

---

Subject: weird reaction with const w\_chart

Posted by [robbyke](#) on Fri, 09 Mar 2012 13:21:38 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

::onchat call

Toggle Spoiler

```
bool Kambot::OnChat(int PlayerID,TextMessageEnum Type,const wchar_t *Message,int
recieverID)
{
    Kambot_Commands(PlayerID,Type,Message,recieverID);

    return true;
}
```

start of the called function

Toggle Spoiler

```
void Kambot_Commands(int PlayerID,TextMessageEnum Type,const wchar_t *Message,int
recieverID)
{
    Console_Input("msg functie ok");
    StringClass Msg;
    Msg.Format("msg %s",Message);
    Console_Input(Msg);
    if (wcsstr(Message,L"!") != NULL)
    {
        Console_Input("msg message ok");
    }
}
```

keyhook function

Toggle Spoiler

```
void KB_keyhook::KeyHook()
{
    if((The_Game()->Get_Game_Duration_S() - LastPress) >= 1){
        StringClass Msg;
        const wchar_t* message = (const wchar_t*)Get_Parameter("Command");
        Kambot_Commands(Get_Player_ID(Owner()),TEXT_MESSAGE_TEAM,message,-1);
        Msg.Format("msg %s",(const wchar_t*)Get_Parameter("Command"));
        Console_Input(Msg);
        LastPress = The_Game()->Get_Game_Duration_S();
    }
}
ScriptRegistrant<KB_keyhook>
```

```
KB_keyhook_Registrant("KB_keyhook","Key:string,Command:string");
```

what happens:

if i use a command in game the command works.  
but the message is just a ! according to the game

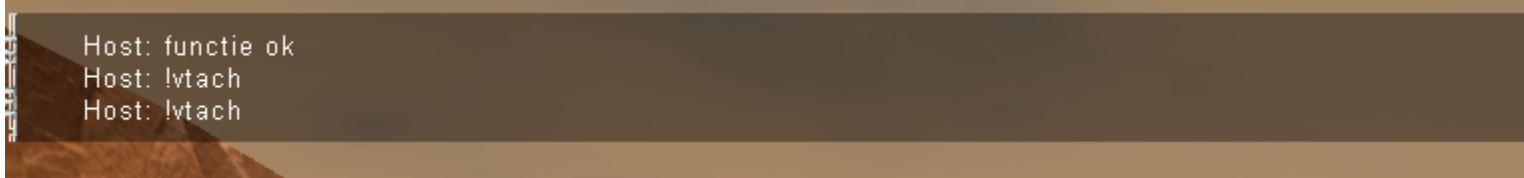
but when i use my chathook the message is more and the command wont work

now i wonder what i do wrong.  
probably my conversion.

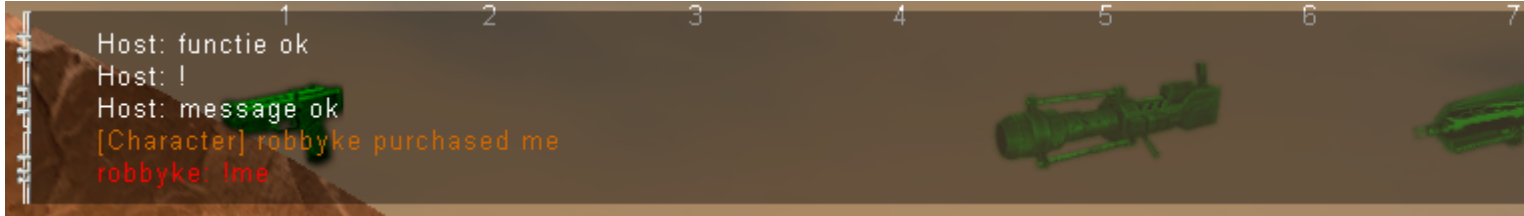
## File Attachments

---

1) [keyhook.png](#), downloaded 937 times



2) [onchat hook.png](#), downloaded 923 times



---

Subject: Re: weird reaction with const w\_chart  
Posted by [iRANian](#) on Fri, 09 Mar 2012 13:53:51 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

You're using %s with a wchar\_t in formatted functions, you need to use %S. Casting a char to wchar\_t (if it works) can cause memory corruption. Instead of using wchar\_t, use StringClass. Instead of using Console\_Input, use a wrapper function that takes formatted input like Console\_Output() does. This is how I would write the code:

Toggle Spoiler// Console\_Input() taking formatted input  
void Console(const char \*Format, ...)

```

{
char buffer[256];
va_list va;
_crt_va_start(va, Format);
vsnprintf(buffer, 256, Format, va);
va_end(va);
Console_Input(buffer);
}

// Call Kambot_Commands() via chat
bool Kambot::OnChat(int PlayerID,TextMessageEnum Type,const wchar_t *Message,int
recieverID)
{
StringClass Msg = Message;
Kambot_Commands(PlayerID, Type, Msg, recieverID);

return true;
}

void Kambot_Commands(int PlayerID,TextMessageEnum Type, StringClass Msg,int recieverID)
{
Console("MSG Debug: Kambot_Commands() called"); // DEBUG CRAP
Console("MSG Debug: Kambot_Commands() MSG == %s", Msg); // DEBUG CRAP
if (Msg[0] == '!')
{
Console("MSG Debug: Kambot_Commands() Command triggered"); // DEBUG CRAP
}
}

// Call Kambot_Commands() via a Keyhook (using keys.cfg keys)
void KB_keyhook::KeyHook()
{
if((The_Game()->Get_Game_Duration_S() - LastPress) >= 1)
{
StringClass Msg = Get_Parameter("Command");

Kambot_Commands(Get_Player_ID(Owner()), TEXT_MESSAGE_TEAM, Msg, -1); // Last
parameter isn't used
Console("MSG Debug: KeyHook() called with %s", Msg); // DEBUG CRAP

LastPress = The_Game()->Get_Game_Duration_S();
}
}

```

Not sure if it actually runs correctly though, didn't bother checking. You can also use the `__FUNCTION__` and `__LINE__` macros to grab the function the code is executing and the line number while debugging.

Subject: Re: weird reaction with const w\_chart  
Posted by [robbyke](#) on Fri, 09 Mar 2012 17:51:18 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

thanks this helped alot

made me understand stringclass better to

---

Subject: Re: weird reaction with const w\_chart  
Posted by [iRANian](#) on Sun, 11 Mar 2012 00:05:06 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

robbyke wrote on Fri, 09 March 2012 10:51thanks this helped alot

made me understand stringclass better to

Cheers no problem. I suggest you take a look at HashTemplateClass and HashTemplateliterator too, they provide hash maps/tables, also known as dictionaries in other languages. If you're gonna be building a bot that responds to commands, it's a nice idea to store them in a hash table, with as key the text that triggers the command and the value being a pointer to a class that inherits a generic chat command class, so you can call a function from a class when it is found in the hash table, which is done almost instantaneously, instead of having to go over a list of all triggers to trigger a specific function, which is very slow and makes the code to check what command to trigger very long-winded code.

I've written an example for this that I haven't tested, I hope this helps you:

derp.cpp:

```
#include "General.h"
```

```
#include "derp.h"
```

```
#include "HashTemplateClass.h"
```

```
#include "HashTemplateliterator.h"
```

```
// This macro creates a new Class on the heap and calls Startup for it
```

```
#define REGISTER_COMMAND(Class, Trigger, registrant) Class *registrant = new Class;
```

```
registrant->Startup(Trigger)
```

```
HashTemplateClass<StringClass, ChatCommand *> ChatCommand::CommandsMap;
```

```
void ChatCommand::Load()
```

```
{
```

```
REGISTER_COMMAND(Help, "!help", Help_Registrant); // Registers a new Help ChatCommand  
objectcalled with !help
```

```
REGISTER_COMMAND(Help, "!h", HelpAlias_Registrant); // Registers a new Help
```

```
ChatCommand object called with !h
```

```
}
```

```
void ChatCommand::Unload()
```

```

{
}

// This simply adds the command to our CommandsMap with the trigger, it can later be expanded
to do more
void ChatCommand::Startup(const char* Trigger)
{
    ChatCommand::CommandsMap.Insert(Trigger, this); // 'this' is a pointer to the current class
}

// This is a virtual method, this is used for the default for classes that don't have the Activate()
method
void ChatCommand::Activate(int ID, int Type, StringClass Msg)
{
    Console("MSG this command hasn't been implemented yet!");
}

// This is the only thing we need to implement for a new class inheriting from ChatCommand
void Help::Activate(int ID, int Type, StringClass Msg)
{
    Console("MSG This is a fancy help command!");
}

// Example chat hook code to get the above to work
bool KamBot::OnChat(int PlayerID, TextMessageEnum Type, const wchar_t *Message, int
recieverID)
{
    StringClass Msg = Message;

    if (ChatCommand::CommandsMap.Exists(Msg)) // Does our hash map contain a trigger that is
the same as the input message?
    { // If so
        ChatCommand* c = ChatCommand::CommandsMap.Get(Msg, 0); // Get the ChatCommand
pointer that's indexed for the chat message
        c->Activate(PlayerID, Type, Msg); // With our ChatCommand pointer called 'c', call the Activate()
function
    }

    return true;
}

derp.h:

class ChatCommand
{
public:

    // Static functions and data, dont need to be called from an object

```

```

static HashTemplateClass<StringClass, ChatCommand *> CommandsMap; // This is our hash
map, we put ChatCommand object pointers here
    // That are triggered by a StringClass trigger text

static void Load(); // This is our loading code, call this from somewhere
static void Unload(); // Unloads all chat commands related stuff

void Startup(const char* Trigger); // Loads a ChatCommand object and adds it to the hash table

virtual void Activate(int ID, int Type, StringClass Msg); // The default Activate() function called if
not defined by an inheriting class
};

// a class that inherits the ChatCommand class, this one implements the !help command
class Help : public ChatCommand
{
public:
void Activate(int ID, int Type, StringClass Msg); // The code to execute when this function is
called in the chat hook by its trigger
};

```

This code should respond to ingame chat that's "!h" or "!help", but I haven't tested it.

---



---

Subject: Re: weird reaction with const w\_chart  
Posted by [Ethenal](#) on Sun, 11 Mar 2012 01:28:48 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

iRANian wrote on Sat, 10 March 2012 18:05 If you're gonna be building a bot that responds to commands, it's a nice idea to store them in a hash table, with as key the text that triggers the command and the value being a pointer to a class that inherits a generic chat command class, so you can call a function from a class when it is found in the hash table, which is done almost instantaneously, instead of having to go over a list of all triggers to trigger a specific function, which is very slow and makes the code to check what command to trigger very long-winded code. Isn't that essentially the same thing? Even if you're using a hash table, does it not internally have to go down the list of values until it finds the one with the name you want?

---



---

Subject: Re: weird reaction with const w\_chart  
Posted by [iRANian](#) on Sun, 11 Mar 2012 12:19:35 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

No, it's basically an array where a hash function is used to index values with based on the key, and when you want to grab a value the index is calculated from the hash of the key.

See: [http://www.pcmag.com/encyclopedia\\_term/0,2542,t=hash+table&i=44129,00.asp](http://www.pcmag.com/encyclopedia_term/0,2542,t=hash+table&i=44129,00.asp)

A very simplified view of it would be like:

```
Array[hash_function(key)] = value;  
printf("%s", Array[hash_function(key)]);
```

Although it requires a lot of internal work and in C++ you need to use `std::unordered_map` (see [http://en.wikipedia.org/wiki/Unordered\\_map\\_\(C%2B%2B\)](http://en.wikipedia.org/wiki/Unordered_map_(C%2B%2B))) or in case of using the `scripts.dll` API `HashTemplateClass` can be used.

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [robbyke](#) on Sun, 11 Mar 2012 13:54:08 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

ok this is pretty funny the code i have used to work like that but with the 4.0 i had to modify it because `chatcommandclass` was gona.

now u understand how that works and ill have remake all the commands again

thanks alot this helps me great deal im really starting to understand everythin bit by bit

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [iRANian](#) on Sun, 11 Mar 2012 14:50:48 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

You'll need a tokenizer class to split the message into tokens, I ported the SSGM 2.0.2 tokenizer class to 4.0, if you need that I can give you it.

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [jonwil](#) on Sun, 11 Mar 2012 15:32:16 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Look at `CommandLineParser.h`, that is part of 4.0 and what the ban system plugin, `extraconsolecommands` plugin, mute plugin, spectate plugin and sudden death plugin are using.

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [robbyke](#) on Sun, 11 Mar 2012 15:34:08 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

i tried to convert it once but i didnt understand vector then(stil not)

so ye if you want to it would be nice

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [jonwil](#) on Sun, 11 Mar 2012 15:45:43 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

commandlineparser is already in 4.0.

---

Subject: Re: weird reaction with const w\_chart  
Posted by [iRANian](#) on Sun, 11 Mar 2012 16:03:15 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

robbyke wrote on Sun, 11 March 2012 08:34i tried to convert it once but i didnt understand vector then(stil not)

so ye if you want to it would be nice

Alright, it should work the same way the stuff in SSGM 2.0.2 worked like (accessing [0] gives you the full string, [1] gives you the first token). Not sure if there's any issues with it.

Tokenizer.cpp:

Toggle Spoiler#include "Tokenizer.h"

```
void Tokenizer::Build(const StringClass &Text, int Pos)
```

```
{  
    Tokens.Clear();  
    VectorSize = 0;  
    StringClass Temp2, All;
```

```
    StringClass Tokenz = Text;  
    char *p = strtok(Tokenz.Peek_Buffer(), " ");
```

```
    if (!Pos)  
    {  
        Tokens.Add(Text);  
    }  
    else  
    {  
        int i = 0;  
        while (i < Pos)  
        {  
            p = strtok(0, " ");  
            ++i;  
        }  
    }  
    while (p)  
    {  
        Temp2 = p;
```

```
Tokens.Add(Temp2);  
p = strtok(0, " ");  
++VectorSize;
```

```
if (Pos)  
{  
    All += Temp2;  
    if (p) All += " ";  
}
```

```
}  
if (Pos)  
{  
    Tokens.Add_Head(All);  
}  
}
```

```
Tokenizer::Tokenizer(const Tokenizer &Copy)  
{  
    Tokens = Copy.Tokens;  
    VectorSize = Copy.VectorSize;  
}
```

```
Tokenizer::Tokenizer()  
{  
}
```

```
Tokenizer::Tokenizer(const StringClass &Text, int Pos)  
{  
    Build(Text, Pos);  
}
```

```
Tokenizer& Tokenizer::operator=(const Tokenizer &Copy)  
{  
    Tokens = Copy.Tokens;  
    VectorSize = Copy.VectorSize;  
    return *this;  
}
```

```
Tokenizer& Tokenizer::operator=(const StringClass &Text)  
{  
    Build(Text,0);  
    return *this;  
}
```

```
StringClass Tokenizer::operator[](int Pos)  
{  
    if (VectorSize < Pos)  
    {
```

```

    return "";
}
return Tokens[Pos];
}

StringClass Tokenizer::operator()(int Start,int End)
{
if (VectorSize < Start || VectorSize < End)
{
return "";
}
StringClass Ret;
if (!End) {
End = Tokens.Count();
}
int i = Start;

while (i <= End && i <= VectorSize)
{
Ret += Tokens[i];
++i;
if (i <= End)
Ret += StringClass(" ");
}
return Ret;
}

int Tokenizer::Size()
{
return VectorSize;
}

void Tokenizer::Erase(int Pos)
{
if (VectorSize < Pos) return;

Tokens.Delete(Pos);
VectorSize--;
}

void Tokenizer::Replace(int Pos, const StringClass &Rep)
{
if (VectorSize < Pos || !Pos) return;

Tokens[Pos] = Rep;
}

/* inline void Tokenizer::Erase_Global(int Pos)

```

```

{
if (VectorSize < Pos) return;

StringClass Temp = Tokens[0];
Temp.Replace(Temp.find(Tokens[Pos]),Tokens[Pos].size()+1,"");
Tokens[0] = Temp;
Erase(Pos);
} */

```

```

inline void Tokenizer::Add(const StringClass &Text, int Pos)
{
if (!Pos)
{
Tokens.Add(Text);
++VectorSize;
}
else if (VectorSize < Pos)
{
return;
}
else
{
Tokens.Insert(Pos, Text);
++VectorSize;
}
}

```

Tokenizer.h:

Toggle Spoiler#pragma once

```
#include "gmpugin.h"
```

```
// First position in the Tokenizer string contains the full string, so use [1] instead of [0]
// for the first string
```

```

class Tokenizer
{
private:
DynamicVectorClass<StringClass> Tokens;
int VectorSize;
void Build(const StringClass &Text, int Pos);

public:
Tokenizer();
Tokenizer(const Tokenizer &Copy);
Tokenizer(const StringClass &Text, int Pos = 0);
Tokenizer& operator=(const Tokenizer &Copy);
Tokenizer& operator=(const StringClass &Text);

```

```
StringClass operator[](int Pos);
StringClass operator()(int Start, int End = 0);
int Size();
void Erase(int Pos);
void Replace(int Pos,const StringClass &Rep);
// void Erase_Global(int Pos);
void Add(const StringClass &Text, int Pos = 0);
};
```

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [iRANian](#) on Sun, 11 Mar 2012 16:07:19 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

If you modify Activate() to take a Tokenizer object instead of StringClass, you could write commands in a similar way to this:

```
void Ping::Activate(int ID, int Type, Tokenizer Msg)
{
if (Msg.Size() > 1)
{
int Count = Functions::Get_Part_Names_Fixed(Msg[2]);
if (Count < 1)
{
Functions::Page(ID,"Player not found.");
}
else if (Count > 1)
{
Functions::Page(ID, "Multiple players found.");
}
else
{
int OtherID = Get_Player_ID(Functions::Get_Part_Name_Fixed(Msg[2]));
Player_t* p = Player::Get(OtherID);
Functions::Page(ID,"%s's ping is %d.", p->Nick, Get_Ping(p->PlayerId));
}
}
else
{
Functions::Page(ID,"Your ping is %d.", Get_Ping(ID));
}
}
```

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [robbyke](#) on Sun, 11 Mar 2012 21:04:37 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

jonwil commandlineparser seems to be for console or at least those plugins use it for console commands while im bussy with ingame commands

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [jonwil](#) on Mon, 12 Mar 2012 02:18:13 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

CommandLineParser is usable for anything where you have a bunch of different items all separated by spaces.

so a string like "a 1.4 ddd 4" can be parsed by it.  
Or any string like that.  
Its not just for console commands.

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [Jerad2142](#) on Mon, 12 Mar 2012 13:26:47 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

This is a real rough/bad example of how I THINK it works; I'm looking at it on my break, have never used it before, and am not sure if you can stick string class into Console\_Input. However, implementing the tokenizer should be at least SOMEWHAT similar to this:

```
Tokenizer TokenString = new Tokenizer();  
TokenString = "Cool Story Bro";  
Console_Input(TokenString[3]);
```

If it works you should see "BRO" as an unknown command popup in pink text.

---

---

Subject: Re: weird reaction with const w\_chart  
Posted by [iRANian](#) on Mon, 12 Mar 2012 18:28:48 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

It just takes a char\* as input, e.g.

```
Tokenizer derp("yo 1 2");  
Console("MSG %s", derp[3]);
```

//Output is "2".

---