darktable - Bug #12098

multiple file export with $(TITLE) creates unneeded subdirectory if an earlier file doesn't have title

03/26/2018 11:58 PM - Dan Torop

Status:	Fixed	**Start date:**	03/26/2018
Priority:	Low	**Due date:**	
Assignee:		**% Done:**	100%
Category:	Lighttable	**Estimated time:**	0.00 hour
Target version:	2.6.0		
Affected Version:	git master branch	**bitness:**	64-bit
System:	Debian	**hardware architecture:**	amd64/x86

Description

STEPS TO REPRODUCE

- In lighttable, select a file, say DSCF0013.RAF, and set its title to blank in the metadata editor.
- Select a second file, whose filename comes later in the alphabet, say DSCF0150.RAF, and set its title to "test" in the metadata editor
- In export, target "file on disk", with filename /tmp/$(TITLE), file format JPEG
- Select DSCF0013.RAF and DSCF0150.RAF files
- Click export

EXPECTED BEHAVIOR

Two files created: /tmp/DSCF0013.jpg and /tmp/test.jpg.

ACTUAL BEHAVIOR

Two files are created, /tmp/DSCF0013.jpg and /tmp/test/DSCF00150.jpg. The subdirectory /tmp/test is created if it didn't already exist.

Output to console:
[export_job] exported to `/tmp//DSCF0013.jpg'
[export_job] exported to `/tmp/test/DSCF0150.jpg'

Associated revisions

Revision 33aceabe - 04/03/2018 07:37 PM - Dan Torop

disk: don't modify filename pattern in dt_imageio_disk_t on store()

The filename pattern may be used by other calls to store() in the same instance of this module. A pattern which needs fixing in one call to store(), say if a variable expands to a null string, may not need fixing in a later call to store().

This fixes #12098.

Also, be sure not to go into an infinite loop if there is no space to append a filename to the pattern.

Revision 6963898a - 04/23/2018 02:24 AM - Dan Torop

disk: don't modify filename pattern in dt_imageio_disk_t on store()

The filename pattern may be used by other calls to store() in the same instance of this module. A pattern which needs fixing in one call to store(), say if a variable expands to a null string, may not need fixing in a later call to store().

This fixes #12098.

Also, be sure not to go into an infinite loop if there is no space to append a filename to the pattern.

(cherry picked from commit 33aceabefdfd37589d96d8cfa4a8c6599c2eb751)

History

#1 - 04/03/2018 05:24 PM - Dan Torop

This from ad3bb1ce0 looks suspicious:

```
modified   src/imageio/storage/disk.c
@@ -240,68 +240,47 @@

     // if filenamepattern is a directory just add ${FILE_NAME} as default..
     char last_char = *(filename + strlen(filename) - 1);
-    if(g_file_test(filename, G_FILE_TEST_IS_DIR) && (last_char == '/' || last_char == '\\'))
+    if(last_char == '/' || last_char == '\\')
     {
-      snprintf(d->filename, sizeof(d->filename), "%s/$(FILE_NAME)", original_filename);
+      snprintf(d->filename, sizeof(d->filename), "%s" G_DIR_SEPARATOR_S "$(FILE_NAME)", original_filename);
       goto try_again;
     }
```

I'm not sure why the g_file_test() was dropped. That commit was around the date I started seeing this. Alternately f542de1f3 touched a lot of code, but hard to imagine the problem lurked that long. I haven't tried git bisect.

The problem here is that when exporting a lot of files with the pattern DIR/$(TITLE) when some are untitled, the titled ones end up written as DIR/$(TITLE)/$(FILE_NAME) and the untitled ones written as DIR/$(FILE_NAME). Prior (and preferable, I think) behavior was be for the titled ones to end up in DIR/$(TITLE) and untitled in DIR/$(FILE_NAME).

#2 - 04/03/2018 06:37 PM - Dan Torop

Here is what seems to be going on:

- dt_variables_expand() will generate a filename ending a slash for patterns with no title (e.g. /tmp/$(TITLE))
- then disk.c:store() appends the filename (e.g. creating a pattern /tmp/$(TITLE)/$(FILE_NAME), which is a bit awkward, but works)
- but store() alters the pattern in d->filename, which changes the pattern for all future invocations of store() by dt_control_export_job_run().

The fix could be to make a copy of the d->filename pattern in store(), and only alter that. I'll try this out.

This is still a bit ad hoc. For example, if the pattern is /tmp/foo-$(TITLE), then there is no fallback to using $(FILE_NAME). Untitled files just get written to /tmp/foo-.jpg, /tmp/foo-01.jpg, etc. So the rule seems to be that if the pattern ends in "/$(VAR)" where VAR is empty, then there is a fallback to $(FILE_NAME), but if it ends in /something-$(VAR)-something where VAR is empty, then the images just get written to soemthing--something with a sequence #.

#3 - 04/03/2018 07:51 PM - Dan Torop

Made PR 1666 for this.

#4 - 04/06/2018 08:52 PM - Tobias Ellinghaus

- % Done changed from 0 to 100

- Status changed from New to Fixed

I agree, the whole thing is a bit messy. I removed the g_file_test because it didn't serve any purpose. When we try to export to a folder name it doesn't matter if that folder already exists, it would result in a hidden file inside that folder.

PR merged. Thank you very much.

#5 - 04/06/2018 09:30 PM - Dan Torop

That makes sense about g_file_test. You're welcome, and thanks for reviewing/merging.

I really have no idea if there's a better way to handle this. It depends how people actually use $(TITLE) and $(FILE_NAME). The way I use it, it would make a lot of sense that if there were no TITLE, that variables.c:get_base_value() would just return FILE_NAME instead. But I'm sure someone else uses patterns like "$(FILE_NAME) $(TITLE)" and absolutely wouldn't want that behavior. At least with PR 1666 things return to a long-standing dt behavior.

#6 - 04/07/2018 01:07 AM - Tobias Ellinghaus

That's more or less why I added the string replacement code to variables. If you want $(TITLE) to return the file name when there is no title set you can use $(TITLE-$(FILE_NAME)).

#7 - 04/09/2018 10:33 PM - Dan Torop

Oh! I didn't figure that out! Good to know.

I guess this fix is still worth it, as the replacement pattern shouldn't vary depending on which prior file has been processed.

#8 - 04/11/2018 08:56 PM - Roman Lebedev

- Target version set to 2.6.0