Subject: [SSGM 4.0 Plugin] RadioCommands
Posted by Xpert on Thu, 05 Jul 2012 02:14:02 GMT
View Forum Message <> Reply to Message

This is a simple SSGM 4.0 plugin that outputs when players use ingame Radio Commands.

This will be the first of many stuff I plan on releasing.


Example from my bot:
Quote:
<CloudyServ2> [Radio] &WNxSmiLey: Affirmative.
<CloudyServ2> [Radio] &WNxSmiLey: Negative.
<CloudyServ2> [Radio] &WNxSmiLey: Don't get in my way!
<CloudyServ2> [Radio] &WNxSmiLey: Don't get in my way!
<CloudyServ2> [Radio] &WNxSmiLey: Watch where you're pointing that!


Enjoy.

File Attachments
1) SSGM4.0_RadioCommands_Plugin_v2.rar, downloaded 214 times

---

Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by Sir Kane on Thu, 05 Jul 2012 04:38:06 GMT
View Forum Message <> Reply to Message

That'll spam you good!

---

Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by iRANian on Thu, 05 Jul 2012 07:06:03 GMT
View Forum Message <> Reply to Message

You could have it log invalid radio commands too.

---

Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by Whitedragon on Thu, 05 Jul 2012 08:40:49 GMT
View Forum Message <> Reply to Message

You're leaking memory. You could rewrite the entire thing as simply:


Console_Output("[Radio] %ls:
%ls\n",Get_Wide_Player_Name_By_ID(PlayerID),TranslateDBClass::Get_String(AnnouncementI

---

D));

---

## Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by iRANian on Thu, 05 Jul 2012 09:09:51 GMT
View Forum Message <> Reply to Message

What's the memory leak?

---

## Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by Whitedragon on Thu, 05 Jul 2012 09:12:13 GMT
View Forum Message <> Reply to Message

Using Get_Player_Name_By_ID.

---

## Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by iRANian on Thu, 05 Jul 2012 09:18:51 GMT
View Forum Message <> Reply to Message

You're right, should use Get_Wide_Player_Name() or Find_Player(PlayerID)->PlayerName, and change the '%s' to a '%S' in the format string.

---

## Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by cAmpa on Thu, 05 Jul 2012 11:23:39 GMT
View Forum Message <> Reply to Message

TT should remove WideCharToChar or replace it with this function.
This one doesn't create memleaks.

Quote:const char* WideCharToChar
 (const wchar_t* string)
{
 struct AutoDelete
 {
 public:
  char* pointer;

   AutoDelete() : pointer(0) {}
  ~AutoDelete()          { delete[] this->pointer; }
 };

 static AutoDelete   _autoDelete[10];

---

```
static unsigned int _index = 0;


 if (!string || !*string)
  return "";

 ++_index;
 if (_index >= 10)
  _index = 0;

 if (_autoDelete[_index].pointer)
  delete[] _autoDelete[_index].pointer;

 int length = ::wcslen (string);
 char* text = new char[length + 1];

 _autoDelete[_index].pointer = text;

 ::wcstombs (text, string, length + 1);

 return text;
}
```

---

## Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by Xpert on Thu, 05 Jul 2012 12:52:47 GMT
View Forum Message <> Reply to Message

Whitedragon wrote on Thu, 05 July 2012 04:40You're leaking memory. You could rewrite the entire thing as simply:


Console_Output("[Radio] %ls:
%ls\n",Get_Wide_Player_Name_By_ID(PlayerID),TranslateDBClass::Get_String(AnnouncementI
D));


If I remember correctly, Get_Wide_Player_Name_By_ID doesn't exist in SSGM 4.0, or am I not seeing it?

iRANian wrote on Thu, 05 July 2012 05:18You're right, should use Get_Wide_Player_Name() or Find_Player(PlayerID)->PlayerName, and change the '%s' to a '%S' in the format string.

I don't know why but in my private code, I did use Get_Wide_Player_Name() instead of Get_Player_Name_By_ID.

Updated 1st post with v2.

---

## Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by iRANian on Thu, 05 Jul 2012 12:56:26 GMT
View Forum Message <> Reply to Message

I didn't see it at first either although I always check scripts.dll API functions that return a string because some allocate on the heap while some don't. :/

---

## Subject: Re: [SSGM 4.0 Plugin] RadioCommands
Posted by Whitedragon on Thu, 05 Jul 2012 20:41:07 GMT
View Forum Message <> Reply to Message

```
const wchar_t *Get_Wide_Player_Name_By_ID(int ID) {
 cPlayer *x = Find_Player(ID);
 if (!x) {
  return L"None";
 }
 return x->PlayerName;
}
```

For things that are stored as wide chars internally, like player names and translated strings, it's always better to use the wide version of their function. This produces cleaner, slightly more efficient code that doesn't have a chance of leaking memory if you forget to delete it.

If you need them in a char for some reason, such as comparison, you can convert them like this:

StringClass Name = Get_Wide_Player_Name_By_ID(ID);

or

StringClass(Get_Wide_Player_Name_By_ID(ID))

And StringClass will automatically handle the conversion and memory allocation.

---