

Forum:[Hack Forums](#)

Thread: [Is it normal in C++ to be constantly converting variable types?](#)

gurlhackerclub - April 23, 2020, 11:52 am

Is it normal in C++ to be constantly converting variable types?

For instance, I want to get the computer name and send that variable through a socket. I find the function GetComputerName which gives me what I need in TCHAR.

I use the send function and it needs the computername variable in const char.

Is it normal to have to be jumping around like this?

Here is an example of code. Doesn't work because tchar to const char issue.

```
Code #ifndef WIN32_LEAN_AND_MEAN
#define WIN32_LEAN_AND_MEAN
#endif
```

```
#include <windows.h>
#include <winsock2.h>
#include <ws2tcpip.h>
#include <iphlpapi.h>
#include <tchar.h>
#include <stdio.h>
```

```
#pragma comment(lib, "Ws2_32.lib")
#define DEFAULT_BUFLen 512
#define DEFAULT_PORT 27015
#define INADDR "127.0.0.1"
```

```
int main(void)
{
    TCHAR compnameBuf[MAX_COMPUTERNAME_LENGTH + 2];
    DWORD compnameBufSize;

    compnameBufSize = sizeof compnameBuf - 1;
    if (GetComputerName(compnameBuf, &compnameBufSize) == TRUE) {
        _tprintf(_T("Your computer name is %s\n"), compnameBuf);
    }

    WSADATA wsaData;
    int iResult;
    SOCKET ConnectSocket = INVALID_SOCKET;
    struct sockaddr_in clientService;

    int recvbuflen = DEFAULT_BUFLen;
    const char* sendbuf = "Client sending data test";
    char recvbuf[DEFAULT_BUFLen] = "";
```

```

// Initialize Winsock
iResult = WSASStartup(MAKEWORD(2, 2), &wsaData);
if (iResult != 0) {
    printf("WSAStartup failed: %d\n", iResult);
    return 1;
}

// Create a SOCKET for connecting to server
ConnectSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
if (ConnectSocket == INVALID_SOCKET) {
    wprintf(L"socket failed with error: %ld\n", WSAGetLastError());
    WSACleanup();
    return 1;
}

//-----
// The sockaddr_in structure specifies the address family,
// IP address, and port of the server to be connected to.
clientService.sin_family = AF_INET;
clientService.sin_addr.s_addr = htonl(INADDR_LOOPBACK);
clientService.sin_port = htons(DEFAULT_PORT);

//-----
// Connect to server.
iResult = connect(ConnectSocket, (SOCKADDR*)&clientService, sizeof(clientService));

if (iResult == SOCKET_ERROR) {
    wprintf(L"connect failed with error: %d\n", WSAGetLastError());
    closesocket(ConnectSocket);
    WSACleanup();
    return 1;
}
printf("test");
//-----
// Send an initial buffer
char sendcomputer = compnameBuf;
iResult = send(ConnectSocket, compnameBuf, (int)strlen(compnameBuf), 0);
if (iResult == SOCKET_ERROR) {
    wprintf(L"send failed with error: %d\n", WSAGetLastError());
    closesocket(ConnectSocket);
    WSACleanup();
    return 1;
}

printf("Bytes Sent: %d\n", iResult);
// shutdown the connection since no more data will be sent
iResult = shutdown(ConnectSocket, SD_SEND);
if (iResult == SOCKET_ERROR) {
    wprintf(L"shutdown failed with error: %d\n", WSAGetLastError());
    closesocket(ConnectSocket);
    WSACleanup();
}

```

```

        return 1;
    }
    // close the socket
    iResult = closesocket(ConnectSocket);
    if (iResult == SOCKET_ERROR) {
        wprintf(L"close failed with error: %d\n", WSAGetLastError());
        WSACleanup();
        return 1;
    }

    WSACleanup();
    return(0);
}

```

HFDone - April 23, 2020, 12:20 pm

and what if the name is unique characters
if you sure its english then ypu can convert it

gurlhackerclub - April 23, 2020, 2:02 pm

I think my answer is to learn C#. C++ would be great if I could find working examples. I found an example in C# in about 5 minutes that does what I need.

```

Code using System;
using System.Net.Sockets;
using System.Text;

namespace Client
{
    class Program
    {
        const int PORT_NO = 5000;
        const string SERVER_IP = "127.0.0.1";
        static void Main(string[] args)
        {
            ///---data to send to the server---
            // string textToSend = DateTime.Now.ToString();
            string textToSend = Environment.MachineName;
            ///---create a TcpClient object at the IP and port no.---
            TcpClient client = new TcpClient(SERVER_IP, PORT_NO);
            NetworkStream nwStream = client.GetStream();
            byte[] bytesToSend = ASCIIEncoding.ASCII.GetBytes(textToSend);

            ///---send the text---
            Console.WriteLine("Sending : " + textToSend);
            nwStream.Write(bytesToSend, 0, bytesToSend.Length);

            ///---read back the text---
            byte[] bytesToRead = new byte[client.ReceiveBufferSize];
            int bytesRead = nwStream.Read(bytesToRead, 0, client.ReceiveBufferSize);
            Console.WriteLine("Received : " + Encoding.ASCII.GetString(bytesToRead, 0, bytesRead));
            Console.ReadLine();
            client.Close();
        }
    }
}

```

```
}  
}  
}
```

dracx - April 23, 2020, 6:07 pm

I would advice to just send the raw data as it is to your server and let the server handle it. Try casting it to (const char *) when passing to the send function.

And btw, use the compnameBufSize on the send function, it contains the size/number of TCHARs in the comp name buffer, and TCHAR may be ascii or wchar (or something else) so do some preprocessor statement to properly handle all this stuff.

saltycracker - April 24, 2020, 12:22 am

You'll always have problems lining up data types when you cross api borders.

Note: You should invest some time investigating marshalling software if you are serious about server programming.

Securifier - April 24, 2020, 5:04 am

TCHAR is an alias to CHAR or WCHAR, following your compiler configuration.

One WCHAR is on 2 bytes, that means sizeof(WCHAR) == sizeof(short). It is UTF16
One CHAR is one byte long. It is UTF8

WCHAR is the standards of Windows, and you should use it.
CHAR is the standard on cross-platform C and C++, and Linux.

That explains the problems of conversion you are getting.

saltycracker - April 24, 2020, 6:27 am

(Apr 24, 2020 05:04 AM)Securifier Wrote: [\[CITE\]](#) TCHAR is an alias to CHAR or WCHAR, following your compiler configuration.

One WCHAR is on 2 bytes, that means sizeof(WCHAR) == sizeof(short). It is UTF16
One CHAR is one byte long. It is UTF8

WCHAR is the standards of Windows, and you should use it.
CHAR is the standard on cross-platform C and C++, and Linux.

That explains the problems of conversion you are getting.
Linux has CHAR? Do you mean char?

Securifier - April 24, 2020, 6:28 am

(Apr 24, 2020 06:27 AM)saltycracker Wrote: [\[CITE\]](#) Linux has CHAR? Do you mean char?
CHAR is a Windows.h alias of char, it is the same yes.

saltycracker - April 24, 2020, 6:36 am

(Apr 24, 2020 06:28 AM)Securifier Wrote: [\[CITE\]](#) CHAR is a Windows.h alias of char, it is the same yes.

You shouldn't use windows specific headers or source code in Linux. They were never meant to run on other platforms... That's why they are called Windows.h.

It would be the same as `unistd.h`(unix standard header file). You won't expect to use anything in `unistd.h` in a Windows program bc it was never designed to work on that platform.

Securifier - April 24, 2020, 7:02 am

(Apr 24, 2020 06:36 AM)saltycracker Wrote: [\[CITE\]](#) You shouldn't use windows specific headers or source code in Linux. They were never meant to run on other platforms... That's why they are called `Windows.h`.

It would be the same as `unistd.h`(unix standard header file). You won't expect to use anything in `unistd.h` in a Windows program bc it was never designed to work on that platform.

He is programming on Windows.

saltycracker - April 24, 2020, 7:51 am

Here, let me respond by quoting this person.

(Apr 24, 2020 05:04 AM)Securifier Wrote: [\[CITE\]](#) `CHAR` is the standard on cross-platform C and C++, and Linux.

gurlhackerclub - April 24, 2020, 1:03 pm

Much love and thanks for help.

I found a different example to get computer name in char format and got it to work.

Problem is I don't know how C++ works. I just copy examples. And C++ has too many old examples of code floating around.

```
Code char compname[MAX_COMPUTERNAME_LENGTH + 1];
    DWORD complength = sizeof(compname);

    bool ok = GetComputerNameExA((COMPUTER_NAME_FORMAT)0, compname, &complength);
    printf(compname);
```

I have no idea why there are three types of functions for everything.
`GetComputerNameEx`, `GetComputerNameExA`, `GetComputerNameExW`

dracx - April 25, 2020, 12:29 am

(Apr 24, 2020 01:03 PM)gurlhackerclub Wrote: [\[CITE\]](#) Much love and thanks for help.

I found a different example to get computer name in char format and got it to work.

Problem is I don't know how C++ works. I just copy examples. And C++ has too many old examples of code floating around.

```
Code
    char compname[MAX_COMPUTERNAME_LENGTH + 1];
    DWORD complength = sizeof(compname);

    bool ok = GetComputerNameExA((COMPUTER_NAME_FORMAT)0, compname, &complength);
    printf(compname);
```

I have no idea why there are three types of functions for everything.

GetComputerNameEx, GetComputerNameExA, GetComputerNameExWMan that's sad af, if you dont understand a concept, spend an hour or a day just to understand it, if you think you'll spend too much time on just learning one thing, I assure you that is wrong. Stop copy pasting code and try and understand why this line is this and why is that, then move onto the next line.

```
Code TCHAR szCompName[MAX_COMPUTERNAME_LENGTH + 1];
DWORD nSize = 0;
GetComputerNameA((char *)szCompName, &nSize);
```

```
#ifdef UNICODE
nSize = nSize * 2;
#endif
```

```
send(sock, (const char *)szCompName, nSize, flag);
Check this out, I've spared error checking and proper initializations.
```