Audacious - OLD, PLEASE USE GITHUB DISCUSSIONS/ISSUES - Bug #498

Not parsing UTF-8 id3v2 tags properly: "Cannot convert from locale (UTF-8): Nightmares On Wax" (maybe glib2-related?)

January 14, 2015 02:36 - CJ Kucera

Status:	Closed	**Start date:**	January 14, 2015
Priority:	Minor	**Due date:**	
Assignee:		**% Done:**	100%
Category:	libaudtag	**Estimated time:**	0.00 hour
Target version:	3.6		
Affects version:	3.5.2		

Description

After an upgrade from Fedora19 to Fedora21 (though I have also verified this problem on a "clean" sourcecode compile on the F21 box), Audacious was no longer reading UTF-8 id3v2 tags properly, with each one outputting the following:

```
Cannot convert from locale (UTF-8): Foo
```

I dug through the code a bit and discovered that in my case, the problem appeared to be line 81 in src/libaudcore/charset.c, which does the following:

```
        if (! g_utf8_validate (str, len, NULL))
        {
            whine_locale (str, len, "from", "UTF-8");
            return NULL;
        }
```

Specifically, the string being passed to g_utf8_validate was null-terminated, and "len" was set to the full length of the string (including the null termination). The current docs for g_utf8_validate at https://developer.gnome.org/glib/stable/glib-Unicode-Manipulation.html#g-utf8-validate state that if the length is specified, the function will return FALSE if any of the specified bytes are NULL. So basically each UTF-8 string being validated included its null termination, and so each UTF-8 string was being discarded.

I can fix this on my end by passing in "len-1" instead (or "-1", I suppose)... I'm not sure if the problem here is a change in the behavior of g_utf8_validate (I'm sure there was a decent glib2 version jump between F19 and F21), or if that string wasn't intended to be NULL-terminated or something so it's not being passed around like it expects.

Anyway, let me know if I can provide any extra info. The glib2 version on this box is currently 2.42.1, if that helps. Thanks!

History

#1 - January 14, 2015 02:43 - CJ Kucera

- File audacious-3.5.2-utf8-fix.patch added

And, I suppose, to be doubly-clear, the attached patch does seem to fix the problem on my end, though I'm not sure if this is the right way to fix it, or if fixing it this way would produce any unintended side effects.

#2 - January 14, 2015 02:45 - CJ Kucera

- File sine.mp3 added

And, for reference, a small little mp3 (just a sine wave) with an id3v2 tag which exhibits this problem for me.

#3 - January 15, 2015 05:26 - John Lindgren

- Category set to libaudtag

- Status changed from New to Closed

- Target version set to 3.6

- % Done changed from 0 to 100

Excellent debugging, thank you! This is no doubt the root problem behind #490 as well.

Subtracting 1 from the length in str_from_locale() isn't quite correct, since the length passed in isn't supposed to include a terminating null. Actually, there shouldn't be a null character in the ID3 text field to start with,* but we can handle it anyway. I've added a check to do so in id3_decode_text():
https://github.com/audacious-media-player/audacious/commit/d90f419288cb8771de42655c2f9807be18672bc7

I didn't see this problem on my machine since I have "ISO-8859-1" set as a fallback character encoding. "UTF-8" would work as well. The idea is to force a call to str_convert(), which will eventually call strlen() and remove the extra null character.

*At least not for text fields that contain only a single string. ("All text information frames supports multiple strings, stored as a null separated list, where null is represented by the termination code for the charater encoding.")

Files

audacious-3.5.2-utf8-fix.patch	547 Bytes	January 14, 2015	CJ Kucera
sine.mp3	40.9 KB	January 14, 2015	CJ Kucera