

TestLink - How to setup bugtracking system integration

Case study mantis

Date	Author	Notes
20051221	Francisco Mancardi	

Overview.....	2
Environment.....	2
Step 1 – Edit config.inc.pph.....	2
Step 2 – Mantis - Configure anonymous access.....	2
Step 3 – Testlink - Configure Mantis Interface Parameters.....	3
Testing the configuration.....	4

Overview

The integration between Test Link and a Bug Tracking System (BTS) has the following characteristics

1. All communication between Test Link and the BTS is done through database tables.
2. Testlink (at the time of this writing) is neither able to send data to the BTS, either able to receive data from the BST, in the traditional model of function call.

After all the configuration is done and running, for a testlink user point of view the process will be:

1. While executing a test, it fails.
2. the user click on the link that opens the BTS web page that allows issue reporting
3. After the issue has been reported, the user has to take note of the issue ID assigned by the BTS, to input it into Testlink.
4. The user returns to the Testlink test execution page, and writes the issue ID in the bug input
5. After the user saves the execution, Testlink will display data taken from the BTS database.

Environment

The enviroment for this example is:

- Testlink and Mantis installed on the same webserver
- Mantis URL: <http://calypso/mantis/>
- Test Link URL: <http://calypso/testlink/>
- Mantis database name: ***mantis_bt***
- mysql user to access mantis_bt: ***mantis_bt_user***
- password of mantis_bt_user: ***mantis_bt_user_password***

Step 1 – Edit config.inc.pph

```
/**
 * @var STRING TL_INTERFACE_BUGS = ['NO', 'BUGZILLA','MANTIS','JIRA']
 * BUGZILLA: edit configuration in TL_ABS_PATH/cfg/bugzilla.cfg.php
 * MANTIS : edit configuration in TL_ABS_PATH/cfg/mantis.cfg.php
 * JIRA : edit configuration in TL_ABS_PATH/cfg/jira.cfg.php
 */
define('TL_INTERFACE_BUGS', 'MANTIS');
require_once(TL_ABS_PATH . 'lib/bugtracking/int_bugtracking.php');
```

Step 2 – Mantis - Configure anonymous access

- anonymous login into mantis has to be turned on,
- a mantis user has to created with viewer rights to all public projects

Change the following in your mantis config_inc.php (replace dummy with your created user)

```
# --- anonymous login -----
# Allow anonymous login
$g_allow_anonymous_login = ON;
$g_anonymous_account = 'dummy';
```

Step 3 – Testlink - Configure Mantis Interface Parameters

You must to edit TL_ABS_PATH/cfg/bugzilla.cfg.php

```
/** The DB host to use when connecting to the mantis db */
define('BUG_TRACK_DB_HOST', 'calypso');

/** The name of the database that contains the mantis tables */
define('BUG_TRACK_DB_NAME', 'mantis_bt');

/** The DB type being used by mantis */
define('BUG_TRACK_DB_USER', 'mantis_bt_user');

/** The DB password to use for connecting to the mantis db */
define('BUG_TRACK_DB_PASS', 'mantis_bt_user_password');

/** link to the bugtracking system, for viewing  bugs */
define('BUG_TRACK_HREF', "http://calypso/www/mantis/view.php?id=");

/** link to the bugtracking system, for entering new bugs */
define('BUG_TRACK_ENTER_BUG_HREF', "http://calypso/www/mantis/");
```

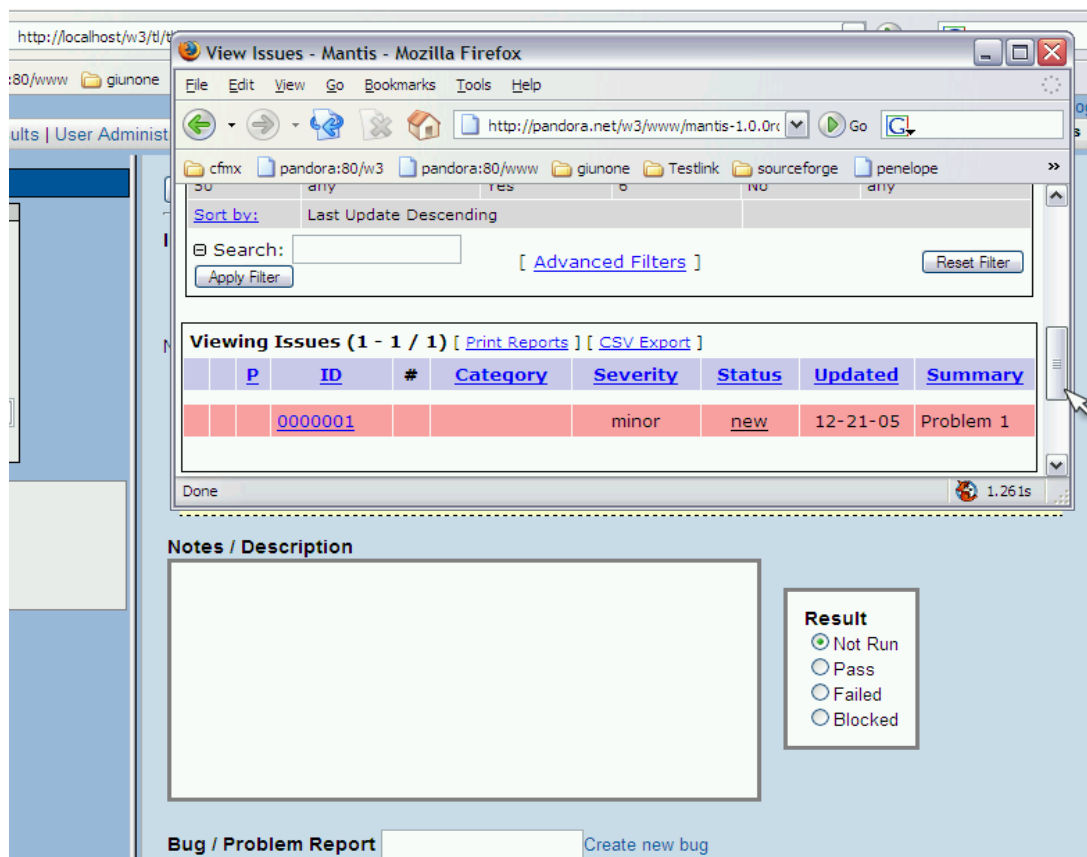
Testing the configuration

Execute a Test Case



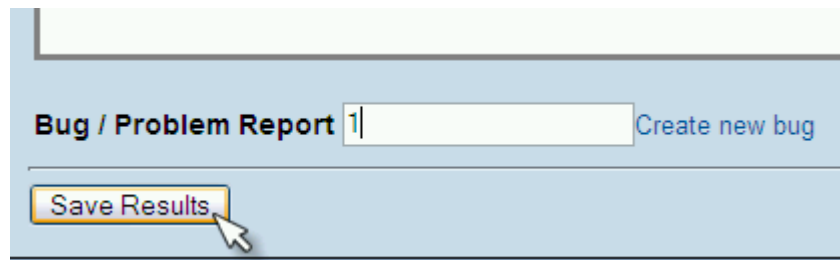
The screenshot shows the 'Bug / Problem Report' form in Mantis. It features a large text area for 'Notes / Description' and a 'Result' section with radio buttons for 'Not Run', 'Pass', 'Failed', and 'Blocked'. At the bottom, there is a 'Bug / Problem Report' label, a text input field, and a 'Create new bug' button. A mouse cursor is pointing at the 'Create new bug' button.

After click on "**Create new bug**", creating the bug using Mantis user interface and reorganizing the windows.



The screenshot shows the 'View Issues' page in a Mozilla Firefox browser. The browser window has a title bar 'View Issues - Mantis - Mozilla Firefox' and a menu bar with 'File', 'Edit', 'View', 'Go', 'Bookmarks', 'Tools', and 'Help'. The address bar shows 'http://localhost/w3/tl/t...'. The page content includes a search bar with 'Sort by: Last Update Descending', a search input field, and buttons for 'Apply Filter', 'Advanced Filters', and 'Reset Filter'. Below the search bar, there is a table titled 'Viewing Issues (1 - 1 / 1)' with columns: P, ID, #, Category, Severity, Status, Updated, and Summary. The table contains one row with the following data: P (empty), ID (0000001), # (empty), Category (empty), Severity (minor), Status (new), Updated (12-21-05), and Summary (Problem 1). Below the table, there is a 'Notes / Description' section with a large text area and a 'Result' section with radio buttons for 'Not Run', 'Pass', 'Failed', and 'Blocked'. At the bottom, there is a 'Bug / Problem Report' label, a text input field, and a 'Create new bug' button. A mouse cursor is pointing at the 'Create new bug' button.

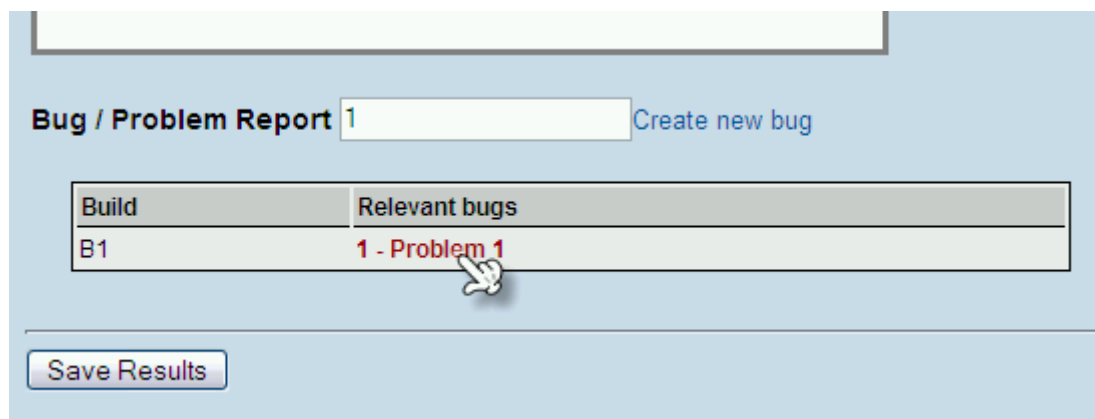
User writes the issue ID on Testlink



Bug / Problem Report 1 [Create new bug](#)

Save Results

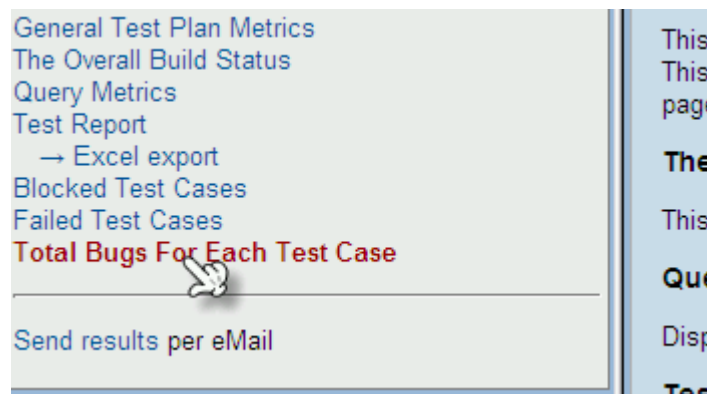
After Saving



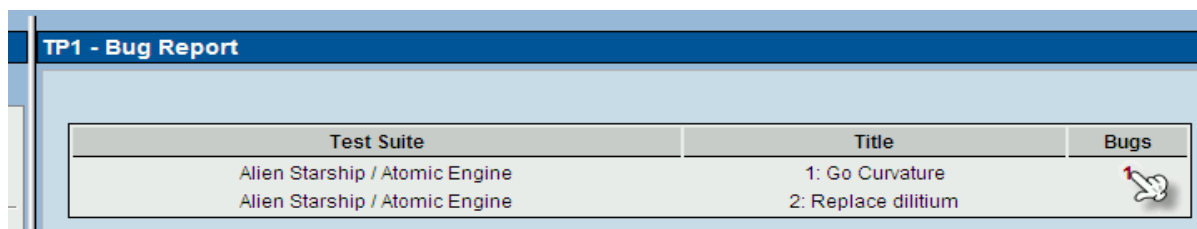
Build	Relevant bugs
B1	1 - Problem 1

Save Results

Using Testlink reports



- General Test Plan Metrics
- The Overall Build Status
- Query Metrics
- Test Report
 - Excel export
- Blocked Test Cases
- Failed Test Cases
- Total Bugs For Each Test Case**
- Send results per eMail



Test Suite	Title	Bugs
Alien Starship / Atomic Engine	1: Go Curvature	1
Alien Starship / Atomic Engine	2: Replace dilitium	