Subject: Writing hexadecimal to files
Posted by tjerk on Mon, 23 May 2011 12:07:45 GMT
View Forum Message <> Reply to Message

For a small utility I wanted to be able to write out hexadecimal values to a file. I can not change that file format (it's a config for a program I have no control over). I have deciphered the file so now I know which positions can be changed to what and I wanted to write a small utility that writes out a new .cfg file with different options selected.

I tried to do this by just making a String like so:

```
String myFile = "C:\\file.cfg";
String str = "\x05\x40\x03\x02\x01"; //just some test entries
SaveFile(myFile, str);
```

This works fine as long as I don't try to use:

```
str += "\x00";
```

This will terminate everything after. But I still need to write out that bit to fill up the spaces in the file that are not used. So ideally I should be able to make a string like this:

```
String str = "\x05\x00\x00\x00\x40\x00\x00\x00\x03\x02\x01";
```

But this does not work. The file only gets written up to the first occurrence of "\x00".

Am I going about it the wrong way? It was the only way I could think of with my limited programming knowledge and would appreciate a nudge into the right direction. Thanks in advance!

Subject: Re: Writing hexadecimal to files
Posted by dolik.rce on Mon, 23 May 2011 16:10:55 GMT
View Forum Message <> Reply to Message

Hi tjerk,

You are doing it almost right The problem is that in Upp::String (unlike std::string) \x00 terminates the string. One possible solution is to use Vector instead: Vector<byte> v;

```
v.Add('a');
v.Add(0);
v.Add(4);
```

```
v.Add('\x00');
v.Add('x');
FileOut f(myfile);
f.Put(v,v.GetCount);
f.Close();
```
Another, sometimes simpler, approach is to use the binary interface of FileOut, the Put* methods (the same goes for FileIn and Get* for binary reading):
```
FileOut f(myfile);
f.Put('a',1);
f.Put('\x00',3);
f.Put('b',1);
f.Close();
```
See Stream documentation for details.

Best regards,
Honza

Subject: Re: Writing hexadecimal to files
Posted by tjerk on Mon, 23 May 2011 20:19:03 GMT
View Forum Message <> Reply to Message

I have picked your second option and it works beautifully, thanks a lot Honza!

Subject: Re: Writing hexadecimal to files
Posted by koldo on Mon, 23 May 2011 20:22:42 GMT
View Forum Message <> Reply to Message

Hello Tjerk

You can also try this:

```
String str;
str.Cat("\x05\x00\x00\x00\x40\x00\x00\x00\x03\x02\x01", 11);
SaveFile("c:\\myfile.txt", str);
```

Subject: Re: Writing hexadecimal to files
Posted by nlneilson on Mon, 23 May 2011 23:51:29 GMT
View Forum Message <> Reply to Message

Something similar to this but not with hex was transferring a considerable amount of data through a socket without sending a line at a time was basically removing the '\0' and adding the text of the next line and \n then another line and \n.
When a set of lines were done then add \0.

Worked great that way once I got it figured out.

Without doing it that way a Sleep(10) was needed between each line so the data would not get scrambled or out of order.

I wrote code to create the string a character at a time and if the char was '\0' (ascii null char(0)) then discard it.

Does the Upp str.Cat work in a similar way to remove the '\0'?

And is the length always required as in your example "11"?

Counting the hex characters is 11 so shouldn't it be 12 including the \0 rather than 11 or am I missing something?

I thought all strings ended with a \0 character.

Subject: Re: Writing hexadecimal to files
Posted by koldo on Tue, 24 May 2011 06:44:28 GMT
View Forum Message <> Reply to Message

Hello

U++ String handle very well binary data.

For example

String binaryC = binaryA + binaryB;

works perfectly.

The problem Tjerk had was when filling the String. Using this:

String a = "text";

does not serve for binary String, only for text, as far as it has no \0 inside.

The reason is that, with:

String a = "abc\0def";

example, Cat();

Subject: Re: Writing hexadecimal to files
Posted by nlneilson on Tue, 24 May 2011 09:27:53 GMT

Good point, I had not checked into that.