Triager Guide

This page is an introductory guide for triagers.

Triagers can be widely defined as anybody that want to help darktable deal with its bugs or feature requests but don't necessarly have coding skills.

As a triager your responsabilities will mainly involve

- collecting as much information as possible on bugs
- analyzing expanding and overall refining feature requests
- deciding what to do with redmine entries
- cleaning up and maintaining the bug database

Overall this requires a lot of common sense, a bit of social skill, and some time.

So, how do you deal deal with entries ? basically you follow them through the whole Bug_Workflow until they are dealt with. You are the first contact point for any bug.

How to deal with bugs

here we are going to discuss bugs. Bugs are reports by users that expected a certain behaviour, but darktable reacted in a way that they consider wrong. The most important thing with bugs is to try to reproduce them, the second most important thing is to try to find duplicates

It is important to try to reproduce them because a lot of bugs are discovered by developers and fixed before being reported and other developers can spend quite some time trying to reproduce a bug that is already fixed. This is something that doesn't need any coding skills but needs to be done. It's a work for the triager.

try the steps that the bug reporter stated.If it doesn't work, make sure you are using the same version as he is, ask him for details, be sure that he reported the exact version number (especially if he reported a bug on a git build)

If you are still unable to reproduce, you will have to guide the bug reporter through all the steps to enrich the bug report. Not as easy as doing it yourself, but again it's a job that needs to be done

The other way of dealing with a new bug is to close it. Don't be shy, as a triager you should feel free to take that action. In particular hunting for duplicates or bugs fixed in master but not in the stable version of darktable is an important job. There are bug status to signal that which allows the developer to quickly find bugs that are easy to port or have a fix ready.

Last but not least, a lot of bugs are due to user errors. By trying to reproduce and starting a dialog with the bug reporter you can find out these and help reduce the load on developers by closing such bugs

Once the bug is reproduced you should try to "explore" the bug. The exact nature of what you do depends on the type of bugs and common sense is your guide here, but here are a few ideas of how to enrich the bug

- get a backtrace and attach it to the bug
- if it's specific to an image, attach the image and the XMP that goes with it
- try to see if it's hardware specific (camera, graphic card, version of OS and distribution)
- reproduce it with different logging levels (through the command line) and attach the corresponding traces
- in some case you might want to bisect the bug using the appropriate git tools to find exactly what commit introduced the bug
- it might be usefull to reproduce with a new library or to make sure the bug is not due to incorrect install. There are specific command line parameters to deal with that
- If you need to, you shouldn't hesitate to join IRC or drop a line on the dev mailing list. More people thinking might help

TODO

- don't be shy
- decision you might have to take
  - closing unreproducible/old bugs
  - closing feature requests
  - dealing with priorities

- tasks to do
  - triage new features
  - finding fixed non-closed bugs
  - finding non-cherrypicked bugs
  - finding forgotten patches
  - finding duplicates
- how to assign a bug to a coder