Audacious - Bug #419

крашится плеер

March 16, 2014 05:44 - Денис Потапов

Status:	Rejected	**Start date:**	March 16, 2014
Priority:	Major	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	3.5		
Affects version:	3.5-alpha1		

Description

запустил плеер. создал плейлист, плеер крашится

```
Имя события проблемы:    APPCRASH
  Имя приложения:    audacious.exe
  Версия приложения:    0.0.0.0
  Отметка времени приложения:    5309071b
  Имя модуля с ошибкой:    libglib-2.0-0.dll
  Версия модуля с ошибкой:    2.38.2.0
  Отметка времени модуля с ошибкой:    53090718
  Код исключения:    c0000005
  Смещение исключения:    0003ea69
  Версия ОС:    6.1.7601.2.1.0.256.1
  Код языка:    1049
  Дополнительные сведения 1:    0a9e
  Дополнительные сведения 2:    0a9e372d3b4ad19135b953a78882e789
  Дополнительные сведения 3:    0a9e
  Дополнительные сведения 4:    0a9e372d3b4ad19135b953a78882e789
```

есть дамп файла при краше

History

#1 - March 19, 2014 01:44 - John Lindgren

- Priority changed from Critical to Major

Please use English on this bug tracker; none of us speak Russian. Google Translate tells me that you created a playlist and you got this crash somewhere in libglib-2.0-0.dll. I can't fix a crash without being able to reproduce it or at least seeing a debugger backtrace. If you have a debugger installed, I can give you a build with debug symbols so that you can get a backtrace.

#2 - March 19, 2014 03:26 - Денис Потапов

I have created *. DMP file when beautiful. Today it is attached to the same bagu.Tak today try to play and flow describe the steps to reproduce the crash. Debugera I net.V brief can you describe the steps play Crush:
1. Player file stored in the archive is downloaded sashego site.
2 opens it in winrar.
3 drag and drop a file folder mu

#3 - March 19, 2014 21:44 - John Lindgren

So the crash occurs when you play a particular file? Can you attach that file to the bug report?

#4 - March 20, 2014 17:27 - Денис Потапов

- File Screen 3-20-2014 (23-21-3).rar added

screen video wish crash steps

#5 - March 20, 2014 23:29 - John Lindgren

Please attach the file you are trying to play to the bug report.

#6 - March 23, 2014 12:08 - Carlo Bramini

On Windows, I verified that the player has two issues when some non latin characters (like cyrillic) are used with local files:

1) if the file to load has some non latin characters in its name, the player will just fail to load it. To reproduce this defect, rename an mp3 file by adding some cyrillic text in its name (you may simply copy and paste some text from the comment made by the user Денис Потапов, if you don't want to waste time in adding new languages and keyboard layouts on your Windows machine ^_-).

2) The player will load its settings from C:/Users/<MY_USER>/AppData/Local/audacious/ on Windows Vista/7 or from the "Documents and settings" tree for previous versions of Windows. If <MY_USER> has some non latin characters, the player will brutally crash when it starts.

Both troubles seem to be caused by the vfs_local plugin, now bundled into libaudcore library: this internal plugin relies on ASCII function _open() or fopen() and, evidently, they are not able to handle this particular condition.

If I can say my opinion, I **STRONGLY** discourage the usage of the "ccs=<encoding>" flag of fopen() because, from my memories, this would mean that the user has to install the recent Visual C++ runtime himself, if it has not been installed by other installers (sometimes, some of them do that).

I did an experiment and I was able to overcome the problems with some hacks, by excluding vfs_local and adding again the "file" scheme to GIO plugin.
These tricks have solved the troubles and perhaps it would be better to do the opposite, GIO plugin bundled inside the core (which it is already GLIB based interely) and the unix I/O plugin outside, which it could even not compiled, at least on Windows.

But I must say that I do not know all drawbacks of such changes, although it seems quite safe for Windows because, from the sources of GLIB, the GIO functions are based on CreateFileW(), ReadFile() and WriteFile(), which they are the key functions for any I/O access.

Sincerely,

Carlo Bramini.

#7 - March 23, 2014 13:36 - John Lindgren

There is nothing inherently ASCII about fopen() and all the other standardized functions, only Microsoft decided not to support the character set that the rest of the world uses (UTF-8). So without using wmain and all the other non-standard UTF-16 functions Microsoft wants you to use (CreateFileW etc.) it is impossible to have full Unicode support. And I'm not aware that GCC/MinGW supports wmain (correct me if I am wrong).

Edit: Using GetCommandLineW might be an option, and then converting to UTF-8 for use with g_open(), g_fopen() wrappers (GIO is different and not necessary).

#8 - March 23, 2014 15:59 - Carlo Bramini

John Lindgren wrote:

> There is nothing inherently ASCII about fopen() and all the other standardized functions, only Microsoft decided not to support the character set that the rest of the world uses (UTF-8).

Yes, that's what I was talking about, perhaps I explained wrongly (sorry).

> So without using wmain and all the other non-standard UTF-16 functions Microsoft wants you to use (CreateFileW etc.) it is impossible to have full Unicode support. And I'm not aware that GCC/MinGW supports wmain (correct me if I am wrong).

I have an up to date MinGW environment and it seems that there is not support of wmain() or wWinMain().
MinGW-w64 claims to have it, but I have not tried.
Unfortunately, I'm not expert of GLIB universe and using an hacked GIO plugin was for me the fastest way to check the origin of the problems...

> Edit: Using GetCommandLineW might be an option, and then converting to UTF-8 for use with g_open(), g_fopen() wrappers (GIO is different and not necessary).

The fact that the player was also using argc/argv escaped to my attention, using wmain() would help although all this can be also obtained without big troubles with GetCommandLineW() and CommandLineToArgvW() as you wrote. If I can say my opinion, it seems strange that the argc/argv encoding problems were not faced by GLIB people, perhaps there are already some utilities for recovering those values as GLIB expects, to be put into something like:
#ifdef _WIN32 ... #endif

#9 - March 23, 2014 16:46 - John Lindgren

GLib handles argc/argv with g_locale_to_utf8(). So, just as in current Audacious, Cyrillic characters are handled fine if the 8-bit code page in use is CP1251 (as would be the case with the Russian edition of Windows), but will not work on an English (or other Latin-based) edition of Windows that is using CP1252.

Anyway, all this discussion is irrelevant to this bug report. There is no indication that the crash in question is related to character sets at all; anyway, it appears from the video that the reporter is using the Russian edition of Windows, so Cyrillic characters would not be a problem for Audacious.

Edit: And you can also see from the video that Audacious started up, opened the files, and read the Russian tags from them without any crash. It was only when trying to play one of the files that it crashed. So the problem is presumably NOT related to file access, but rather decoding or output.

#10 - March 30, 2014 18:33 - John Lindgren

- Status changed from New to Rejected

Closing: not enough information to reproduce.

Files

Screen 3-20-2014 (23-21-3).rar	1.56 MB	March 20, 2014	Денис Потапов