件, 在 properties 中设置取消预编译或者直接注释掉这一行就行, 这一点在 tcpclient 和 tcpserver 的 demo 中也一样。
3. 第一次实验, 程序内容比较简单, 感谢老师的讲解! 祝疫情期间一切平安顺利!