https://www.cnblogs.com/L-hq815/archive/2012/08/04/2622829.html

一台机器上可能不只有一个网卡，但每一个网卡只有一个MAC地址，而每一个网卡可能配置有多个IP地址；如平常的笔记本电脑中，就会有无线网卡和有线网卡（网线接口）两种；因此，如果要获得本机所有网卡的IP和MAC地址信息，则必须顺序获得每个网卡，再依次获取其信息等；在windows sdk中，用IP\_ADAPTER\_INFO结构体存储网卡信息，包括网卡名、网卡描述、网卡MAC地址、网卡IP等，该结构体的主要描述如下所示：

[复制代码](javascript:void(0);)

typedef struct \_IP\_ADAPTER\_INFO {

　　struct \_IP\_ADAPTER\_INFO\* Next;//指向链表中下一个适配器信息的指针

　　DWORD ComboIndex;//预留值

　　char AdapterName[MAX\_ADAPTER\_NAME\_LENGTH + 4];//使用ANSI字符串表示的适配器名称

　　char Description[MAX\_ADAPTER\_DESCRIPTION\_LENGTH + 4];//使用ANSI字符串表示的适配器描述

　　UINT AddressLength;//适配器硬件地址以字节计算的长度

　　BYTE Address[MAX\_ADAPTER\_ADDRESS\_LENGTH];//硬件地址以BYTE数组所表示

　　DWORD Index;//适配器索引

UINT Type;//适配器类型,主要有以下几种：

/\*

\* MIB\_IF\_TYPE\_OTHER 1

\* MIB\_IF\_TYPE\_ETHERNET 6

\* MIB\_IF\_TYPE\_TOKENRING 9

\* MIB\_IF\_TYPE\_FDDI 15

\* MIB\_IF\_TYPE\_PPP 23

\* MIB\_IF\_TYPE\_LOOPBACK 24

\* MIB\_IF\_TYPE\_SLIP 28

\*/

　　UINT DhcpEnabled;//指定这个适配器是否开启DHCP

　　PIP\_ADDR\_STRING CurrentIpAddress;//预留值

　　IP\_ADDR\_STRING IpAddressList;//该适配器的IPv4地址链表

　　IP\_ADDR\_STRING GatewayList;//该适配器的网关IPv4地址链表

　　IP\_ADDR\_STRING DhcpServer;//该适配器的DHCP服务器的IPv4 地址链表

　　BOOL HaveWins;

　　IP\_ADDR\_STRING PrimaryWinsServer;

　　IP\_ADDR\_STRING SecondaryWinsServer;

　　time\_t LeaseObtained;

　　time\_t LeaseExpires;

　　} IP\_ADAPTER\_INFO,\*PIP\_ADAPTER\_INFO;

[复制代码](javascript:void(0);)

由于可能有多个网卡，因此struct \_IP\_ADAPTER\_INFO\* Next字段为一个链表结构指针，由于一个网卡可能有多个IP，因此IP\_ADDR\_STRING字段应该也是一个链表结构，其信息如下所示：

[复制代码](javascript:void(0);)

typedef struct \_IP\_ADDR\_STRING

{

struct \_IP\_ADDR\_STRING\* Next; //指向同类型节点，即下一个IP（如果有多IP的话）

IP\_ADDRESS\_STRING IpAddress; //IP地址信息

IP\_MASK\_STRING IpMask; //IP子网掩码

DWORD Context;// 网络表入口。这个值对应着AddIPAddredd和DeleteIPAddress函数中的NTEContext参数

} IP\_ADDR\_STRING;

[复制代码](javascript:void(0);)

在基本了解以上信息后，就可以调用GetAdaptersInfo函数来获取相关网卡信息了，其通用的代码如下所示：

[复制代码](javascript:void(0);)

#include <WinSock2.h>

#include <Iphlpapi.h>

#include <iostream>

using namespace std;

#pragma comment(lib,"Iphlpapi.lib") //需要添加Iphlpapi.lib库

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

{

//PIP\_ADAPTER\_INFO结构体指针存储本机网卡信息

PIP\_ADAPTER\_INFO pIpAdapterInfo = new IP\_ADAPTER\_INFO();

//得到结构体大小,用于GetAdaptersInfo参数

unsigned long stSize = sizeof(IP\_ADAPTER\_INFO);

//调用GetAdaptersInfo函数,填充pIpAdapterInfo指针变量;其中stSize参数既是一个输入量也是一个输出量

int nRel = GetAdaptersInfo(pIpAdapterInfo,&stSize);

//记录网卡数量

int netCardNum = 0;

//记录每张网卡上的IP地址数量

int IPnumPerNetCard = 0;

if (ERROR\_BUFFER\_OVERFLOW == nRel)

{

//如果函数返回的是ERROR\_BUFFER\_OVERFLOW

//则说明GetAdaptersInfo参数传递的内存空间不够,同时其传出stSize,表示需要的空间大小

//这也是说明为什么stSize既是一个输入量也是一个输出量

//释放原来的内存空间

delete pIpAdapterInfo;

//重新申请内存空间用来存储所有网卡信息

pIpAdapterInfo = (PIP\_ADAPTER\_INFO)new BYTE[stSize];

//再次调用GetAdaptersInfo函数,填充pIpAdapterInfo指针变量

nRel=GetAdaptersInfo(pIpAdapterInfo,&stSize);

}

if (ERROR\_SUCCESS == nRel)

{

//输出网卡信息

//可能有多网卡,因此通过循环去判断

while (pIpAdapterInfo)

{

cout<<"网卡数量："<<++netCardNum<<endl;

cout<<"网卡名称："<<pIpAdapterInfo->AdapterName<<endl;

cout<<"网卡描述："<<pIpAdapterInfo->Description<<endl;

switch(pIpAdapterInfo->Type)

{

case MIB\_IF\_TYPE\_OTHER:

cout<<"网卡类型："<<"OTHER"<<endl;

break;

case MIB\_IF\_TYPE\_ETHERNET:

cout<<"网卡类型："<<"ETHERNET"<<endl;

break;

case MIB\_IF\_TYPE\_TOKENRING:

cout<<"网卡类型："<<"TOKENRING"<<endl;

break;

case MIB\_IF\_TYPE\_FDDI:

cout<<"网卡类型："<<"FDDI"<<endl;

break;

case MIB\_IF\_TYPE\_PPP:

printf("PP\n");

cout<<"网卡类型："<<"PPP"<<endl;

break;

case MIB\_IF\_TYPE\_LOOPBACK:

cout<<"网卡类型："<<"LOOPBACK"<<endl;

break;

case MIB\_IF\_TYPE\_SLIP:

cout<<"网卡类型："<<"SLIP"<<endl;

break;

default:

break;

}

cout<<"网卡MAC地址：";

for (DWORD i = 0; i < pIpAdapterInfo->AddressLength; i++)

if (i < pIpAdapterInfo->AddressLength-1)

{

printf("%02X-", pIpAdapterInfo->Address[i]);

}

else

{

printf("%02X\n", pIpAdapterInfo->Address[i]);

}

cout<<"网卡IP地址如下："<<endl;

//可能网卡有多IP,因此通过循环去判断

IP\_ADDR\_STRING \*pIpAddrString =&(pIpAdapterInfo->IpAddressList);

do

{

cout<<"该网卡上的IP数量："<<++IPnumPerNetCard<<endl;

cout<<"IP 地址："<<pIpAddrString->IpAddress.String<<endl;

cout<<"子网地址："<<pIpAddrString->IpMask.String<<endl;

cout<<"网关地址："<<pIpAdapterInfo->GatewayList.IpAddress.String<<endl;

pIpAddrString=pIpAddrString->Next;

} while (pIpAddrString);

pIpAdapterInfo = pIpAdapterInfo->Next;

cout<<"--------------------------------------------------------------------"<<endl;

}

}

//释放内存空间

if (pIpAdapterInfo)

{

delete pIpAdapterInfo;

}

return 0;

}

[复制代码](javascript:void(0);)

执行结果：

