
Software Quality Assurance: SOFT3302

Tutorial – Week 7

Objectives

This tutorial gives experience in the use of Bugzilla. By the end of this tutorial you should have created a Bugzilla account and had practice at entering, finding and updating bugs.

Pework

Note that we will be using Bugzilla 3.0.1 so make sure you read the correct documentation.

You will be ‘test driving’ Bugzilla. You will need to use this URL <http://landfill.bugzilla.org/bugzilla-3.0-branch/> to ‘test drive’ Bugzilla.

1. Create an account for yourself using your School of IT or UniKey email address. You will need to use an email address you can access as your password will be mailed to you.
2. Read the Bugzilla documentation at <http://www.bugzilla.org/docs/3.0/html/myaccount.html>. Make sure you familiarise yourself with **§5.4 The Life Cycle of a Bug**.
3. Read **§5.5 Searching for Bugs** and familiarise yourself with the bug query page.
4. Note that many of the options on this page contain links to further explanation. Use these links freely in your exploration.
5. Try the following sample queries:
 - a. Search for NEW, ASSIGNED or REOPENED bugs currently assigned to anyone at bluemartini.com: leave all the fields at their default setting (reload the page if you have to) and enter ‘bluemartini.com’ in the left-hand text entry field in the ‘Email and Numbering’ box. Make sure only ‘Bug Owner’ is ticked in that column. Press the ‘Search’ button (you can most likely just press ‘Enter’ as well if your browser is sensible). You should find something in the order of 1285 bugs.
 - b. Try the same search but this time limit your query to just bugs reported for the ‘WorldControl’ product. Select this in the ‘Product’ menu list. Notice that when you do this the list in the ‘Component’ menu list changes to include only those components of the selected product. Search again. You should get around 640 hits.

A note about versioning: versions are not decimals they are components separated by a dot. This means 1.12 is a later version than 1.6 and 1.12 is different to 1.1.2. Often the major version represents substantial (often non-compatible) changes in functionality or data formats, the minor version represents updates adding functionality in a compatible manner within versions and the third identifier represents patch levels (perhaps with no visible change beyond bug fixes). Sometimes whole versions are reserved. For example, until recently odd minor versions of the Linux kernel were development versions, i.e. 2.1, 2.3, 2.5 and even versions were release versions, i.e. 2.0, 2.2, 2.4. (This convention seems to have been changed with the 2.6 release.) Other projects have adopted this type of version numbering convention but don’t assume it. Some projects adopt a versioning scheme involving the date of release, either for final versions or to mark development releases. With the rise in popularity of subversion it is not uncommon to see the repository revision somewhere in a release

version identifier. All versioning is by convention and when developing a version numbering scheme it is important to come up with something that is consistent and extensible.

6. Try a too-specific query that will find no bugs. You will find Bugzilla will tell you ‘Zarro Boogs found.’ (Zero bugs found.)

Since this is a fairly involved pre-work, just take some time to make sure everyone is comfortable with querying Bugzilla.

Labwork

1. Writing good incident reports is an art one needs to develop. Read the documentation on writing a bug report in **§5.6 Filing Bugs** of the Bugzilla documentation at <http://www.bugzilla.org/docs/3.0/html/bugreports.html>. The bug writing guidelines can be found at <http://landfill.bugzilla.org/bugzilla-3.0-branch/page.cgi?id=bug-writing.html>.
2. You will also need to be able to find if bugs have already been reported so you don’t end up resubmitting the same bugs over and over again. It’s Mozilla-specific but have a look at <http://www.mozilla.org/quality/help/beginning-duplicate-finding.html>.
3. File a (fictitious) bug on landfill.
4. Find your bug (try finding it using the search tool and by entering the bug number directly—you did remember the number, didn’t you? ☺). Change the bug state to INVALID. (Normally one would accept the bug by changing the status to ACCEPTED.) Initially, try changing the status without entering a comment. You will find Bugzilla will force you to enter a comment. (This is actually configurable by the administrator.)

There is not a huge amount of set hands-on work this week but there is a lot of reading and exercises associated with that. It’s mainly a chance for students to familiarise themselves with an extremely important and ubiquitous tool. Don’t let them think it’s just reading and that they need not do it. They need to know this stuff and they should spend time familiarising themselves properly with it. They should definitely *try* the exercises not just read them and say ‘yeah, I’d know how to do that.’ (You can draw a comparison between reading a mathematical problem and the difference between thinking one knows a solution and actually trying to work one out. Until one actually tries it one may be gravely mistaken.)

Further Reading

Bugzilla etiquette: <http://bugzilla.mozilla.org/page.cgi?id=etiquette.html>.

What do to and what not to do in Bugzilla: <http://www.mozilla.org/quality/help/bugzilla-privilege-guide.html> (deals with having administrator privileges).

These are for further reading outside the tutorial to help make you ‘good bug-hunting citizens’. They are not optional.