BUG REPORTING

Top Seven List of Items Included in an Ideal Bug Report

- [Feature Name] Title.
- Environment.
- Steps to reproduce.
- Expected Result.
- Actual Result.
- Visual Proof (screenshots, videos, text)
- Severity/Priority.

Required fields for the bug report

Note that the required fields of the bug report are: Bug Summary, Severity, Steps to reproduce, Actual Result, Expected Result. Below are the requirements and examples for completing these fields.

Bug Summary

The name speaks for itself. In one sentence, you need to fit the meaning of the entire bug report, namely: briefly and clearly, using the correct terminology to say what and where does not work. For example:

1. The application hangs when trying to save a text file larger than 50 MB.
1. Data on the “Profile” form is not saved after clicking the “Save” button.

In addition, we suggest that you study the Principle “Where? What? When?”:

“What is this principle?

Make a proposal, in which the facts of the defect are stated in the following sequence:

- *Where ?: Where in the user interface or the architecture of the software product is the problem. And, start the sentence with a noun, not a preposition.*
- What ?: What happens or does not happen according to the specification or your view of the normal operation of the software product. In this case, indicate the presence or absence of the object of the problem, and not its content (it is indicated in the description). If the content of the problem varies, all known options are listed in the description.
- *When ?: At what point in the software product, at the onset of an event or under what conditions is the problem manifested.*

Why should the sequence be exactly like this?

In this form unfamiliar defects are more conveniently sorted by summary as practice shows (after all, most likely, it is among the defects of other engineers that duplicates will be searched). If you have a different opinion – come up with your own sequence, but it should become one for all members of the project without exception, otherwise you will not achieve the necessary result. “

Severity

In a nutshell, it can be noted that if the problem is found in the key functionality of the application and after its occurrence the application becomes completely inaccessible, and further work with it is impossible, then it is blocking. Typically, all blocking problems are during the initial check of the new version of the product (Build Verification Test, Smoke Test), because their presence does not allow full-scale testing. If testing can be continued, then the severity of this defect will be critical. At the expense of significant, minor and trivial errors, the question is fairly transparent and, in our opinion, does not require superfluous explanations.

Steps to reproduce / Actual Result / Expected Result

It is very important to clearly describe all the steps, with mention of all input data (user name, data to fill in the form) and intermediate results.

For example:

Steps to reproduce

1. Log into the system: User Tester1, password xxxXXX
   -> Login to the system
2. Click the link.
   -> Profile Page has opened
3. Enter the New user name: Tester2
4. Click the Save button.

Actual Result

An error appeared on the screen. The new user name was not saved.

Expected Result

The profile page has been reloaded. The new value for the user name is saved.

The main mistakes in writing bug reports

Lack of data provided

Not always the same problem occurs with all entered values and under any logged in user, so it is strongly recommended that you enter all the necessary data into the bug report

Definition of severity

Very often, either an overestimation or an underestimation of the severity of the defect occurs, which can lead to an incorrect sequence in solving the problem.

Description Language

Often when describing the problem, incorrect terminology or complicated speech turns are used, which can mislead the person responsible for solving the problem.

Lack of Expected Result

In cases where you did not specify what should be the required behavior of the system, you spend the developer's time searching for this information, thereby slowing the correction of the defect. You must specify the item in the requirements, the written test case or your personal opinion if this situation has not been documented.

Filling in the bug report fields The table below describes the main fields of the bug report and the role of the employee responsible for completing this field. Red color indicates the

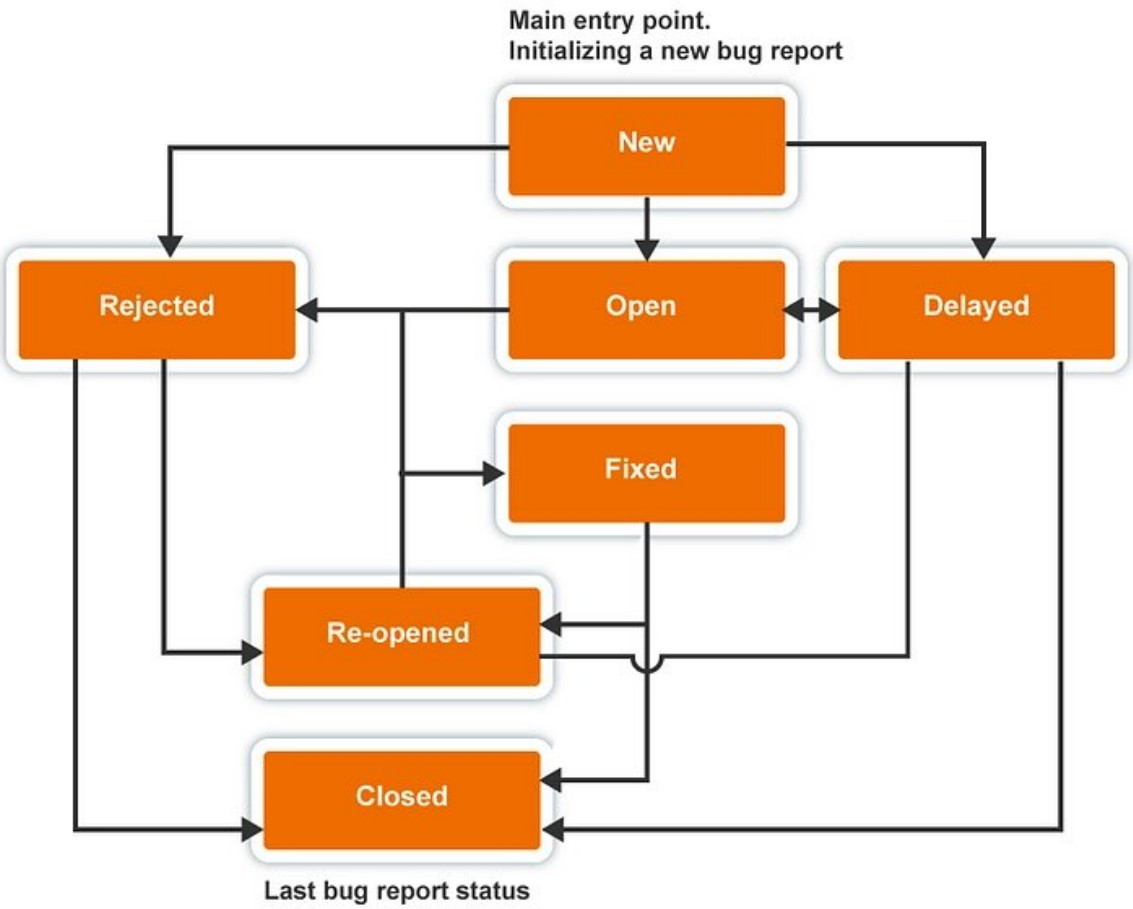
required fields:

Field	Responsible for filling the field
Summary	The author of the bug report (usually Tester)
Project	The author of the bug report (usually Tester)
Component	The author of the bug report (usually Tester)
Version	The author of the bug report (usually Tester)
Severity	The author of the bug report (usually Tester), but this attribute can be changed by a higher-level Manager
Priority	Project Manager or Manager responsible for developing the component to which the bug report was written
Status	The author of the bug report (usually Tester), but many bug tracking systems set the default status
Author	Installed by default, if not, the bug report author's name is indicated
Assignet To	The author of the bug report (usually Tester), as well as any member of the team that believes that the attached data will help in correcting the bug
OS/Service Pack/Browser + ver/...	The author of the bug report (usually Tester)
Steps to Reproduse	The author of the bug report (usually Tester)
Result	The author of the bug report (usually Tester)
Expected Result	The author of the bug report (usually Tester)
Attachment	The author of the bug report (usually Tester), as well as any member of the team that believes that the attached data will help in correcting the bug

Bug Life Cycle

The bug life cycle or defect life cycle in testing is a process that involves different stages through which a defect progresses during its lifetime. It starts when a tester identifies a new defect while testing the application. The cycle continues until the tester finds a resolution and closes the bug to prevent its recurrence.

Before starting the description of elementary bug life cycle, we suggest to consider the following block diagram showing the main statuses and possible transitions from status to status in the process of its existence:



The developer first identifies the bug, then moves to the tester for testing, and the tester marks the stages based on the priority of the bug that needs to be fixed.

Finally, they fix the bug, develop error-free software, and deliver it to the customer. But following the bug life cycle to improve the quality of the software has its upsides and downsides.

Here are some advantages of a bug life cycle:

- It enables you to produce a high-quality, dependable product.
- It lowers development costs because each process step is marked and well-documented, allowing you to use fewer resources to find the problem.
- It promotes better communication among members of the development and testing teams, increasing understanding and empathy.
- It enables you to detect issues earlier, significantly lowering the cost of finding a bug.
- It enables better services and raises customer satisfaction.

Disadvantages of the bug life cycle:

- Because all issues are marked and tracked in real-time, and the development environment is dynamic, there is no single way to control the entire process.
- Other life cycle variations enable you to better track issues and increase work throughout, making this procedure obsolete.

What is a Bug Status?

During the entire bug life cycle, the status of the bugs can change multiple times depending upon the situation. Bug or defect status refers to the current state of a defect in the bug life cycle. The goal of defect status is to convey a bug's current state or progress so that it can be tracked and understood better.

During the BLC, testers create an effective bug report containing the bug's status and other relevant information enabling the developers to understand what is wrong.

How to Create a Bug Report?

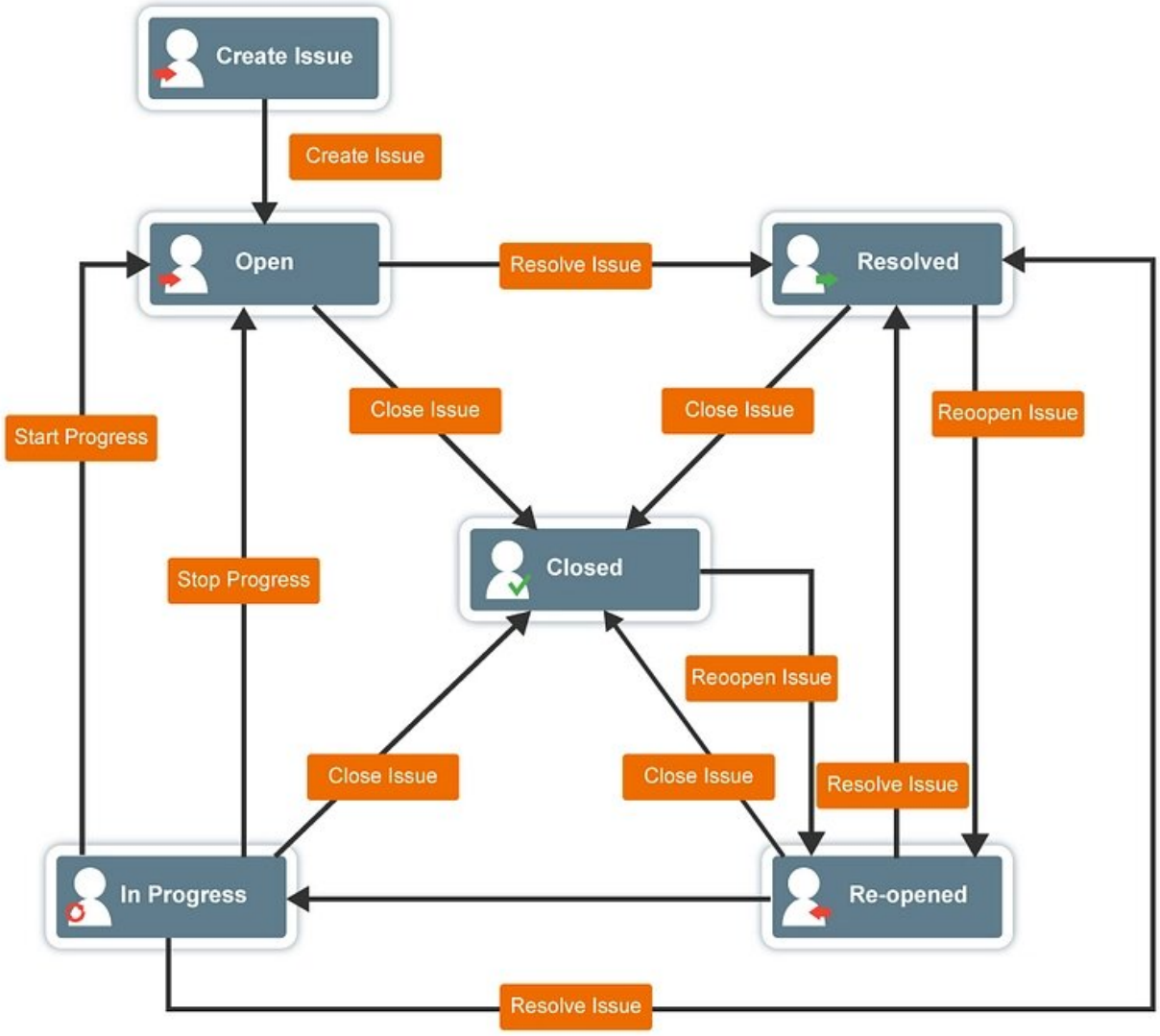
A bug report is a document containing the problem the testers discovered so the development team can fix it. When writing a bug report, you must keep it simple and to the point because a good bug report is detailed and repeatable.

Also, ensure you reproduce the bug yourself before filing a bug report to ensure it exists. The last thing your development team needs is to spend time looking for bug fixes that were only a one-time occurrence.

How to Report and Track a Bug?

Reporting and tracking a bug depends upon the tool you use. Using a professional tool, you can record, report, and monitor bugs during the software testing process.

A bug-tracking tool ensures that all the bugs are detected and fixed. You can use BrowserStack integrations with Slack, Jira, GitHub, and Trello to create a robust testing ecosystem and improve the results of your bug-tracking lifecycle. BrowserStack also offers a debugging toolkit, making verifying, fixing, and debugging errors easy.



Now we turn to the description of this scheme.

Suppose you found a bug and registered it in a bug tracking system. According to our block diagram, it will get the status “New”. The tester responsible for validating new bug reports, or the project coordinator (depending on the distribution of roles in your team) can translate it into one of the following statuses:

- “Rejected” if the given bug is invalid or repeated, or it simply could not be reproduced
- “Delayed” if this bug does not need to be fixed in this iteration
- “Opened” if bug fix is needed

Let’s consider now in order each of variants.

1. Rejected. In this case, you can either argue about the fate of your bug report by changing the status to “Re-opened” or close it – the status “Closed”
1. Opened. It is in this condition that the developer receives a bug report for correction. It can reject (for further actions, see step 1) or correct the bug. The bug report in the status “Fixed” is transferred to the tester for verification. In case the problem is still reproduced, the status “Re-opened” is displayed and the bug report is sent back for revision to the developer. If the fix was successful, then the bug report is translated to the “Closed” status.

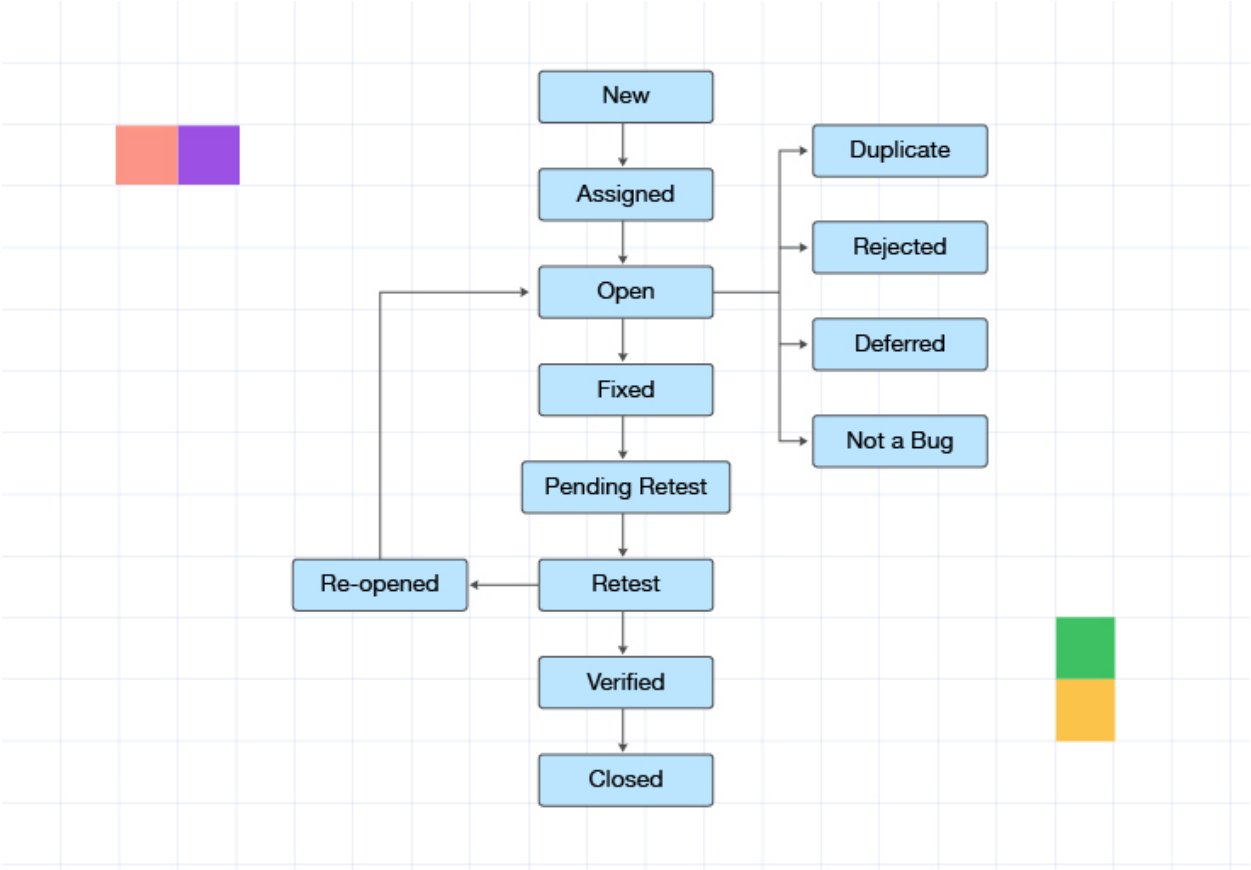
   * * *

   Note that this scheme is greatly simplified. For more visibility and, perhaps, the convenience of working on the project, you can add additional statuses and transitions, especially since modern bug tracking systems allow you to do this. However, keep in mind that unnecessarily intricate transition schemes and redundant statuses can make life much more difficult.

   Note 1: in some bug tracking systems the created bug report immediately gets the “Opened” status without additional validation

   Note 2: many bug tracking systems allow you to rediscover closed bugs, but personally we do not support this practice, and therefore did not describe such a transition in the above life cycle

   Note 3: The above life cycle is based on the fact that there is someone in the team responsible for assigning bug reports. If there is no such role on the project, the bugs are assigned by the developers themselves, and in order to avoid confusion, it makes sense to introduce one more intermediate status “In progress”, indicating that this bug report has already been assigned and is at the stage corrections. An example of implementing such a life cycle on the basis of JIRA can be seen in the following figure:


New
Assigned
Open
Fixed
Pending Retest
Retest
Re-opened
Verified
Closed
Duplicate
Rejected
Deferred
Not a Bug

•