darktable - Feature #8387

gnome-color-manager interoperability

01/06/2011 11:15 AM - Johannes Hanika

Status:	Triaged	**Start date:**	
Priority:	Low	**Due date:**	
Assignee:		**% Done:**	20%
Category:	General	**Estimated time:**	0.00 hour
Target version:	Future		
Affected Version:	git development version	**bitness:**	64-bit
System:	all	**hardware architecture:**	amd64/x86

Description

we should abandon the /usr/share/darktable/color/{in,out} madness for some cool filtering via g-c-m.

pasting a mail by richard hughes:

> 3. Implement D-BUS client to talk to GNOME Color Manager.
>
> didn't richard hughes already do that and stopped because the dbus interface implementation will change from glib-dbus and merge into glib directly soon?

Well, as you already depend on dbus-glib it would be really easy to use this in dt. You can see an example of how to get different attributes in an old version of GCM:
http://git.gnome.org/browse/gnome-color-manager/tree/src/gcm-inspect.c?h=gnome-2-32

If you want to raise your GLib2 dep to 2.27 (which is quite aggressive) you can use GDBus, which results in a lot less code and a lot nicer code. You can see an example of the new code in the same file, on the git master branch:
http://git.gnome.org/browse/gnome-color-manager/tree/src/gcm-inspect.c

I'm guessing, at least for the short term, the former is what you'll want to do.

Now, I want to make your life as easy as possible to integrate. From a high level perspective, GCM offers the following things:

- Choosing per-session default colorspaces for import and export
- Choosing rendering intent defaults
- Getting the profile to use for output, either per-screen or per-window
- Mapping color profiles to devices
- Mapping color profiles to exif data

You can see the full list of API's here:
http://git.gnome.org/browse/gnome-color-manager/tree/src/org.gnome.ColorManager.xml

If we need to add or change API, that's not really a problem, as I'm pretty open minded about changing things at this stage if we have to. Already there are a handful of projects using the GCM DBus API and a lot more have promised support for GNOME 3.0.

At a later stage (post 3.0) we'll want to be using a compositor for

full screen color management, probably using GLSL. By using the GCM API you'll be future proofing dt to work with or without a FSCM as we'll provide fallbacks on all APIs.

Richard.

History

#1 - 01/06/2011 11:42 AM - Tobias Ellinghaus

Supporting g-c-m makes sense for those who use gnome software and have g-c-m installed. However we should not enforce users to use it, so abandoning direct input of profiles should still be possible.

#2 - 01/07/2011 02:48 PM - Jose Carlos Garcia Sogo

This is always a tradeoff, do you want a command line interface tool, or a powerful desktop tool? Anyways, I don't see as a bad thing that if dt is compiled without gnome-color-manager support, you can use direct profiles, but only in that situation.
The only change in the latter I would suggest is to install those profiles in a $HOME directory, so users can add them without being root in their systems.

#3 - 03/18/2011 03:19 AM - Russell Harrison

g-c-m stores its color profiles in ~/.color/icc/ and /usr/share/color/icc/ which is the standard locations for storing color information. Shouldn't dt just use these locations to be compatible with other applications and desktop environments?

#4 - 07/03/2012 02:09 PM - Tobias Ellinghaus

- Status changed from New to Triaged

- % Done changed from 0 to 20

- Target version set to Future

#5 - 07/25/2014 02:28 PM - Pascal de Bruijn

- bitness set to 64-bit

- Affected Version set to git development version

- System set to all

We already have some interporability level due our colord integration.

That said, both colord and gnome-color-manager potentially have lots of irrelevant profiles in those directories, which would painfully clutter our menus. So using those locations would clutter these menu's for all our users (most of which who don't use them). And it would benefit only a very tiny few. Which makes it a very poor trade-off.

As far as GCM goes, if I recall correctly only GCM's display calibration is fairly well tested, and commonly used. So I don't think we should actively try to depend on other parts.