QA Bug Triage/Investigation Basics

Editing and commenting are now disabled as the text has been moved to the Wiki.

Bug Verification

This is the first step of bug triage and involves reading the description of a bug report and attempting to verify that you can reproduce the problem described. You should ideally test that the bug is reproducible on a daily Master build or alternatively on the latest release, as the bug may have already been fixed. If you don't want to affect the LibreOffice version that you currently run on your system, you can install it as a parallel installation. If you are on Linux and want to use the build/release without installing it, you can use this bash script.

When replying to bug reports, please reply in a polite and helpful manner. You can use the available list of pre-written responses, so that you don't need to retype the same responses multiple times. It is important to include with your response, the version details you tested with, which can be copied from the Help > About LibreOffice dialog.

If the bug was reproducible, then you should set the bug status to **NEW**. If the bug report description is too difficult to understand or references a document the bug reporter has, suggest that the bug reporter provide simpler instructions or attach a document to make it easier to verify the bug, then set the bug status to **NEEDINFO**.

As some bugs are operating system specific, it is always a good idea to test against the same operating system, but if that is not possible, do test it on your current operating system, as most bugs are not operating system specific.

Update the bug summary so it better explains the real/main problem being addressed and makes it easier to find duplicates. The **Component** field can also be updated to better specify which component the problem found in or originates from.

Enhancement bug reports requesting the addition of new features need to be first evaluated by the UX team to determine whether it should be implemented or not, so simply set the bug status to **NEW** and set the **Component** field to 'ux-advise'.

Bug Duplication

There is a high probability that an unconfirmed bug report has already been reported previously, so it is always a good idea to search for duplicates of a bug before attempting to confirm it.

Doing this will reduce additional investigation efforts and streamline all discussions of the bug in the first bug report it was identified in. If you have identified a bug report to be a duplicate, click the ‘Mark as Duplicate’ link below the **Status** drop down menu and then insert the bug report number in the field presented.

When marking a bug as...
(this is annoying. Every word is a new modification. Pedro)

It isn't always easy to find duplicate bugs because bug summaries may be incomplete or the bugs was filed under a different component.

Bug Prioritizing

New active QA team members can be given additional bugzilla rights to modify the **Priority** and **Severity** fields, as regular bugzilla users don't have access to change these. To request these privileges, ask members of the QA team (e.g. beluga, jmadero) on IRC or email the team on the mailing list. If you want to take on this task, it would be good for you to follow other members who are doing it and see if you would have set it in a similar fashion.
https://wiki.documentfoundation.org/images/0/06/Prioritizing_Bugs_Flowchart.jpg
It is usually easier to determine the severity of an issue than try to think of a suitable priority. Small aesthetic tweaks to the UI can be considered low priority and trivial severity.

Regression Testing

Testing older versions of LibreOffice to find out whether a bug is also found in earlier versions makes it easier to identify whether the bug was introduced within the lifetime LibreOffice. Use the **Version** field to identify the earliest version the bug can be found in. If the bug is found in the first version of LibreOffice (version 3.3.0) then set the **Version** field to ‘Inherited from OOo’. If the bug was introduced in a particular version of LibreOffice, then add the keywords ‘regression’ and ‘bibisectRequest’ the **Keywords** field.

https://wiki.documentfoundation.org/SI-GUI
https://wiki.documentfoundation.org/Installing_in_parallel
http://downloadarchive.documentfoundation.org/libreoffice/old/

Crash Reports

Having a crash report (backtrace) for a bug makes it easy for a developer to identify what is causing the crash. If a crash report is added to a bug report, the ‘have-backtrace’ keyword should be added to the **Keywords** field.

https://wiki.documentfoundation.org/QA/BugReport/Debug_Information
https://wiki.documentfoundation.org/How_to_get_a_backtrace_with_WinDbg

Bibisecting

Bugs that were introduced into LibreOffice after version 3.5 can be easily tracked down by bibisecting.

Effective Bisection and Bibisection (video) - Matthew Francis
https://wiki.documentfoundation.org/QA/Bibisect
https://wiki.documentfoundation.org/QA/Bibisect/Implementation

Marking for further work

Doing the first triage is of great importance! But do not forget to mark for further work to be done. Do this with XXX in the field Whiteboard (??)

Summary and must haves

- New or needinfo
- Regression / bibisectrequest?
- Bug summary OK?
- Marking for further work
- ...

https://wiki.documentfoundation.org/QA/Triage_For_Beginners
https://wiki.documentfoundation.org/QA/BugTriage