Mantis Bug Tracker Administration Guide

Mantis Bug Tracker Administration Guide

Reference manual for the Mantis Bug Tracker.

Build Date: 15 January 2009

Table of Contents

Chapter 1. About MantisBT

What is MantisBT?

MantisBT is a web based bug tracking system that was first made available to the public in November 2000. Over time it has matured and gained a lot of popularity, and now it has become one of the most popular open source bug trackering systems. MantisBT is developed in PHP, with support to multiple database backends including MySQL, MS SQL, PostgreSQL and DB2.

MantisBT, as a PHP script, can run on any operating system that is supported by PHP and has support for one of the DBMSes that are supported. MantisBT is known to run fine on Windows, Linux, OS/2, System i and a variety of Unix operating systems.

Who should read this manual?

This manual is targetted for the person responsible for evaluating, installing and maintaing MantisBT in a company. Typically we refer to this person as the MantisBT administrator.

License

MantisBT is released under the terms of GNU General Public License (GPL)[1]. MantisBT is free to use and modify. It is free to redistribute as long as you abide by the distribution terms of the GPL[2].

Minimum Requirements

MantisBT has modest software and hardware requirements. It requires a computer that is able to run the server software. All of the required software is free for commercial or non-commercial use. The server can be a shared public web server or a dedicated co-located box. The disk space required will depend on the size of the database, however, it is typically driven by the expected number and size of the attachments.

- Operating System: MantisBT runs on Windows, MacOS, OS/2, Linux, Solaris, the BSDs, and just about anything that supports the required server software.
- Web Server: MantisBT is mainly tested with Microsoft IIS[3] and Apache[4]. However, it is expected to work with any decent web server software.
- PHP[5]: The web server must have PHP installed on it. It can be installed as CGI or whatever other integration technology that is supported by PHP and the web server. Required version is PHP 5.2.x or higher.
- Database: MantisBT requires a database to store its data. The supported DBMSes include MySQL (4.1.x or higher), MS SQL, PostgreSQL and DB2.
- Browser: MantisBT aims to support most of the browsers in the market. The mainly supported ones are Internet Explorer and Mozilla Firefox. However, browsers like Safari and Opera should also work fine although they are not used by most developers during development and testing.

How to get it?

MantisBT is available in several Linux distributions including: Debian, Ubuntu, Fedora, Gentoo, Frugalware and others. Hence, if you are running Linux, start by checking if your

distribution has a package for MantisBT. If not, or if the package is not up-to-date with the latest MantisBT version, then you may want to download it directly from here[6].

For Windows, Mac and other operating systems, use the link provided above to download MantisBT. The download is compressed in tar.gz or zip format. Both formats can be unpacked using tools like 7-Zip[7] (in case of Windows).

Note that at any point in time there are typically two "latest" MantisBT releases that are available for download. The latest production release (stable), and the latest development release which can be an alpha or a release candidate. It is not recommended to use development releases in production specially if it is still in the alpha stage unless the administrator is familiar with PHP and is able to troubleshoot and fix any issues that may arise.

About the Name

When initially seeking to name this project Ken ran into a problem every programmer encounters. What is a good name? It has to be descriptive, unique, and not too verbose. Additionally having multiple meanings would be a nice touch. Quickly ruled out were php*Something* names which, incidentally, although popular, do not seem to be condoned by the PHP Group developers. Drawing inspiration from Open Source projects like Apache, Mozilla, Gnome, and so forth resulted in two eventual choices: Dragonfly and Mantis. Dragonfly was already the name of a webmail package. So the name became Mantis. Praying Mantis are insects that feed primarily on other insects and bugs. They are extremely desirable in agriculture as they devour insects that feed on crops. They are also extremely elegant looking creatures. So, we have a name that is fairly distinctive and descriptive in multiple ways. The BT suffix stands for "Bug Tracker" and distinguishes this project from general usage of the word Mantis. However, over time the project was typically referred to as Mantis.

History

Kenzaburo Ito and a friend originally created a bug tracker as an internal tool for their pet project. A search for good, free packages came up with nothing suitable so they wrote their own. After a rewrite and cleanup it was made available to the public via the GNU General Public License (GPL). The GPL was chosen partly because of his belief that development tools should be cheap or free. In 2002, Ken was joined by Jeroen Latour, Victor Boctor and Julian Fitzell to be the administrators and the core development team of MantisBT. This marks a new era in MantisBT lifetime where it is now a team project.

Support

There are plenty of resources to help answer support queries. Following are the main ones:

- Forums[8] - The forums are one of the most popular destinations for getting MantisBT support. Start off by searching the forums for your questions, if not found, then go ahead and submit a question.
- Mailing lists[9] - Available mailing lists are "mantisbt-announce" for announcements, "mantisbt-dev" for development issues, mantisbt-lang for localization () and "mantisbt-help" for general help/support questions. There are public archives for such mailing lists. Note that only members of the mailing lists can post to them, hence, subscribe to the lists before you attempt to email them.
- IRC[10] - The IRC channel is mainly used by developers to engage in in-person discussion. The recommended tool for IRC is XChat (for Linux), XChat 2 (for Windows). However, you can also use Mibbit[11] to connect to IRC via your web browser. This is also useful when your work firewall blocks the IRC port (although there are other workarounds

involving tunneling to fix this issue). Many people prefer to use IRC to ask questions to the developers and other users who are in the IRC channel.

- Wiki[12] - The MantisBT Wiki has information related to "How To (recipes)", FAQ, feature requirements, etc.
- Search - A good way for locating an answer for your question or finding more information about a topic is to search accross all MantisBT website and the Internet via Google[13] or Live[14].

It is important to note that support questions should not be sent directly to MantisBT developers or through the MantisBT contact us pages. Use of "Contact Us" page or emailing the developer directly is available if you are after a paid support or consulting service.

MantisBT News

There are several ways to keep up to date with MantisBT news. These include:

- mantisbt-announce mailing list is a very low traffic list that is used for major announcements, typically announcements about releases. All Mantis users are encouraged to subscribe to this mailing list. The average traffic should be no more than one to two posts per month.
- Mantis Blog[15] is used to communicate announcements about new releases, topics relating to MantisBT, etc. Users are encouraged to subscribe to the RSS feed to know when new posts are posted there.
- Twitter[16] is used to notify users about up-to-date details about what is happening with MantisBT development. For example, a Twitter update is automatically posted by the official bug tracker whenever an issue is resolved. Twitter users are encouraged to follow "mantisbt".

Goals

The goals for this project are to produce and maintain a lightweight, simple bugtracking system. Additions of complexity/features are modular so that users can be shielded from unwanted clutter. Thus, much of the package has a simple version of a feature along with a more fully developed version. In the 'core' package the goal is to have the most important, most used, most time saving portions of a bug tracking system. The product is designed to be easily modifiable, customizable, and upgradeable. Anyone with intermediate PHP and MySQL experience should be able to customize MantisBT to suit their needs. Here are some of the guidelines that are followed in MantisBT:

- Quick access to "What I want to do?"
- Simple navigation
- Layered complexity
- Consistency
- Scale to browser window size
- Minimal clutter
- Minimal graphics
- No frames
- No animations
- Work with Javascript turned off.

Versioning

The release numbering convention we use is major.minor.micro (eg. 1.0.8).

- Major - Indicates a very large change in the core package. Rewrites or major milestones.
- Minor - Significant amount of feature addition/modification.
- Micro - Mostly bug fixes and maintenance releases.

Notes

1. http://www.gnu.org/copyleft/gpl.html
2. http://www.gnu.org/copyleft/gpl.html
3. http://www.microsoft.com/iis
4. http://www.apache.org/
5. http://www.php.net/
6. http://www.mantisbt.org/download.php
7. http://www.7-zip.org/
8. http://www.mantisbt.org/forums/
9. http://www.mantisbt.org/mailinglists.php
10. http://www.mantisbt.org/irc.php
11. http://www.mibbit.com/
12. http://www.mantisbt.org
13. http://www.google.com
14. http://www.live.com
15. http://www.mantisbt.org/blog/
16. http://twitter.com/mantisbt

Chapter 2. Installation

Following are the steps for a new Mantis installation:

- Download[1] Mantis
- Go through MantisBT Configuration[2] and set the database options + whatever options where you need to override the default values.
- Test your configuration through the admin folder
- Create a new administrator account and remove the standard user 'administrator'

Following are the steps to upgrade a MantisBT installation:See Upgrading[3].The following details the basic steps for installation on any system. Theinstructions may seem unix-centric but should work fine on Windows systems.Barring complications, it should take you about 10-20 minutes to install,configure, and be using MantisBT.

Summary

1. Tranfer files
2. Uncompress files
3. Generate database tables
4. Edit configuration file, if needed
5. PHP File extensions
6. Login
7. Add projects and users

New Installations

1. First, transfer the file to your webserver using whatever method you likebest (ftp, scp, etc). You will need to telnet/ssh into the server machine forthe next steps.
2. Next, untar/gunzip it to the directory that you want.The usual command is (1 step): tar zxvf <filename.tar.gz> OR (2 steps): gunzip <filename.tar.gz> tar xvf <filename.tar> Winzip, Stuffit, and other programs should also be able to handledecompression of the archive.At this point you may want to rename the directory to something simpler like 'mantisbt'. You will use the mv command to rename a directory (Windows users substitute the "ren" command or use explorer). mv <directoryname> mantisbt
3. Next we will create the necessary database tables and a basic configurationfile.From your web server, access http://yoursite/mantisbt/admin/install.php. This page will walk through the following steps:
    a. check basic parameters for the web server
    b. prompt for the database type and location, and a database user/passwordpair. For installion, an administrative user/password pair can also be provided. The operating user requires SELECT, INSERT, UPDATE, and DELETE privileges. For installation, INDEX, CREATE, ALTER, and DROP privileges arealso required.
    c. create the database and tables. WARNING: A DEFAULT ADMINISTRATOR level account is created. The account name and password are administrator / root. Use this when you first login to MantisBT. Immediately go to Manage and create at least one administrator level account. Immediately after that DISABLE

or DELETE the administrator account. You can recreate it but you should delete the account to prevent the cookie_string from being used to trick the package. It would be even better to rename the account or delete it permanently. REMEMBER: After setting up the package, REMOVE the default administrator account.

d. write a basic "config_inc.php file to define the database.

e. perform some post installation checks on the system.

4. The next part involves configuring the installation to work with yourspecific setup.Open the file in an editor and add anyother values that are required. There aremany more that you can use to customize your MantisBT installation. SeeConfiguration[4] for in depth explanations.The file will overwrite the default values with those necessary for setup.You can load up admin/check.php to see if you set things up correctly. NOTE: check.php sometimes reports the value of register_globalsincorrectly. Create a page with this line in it: <? phpinfo() ?>, save itwith a .php extension and load it up in your web browser. It will, among amultitude of other things, have the correct value of register_globals that youare using.
5. MantisBT now uses only .php files.If your webserver is configured for other extensions (.PHP3, .PHTML) then youwill have to have the administrator add support for .PHP files. This shouldbe a trivial modification.Documentation can be found at: http://www.php.net/manual/en/installation.php
6. Login to your bugtracker and go to the manage section. Click on the projects link. You will need to ADD a new project. Then EDIT the new project and remember to ADD at least one category. Otherwise you won't be able to add any issues. That should be it. You're off and running.

Requirements

The following versions are required for proper operation:

Package	Minimum Version	Tested with
MySQL	4.1.x	
PostgreSQL (experimental)	7.0	8.0
PHP	5.2.x	
Web Server		Apache 1.3, Apache 2.0.54, IIS 6.0

MantisBT is designed in a way to work in as many environments as possible. Hence the required extensions are minimal and many of them are optional affecting only one feature. The following PHP extensions are the ones used:

- mysqli (or the extension for the DBMS being used) - mandatory.
- Curl - if the Twitter integration feature is required.
- GD - if the graphs feature is required.

Backups

It is recommended to backup your MantisBT database on a regular basis. This is easy to accomplish using the mysqldump command: mysqldump -u<username> -p<password> <database name> > <output file> To restore a backup you will need to have a clean database. Then run: mysql -u<username> -p<password> <database name> < <input

file> You can also perform both of these tasks using phpMyAdmin[5] A good idea is to make a backup script and run it regularly through cron or a task scheduler (for Windows see WinCron[6]). Using the current date in the filename can prevent overwriting and make cataloguing easier. !!! Backups should always be performed before an upgrade !!! Make sure to backup MantisBT code (which includes your configs + possibly customization), issue attachments / project documents, and database contents.

CVS Integration

CVS integration allows MantisBT to register commits to the CVS source control system into corresponding issue notes in the issue tracker. The setup requires that the MantisBT installation be accessible on the computer running the CVS server. A copy of the MantisBT config_inc.php file must be present.Note that the mysql database also needs to be accessible from the cvs machine. That is, "localhost" for $g_hostname won't work unless CVS and MantisBT are hosted on the same machine.To activate the integration, the following line to the cvs "commitinfo" file. (Instructions to edit this file are in any number of CVS primers). ALL /usr/bin/php /path_to_mantis/core/checkin.php This will pass the commit message to checkin.php for all commits. If the stringissue #nnnn is found in the commit message, the MantisBT corresponding to "nnnn" will have the CVS commit message added as an issue note to the issue. Multiple issues can be listed.This feature is configured through config_inc.php and through custom functions.

See also: Source Control Integration [7] for configuration, and Custom Functions [8]

Uninstall

It is recommended that you make an backup in case you wish to use your data in the future. See the Backups[9] page for details. To uninstall MantisBT:

- Delete the MantisBT directory and all files and subdirectories.
- Drop all MantisBT tables from the database, these can be identified by the configured prefix for the installation. The default prefix is 'mantis'.
- Remove any customizations or additions that you may have made.

If you have the permissions to create/drop databases and you have a specific database for MantisBT that does not contain any other data, you can drop the whole database.

Notes

1. manual.about.mantis.download.html
2. manual.configuration.html
3. manual.installation.upgrading.html
4. manual.configuration.html
5. http://www.phpmyadmin.net/
6. http://www.wincron.com/
7. manual.configuration.source.control.integration.html
8. manual.customizing.mantis.custom.functions.html
9. manual.installation.backups.html

Chapter 3. User Management

Creating User Accounts

In MantisBT, there is no limit on the number of user accounts that can be created. Typically, installations with thousands of users tend to have a limited number of users that have access level above REPORTER.

By default users with ADMINISTRATOR access level have access to create new user accounts. The steps to do that are:

- Click "Manage" on Main Menu.
- Click "Manage Users" (if not selected by default).
- Click "Create New Account" button just below the alphabet key.
- Enter user name, email address, global access level (more details about access levels later). Other fields are optional.
- Click "Create Users".

Creating a user triggers the following actions:

- Creating a user in the database.
- If email notifications ($g_enable_email_notification) is set to ON, then the user will receive an email allowing them to activate their account and set their password. Otherwise, the account will be created with a blank password.
- If email notifications ($g_enable_email_notification) is set to ON, users with access level about $g_notify_new_user_created_threshold_min will get a notification that a user account has been created. Information about the user like user name and email address are provided. The IP of the user that created the account is also included.

When the 'Protected' flag is set on a user account, it indicates that the account is a shared account (e.g. demo account) and hence users logged using such account will not be allowed to change account preferences and profile information.

Enabling/Disabling User Accounts

The recommended way of retiring user accounts is to disable them. Scenarios where this is useful is when a person leaves the team and it is necessary to retire their account.

Once an account is disabled the following will be enforced:

- All currently active sessions for the account will be invalidated (i.e. automatically logged out).
- It will no longer be possible login using this account.
- No further email notifications will be sent to the account once it is disabled.
- The user account will not show anymore in lists like "assign to", "send reminder to", etc.

The disabling process is totally reversible. Hence, the account can be re-enabled and all the account history will remain intact. For example, the user will still have issues reported by them, assigned to them, monitored by them, etc.

Deleting User Accounts

Another way to retire user accounts is by deleting them. This approach is only recommended for accounts that have not been active (i.e. haven't reported issues). Once the account is deleted, any issues or actions associated with such account, will be associated with user123 (where 123 is the code of the account that was deleted). Note that associated issues or actions are not deleted.

As far as the underlying database, after the deletion of a user, records with the user id as a foreign key will have a value that no longer exists in the users table. Hence, any tools that operate directly on the database must take this into consideration.

By default administrators are the only users who can delete user accounts. They can delete accounts by clicking Manage, Manage Users, locating the user to be deleted and opening it details page, then clicking on the "Delete User" button which deletes the user.

Note that "Deleting Users" is not a reversible process. Hence, if it is required to re-add the user account, it is not possible to recreate the user account so that it gets the same ID and hence retains its history. However, manually creating a record in the users table with the same id, can possibly do that. However, this approach is not recommended or supported.

User Signup

For open source and freeware projects, it is very common to setup MantisBT so that users can signup for an account and get a REPORTER access by default (configurable by the $g_default_new_account_access_level configuration option). The signup process can be enabled / disabled using the $g_allow_signup configuration option, which is enabled by default.

If email notifications ($g_enable_email_notification) is set to ON, users with access level about $g_notify_new_user_created_threshold_min will get a notification that a user account has been created. Information about the user like user name, email address, IP address are included in the email notification.

Forgot Password and Reset Password

It is pretty common for users to forget their password. MantisBT provides two ways to handle such scenario: "Forgot Password" and "Reset Password".

"Forgot Password" is a self service scenario where users go to the login page, figure out they don't remember their password, and then click the "Lost your password?" link. Users are then asked for their user name and email address. If correct, then they are sent an email with a link which allows them to login to MantisBT and change their password.

"Reset Password" scenario is where a user reports to the administrator that they are not able to login into MantisBT anymore. This can be due to forgetting their password and possibly user name or email address that they used when signing up. The administrator then goes to Manage, Manage Users, locates the user account and opens its details. Under the user account details, there is a "Reset Password" button which the administrator can click to reset the password and trigger an email to the user to allow them to get into MantisBT and set their password. In the case where email notifications are disabled, resetting password will set the password to an empty string.

Changing Password

Users are able to change their own passwords (unless their account is "protected"). This can be done by clicking on "My Account", and then typing the new password in the "Password" and "Confirm Password" fields, then clicking "Update User". Changing the password automatically invalidates all logged in sessions and hence the user will be required to re-login. Invalidating existing sessions is very useful in the case where a user going onto

a computer, logs into MantisBT and leaves the computer without logging out. By changing the password from another computer, the session on the original computer automatically becomes invalidated.

Pruning User Accounts

The pruning function allows deleting of user accounts for accounts that have been created more than a week ago, and they never logged in. This is particulary useful for users who signed up with an invalid email or with a typo in their email address address.

The account pruning can be done by administrators by going to "Manage", "Manage Users", and clicking the "Prune Accounts" button inside the "Never Logged In" box.

Authorization and Access Levels

MantisBT uses access levels to define what a user can do. Each user account has a global or default access level that is associated with it. This access level is used as the access level for such users for all actions associated with public projects as well as actions that are not related to a specific project. Users with global access level less than $g_private_project_threshold will not have access to private projects by default.

The default access levels shipped with MantisBT out of the box are VIEWER, REPORTER, UPDATER, DEVELOPER, MANAGER and ADMINISTRATOR. Each features has several configuration options associated with it and identifies the required access level to do certain actions. For example, viewing an issue, reporting an issue, updating an issue, adding a note, etc.

For example, in the case of reporting issues, the required access level is configurable using the $g_report_bug_threshold configuration option (which is defaulted to REPORTER). So for a user to be able to report an issue against a public project, the user must have a project-specific or a global access level that is greater than or equal to REPORTER. However, in the case of reporting an issue against a private project, the user must have project specific access level (that is explicitly granted against the project) that is higher than REPORTER or have a global access level that is higher than both $g_private_project_threshold and $g_report_bug_threshold.

Note that project specific access levels override the global access levels. For example, a user may have REPORTER as the global access level, but have a MANAGER access level to a specific project. Or a user may have MANAGER as the global access level by VIEWER access to a specific project. Access levels can be overriden for both public and private projects. However, overriding access level is not allowed for users with global access ADMINISTRATOR.

Each feature typically has multiple access control configuration options to defines what access level can do certain operations. For example, adding a note may require REPORTER access level, updating a note my require DEVELOPER access level, unless the own was owned by the same user and in this case REPORTER access level. Such threshold configuration options can be set to a single access level, which means users with such threshold and above are authorized to do such action. The other option is to specify an array of access level which indicates that users with the explicitly specific thresholds are allowed to do such actions.

It is also worth mentioning that the access levels are defined by the $g_access_levels_enum_string configuration option, and it is possible to customize such list. The default value for the available access levels is '10:viewer, 25:reporter, 40:updater, 55:developer, 70:manager, 90:administrator'. The instructions about how to customize the list of access levels will be covered in the customization section.

Auto Creation of Accounts on Login

In some cases MantisBT is setup in a way, where it allows users that already exists in a directory or another application to be automatically authenticated and added to MantisBT. For example, a company may setup their MantisBT installation in a way, where its staff members that are already registered in their LDAP directory, should be allowed to login into MantisBT with the same user name and password. Another example, is where MantisBT is integrated into some content management system, where it is desired to have a single registration and single sign-on experience. In such scenarios, once a user logs in for the first time, a user account is automatically created for them, although the password verification is still done against LDAP or the main users repository.

User Preferences

Users can fine tune they way MantisBT interacts with them via modifying their user preferences. User preferences can only be managed by users and are not available for the administrators to tweak. The administrators can only tweak the default value for such preferences. However, once a user account is created, it is then the responsibility of the user to manage their own preferences. The user preferences include the following:

- Default Project: A user can choose the default project that is selected when the user first logs in. This can be a specific project or "All Projects". For users that only work on one project, it would make sense to set such project as the default project (rather than "All Projects"). The active project is part of the filter applied on the issues listed in the "View Issues" page. Also any newly reported issues will be associated with the active project.
- Advanced Report: If selected, then the user will use the advanced report issue page by default (rather than the simple one). This preference is only applicable when the administrator configures MantisBT to support both simple and advanced views (i.e. $g_show_report is set to BOTH). If $g_show_report is set to SIMPLE_ONLY or ADVANCED_ONLY, then this preference has no effect.
- Advanced View: If selected, then the user will get the issue advanced view page by default. This preference is only applicable when the administrator configures MantisBT to support both simple and advanced view pages (i.e. $g_show_view is set to BOTH). If $g_show_view is set to SIMPLE_ONLY or ADVANCED_ONLY, then this preference has no effect.
- Advanced Update: If selected, then the user will get the issue advanced update page by default. This preference is only applicable when the administrator configures MantisBT to support both simple and advanced update pages (i.e. $g_show_update is set to BOTH). If $g_show_update is set to SIMPLE_ONLY or ADVANCED_ONLY, then this preference has no effect.
- Refresh Delay: The refresh delay is used to specify the number of seconds between auto-refreshes of the View Issues page.
- Redirect Delay: The redirect delay is the number of seconds to wait after displaying flash messages like "Issue created successfully", and before the user gets redirected to the next page.
- Notes Sort Order: The preference relating to how notes should be ordered on an issue is viewed or in email notifications. The ascending order is where notes are ordered so that ordered notes appear before newer notes, the descending order is the reverse.
- Email on New: If unticked, then email notifications relating to creation of a new issue would be disabled. Note that the preference is only used to disabled notifications that as per the administrator's configuration, this user would have qualified to receive them.
- Email on Change of Handler: TODO - is this preference used?
- Email on Feedback: TODO - is this preference used?

- Email on Resolved: TODO
- Email on Closed: TODO
- Email on Reopened: TODO
- Email on Note Added: TODO
- Email on Status Change: TODO
- Email on Priority Change: TODO - is this preference used?
- Email Notes Limit: This preference can be used to limit the number of issue notes to view or to be included in an email notifications. Specifying N here means that the latest N notes will be included. The value 0 causes all notes to be included.
- Language: The preferred language of the user. This language is used by the GUI and in email notifications. Note that MantisBT uses UTF8 for encoding the data, and hence, the user can be interacting with MantisBT user interface in Chinese while logging issue data in German.

User Profiles

A user profile describes an environment that the user uses to run the software for which issues are being tracked. The profile information include "Platform", "Operating System", "OS Version", and "Additional Description". Each user has access to profiles that they create (can be multiple), in addition to global ones that are shared created by other users. When reporting issues, users can elect to enter information like platform, operating system, version manually, or they can choose from the list of profiles that are already defined.

Global profiles are typically used by the administrator to define a set of standard profiles that are typically used by the MantisBT users. This makes it easier for the users to use such profiles without having to define create them. The access level required for users to be able to create global profiles is configured by the $g_manage_global_profile_threshold configuration option and it is defaulted to MANAGER.

Chapter 4. Issue Lifecycle and Workflow

Issue Creation

The life cycle of an issue starts with its creation. An issue can be created via one of the following channels:

- MantisBT Web Interface - This is where a user logs into MantisBT and reports a new issue.
- SOAP API - Where an application automatically reports an issue into MantisBT using the SOAP API web services interfaces. For example, the nighlty build script can automatically report an issue if the build fails.
- Email - This is not supported out of the box, but there are existing MantisBT patches that would listen to emails on pre-configured email addresses and adds them to the MantisBT database.
- Others - There can be several other ways to report issues. For example, applications / scripts that directly injects issues into MantisBT database (not recommended, except for one-off migration scripts), or PHP scripts that use the core MantisBT API to create new issues.

Issue Statuses

An important part of issue tracking is to classify issues as per their status. Each team may decide to have a different set of categorization for the status of the issues, and hence, MantisBT provides the ability to customize the list of statuses. MantisBT assumes that an issue can be in one of three stages: opened, resolved and closed. Hence, the customized statuses list will be mapped to these three stages. For example, MantisBT comes out of the box with the following statuses: new, feedback, acknowledged, confirmed, assigned, resolved and closed. In this case "new" -> "assigned" map to opened, "resolved" means resolved and "closed" means closed.

Following is the explanation of what the standard statuses that are shipped with MantisBT means.

- New - This is the landing status for new issues. Issues stay in this status until they are assigned, acknowledged, confirmed or resolved. The next status can be "acknowledged", "confirmed", "assigned" or "resolved".
- Acknowledged - This status is used by the development team to reflect their agreement to the suggested feature request. Or to agree with what the reporter is suggesting in an issue report, although they didn't yet attempt to reproduce what the reporter is referring to. The next status is typically "assigned" or "confirmed".
- Confirmed - This status is typically used by the development team to mention that they agree with what the reporter is suggesting in the issue and that they have confirmed and reproduced the issue. The next status is typically "assigned".
- Assigned - This status is used to reflect that the issue has been assigned to one of the team members and that such team member is actively working on the issue. The next status is typically "resolved".
- Resolved - This status is used to reflect that the issue has been resolved. An issue can be resolved with one of many resolutions (customizable). For example, an issue can be resolved as "fixed", "duplicate", "won't fix", "no change required", etc. The next statuses are typically "closed" or in case of the issue being re-opened, then it would be "feedback".

- Closed - This status reflects that the issue is completely closed and no further actions are required on it. It also typically hides the issue from the View Issues page. Some teams use "closed" to reflect sign-off by the reporter and others use it to reflect the fact that the fix has been released to customers.

Workflow

Now that we have covered how an issue gets created, and what are the different statuses during the life cycle of such issues, the next step is to define the workflow. In other words, how issues move from one status to another and who has access to trigger such transitions. MantisBT provides the ability for teams to define their own custom workflow which works on top of their custom status. The workflow dictates the valid transitions between statuses and the user access level required of the user who triggers such transition.

Workflow Transitions

This "Manage > Manage Configuration > Workflow Transitions" page allows users with ADMINISTRATOR access level to do the following tasks.

- Define the valid next statuses for each status.
- Define the default next status for each status.
- Define the minimum access level required for a user to transition to each status.
- Define the default status for newly created issues.
- Define the status at which the issue is considered resolved. Any issues with this status code greater than or equal to the specified status will be considered resolved.
- Define the status which is assigned to issues that are re-opened.
- Define the required access level to change the workflow.

Note that the scope of the applied change is dependent on the selected project. If "All Projects" is selected, then the configuration is to be used as the default for all projects, unless overidden by a specific project. To configure for a specific project, switch to the project via the combobox at the top right corner of the screen.

Workflow Thresholds

This "Manage > Manage Configuration > Workflow Thresholds" page allows users with ADMINISTRATOR access level to define the thresholds required to do certain actions. Following is a list of such actions and what they mean:

- Report an issue - The access levels that are allowed to report an issue.
- Update an issue - The access levels that are allowed to update the header information of an issue.
- Allow issue to be closed on resolved - The access levels that are allow to resolve and close an issue in one step.
- Allow reporter to close issue - Indicates if reporters should be allowed to close issues reported by them.
- Monitor an issue - The access levels required for a user to be able to monitor an issue. Once a user monitors an issue, the user will be included in all future email notifications relating to changes in the issue.

- Handle an issue - The access levels required for a user to be shown in the list of users that can handle an issue.
- Assign an issue - The access levels required for a user to be able to change the handler (i.e. assign / unassign) an issue.
- Move an issue - The access levels required for a user to be able to move an issue from one project to another. (TODO: are these access levels evaluated against source or destination project?).
- Delete an issue - The access levels required for a user to be able to delete an issue.
- Reopen an issue - The access levels required for a user to be able to re-open a resolved or closed issue.
- Allow Reporter to re-open Issue - Whether the reporter of an issue can re-open a resolved or closed issue, independent of their access level.
- Status to which a reopened issue is set - This is the status to which an issue is set after it is re-opened.
- Resolution to which a reopen issue is set - The resolution to set on issues that are re-opened.
- Status where an issue is considered resolved - The status at which an issue is considered resolved.
- Status where an issue becomes readonly - Issues with such status and above are considered read-only. Read-only issues can only be modified by users with a configured access level. Read-only applies to the issue header information as well as other issue related information like relationships, attachments, notes, etc.
- Update readonly issues - The access levels required for a user to be able to modify a readonly issue.
- Update issue status - The access levels required for a user to be able to modify the status of an issue.
- View private issues - The access levels for a user to be able to view a private issue.
- Set view status (public vs. private) - The access level for a user to be able to set whether an issue is private or public, when reporting the issue. If the user reporting the issues doesn't have the required access, then the issue will be created with the default view state.
- Update view status (public vs private) - The access level required for a user to be able to update the view status (i.e. public vs. private).
- Show list of users monitoring issue - The access level required for a user to be able to view the list of users monitoring an issue.
- Set status on assignment of handler - The access levels required for a user to be able to re-assign an issue when changing its status.
- Status to set auto-assigned issues to - The status - This is the status that is set on issues that are auto assigned to users that are associated with the category that the issuer is reported under.
- Limit reporter's access to their own issues - When set, reporters are only allow to view issues that they have reported.
- Add notes - The access levels required for users to be able to add notes.
- Update notes - The access levels required for users to be able to update issue notes.
- Allow user to edit their own issue notes - A flag that indicates the ability for users to edit issue notes report by them.
- Delete note - The access levels required for a user to delete a note that they may or may not have reported themselves.

- View private notes - The access levels required for a user to be able to view private notes associated with an issue that they have access to view.
- View Change Log - The access levels required for a user to be able to view the change log.
- View Assigned To - The access levels required for a user to be able to know the handler of an issue that they have access to.
- View Issue History - The access levels required for a user to be able to view the history of changes of an issue.
- Send reminders - The access levels required for a user to be able to send reminders to other users relating to an issue that they have access to.

Chapter 5. Configuration

Database

The database settings must be set in order for the package to work properly. These settings should be provided to you by your system administrator or your hosting company.

$g_hostname

Host name or connection string for Database server. The default value is localhost. For MySql, this should be hostname or hostname:port (e.g. localhost:3306).

$g_db_username

User name to use for connecting to the database. The user needs to have read/write access to the MantisBT database. The default user name is "root".

$g_db_password

Password for the specified user name. The default password is empty.

$g_database_name

Name of database that contains MantisBT tables. The default name is 'bugtracker'.

$g_db_schema

The database schema (used in case of DB2), otherwise should be left blank.

$g_db_type

The supported database types include: 'mysql' or 'mysqli' for MySQL, 'pgsql' for PostgreSQL, 'mssql' for MS SQL Server, 'oci8' for Oracle, and 'db2' for DB2. It is important to make sure that the PHP extension corresponding to the database type is enabled.

Path

These path settings are important for proper linking within MantisBT. In most scenarios the default values should work fine, and you should not need to override them.

$g_path

URL to your installation as seen from the web browser; this is what you type into the URL field. Requires trailing '/' character. eg. 'http://www.example.com/mantisbt/'. In the following example https protocol is used: eg. 'https://www.example.com/mantisbt/'. MantisBT will default this to the correct value. However, in some cases it might be necessary to override the default. This is typically needed when an installation can be accessed by multiple URLs (internal vs external).

$g_icon_path

This is the URL to the icons (images) directory as seen from the web browser. All MantisBT images/icons are loaded from this URL. The default value for this URL is based on $g_path (i.e. '%path%images/'). Note that a trailing '/' is required.

$g_short_path

Short web path without the domain name. This requires the trailing '/'.

$g_absolute_path

This is the absolute file system path to the MantisBT installation, it is defaulted to the directory where config_defaults_inc.php resides. Requires trailing '/' character (eg. '/usr/apache/htdocs/mantisbt/').

$g_core_path

This is the path to the core directory of your installation. The default value is usually OK, unless you move the 'core' directory out of your webroot. Requires trailing DIRECTORY_SEPARATOR. character.

$g_class_path

This is the path to the classes directory which is a sub-directory of core by default. The default value is typically OK. Requires trailing DIRECTORY_SEPARATOR. character.

$g_manual_url

This is the url to the MantisBT online manual. Requires trailing '/' character.

Webserver

$g_use_iis

Indicates the IIS (Microsoft Internet Information Server) is the web server on which MantisBT is hosted.

Configuration Settings

$g_global_settings

This option contains the list of regular expressions that are used to determine if it is allowed for a specific configuraiton option to be saved to or loaded from the database. Configuration options that matches the regular expressions are considered global only and hence are only configurable via the config_inc.php file and defaulted by config_defaults_inc.php file.

Signup and Lost Password

$g_allow_signup

Allow users to signup for their own accounts. Default is ON.

$g_max_failed_login_count

Maximum failing login attempts before the account is locked. Once locked, it's required to reset the password (lost password). Value resets to zero at each successfully login. Default is OFF.

$g_notify_new_user_created_threshold_min

The minimum global access level required to be notified when a new user registers via the "signup form". To pick specific access levels that are not necessarily at the higher end of access levels, use an array of access levels. Default is ADMINISTRATOR.

$g_send_reset_password

When set to ON, MantisBT will email the users their new passwords when their accounts are reset. If set to OFF, the password will be reset to blank and no e-mail will be sent. Default is ON.

$g_password_confirm_hash_magic_string

TODO

$g_signup_use_captcha

TODO

$g_system_font_folder

TODO

$g_font_per_captcha

TODO

$g_lost_password_feature

TODO

$g_max_lost_password_in_progress_count

TODO

Open Id

$g_openid_enabled

Controls whether Open Id authentication should be supported or not (ON: enabled, OFF: disabled). Default is OFF.

$g_openid_api_key

The RpxNow API key for the site. Note that each site should be registered separately and get its own api key, otherwise, user logins will be mixed up. This is because the mapping between the open ids and MantisBT database id is stored in rpxnow.

Register on https://rpxnow.com.[1] and create a site to get the API key.

$g_openid_site_name

The name of the site as registered on https://rpxnow.com.[2].

$g_openid_ca_bundle

SSL CA Certificate Bundle. If you get "Error performing HTTP request: SSL certificate problem, verify that the CA cert is OK.", see http://curl.haxx.se/docs/caextract.html to create a recent certificate file.

Email

$g_administrator_email

The administrator's e-mail address. This is mainly prompted to the user in case of errors that might require the intervention of the system administrator. For example, SQL errors. sysadmin@example.com

$g_webmaster_email

The webmaster's e-mail address. This address is displayed in the bottom of all MantisBT pages. webmaster@example.com

$g_from_email

The email address to be used as the source of all emails sent by MantisBT. noreply@example.com

$g_return_path_email

Email address to receive bounced emails.

$g_enable_email_notification

Set to ON to enable e-mail notifications, OFF to disable them. Default is ON. Note that disabling email notifications has no effect on emails generated as part of the user signup process.

$g_default_notify_flags

Associated with each action a list of flags to control who should be notified.The default will be used if the action is not included in $g_notify_flags orif the flag is not included in the specific action definition. The list of actions include: new, assigned, resolved, bugnote, reopened, closed, deleted, feedback.The default is: $g_default_notify_flags = array('reporter' => ON, 'handler' => ON, 'monitor' => ON, 'bugnotes' => ON, 'explicit' => ON, 'threshold_min' => NOBODY, 'threshold_max' => NOBODY); threshold_min and threshold_max are used to send messages to all members of the project whose status is greater than or equal to "threshold_min" and less than or equal to "threshold_max". Sending messages to everyone would set "threshold_min" to ANYBODY and "threshold_max to "NOBODY". To send to all DEVELOPERS and above (as 0.17.5), use DEVELOPER and NOBODY respectively.

$g_notify_flags

Defines the notification flags that are different from the defaults that are defined in $g_default_notify_flags. The following code overrides the default by disabling notifications to bugnote authors and users monitoring the bug on submitting a new bug: $g_notify_flags['new'] = array('bugnotes' => OFF, 'monitor' => OFF); Available actions include:

- 'new': a new bug has been added
- 'reopened': the bug has been reopened
- 'deleted': a bug has been deleted
- 'owner': the bug has been assigned a new owner
- 'bugnote': a bugnote has been added to a bug
- 'sponsor': the sponsorship for the bug has changed (added, deleted or updated)
- 'relation': a relationship for the bug has changed (added, deleted or updated)
- 'monitor': a user is added to the monitor list.

In addition, an action can match the bug status in $g_status_enum_string. Note that spaces in the string are replaced with underscores ('_') in creating the action. Thus, using the defaults, 'feedback' would be a valid action.

$g_email_receive_own

This defines whether users should receive emails for their own actions. This option is defaulted to OFF, hence, users do not receive email notification for their own actions. This can be a source for confusions for users upgrading from MantisBT 0.17.x versions, since in these versions users used to get notified of their own actions.

$g_validate_email

Set to OFF to disable email checking. Default is ON.

$g_check_mx_record

Set to OFF to disable email checking. Default is OFF.

$g_allow_blank_email

If ON, allows the user to omit an email address field. If you allow users to create their own accounts, they must specify an email at that point, no matter what the value of this option is. Otherwise they wouldn't get their passwords.

$g_limit_email_domain

Only allow and send email to addresses in the given domain. This is useful as a security feature and it is also useful in cases like Sourceforge where its servers are only limited to send emails to SourceForge email addresses in order to avoid spam. $g_limit_email_domain = 'users.sourceforge.net';

$g_show_user_email_threshold

This specifies the access level that is needed to have user names hyperlinked with mailto: links. The default value is NOBODY, hence, even administrators won't have this feature enabled.

$g_mail_priority

If use_x_priority is set to ON, what should the value be? Urgent = 1, Not Urgent = 5, Disable = 0 . Default is 3 Some MTAs interpret X-Priority = 0 to mean 'Very Urgent'

$g_phpMailer_method

Select the method to mail by: PHPMAILER_METHOD_MAIL for use of mail() function, PHPMAILER_METHOD_SENDMAIL for sendmail (or postfix), PHPMAILER_METHOD_SMTP for SMTP. Default is PHPMAILER_METHOD_MAIL.

$g_smtp_host

This option specifies the SMTP server to submit messages to. The SMTP server (MTA) then takes on the responsibility of deliverying such messages to their final destinations. To use the local SMTP (if available) set this to 'localhost', otherwise use the fully qualified domain name of the remote SMTP server. Default value is 'localhost'.

$g_smtp_port

The smtp port to use. The typical SMTP ports are 25 and 587. The port to use will depend on the SMTP server configuration and hence others may be used. The default is 25.

$g_smtp_connection_mode

This option allows you to specify the connection mode to the SMTP server. Possible values are '', 'ssl', 'tls'. The default value is ''.

$g_smtp_username

This option allows the use of SMTP Authentication when using a remote SMTP host with PHPMailer. If smtp_username is not '' then the username and password will be used when logging in to the SMTP server. Default is ''.

$g_smtp_password

This is the password that is used in SMTP Authentication . Default is ''.

$g_email_send_using_cronjob

TODO

$g_email_set_category

Specify whether e-mails should be sent with the category set or not. This is tested with Microsoft Outlook. More testing for this feature + other formats will be added in the future. OFF, EMAIL_CATEGORY_PROJECT_CATEGORY (format: [Project] Category). Default is OFF.

$g_email_separator1

Default is str_pad('', 70, '='); This means 70 equal signs.

$g_email_separator2

Default is str_pad('', 70, '-'); This means 70 minus signs.

$g_email_padding_length

Default is 28.

MantisBT uses flags and a threshold system to generate emails on events. For each new event, email is sent to:

- the reporter, qualified by the notify flag 'reporter' below
- the handler (or Assigned to), qualified by the notify flag 'handler' below
- anyone monitoring the bug, qualified by the notify flag 'monitor' below
- anyone who has ever added a bugnote the bug, qualified by the notify flag 'bugnotes' below
- anyone assigned to the project whose access level is greater than or equal to the notify flag 'threshold_min' and less than or equal to the notify flag 'threshold_max' below

From this list, those recipients who meet the following criteria are eliminated:

- the originator of the change, if $g_email_receive_own is OFF
- the recipient either no longer exists, or is disabled
- the recipient has turned their email_on_<new status> preference OFF
- the recipient has no email address extered

Version

$g_show_version

Whether to show the MantisBT version at the bottom of each page or not. Default is ON.

Language

$g_default_language

This is the language used by default in MantisBT. This may be set to 'auto' where MantisBT will try to determine the language from the browser.

$g_language_choices_arr

This is to be set to an array of languages that are available for users to choose from. The default value includes all languages supported by MantisBT. The administrator can limit the languages available for users to choose from by overriding this value. For example, to support English, French and German include the following code: array('english', 'french', 'german'); Of course, administrators can also add their own languages by translating the strings and creating their own language files. You are encouraged to share any translation work that you do with the MantisBT team. This will ensure that the newly created language file is maintained with future MantisBT releases.All language files reside in the lang/ folder. They are all named according to the following pattern: strings_<language>.txt.

$g_fallback_language

This is the language used if MantisBT cannot determine the language from the browser. It defaults to 'english'.As of 0.19.0, this may be set to 'auto' where MantisBT will try to determine the language from the browser.

Note: If a string does not exist in the active language, the English string is used instead.

Display

$g_window_title

This is the browser window title (<TITLE> tag).

$g_page_title

This is a heading that is displayed in the viewing area of the page.

$g_favicon_image

Path to the favorites icon relative to MantisBT root folder (default 'images/favicon.ico').

$g_logo_image

Path to the logo image relative to MantisBT root folder (default 'images/mantis_logo.gif').

$g_logo_url

The default URL to be associated with the logo. By default this is set to $g_default_home_page (which defaults to My View page). Clicking on the logo from any page in the bug tracker will navigate to the URL specified in this configuration option.

$g_show_report

This option is used as a default value for user preferences. This field indicates whether users should get the simple bug report page, advanced bug report page, or both. Possible values are: BOTH, SIMPLE_ONLY, or ADVANCED_ONLY. The default is BOTH.

$g_show_update

This option is used as a default value for user preferences. This field indicates whether users should get the simple bug update page, or the advanced bug update page, or both. Possible values are: BOTH, SIMPLE_ONLY, or ADVANCED_ONLY. The default is BOTH.

$g_show_view

This option is used as a default value for user preferences. This field indicates whether users should get the simple bug view page, or advanced bug view page, or both. Possible values are: BOTH, SIMPLE_ONLY, or ADVANCED_ONLY. The default is BOTH.

> **Note:** Users can set their own default preferences for the show report/update/view if the configuration value is BOTH. However, if the value is set to SIMPLE or ADVANCED, then the users should be forced to use the configured values. Note that these settings apply to all projects.

$g_show_footer_menu

Show the menu at the bottom of the page as well as at the top. Default value is OFF.

$g_show_project_menu_bar

This option specifies whether to add menu at the top of the page which includes links to all the projects. The default value is OFF.

$g_show_assigned_names

When a bug is assigned then replace the word "assigned" with the name of the developer in parenthesis. Default is ON.

$g_show_priority_text

Specifies whether to show priority as text (ON) or icon (OFF) in the view all bugs page. Default is OFF (icon).

$g_view_issues_page_columns

This configuration option is used to select the columns to be included in the View Issues page and in which order. If one of the column is not accessible to the logged in user, or corresponds to a disabled feature, then it will be automatically removed from the list at runtime. Hence, the same column list may show a different set of columns based on the logged in user, the currently selected project and enabled features (e.g. sponsorship_total is only shown if the sponsorship feature is enabled).

The supported columns are: selection, edit, id, project_id, reporter_id, handler_id, priority, reproducibility, projection, eta, resolution, fixed_in_version, view_state, os, os_build, build (for product build), platform, version, date_submitted, attachment, category, sponsorship_total, severity, status, last_updated, summary, bugnotes_count, description, steps_to_reproduce, additional_information. As for custom fields they can be referenced by adding a 'custom_' to their name (e.g. xyz would be custom_xyz).

By default the following columns are selected: selection, edit, priority, id, sponsorship_total, bugnotes_count, attachment, category_id, severity, status, last_updated, summary.

$g_print_issues_page_columns

This configuration option is used to select the columns to be included in the Print Issues page and in which order. See $g_view_issues_page_columns for more details about the supported fields.

By default the following columns are selected: selection, priority, id, sponsorship_total, bugnotes_count, attachment, category_id, severity, status, last_updated, summary

$g_csv_columns

This configuration option is used to select the columns to be included in the CSV export and in which order. See $g_view_issues_page_columns for more details about the supported fields.

By default the following columns are selected: id, project_id, reporter_id, handler_id, priority, severity, reproducibility, version, build, projection, category_id, date_submitted, eta, os, os_build, platform, view_state, last_updated, summary, status, resolution, fixed_in_version, duplicate_id.

$g_excel_columns

This configuration option is used to select the columns to be included in the CSV export and in which order. See $g_view_issues_page_columns for more details about the supported fields.

By default the following columns are selected: id, project_id, reporter_id, handler_id, priority, severity, reproducibility, version, build, projection, category_id, date_submitted, eta, os, os_build, platform, view_state, last_updated, summary, status, resolution, fixed_in_version, duplicate_id.

$g_show_bug_project_links

Show project links when in All Projects mode. Default is ON.

$g_status_legend_position

Specifies the position of the status colour legend, can be: STATUS_LEGEND_POSITION_TOP or STATUS_LEGEND_POSITION_BOTTOM. Default is STATUS_LEGEND_POSITION_BOTTOM.

$g_show_attachments_indicator

In view all bug page, show a clip icon next to bugs that has one or more attachments. The default value is OFF. The reason why this is defaulted to OFF is that it adds an extra query for every bug dispayed in the list.

$g_show_product_version

This controls display of the version dropdown menus in the report, view and update pages. Valid values are ON, OFF, and AUTO. ON forces the display of the dropdown even if no versions are defined. OFF will suppress the dropdown always. AUTO will suppress the dropdown, if no versions are defined for the project.

$g_show_realname

This control will replace the user's userid with their realname. If it is set to ON, and the real name fiels has been populated, the replacement will occur. It defaults to OFF.

$g_show_avatar

Show user avatar (default OFF); the current implementation is based on http://www.gravatar.com, users will need to register there with the same email address used in this MantisBT installation to have their avatar shown.

> **Note:** Upon registration or avatar change, it takes some time for the updated gravatar images to show on sites.

$g_show_avatar_threshold

The threshold of users for which MantisBT should attempt to show the avatar (default DEVELOPER). Note that the threshold is related to the user for whom the avatar is being shown, rather than the user who is currently logged in.

$g_default_avatar

The full URL to the image to be used when a user doesn't have an avatar account.

Time

$g_cookie_time_length

Time for 'permanent' cookie to live in seconds. This is what is used when a user selects "save login". Default is the equivalent of 1 year (30000000).

$g_wait_time

Time to delay between page redirects (in seconds). Users can override this setting in their user preferences. Default is 2 seconds.

$g_content_expire

Time to wait before document is stale (in minutes). This is used in meta_inc.php. Default is 0 (expires right away).

$g_long_process_timeout

This timeout is used by pages which does time consuming operations like upgrading the database. The default value of 0 disables timeout. Note that this timeout is specified in seconds.

JpGraph

Jpgraph is a package that is used to render graphs. It is used by MantisBT to provide the users with graphs that capture the state of the bugs database. Following are the configuration options that are related to configuring it:

$g_use_jpgraph

Enable the use of jpgraph. Default is OFF.

$g_jpgraph_path

Path to jpgraph base directory. Don't forget to add the trailing '/'.

Note: To use the Jpgraph addon you need the JpGraph[4] package. You can place the package whereever you want, but you have to set the var in jpgraph.php eg. DEFINE("DIR_BASE","/www/mantisbt/jpgraph/");

Warning

Please note that JpGraph is free to use only for non-commercial, open-source or educational use. All other uses are allowed with the professional version[5].

Date

These variables control how the date is displayed (default is 'US' formatting). Go to the date()[6] function in PHP online manual for detailed instructions on date formatting.

$g_short_date_format

This format is used in the bug listing pages (eg: View Bugs). Default is 'm-d-y'.

$g_normal_date_format

This format is used in the view/update bug pages, bug notes, manage section, and news section. Default is 'm-d-y H:i'.

$g_complete_date_format

This format is used on the top of each page (current time) and the emails that are sent out. Default is 'm-d-y H:i T'.

News

These options are used to control the query that selects the news entries to be displayed.

$g_news_limit_method

Limit the news entry that are displayed by number of entries (BY_LIMIT) or by date (BY_DATE). The default is BY_LIMIT.

$g_news_view_limit

The limit for the number of news entries to be displayed. This option is only used if $g_news_limit_method is set to BY_LIMIT.

$g_news_view_limit_days

Specifies the number of dates after which the news are not displayed. This option is only used if $g_news_limit_method is set to BY_DATE.

$g_private_news_threshold

Specifies the access level required to view private news. The default is DEVELOPER.

Default Preferences

$g_default_new_account_access_level

This is the default access level users are given when their account is created by email. The default access level is REPORTER. Look in constant_inc.php for other values.

$g_default_bug_view_status

The default viewing status for the new bug (VS_PUBLIC or VS_PRIVATE). The default is VS_PUBLIC.

$g_default_bugnote_view_status

The default viewing status for the new bugnote (VS_PUBLIC or VS_PRIVATE). The default is VS_PUBLIC.

$g_default_reminder_view_status

The default viewing status for the new reminders (VS_PUBLIC or VS_PRIVATE). The default is VS_PUBLIC.

$g_reminder_receive_threshold

The minimum access level for a user to show up in the reminder user picker. Note that this is the access level for the project for which the issue belongs. The default is DEVELOPER.

$g_default_bug_severity

The severity for a newly created issue. The default is MINOR. Look in constant_inc.php for other values.

$g_default_bug_priority

The priority for a newly created issue. The default is NORMAL. Look in constant_inc.php for other values.

$g_default_limit_view

Number of bugs to show in the View Bugs page. The default value is 50.

$g_default_show_changed

Highlight bugs that have changed during the last N hours. The default value is 6.

$g_hide_status_default

Controls which issues will be displayed in the View Issues page. Default value is CLOSED, implying that all issues at "closed" or higher state will not be shown.

$g_min_refresh_delay

This is the delay between automatic refreshes of the View Issues page in minutes. Make sure refresh delay in user preferences isn't too short. If a users set their preferences to be lower then it is bumped back up to this minimum value. The default value is 10 minutes.

These settings are used as the default values for preferences for new users. Each user can override these settings through the user preferences form. Default language is set to default site language ($g_default_language).

$g_default_advanced_report

Default user preferences to use the advanced page for reporting bugs. Default is OFF.

$g_default_advanced_view

Default user preferences to use the advanced page for view bugs. Default value is OFF.

$g_default_advanced_update

Default user preferences to use the advanced page for updating bugs. Default value is OFF.

$g_default_refresh_delay

Default page refresh delay (in minutes). This is for the bug listing pages. Default value is 30 minutes.

$g_default_redirect_delay

Default delay before a user is redirected to a page after being prompted by a message (eg: operational successful). Default value is 2 seconds.

$g_default_bugnote_order

This controls the time order in which bug notes are displayed. It can be either ASC (oldest first, the default) or DESC (newest first).

$g_default_email_on_new$g_default_email_on_assigned$g_default_email_on_feedback$g_default_email_on_r

Default user preferences to enable receiving emails when a bug is set to the corresponding status. This option only has an effect if users have the required access level to receive such emails. Default value is ON.

$g_default_email_on_reopened

Default user preferences to enable receiving emails when bugs are re-opened. Default value is ON.

$g_default_email_on_bugnote

Default user preferences to enable receiving emails when bugnotes are added to bugs. Default value is ON.

$g_default_email_on_status$g_default_email_on_priority

Default user preferences to enable receiving emails when status or priority is changed. Default is ON. Note that this option is not implemented.

$g_default_email_on_new_minimum_severity$g_default_email_on_assigned_minimum_severity$g_default_er

Default user preferences to enable filtering based on issue severity. These correspond to the email_on_<status> settings. Default is 'any'.

$g_default_email_on_bugnote_minimum_severity

Default user preference to enable filtering based on issue severity. These corresponds to the email_on_bugnote setting. Default is 'any'.

$g_default_email_on_status_minimum_severity$g_default_email_on_priority_minimum_severity

Default user preferences to enable filtering based on issue severity. These correspond to the email_on_status and email_on_priority settings. Default is 'any'. Note that this option is not yet implemented.

See also: Email Notifications [7]

Summary

These are the settings that are used to configuration options related to the Summary page. This page contains statistics about the bugs in MantisBT.

$g_reporter_summary_limit

Limit how many reporters to show in the summary page. This is useful when there are dozens or hundreds of reporters. The default value is 10.

$g_date_partitions

An array of date lengths to count bugs by (in days) for the summary by date. The default is to count for 1, 2, 3, 7, 30, 60, 90, 180, and 365.

$g_summary_category_include_project

Specifies whether category names should be preceeded by project names (eg: [Project] Category) when the summary page is viewed for all projects. This is useful in the case where category names are common accross projects. The default is OFF.

$g_view_summary_threshold

Specifies the access level required to view the summary page. Default is VIEWER.

Bugnote

$g_bugnote_order

Order to use for sorting bugnotes by submit date. Possible values include ASC for ascending and DESC for descending order. The default value is ASC.

File Upload

MantisBT allows users to upload file attachements and associated them with bugs as well as projects. Bug attachments / project documents can be uploaded to the webserver, database, or an FTP server. When bugs are uploaded to the webserver they are uploaded to the path that is configured in the project properties.In case of problems getting the file upload feature to work, check the following resources: PHP Manual [8] .

MantisBT FAQ

$g_allow_file_upload

Whether to allow/disallow uploading of attachments. Default value is ON.

$g_file_upload_method

Specify the location for uploading attachements. This can be DISK, DATABASE, or FTP. In case of FTP, the files are saved on the webserver (same as disk) as well as on the specified FTP server. Default value is DATABASE. In case of DISK / FTP upload methods you need to provide the webserver with write access rights to the configured upload path (configured in the project) and temporary upload path (used by PHP).

$g_max_file_size

The maximum file size to allow as an attachment. You may also have to configure your php.ini file to increase the execution time, memory limit, max post size, and max upload size.

$g_file_upload_ftp_server

Address of the FTP server to write to (eg: ftp.example.com). This option is only effective if upload method is FTP.

$g_file_upload_ftp_user

FTP user name for account to be used in uploading files to the FTP server. This account must have read/write access to the FTP server. The default path for the account is used for uploading the files.

$g_file_upload_ftp_pass

Password to use when loggin in to the FTP server.

$g_max_file_size

Maximum file size that can be uploaded. Default value is about 5MB. The max file upload size is also affected by the value specified in php.ini. The PHP value is usually defaulted to 2MB.

$g_allowed_files

Files that are allowed. Separate items by commas. eg. "zip,bmp,gif,jpg,txt" If $g_allowed_files is filled in NO other file types will be allowed. If empty it will assume any files are accepted that pass the $g_disallowed_files list.

$g_disallowed_files

Files that are not allowed. Separate items by commas. eg. "php,php3,phtml,html,class,java,exe,pl" $g_disallowed_files takes precedence over $g_allowed_files. It is recommended to disable all extensions that can be executed by your server.

$g_document_files_prefix

Prefix to give to uploaded files when saved to the upload directory. This is used for documents that are attached to projects in order to be able to differentiate them from files that are attached to bugs. The name of the file has the following format prefix-projectcode-filename. The default value is 'doc'.

$g_preview_attachments_inline_max_size

This limit applies to previewing of image / text attachments. If the attachment size is smaller than the specified value, the attachment is previewed with the issue details. The previewing can be disabled by setting this configuration to 0. The default value is 256 * 1024 (256KB).

HTML

$g_html_tags

This is the list of HTML tags that are allowed.Do NOT include href or img tags here.Do NOT include tags that have parameters (eg.)The HTML code is allowed to enter the database as is. The $g_allow_href_tags does not have to be enabled to make URL links. The package will automatically hyperlink properly formatted URLs eg. http://blah.blah/ or mailto://me@more.com/

$g_hr_size

hr size.

$g_hr_width

hr width. Leave off the percentage (%) symbol.

$g_bottom_include_page

If this page eixsts it will be displayed at the bottom of every page. It makes a good company branding include page.

$g_top_include_page

If this page eixsts it will be displayed at the top of every page. It makes a good company branding include page.

$g_css_include_file

Set this to point to the CSS file of your choice.

$g_meta_include_file

Set this to point to the META tag file of your choice.

$g_main_menu_custom_options

This option will add custom options to the main menu. It is an array of arrays listing the caption, access level required, and the link to be executed. For example: $g_main_menu_custom_options = array(array("My Link", MANAGER, 'my_link.php'), array("My Link2", ADMINISTRATOR, 'my_link2.php')); Note that if the caption is found in custom_strings_inc.php, then it will be replaced by the

translated string. Options will only be added to the menu if the current logged in user has the appropriate access level.

Authentication

$g_login_method

- MD5
- LDAP
- PLAIN
- CRYPT
- CRYPT_FULL_SALT
- BASIC_AUTH
- Some systems (mostly non-unix) do not have crypt support in PHP. MD5 will accomplish almost the same thing. PLAIN is plain text and there is no attempt to secure the password in the database. You will not be able to easily convert between encryption methods so this needs to be chosen at install time. MD5 is the default.

$g_reauthentication

TODO

$g_reauthentication_expiry

TODO

LDAP authentication method parameters

$g_ldap_server

The ldap server (eg: ldaps://ldap.example.com)

$g_ldap_port

LDAP port (default 636).

$g_ldap_root_dn

"dc=example, dc=com"

$g_ldap_organisation

"organizationname=*Example)"

$g_use_ldap_email

Use email address in LDAP rather than the email stored in the database.

$g_ldap_bind_dn

"cn=Manager, dc=example, dc=com"

$g_ldap_bind_passwd

Status Settings

$g_bug_submit_status

Status to assign to the bug when submitted. Default value is NEW_.

$g_bug_assigned_status

Status to assign to the bug when assigned. Default value is ASSIGNED.

$g_bug_reopen_status

Status to assign to the bug when reopened. Default value is FEEDBACK.

$g_bug_reopen_resolution

Resolution to assign to the bug when reopened. Default value is REOPENED.

$g_auto_set_status_to_assigned

Automatically set status to $g_bug_assigned_status whenever a bug is assigned to a person. Installations where assigned status is to be used when the defect is in progress, rather than just put in a person's queue should set it to OFF. Default is ON.

$g_bug_resolved_status_threshold

Bug is resolved, ready to be closed or reopened. In some custom installations a bug maybe considered as resolved when it is moved to a custom (FIXED OR TESTED) status.

$g_bug_readonly_status_threshold $g_update_readonly_bug_threshold

Bug becomes readonly if its status is >= $g_bug_readonly_status_threshold. The bug becomes read/write again if re-opened and its status becomes less than this threshold. The default is RESOLVED. Once the bug becomes readonly, a user with an access level greater than or equal to $g_update_readonly_bug_threshold can still edit the bug.

$g_status_enum_workflow

'status_enum_workflow' defines the workflow, and reflects a simple 2-dimensional matrix. For each existing status, you define which statuses you can go to from that status, e.g. from NEW_ you might list statuses '10:new,20:feedback,30:acknowledged' but not higher ones.The default is no workflow, where all states are accessible from any others.

$g_report_bug_threshold

This is the access level required to open a bug. The default is REPORTER.

$g_update_bug_threshold

This is the access level generally required to update the content of a bug. The default is UPDATER.

$g_handle_bug_threshold

This is the access level generally required to be access level needed to be listed in the assign to field. The default is DEVELOPER. If a more restrictive setting can be determined from $g_set_status_threshold, it will be used.

$g_update_bug_status_threshold $g_set_status_threshold

These settings control the access level required to promote a bug to a new status once the bug is opened.$g_set_status_threshold is an array indexed by the status value that allows a distinct setting for each status. It defaults to blank.If the appropriate status is not defined above, $g_update_bug_status_threshold is used instead. The default is DEVELOPER.

$g_allow_close_immediately

If set, bugs are allowed to be resolved and closed in one action. The default is OFF.

$g_allow_reporter_close

If set, the bug reporter is allowed to close their own bugs, regardless of their access level. The default is OFF.

$g_allow_reporter_reopen

If set, the bug reporter is allowed to reopen their own bugs once resolved, regardless of their access level. This allows the reporter to disagree with the resolution. The default is ON.

See also: Customizing Status Values [9]

Filters

$g_filter_by_custom_fields

Show custom fields in the filter dialog and use these in filtering. Defaults to ON.

$g_filter_custom_fields_per_row

The number of custom fields to display per row. The default is 7. The value should be greater than or equal to 7.

$g_view_filters = SIMPLE_DEFAULT;

Controls the display of the filter pages. Possible values are:

- SIMPLE_ONLY - only simple view
- ADVANCED_ONLY - only advanced view (allows multiple value selections)
- SIMPLE_DEFAULT - defaults to simple view, but shows a link for advanced
- ADVANCED_DEFAULT - defaults to advanced view, but shows a link for simple

$g_dhtml_filters = OFF;

Controls the use of DHTML filters that will display the filter in view page using DHTML and javascript. Default is OFF. This requires $g_use_javascript to ne set to ON. This may not work in all browsers as it requires xmlhttprequest functionality.

$g_create_permalink_threshold

The threshold required for users to be able to create permalinks (default DEVELOPER). To turn this feature off use NOBODY.

$g_create_short_url

The service to use to create a short URL. The %s will be replaced by the long URL. By default http://www.tinyurl service is used to shorten URLs.

Misc

$g_limit_reporters

Limit reporters to only viewing bugs that they report.

$g_allow_reporter_close

Allow reporters to close the bugs they reported.

$g_allow_close_immediately

Allow developers and above to close bugs immediately when resolving bugs.

$g_allow_bug_delete_access_level

Allow the specified access level and above to delete bugs.

$g_bug_move_access_level

Allow the specified access level and above to move bugs between projects.

$g_allow_account_delete

Allow users to delete their own accounts.

$g_allow_anonymous_login

Allow easy anonymous access.

$g_anonymous_account

Set the account that users will login as. Make sure this is a viewer or reporter account.

$g_cvs_web

This allows for quick linking to CVS files via CVSweb or ViewCVS.

$g_bug_link_tag

If a number follows this tag it will create a link to a bug. eg. for # a link would be #45 eg. for bug: a link would be bug:98

$g_show_timer

Time page loads. Shows at the bottom of the page.

$g_show_queries_count

Shows the total number/unique number of queries executed to serve the page. Default is ON.

$g_show_queries_list

Shows the list of all queries that are executed in chronological order from top to bottom. This option is only effective when $g_show_queries_count is ON. Default is OFF. WARNING: Potential security hazard. Only turn this on when you really need it (for debugging or profiling)

$g_register_globals

If your register_globals is Off then set this to OFF. Check your register_globals setting in php.ini or phpinfo().

$g_enable_project_documentation

Specifies whether to enable support for project documents or not. Default is ON.

Cookies

The configuration variables $g_string_cookie, $g_project_cookie, $g_view_all_cookie, $g_manage_cookie are calculated based on $g_cookie_prefix. When you change the

cookie prefix in config_inc.php, you need to follow it with a copy of the four lines that calculate the names for these cookies.

$g_cookie_path

This is specifies to the path under which a cookie is visible. All scripts in this directory and its sub-directories will be able to access MantisBT cookies. Default value is '/'. It is recommended to set this to the actual MantisBT path.

$g_cookie_domain

Unused

$g_cookie_version

Cookie version is used as a prefix for cookies that should be expire when the code is changed in a certain way. The developers would increase this version when necessary, which in effect will cause the creation of new cookies that are compatible with the new code. It is not expected for the user to need to change this setting.

$g_cookie_prefix

This should be set to a unique identifier which does not include spaces. Again, this should be unique per MantisBT installation, specially if the $g_cookie_path is not restricting the cookies scope to the actual MantisBT directory.

Database Tables

MantisBT enables users to configure a table prefix for all its tables. This is useful to be able to have multiple MantisBT installation in the same database. The advantage of that is for users who are limited by their ISP to have one database.

$g_db_table_prefix

Specifies the prefix to be use for all table names. The default value is 'mantis'. If you override the default prefix, make sure to update doc/db_generate.sql file before generating your database. The other option is to import db_generate.sql as is, then rename the tables to match the new prefix.

The prefix is used to help make sure table names are unique. This is useful for users who are limited to one database.

> **Note:** The table name for each of the tables is stored in a variable which is calculated based on this configuration option. If you change the prefix you have to make sure these variables are re-calculated (by adding these calculation statements to config_inc.php after assigning the new prefix). An example of these variables is: $g_mantis_bug_file_table

Speed Optimisation

$g_compress_html

This option is used to enable buffering/compression of HTML output if the user's browser supports it. Default value is ON. This option will be ignored in the following scenarios:

- If php.ini has zlib.output_compression enabled.

- If php.ini has output_handler set to a handler.
- If PHP does not include the zlib extension (PHP 4.3.0 and later include zlib extension by default).

You can check the loaded modules in your PHP by running "php -m" on the command line, or by using php_info() command in a php script.

$g_use_persistent_connections

Use persistent database connections, setting this to ON will open the database once per connection, rather than once per page. There might be some scalability issues here and that is why it is defaulted to OFF.

Reminders

Sending reminders is a feature where a user can notify / remind other users about a bug. In the past, only selected users like the managers, or developers would get notified about bugs. However, these people can not invite other people (through MantisBT) to look at or monitor these bugs.

This feature is useful if the Manager needs to get feedback from testers / requirements team about a certain bug. It avoid needing this person to do this manual outside MantisBT. It also records the history of such reminders.

$g_store_reminders

Specifies if reminders should be stored as bugnotes. The bugnote will still reflect that it is a reminder and list the names of users that got it. Default is ON.

$g_reminder_recipients_monitor_bug

Specifies if users who receive reminders about a bug, should be automatically added to the monitor list of that bug. Default is ON.

Bug History

Bug history is a feature where MantisBT tracks all modifications that are made to bugs. These include everything starting from its creation, till it is closed. For each change, the bug history will record the time stamp, user who made the change, field that changed, old value, and new value.

Independent of the these settings, MantisBT will always track the changes to a bug and add them to its history.

$g_history_default_visible

Make the bug history visible by default. If this option is not enabled, then the user will have to click on the Bug History link to see the bug history. Default is ON.

$g_history_order

Show bug history entries in ascending or descending order. Default value is 'ASC'.

Sponsorship

$g_enable_sponsorship

enable/disable the whole issue sponsorship feature. The default os OFF.

$g_sponsorship_currency

The currency string used for all sponsorships. The default is 'US$'.

$g_minimum_sponsorship_amount

The minimum sponsorship amount that can be entered. If the user enters a value less than this, an error will be flagged. The default is 5.

$g_view_sponsorship_total_threshold

The access level threshold needed to view the total sponsorship for an issue by all users. The default is VIEWER.

$g_view_sponsorship_details_threshold

The access level threshold needed to view the details of the sponsorship (i.e., who will donate what) for an issue by all users. The default is VIEWER.

$g_sponsor_threshold

The access level threshold needed to allow user to sponsor issues. The default is REPORTER. Note that sponsoring user must have their email set in their profile.

$g_handle_sponsored_bugs_threshold

The access level required to be able to handle sponsored issues. The default is DEVELOPER.

$g_assign_sponsored_bugs_threshold

The access level required to be able to assign a sponsored issue to a user with access level greater or equal to 'handle_sponsored_bugs_threshold'. The default is MANAGER.

Source Control Integration

$g_source_control_account

Account to be used by the source control script. The account must be enabled and must have the appropriate access level to add notes to all issues even private ones (DEVELOPER access recommended). The default is '' (not set).

$g_source_control_notes_view_status

This sets whether the note added will be public or private (VS_PUBLIC or VS_PRIVATE). For open source projects it is expected that the notes be public, however, for non-open source it will probably be VS_PRIVATE. The default is VS_PRIVATE.

$g_source_control_set_status_to

If set to a status, then after a checkin, the issue status is set to the specified status, otherwise if set to OFF, the issue status is not affected. The default is OFF.

$g_source_control_regexp

Regular expression used to detect issue ids within checkin comments. See preg_match_all()[10] documentation for more details on setting a pattern. The default is "/\bissue [#]{0,1}(\d+)\b/i" (e.g., issue #745).

Custom Fields

$g_manage_custom_fields_threshold

Access level needed to manage custom fields. The default is ADMINISTRATOR.

$g_custom_field_link_threshold

Access level needed to link a custom field to a project. The default is MANAGER.

$$g_custom_field_edit_after_create

This flag determines whether to start editng a custom field immediately after creating it, or return to the definition list. The default is ON (edit the custom field after creating).

My View Settings

$g_my_view_boxes

This is an array of values defining the order that the boxes to be shown. A box that is not to be shown can have its value set to 0. The default is:

```
$g_my_view_boxes = array ( 'assigned' => '1',
                           'unassigned' => '2',
                           'reported' => '3',
                           'resolved' => '4',
                           'recent_mod' => '5',
                           'monitored' => '6'
                           );
```

If you want to change the definition, copy the default value and apply the changes.

$g_my_view_bug_count

Number of bugs shown in each box. The default is 10.

$g_default_home_page

Default page to transfer to after Login or Set Project. The default is 'my_view_page.php'. An alternative would be 'view_all_bugs_page.php' or 'main_page.php'.

Relationships

TODO

System Logging

The system logging interface is used to extract detailed debugging information for the MantisBT system. It can also serve as an audit trail for user actions.

$g_log_level

This controls the type of logging information recorded. Possible values include:

LOG_EMAIL

logs issue id, message type and recipients for all emails sent

LOG_EMAIL_RECIPIENT

logs the details of email recipient determination. Each user id is listed as well as why they are added, or deleted from the recipient list

$g_log_destination

specifies the file where the log data goes. This file must be writable by the web server userid running MantisBT. Right now, only "file:<file path>" is supported. For example, $g_log_destination = 'file:/tmp/mantis_log'; See http://www.php.net/error_log for details.

Notes

1. https://rpxnow.com
2. https://rpxnow.com
3. http://curl.haxx.se/docs/caextract.html
4. http://www.aditus.nu/jpgraph/index.php
5. http://www.aditus.nu/jpgraph/proversion.php
6. http://www.php.net/manual/en/function.date.php
7. manual.customizing.mantis.email.notifications.html
8. http://www.php.net/manual/en/features.file-upload.php
9. manual.customizing.mantis.customizing.status.values.html
10. http://www.php.net/manual/en/function.preg-match-all.php

Chapter 6. Page descriptions

Login page

Just enter your username and password and hit the login button. There is also a Save Login checkbox to have the package remember that you are logged in between browser sessions. You will have to have cookies enabled to login.If the account doesn't exist, the account is disabled, or the password is incorrect then you will remain at the login page. An error message will be displayed.The administrator may allow users to sign up for their own accounts. If so, a link to Signup for your own account will be available.The administrator may also have annonymous login allowed. Annonymous users will be logged in under a common account.You will be allowed to select a project to work in after logging in. You can make a project your default selection from the Select Project screen or from your Account Options.SignupHere you can signup for a new account. You must supply a valid email address and select a unique username. Your randomly generated password will be emailed to your email account. If MantisBT is setup so that the email password is not to be emailed, newly generated accounts will have an empty password.

Main page

This is the first page you see upon logging in. It shows you the latest news updates for the bugtracker. This is a simple news module (based off of work by Scott Roberts) and is to keep users abreast of changes in the bugtracker or project. Some news postings are specific to projects and others are global across the entire bugtracker. This is set at the time of posting in the Edit News section.The number of news posts is controlled by a global variable. When the number of posts is more than the limit, a link to show "older news" is displayed at the bottom. Similarly a "newer news" is displayed when you have clicked on "older news".There is an Archives option at the bottom of the page to view all listings.ArchivesA title/date/poster listing of ALL past news articles will be listed here. Clicking on the link will bring up the specified article. This listing will also only display items that are either global or specific to the selected project.

View Issues page

Here we can view the issue listings. The page has a set of viewing filters at the top and the issues are listed below.FiltersThe filters control the behavior of the issues list. The filters are saved between browsing sessions but do not currently save sort order or direction.If the number of issues exceeds the "Show" count in the filter a set of navigation to go to "First", "Last", "Previous", "Next" and specific page numbers are added.The Search field will look for simple keyword matches in the summary, description, steps to reproduce, additional information, issue id, or issue text id fields. It does not search through issue notes. Issue List - The issues are listed in a table and the attributes are listed in the following order: priority, id, number of issue notes, category, severity, status, last updated, and summary. Each (except for number of issue notes) can be clicked on to sort by that column. Clicking again will reverse the direction of the sort. The default is to sort by last modification time, where the last modified issue appears at the top. The issue id is a link that leads to a more detailed report about the issue. Depending on what you have set in your Account Preferences you will be sent to the simple or advanced view. You can also add issue notes here.The number in the issue note count column will be bold if an issue note has been added in the specified time frame. The addition of an issue note will make the issue note link of the issue appear in the unvisited state.The text in the "Severity" column will be bold if the severity is major, crash, or block and the issue not resolved.The text in the "Updated" column will be bold if the issue has changed in the last "Changed(hrs)" field which is specified in the viewing filters.Each table row is color coded according to the issue status. The colors can be customised through MantisBT Configuration[1] .Severitiesblock - prevents further work/progress from being made crash - crashes the application or blocking, major

- major issue, minor - minor issue, tweak - needs tweaking, text - error in the text, trivial - being nit picky, feature - requesting new feature - Status new - new issue, feedback - issue requires more information from reporter, acknowledged - issue has been looked at but not confirmed or assigned, confirmed - confirmed and reproducible (typically set by an Updater or other Developer), assigned - assigned to a Developer, resolved - issue should be fixed, waiting on confirmation of fix, closed - issue is closed, Moving the mouse over the status text will show the resolution as a title. This is rendered by some browsers as a bubble and in others as a status line text.

Issue View Simple page

Here is the simple listing of the issue report. Most of the fields are self-explanatory. "Assigned To" will contain the developer assigned to handle the issue. Priority is fully functional but currently does nothing of importance. Duplicate ID is used when an issue is a duplicate of another. It links to the duplicate issue which allows users to read up on the original issue report. Below the issue report is a set of buttons that a user can select to work on the issue.

- Update Issue - brings up a page to edit all aspects of the issue
- Assign to - in conjunction with the dropdown list next top the button, this is a shortcut to change the assignment of an issue
- Change Status to - in conjunction with the dropdown list next top the button, this is a shortcut to change the status of an issue. Another page (Change Status) will be presented to allow the user to add notes or change relevant information
- Monitor / Unmonitor Issue - allows the user to monitor any additions to the issue by email
- Create Clone - create a copy of the current issue. This presents the user with a new issue reporting form with all of the information in the current issue filled in. Upon submission, a new issue, related to the current issue, will be created.
- Reopen Issue - Allows the user to re-open a resolved issue
- Move Issue - allows the user to move the issue to another project
- Delete Issue - Allows the user to delete the issue permanently. It is recommended against deleting issues unless the entry is frivolous. Instead issues should be set to resolved and an appropriate resolution category chosen.

A panel is provided to view and update the sponsorship of an issue.Another panel is provided to view, delete and add relationships for an issue. Issues can have a parent/child relationship, where the user is warned about resolving a parent issue before all of the children are resolved. A peer relationship is also possible.Below this, there may be a form for uploading file attachments. The Administrator needs to configure the bugtracker to handle file uploads. If uploading to disk is selected, each project needs to set its own upload path. Issue notes are shown at the bottom of the issue report. A panel to add issue notes is also shown.

Issue View Advanced page

The advanced view is much the same as the simple view with a few additional fields. Here you can see Projection, ETA, Platform, OS, OSBuild, Product Version, Product Build, and Steps to Reproduce.

See also: Issue View Simple page.

Issue Change Status page

This page is used to change the status of an issue. A user can add an issue note to describe the reason for change.In addition, the following fields may be displayed for update:

- Resolution and Duplicate ID - for issues being resolved or closed
- Issue Handler (Assigned to)
- any Custom Fields that are to be visible on update or resolution
- Fixed in Version - for issues being resolved
- Close Immediately - to immediately close a resolved issue

Issue Update Simple page

The layout of this page resemples the Simple Issue View page, but here you can update various issue fields. The Reporter, Category, Severity, and Reproducibility fields are editable but shouldn't be unless there is a gross mis-categorization.Also modifiable are the Assigned To, Priority, Projection, ETA, Resolution, and Duplicate ID fields.As per version 0.18.0, the user can also add an issue note as part of an issue update.

Issue Update Advanced page

Similar to Issue Update Simple page but has the extra advanced fields. The difference between the simple/advanced update pages should be consistent with the difference between the simple/advanced view pages.

My Account Page

This page changes user alterable parameters for the system. These selections are user specific.My AccountThis allows the user to change their password, screen name, and email address. It also reports the user's access levels on the current and other projects.

Preferences

This sets the following information:

- Default project
- whether the pages used for reporting, viewing, and updating are the simple or advanced views
- the delay in minutes between refreshes of the view all issues page
- the delay in seconds when redirecting from a confirmation page to the display page
- the time order in which notes will be sorted
- whether to filter email messages based on type of message and severity
- the number of notes to append to notification emails
- the default language for the system. The additional setting of "auto" will use the browser's default language for the system.

Profiles

Profiles are shortcuts to define the values for Platform, OS, and version. This page allows you to define and edit personal shortcuts.

System Management Pages

A number of pages exist under the "Manage" link. These will only be visible to those who have an appropriate access level.

Manage Users

This page allow an administrator to manage the users in the system.It essentially supplies a list of users defined in the system. The user names are linked to a page where you can change the user's name, access level, and projects to which they are assigned. You can also reset their passwords through this page.At the top, there is also a list of new users (who have created an account in the last week), and accounts where the user has yet to log in.New users are created using the "Create User" link above the list of existing users. Note that the username must be unique in the system. Further, note that the user's real name (as displayed on the screen) cannot match another user's user name.

Manage Projects Page

This page allows the user to manage the projects listed in the system.Each project is listed along with a link to manage that specific project. The specific project pages allow the user to change:

- the project name
- the project description
- its status
- whether the project is public or private. Private projects are only visible to users who are assigned to it or users who have the access level to automatically have access to private projects (eg: administrators).
- afile directory used to store attachments for issues and documents associated with the project. This folder is located on the webserver, it can be absolute path or path relative to the main MantisBT folder. Note that this is only used if the files are stored on disk or via FTP. In case of FTP, the cached version that is saved on the webserver, is stored in the specified path.
- common subprojects. These are other projects who can be considered a sub-project of this one. They can be shared amongst multiple projects. For example, a "documentation" project may be shared amongst several development projects.
- project categories. These are used to sub-divide the issues stored in the system.
- project versions. These are used to create ChangeLog reports and can be used to filter issues. They are used for both the Found In and Fixed In versions.
- Custom Fields linked to this project
- Users linked to this project. Here is the place where a user's access level may be upgraded or downgraded depending on their particular role in the project.

Manage Custom Fields

This page is the base point for managing custom fields. It lists the custom fields defined in the system. There is also a place to enter a new field name to create a new field.The "Edit" links take you to a page where you can define the details of a custom field. These include it's name, type, value, and display information. On the edit page, the following information is defined to control the custom field:

- name
- type. Possible values are listed below.
- Value constraints (Possible values, default value, regular expression, minimum length, maximum length).
- Access (who can read and write the field based on their access level).
- Display control (where the field will show up and must be filled in

All fields are compared in length to be greater than or equal to the minimum length, and less than or equal to the manimum length, unless these values are 0. If the values are 0, the check is skipped. All fields are also compared against the regular expression. If the value matches the expression, then the value is stored. For example, the expression "/^-?([0-9])*$/" can be used to constrain an integer.The table below describes the field types and the value constraints.

Type	Field Contents	Value Constraints
String	text string up to 255 characters	
Numeric	an integer	
Float	a floating point number	
Enumeration	one of a list of text strings	Enter the list of text strings separated by "|" (pipe character) in the Possible Values field. The Default value should match one of these strings as well. This will be displayed as a dropdown menu.
Email	an email address string up to 255 characters	When displayed, the value will also be encapsulated in a mailto: reference.
Checkbox	zero or more of a list of text strings	Enter the list of text strings separated by "|" (pipe character) in the Possible Values field. The Default value should match one of these strings as well. This will be displayed as a list of text strings with a checkbox beside them.

List	one of a list of text strings	Enter the list of text strings separated by "|" (pipe character) in the Possible Values field. The Default value should match one of these strings as well. This will be displayed as a multi-line dropdown menu.
Multiselection List	zero or more of a list of text strings	Enter the list of text strings separated by "|" (pipe character) in the Possible Values field. The Default value should match one of these strings as well. This will be displayed as a multi-line dropdown menu.
Date	text string defining a date	This is displayed as a set of dropdown menus for day, month, and year. Defaults should be defined in yyyy-mm-dd format.

The display entries are used as follows:

Entry	Meaning
Display Only On Advanced Page	If checked, the field will NOT be shown on the simple issue displays
Display When Reporting Issues	If checked, the field will be shown on the report issues displays
Display When Updating Issues	If checked, the field will NOT be shown on the update issue and change status displays
Display When Resolving Issues	If checked, the field will NOT be shown on the update issue displays and change status displays, if the new status is resolved.
Display When Closing Issues	If checked, the field will NOT be shown on the update issue displays and change status displays, if the new status is closed.
Required On Report	If checked, the field must be filled in on the issue reports.
Required On Update	If checked, the field must be filled in on the update issue and change status displays.
Required On Resolve	If checked, the field must be filled in on the update issue and change status displays, if the new status is resolved.
Required On Close	If checked, the field must be filled in on the update issue and change status displays, if the new status is closed.

Notes on Display

- Be careful not to set both a required attribute and show only on advanced display. It may be possible to trigger a validation error that the user cannot recover from (i.e., field is not filled in).

Manage Global Profiles

This page allows the definition of global profiles accessible to all users of the system. It is similar to the user definition of a profile consisting of Platform, OS and Version.

Manage Configuration

This set of pages control the configuration of the MantisBT system. Note that the configuration items displayed may be on a project by project basis.These pages serve two purposes. First, they will display the settings for the particular aspects of the system. If authorized, they will allow a user to change the parameters. They also have settings for what access level is required to change these settings ON A PROJECT basis. In general, this should be left alone, but administrators may want to delegate some of these settings to managers.

Workflow Thresholds

This page covers the adjustment of the settings for many of the workflow related parameters. For most of these, the fields are self explanatory and relate to a similarly named setting in the configuration file. At the right of each row is a selector that allows the administrator to lower the access level required to change the particular parameter.The values changeable on this page are:

Issues

Title	Variable	Description
Report an Issue	$g_report_bug_threshold	threshold to report an issue
Status to which a new issue is set	$g_bug_submit_status	status issue is set to when submitted
Update an Issue	$g_update_bug_threshold	threshold to update an issue
Allow Issue to be closed on Resolve	$g_allow_close_immediately	allow close immediately on resolve
Allow Reporter to close an issue	$g_allow_reporter_close	allow reporter to close issues they reported
Monitor an issue	$g_monitor_bug_threshold	threshold to monitor an issue
Handle Issue	$g_handle_bug_threshold	threshold to handle (be assigned) an issue
Assign Issue	$g_update_bug_assign_threshold	threshold to be in the assign to list
Move Issue	$g_move_bug_threshold	threshold to move an issue to another project. This setting is for all projects.

Delete Issue	$g_delete_bug_threshold	threshold to delete an issue
Reopen Issue	$g_reopen_bug_threshold	threshold to reopen an issue
Allow reporter to reopen Issue	$g_allow_reporter_reopen	allow reporter to reopen issues they reported
Status to which a reopened Issue is set	$g_bug_reopen_status	status issue is set to when reopened
Resolution to which a reopened Issue is set	$g_bug_reopen_resolution	resolution issue is set to when reopened
Status where an issue is considered resolved	$g_bug_resolved_status_threshold	status where bug is resolved
Status where an issue becomes read-only	$g_bug_readonly_status_threshold	status where bug is read-only (see update_readonly_bug_threshold)
Update readonly issue	$g_update_readonly_bug_threshold	threshold to update an issue marked as read-only
Update Issue Status	$g_update_bug_status_threshold	threshold to update an issue's status
View Private Issues	$g_private_bug_threshold	threshold to view a private issue
Set View Status	$g_set_view_status_threshold	threshold to set an issue to Private/Public
Update View Status	$g_change_view_status_threshold	threshold needed to update the view status while updating an issue or an issue note
Show list of users monitoring issue	$g_show_monitor_list_threshold	threshold to see who is monitoring an issue
Set status on assignment of handler	$g_auto_set_status_to_assigned	change status when an issue is assigned
Status to set auto-assigned issues to	$g_bug_assigned_status	status issue is set to when assigned
Limit reporter's access to their own issues	$g_limit_reporters	reporters can see only issues they reported. This setting is for all projects.

Notes

Title	Variable	Description
Add Notes	$g_add_bugnote_threshold	threshold to add an issue note
Update Notes	$g_update_bugnote_threshold	threshold to edit an issue note
Allow users to edit their own bugnotes	$g_bugnote_allow_user_edit_delete	allow user edit/delete their own issue notes
Delete Note	$g_delete_bugnote_threshold	threshold to delete an issue note

View private notes	$g_private_bugnote_threshold	threshold to view a private issue note

Others

View Change Log	$g_view_changelog_threshold	threshold to view the changelog
View Assigned To	$g_view_handler_threshold	threshold to see who is handling an issue
View Issue History	$g_view_history_threshold	threshold to view the issue history
Send Reminders	$g_bug_reminder_threshold	threshold to send a reminder

Workflow Transitions

This page covers the status workflow. For most of these, the fields are self explanatory and relate to a similarly named setting in the configuration file. At the right of each row is a selector that allows the administrator to lower the access level required to change the particular parameter.The values changeable on this page are:

Table 6-1. Issues

Title	Variable	Description
Status to which a new issue is set	$g_bug_submit_status	status issue is set to when submitted
Status where an issue is considered resolved	$g_bug_resolved_status_threshold	status where issue is resolved
Status to which a reopened Issue is set	$g_bug_reopen_status	status issue is set to when reopened

The matrix that follows has checkmarks where the transitions are allowed from the status on the left edge to the status listed across the top. This corresponds to the $g_enum_workflow array.At the bottom, there is a list of access levels that are required to change the status to the value listed across the top. This can be used, for instance, to restrict those who can close an issue to a specific level, say a manager. This corresponds to the $g_set_status_threshold array and the $g_report_bug_threshold setting.

Email Notifications

This page sets the system defaults for sending emails on issue related events. MantisBT uses flags and a threshold system to generate emails on events. For each new event, email is sent to:

- the reporter
- the handler (or Assigned to)
- anyone monitoring the issue

- anyone who has ever added a issue note the issue
- anyone assigned to the project whose access level matches a range

From this list, those recipients who meet the following criteria are eliminated:

- the originator of the change, if $g_email_receive_own is OFF
- the recipient either no longer exists, or is disabled
- the recipient has turned their email_on_<new status> preference OFF
- the recipient has no email address extered

The matrix on this page selects who will receive messages for each of the events listed down the left hand side. The first four columns correspond to the first four points listed above. The next columns correspond to the access levels defined. Note that because a minimum and maximum threshold are used, a discontinuous selection is not allowed.

Monitor Issue

The monitor issues feature allows users to subscribe to certain issues and hence get copied on all notification emails that are sent for these issues.Depending on the configuration, sending a reminder to a user about an issue can add this issue to the user's list of monitored issues. Users who reported the issue or are assigned the issue typically don't need to monitor the issue to get the notifications. This is because by default they get notified on changes related to the issue anyway. However, administrators can change the configuration to disable notifications to reporters or handlers in specific scenarios.

Reopen Issue

Re-open issue button is visible in the issue view pages if the user has the appropriate access level and the issue is resolved/closed. Re-opening a issue will allow users to enter issue notes for the re-opening reason. The issue will automatically be put into the Feedback status.

Delete Issue

The delete issues button appears on the issue view pages for the users who have the appropriate access level. This allows you to delete an existing issue. This should only be used on frivolous or test issues. A confirmation screen will prompt you if you really want to delete the issue. Updaters, Developers, Managers, and Administrators can remove issues (you can also configure this).

Close Issue

This is a button that appears on the issue view pages for users that are authorized to close issues. Depending on the configuration, users may be able to close issues without having to resolve them first, or may be able to only close resolved issues. After the button is clicked, the user is redirected to a page where an issue note maybe added.

Assign to Me

This button appears in the issue view pages in case of users with access level that is equal to handle_bug_threshold or higher. When this button is clicked the issue is assigned to the user.

Resolve Issue

This option on the View Issues page allows you to resolve the issue. It will lead you to a page where you can set the resolution state and a duplicate id (if applicable). After choosing that the user can choose to enter an issue note detailing the reason for the closure. The issue is then set to the Resolved state. The reporter should check off on the issue by using the Close Issue button.

News Syndication

MantisBT supports news syndication using RSS v2.0 protocol. MantisBT also supports authenticated news feeds for private projects or installations where anonymous access is not enabled. Authenticated feeds takes a user name and a key token that are used to authenticate the user and generate the feed results in the context of the user's access rights (i.e. the same as what the user would see if they were to logged into MantisBT).To get access to the News RSS as anonymous user, visit the following page: http://www.example.com/mantisbt/news_rss.php While a user is logged in, the RSS links provided in the UI will always provide links to the authenticated feeds, if no user is logged in (i.e. anonymous), then anonymous links will be provided.

Notes

1. manual.configuration.html

Chapter 7. Customizing MantisBT

Custom Fields

Overview

Different teams typically like to capture different information as users report issues, in some cases, the data required is even different from one project to another. Hence, MantisBT provides the ability for managers and administrators to define custom fields as way to extend MantisBT to deal with information that is specific to their teams or their projects. The aim is for this to keep MantisBT native fields to a minimum. Following are some facts about the implementation of custom fields in MantisBT:

- Custom fields are defined system wide.
- Custom fields can be linked to multiple projects.
- The sequence of displaying custom fields can be different per project.
- Custom fields must be defined by users with access level ADMINISTRATOR.
- Custom fields can be linked to projects by users with access level MANAGER or above (by default, this can be configurable).
- Number of custom fields is not restricted.
- Users can define filters that include custom fields.
- Custom fields can be included in View Issues, Print Issues, and CSV exports.
- Enumeration custom fields can have a set of static values or values that are calculated dynamically based on a custom function.

Custom Field Definition

The definition of a custom field includes the following logical attributes:

- Caption variable name (eg: This is the value that is supplied to lang_get() API, or displayed as-is if not found in language file). This should not include any space or any character that would be an invalid PHP identifier.
- Custom field type (string, numeric, float, enumeration, email, checkbox, radio, list, multi-selection list, date).

  Type 'string' is used for strings of up to 255 characters.

  Type 'numeric' is used for numerical integer values.

  Type 'float' is used for real (float / double) numbers.

  Type 'enumeration' is used when a user selects one entry from a list. The user interface for such type is a combo-box.

  Type 'email' is used for storing email addresses.

  Type 'checkbox' is like enumeration but the list is shown as checkboxes and the user is allowed to tick more than one selection. The default value and the possible value can contain multiple values like 'RED|YELLOW|BLUE' (without the single quote).

  Type 'radio' is like enumeration but the list is shown as radio buttons and th euser is allowed to tick on of the options. The possible values can be 'RED|YELLOW|BLUE',

where the default value can be 'YELLOW'. Note that the default value can't contain multiple values.

Type 'list' is like enumeration but the list is shown as a list box where the user is only allowed to select one option. The possible values can be 'RED | YELLOW | BLUE', where the default value can be 'YELLOW'. Note that the default value can't contain multiple values.

Type 'multi-selection list' is like enumeration but the list is shown as a list box where the user is allowed to select multiple options. The possible values can be 'RED | YELLOW | BLUE', where the default value can be 'RED | BLUE'. Note that in this case the default value contains multiple values.

Type 'date' is for date values. The default value can be empty, or {tomorrow}, {yesterday}, {next week}, {last week}, {+3 days}, {-2 days}.

- Enumeration possible values (eg: RED | YELLOW | BLUE). Use the pipe ('|') character to separate possible values for an enumeration. One of the possible values can be an empty string. The set of possible values can also be calculated at runtime. For example, "=versions" would automatically resolve into all the versions defined for the current project.
- Default value - see details above for a sample default value for each type.
- Minimum/maximum length for the custom field value (use 0 to disable). Note that these metrics are not really relevant to custom fields that are based on an enumeration of possible values.
- Regular expression to use for validating user input (use ereg()[1] syntax).
- Read Access level: Minimum access level for users to be able to see the value of the custom field.
- Write Access level: Minimum access level for users to be able to edit the value of the custom field.
- Display only on Advanced Page? - If set, then this custom field will only show on the advanced pages.
- Display when reporting issues? - If this custom field should be shown on the Report Issue page.
- Display when updating issues? - If this custom field should be shown on the Update Issue page.
- Display when resolving issues? - If this custom field should be shown when resolving an issue. For example, a "root cause" custom field would make sense to set when resolving the issue.
- Display when closing issues? - If this custom field should be shown when closing an issue.
- Required on Report - If this custom field is a mandatory field on the Report Issue page.
- Required on Update - If this custom field is a mandatory field on the Update Issue page.
- Required on Resolve - If this custom field is a mandatory field when resolving an issue.
- Required on Close - If this custom field is a mandatory field when closing an issue.

All custom fields are currently saved to a field of type VARCHAR(255) in the database. However, in future releases, it is possible to support custom fields of different types (eg: memo, file).

If the value of a custom field for a certain defect is not found, the default value is assumed.

Adding/Editing Custom Fields

- The logged in user needs $g_manage_custom_fields_threshold access level.
- Select "Manage" from the main menu.
- Select "Manage Custom Fields" from the management menu.
- In case of edit, click on the name of an existing custom field to edit its information.
- In case of adding a new one, enter the name of the new custom field then click "New Custom Field".

Note: Added custom fields will not show up in any of the issues until the added custom field is linked to the appropriate projects.

Linking/Unlinking/Ordering Existing Custom Fields in Projects

- The logged in user needs to have access level that is greater than or equal to $g_custom_field_link_threshold and $g_manage_project_threshold.
- Select "Manage" from the main menu.
- Select "Manage Projects".
- Select the name of the project to manage.
- Scroll down to the "Custom Fields" box.
- Select the field to add from the list, then click "Add This Existing Custom Field".
- To change the order of the custom fields, edit the "Sequence" value and click update. Custom fields with smaller values are displayed first.
- To unlink a custom field, click on "Remove" link next to the field. Unlinking a custom field will not delete the values that are associated with the issues for this field. These values are only deleted if the custom field definition is removed (not unlinked!) from the database. This is useful if you decide to re-link the custom field. These values may also re-appear if issues are moved to another project which has this field linked.

Moving Issues

When an issue is moved from one project to another, custom fields that are not defined for the new project are not deleted. These fields will re-appear with their correct values if the issue is moved back to the original project, or if these custom fields are linked to the new project.

Localizing Custom Field Names

It is possible to localize the label for the custom fields. This can be as follows:

- Give the custom field a valid variable name (i.e. start with an alpha character, no spaces, etc) - For example, we will use "my_start_date" for a custom field of type "Date" which stores the "Start Date" for working on an issue.

- Add the localized string for "my_start_date" - This can be done by creating custom_strings_inc.php in the MantisBT root folder and adding the following code to it:

```
<?php
if ( lang_get_current() == 'german' ) {
    $s_my_start_date = 'Start Date XX';  // German translation of Start Date
} else {
    # Default (use your preferred language as the default)
    $s_my_start_date = 'Start Date';
}
?>
```

Note: If we would have decided to use start_date as the name of the custom field, then we have used an already localized string from MantisBT standard strings. In this case, there is no need to create the custom_strings_inc.php or to add any strings to it. To check for standard strings, inspect lang/strings_english.txt.

Dynamic default values

Dynamic defaults for Date fields

Custom fields of type date can be defaulted to a specific dates or to relative dates. Typically relative dates is the scenario that makes sense in most of the cases. The format for specific dates is an integer which indicates the number of seconds since the Unix Epoch (January 1 1970 00:00:00 GMT), which is the format consumed by the PHP date()[2] method. The relative scenario expects default values like {tomorrow}, {yesterday}, {+2 days}, {-3 days}, {next week}, etc. The curly brackets indicate that this is a logical value which is then evaluated using the PHP strtotime()[3] function.

Dynamic values for Enumeration Custom Fields

As discussed earlier, one of the possible types of a custom field is "enumeration". This type of custom field allows the user to select one value from a provided list of possible values. The standard way of defining such custom fields is to provide a '|' separated list of possible values. However, this approach has two limitations: the list is static, and the maximum length of the list must be no longer than 255 characters. Hence, the need for the ability to construct the list of possible values dynamically.

Dynamic possible values included by default

MantisBT ships with some dynamic possible values, these include the following:

- =categories - a list of categories defined in the current project (or the project to which the issue belongs).
- =versions - a list of all versions defined in the current project (or the project to which the issue belongs).
- =future_versions - a list of all versions that belong to the current project with released flag set to false.
- =released_versions - a list of all versions that belong to the current project with released flag set to true.

Note: The '=' before the name of the dynamic list of options is used to tell MantisBT that this is a dynamic list, rather than a static list with just one option.

Defining Custom Dynamic Possible Values

If the user selects =versions, the actual custom function that is executed is custom_function_*_enum_versions(). The reason why the "enum_" is not included is to have a fixed prefix for all custom functions used for this purpose and protect against users using custom functions that were not intended for this purpose. For example, you don't want the user to use custom_function_*_issue_delete_notify() which may be overridden by the web master to delete associated data in other databases.

Following is a sample custom function that is used to populate a field with the categories belonging to the currently selected project:

```
# --------------------
# Construct an enumeration for all categories for the current project.
# The enumeration will be empty if current project is ALL PROJECTS.
# Enumerations format is: "abc|lmn|xyz"
# To use this in a custom field type "=categories" in the possible values field.
function custom_function_override_enum_categories() {
    $t_categories = category_get_all_rows( helper_get_current_project() );

    $t_enum = array();
    foreach( $t_categories as $t_category ) {
        $t_enum[] = $t_category['category'];
    }

    $t_possible_values = implode( '|', $t_enum );

    return $t_possible_values;
}
```

Notice the following:

- The custom function doesn't take any parameters.
- The custom function returns the possible values in the format (A|B|C).
- The custom function uses the current project.
- The custom function builds on top of the already existing APIs.

To define your own function \u201c=mine\u201d, you will have to define it with the following signature:

```
# --------------------
# To use this in a custom field type "=mine" in the possible values field.
function custom_function_override_enum_mine() {
    $t_enum = array();

    :

    $t_possible_values = implode( '|', $t_enum );

    return $t_possible_values;
}
```

Notice "override" in the function name. This is because this method is defined by the MantisBT adminstrator/webmaster and not part of the MantisBT source. It is OK to override a method that doesn't exist.

As usual, when MantisBT is upgraded to future releases, the custom functions will not be overwritten. The difference between the "default" implementation and the "override" implementation is explained in more details in the custom functions section.

Enumerations

Enumerations are used in MantisBT to represent a set of possible values for an attribute. Enumerations are used for access levels, severities, priorities, project statuses, project view state, reproducibility, resolution, ETA, and projection. MantisBT provides the administrator with the flexibility of altering the values in these enumerations. The rest of this topic explains how enumerations work, and then how they can be customised.

How enumerations work?

`core/constant_inc.php` defines the constants that correspond to those in the enumeration. These are useful to refer to these enumerations in the configs and the code.

```
define( 'VIEWER', 10 )
define( 'REPORTER', 25 )
define( 'UPDATER',  40 )
define( 'DEVELOPER', 55 )
define( 'MANAGER', 70 )
define( 'ADMINISTRATOR', 90 )
```

`config_defaults_inc.php` includes the defaults for the enumerations. The configuration options that are defaulted here are used in specifying which enumerations are active and should be used in MantisBT. However, the strings included in the enumerations here are just for documentation purpose, they are not shown to the user (due to the need for localisation). Hence, if an entry in this enumeration is not found in the corresponding localised enumeration (i.e. 70:manager), then it will be printed to the user as @70@.

```
$g_access_levels_enum_string =
'10:viewer,25:reporter,40:updater,55:developer,70:manager,90:administrator'
```

`lang/strings_german.txt` provide the localised strings (in this case, in german) for enumerations. But again, the master list is the enumeration in the configs, the ones in the language files are just used for finding the localised equivalent for an entry. Hence, if a user changes the config to have only two types of users developers and administrators, then only those will be prompted to the users even if the enumerations in the language files still includes the full list.

```
$s_access_levels_enum_string =
'10:Betrachter,25:Reporter,40:Updater,55:Entwickler,70:Manager,90:Administr
```

How can they be customised?

Let say we want to remove access level "Updater" and add access level "Senior Developer".

The file `custom_constant_inc.php` is supported for the exclusive purpose of allowing administrators to define their own constants while maintaining a simple upgrade path for future releases of MantisBT. Note that this file is not distributed with MantisBT and you will need to create it if you need such customisation. In our example, we need to define a constant for the new access level.

```
define ( 'SENIOR_DEVELOPER', 60 );
```

In `config_inc.php`

```
// Remove Updater and add Senior Developer
$g_access_levels_enum_string =
'10:viewer,25:reporter,55:developer,60:senior_developer,70:manager,90:admin
// Give access to Senior developers to create/delete custom field.
$g_manage_custom_fields_threshold = SENIOR_DEVELOPER;
```

The file `custom_strings_inc.php` is introduced for a similar reason to that of custom_constant_inc.php, which is to define custom strings. The advantage of defining them here is to provide a simple upgrade path, and avoid having to re-do the changes when upgrading to the next MantisBT release. Note that you will need to create this file if you need such customisation. The file is automatically detected and included by MantisBT code.

```
# Note that we don't have to remove the Updater entry from the
localisation file if ( lang_get_current() === 'english' ) {
$s_access_levels_enum_string =
'10:Betrachter,25:Reporter,40:Updater,55:Entwickler,60:Senior
Developer,70:Manager,90:Administrator'; }
```

Conclusion

We have covered how enumerations work in general, and how to customise one of them. If you are interested in customising other enumerations, a good starting point would be to go to "MantisBT Enum Strings" section in config_defaults_inc.php. This section defines all enumerations that are used by MantisBT. For versions that are older than 0.18.0, custom_*_inc.php files are not supported, and hence you will need to change in the actual constants / language files directly.

Email Notifications

See Email in the Configuration section.

Examples:

- Notify only managers of new issues.

```
$g_notify_flags['new']['threshold_min'] = MANAGER;
$g_notify_flags['new']['threshold_max'] = MANAGER;
```

- Notify Developers and managers of all project events, except, exclude developers from the 'closed' events.

```
$g_default_notify_flags['threshold_min'] = DEVELOPER;
                              $g_default_notify_flags['threshold_max'] = MANAGER;
                              $g_notify_flags['closed']['threshold_max'] = MANAGER;
                              $g_notify_flags['closed']['threshold_max'] = MANAGER;
```

- Exclude those who contributed issue notes from getting messages about other changes in the issue.

  ```
  $g_default_notify_flags['bugnotes'] = OFF;
  ```

- Exclude those monitoring issues from seeing the 'closed' message

  ```
  $g_notify_flags['closed']['monitor'] = OFF;
  ```

- Only notify developers when issue notes are added.

  ```
  $g_notify_flags['bugnote']['threshold_min'] = DEVELOPER;
                              $g_notify_flags['bugnote']['threshold_max'] = DEVELOPER;
  ```

- Notify managers of changes in sponsorship.

  ```
  $g_notify_flags['sponsor']['threshold_max'] = MANAGER;
                              $g_notify_flags['sponsor']['threshold_max'] = MANAGER;
  ```

- Notify originator and managers of changes in ownership ("Assigned To:").

  ```
  $g_notify_flags['owner']['threshold_max'] = MANAGER;
                              $g_notify_flags['owner']['threshold_max'] = MANAGER;
                              $g_notify_flags['owner']['reporter'] = ON;
  ```

- I'm paranoid about mail. Only send information on issues to those involved in them. Don't send mail people already know about. Also send new issue notifications to managers so they can screen them.

  ```
  $g_mail_receive_own = OFF;
                              $g_default_notify_flags =
                              array('reporter' => ON, 'handler' => ON, 'monitor' => ON,
                              'bugnotes' => ON, 'threshold_min' => NOBODY, 'threshold_max'
                              => NOBODY);
                              $g_notify_flags['new']['threshold_min'] = MANAGER;
                              $g_notify_flags['new']['threshold_max'] = MANAGER;
  ```

- How do I replace the $g_to_email configuration variable to log all messages to an email logger.

  You will need to create a dummy user with the appropriate access level for the notices you want to log. Once this user is added to projects, they will receive mail using the appropriate rules.

Customizing Status Values

The default is no workflow, where all states are accessible from any others. The following example can be transferred to config_inc.php. The workflow needs to have a path from the statuses greater than or equal to the resolved state back to the feedback state. Otherwise, the re-open operation won't work.

```
$g_status_enum_workflow[NEW_]=
                '10:new,20:feedback,30:acknowledged,40:confirmed,50:assigned,80:resolved';
                $g_status_enum_workflow[FEEDBACK] =
                '10:new,20:feedback,30:acknowledged,40:confirmed,50:assigned,80:resolved';
                $g_status_enum_workflow[ACKNOWLEDGED] =
                '20:feedback,30:acknowledged,40:confirmed,50:assigned,80:resolved';
                $g_status_enum_workflow[CONFIRMED] =
```

```
'20:feedback,40:confirmed,50:assigned,80:resolved';
$g_status_enum_workflow[ASSIGNED] =
'20:feedback,50:assigned,80:resolved,90:closed';
$g_status_enum_workflow[RESOLVED] =
'50:assigned,80:resolved,90:closed';
$g_status_enum_workflow[CLOSED] = '50:assigned';
```

To add a status:

1. Define a constant to map the new status to.In a new file custom_constant_inc.php in the main mantisbt directory:

   ```
   <?php define ( 'TEST', 60 ); ?>
   ```

2. Define the language strings required. This may need to be defined in several languages.In a new file custom_strings_inc.php in the main mantisbt directory:

   ```
   <?php if ( lang_get_current() == 'german' ) {
                              $s_status_enum_string =
                              '10:neu,20:Rckmeldung,30:anerkannt,40:best%tigt,50:zugewies
                              60:zu testen,80:behoben,90:geschlossen'; } else {
                              $s_status_enum_string =
                              '10:new,20:feedback,30:acknowledged,40:confirmed,50:assigne
                              be tested,80:resolved,90:closed'; $s_to_be_tested_bug_butto
                              "Issue Ready to Test"; $s_to_be_tested_bug_title = "Set Iss
                              to Test"; $s_email_notification_title_for_status_bug_to_be_
                              "The following issue is ready TO BE TESTED."; } ?>
   ```

3. Define any configurations required.In the existing file config_inc.php in the main mantisbt directory:

   ```
   $g_status_enum_string =
                              '10:new,20:feedback,30:acknowledged,40:confirmed,50:assigne
                              be tested,80:resolved,90:closed'; # Status color additions
                              $g_status_colors['to be tested'] = '#ACE7AE';
   ```

4. Add the status to any workflow defined.In config_inc.php:

   ```
   $g_status_enum_workflow[NEW_]=
                              '10:new,20:feedback,30:acknowledged,40:confirmed,50:assigne
                              be tested'; $g_status_enum_workflow[FEEDBACK] =
                              '10:new,20:feedback,30:acknowledged,40:confirmed,50:assigne
                              be tested'; $g_status_enum_workflow[ACKNOWLEDGED] =
                              '20:feedback,30:acknowledged,40:confi rmed,50:assigned,60:t
                              tested'; $g_status_enum_workflow[CONFIRMED] =
                              '20:feedback,40:confirmed,50:assigned,60:to be tested';
                              $g_status_enum_workflow[ASSIGNED] = '20:feedback,50:assigne
                              be tested,90:closed'; $g_status_enum_workflow[TEST] =
                              '10:new,20:feedback,50:assigned,60:to be
                              tested,80:resolved,90:closed'; $g_status_enum_workflow[RESO
                              '50:assigned,60:to be tested,80:resolved,90:closed';
                              $g_status_enum_workflow[CLOSED] = '50:assigned,90:closed';
   ```

Custom Functions

Custom functions are used to extend the functionality of MantisBT by integrating user writtenfunctions into the issue processing at strategic places. This allows the system administrator to changethe functionality without re-writing parts of the internals of the code.

User versions of these functions are placed in a file called custom_functions_inc.php in the root directory of MantisBT. This is the same place that the "config_inc.php" file modifying MantisBT defaults is placed. In normal processing, MantisBT will look for override functions and execute them instead of the provided default functions.

Custom functions have names like custom_function_override_descriptive_name where descriptive namedescribed the particular function. The specific functions are described below. The simplest way tocreate a custom function is to copy the default function, named custom_function_default_descriptive_namefrom the core/custom_function_api.php file to your override file (custom_functions_inc.php), andrename it. The specific functionality you need can then be coded into the override function.

Defined Functions

custom_function_default_changelog_include_issue($p_issue_id) returns true or false if the issue if to be included in the Changelogcustom_function_default_changelog_print_issue($p_issue_id) returns a formatted string to be included for the issue in the Changelogcustom_function_default_checkin($p_issue_id, $p_comment, $p_file, $p_new_version) registers a checkin in source control in MantisBT custom_function_default_issue_update_validate($p_issue_id, $p_new_bug, $p_bug_note_text) validate issue field settings before an update occurs. It returns true or fails with an error.custom_function_default_issue_update_notify($p_issue_id) notify after an issue has been updatedcustom_function_default_issue_create_validate($p_new_bug) validate issue field settings before an issue is created. It returns true or fails with an error.custom_function_default_issue_create_notify($p_issue_id) notify after an issue has been openedcustom_function_default_issue_delete_validate($p_issue_id) validate issue field settings before an issue can be deleted. It returns true or fails with an error.custom_function_default_issue_delete_notify($p_issue_id) notify after an issue has been deleted

Example Custom Function

The following function is used to validate an issue before it is resolved.

```
<?php
# --------------------
# Hook to validate Validate field settings before resolving
# verify that the resolution is not set to OPEN
# verify that the fixed in version is set (if versions of the product exist)

function custom_function_override_issue_update_validate( $p_issue_id, $p_bug_data, $p_bugnote_text ) {
if ( $p_bug_data->status == RESOLVED ) {
if ( $p_bug_data->resolution == OPEN ) {
error_parameters( 'the resolution cannot be open to resolve the issue' );
trigger_error( ERROR_BUG_VALIDATE_FAILURE, ERROR );
}
$t_version_count = count( version_get_all_rows( $p_bug_data-&gt;project_id ) );
if ( ( $t_version_count > 0 ) && ( $p_bug_data->fixed_in_version == '' ) ) {
error_parameters( 'fixed in version must be set to resolve the issue' );
trigger_error( ERROR_BUG_VALIDATE_FAILURE, ERROR );
}
}
}
?>
```

The errors will also need to be defined by adding the following to `custom_constant_inc.php`

```
define ( 'ERROR_VALIDATE_FAILURE', 2000 );
To custom_strings_inc.php
$MANTIS_ERROR['ERROR_VALIDATE_FAILURE'] = 'This change cannot be ma
```

Notes

1. http://www.php.net/ereg
2. http://www.php.net/date
3. http://www.php.net/strtotime

Chapter 8. Authentication

MantisBT supports several authentication techniques out of the box. In addition, there is work in progress relating to supporting authentication plug-ins. Once authentication plug-ins are implemented, then authentication against any protocol or repository of user names and passwords can be done without having to touch MantisBT core code.

Although MantisBT supports multiple authentication techniques, it is important to note that MantisBT doesn't yet support hybrid authentication scenarios. For example, internal staff authentications against LDAP where customer authentications against MantisBT database.

Standard Authentication

Standard, or native, authentication is where MantisBT users are authenticated against user records in the MantisBT database. The passwords are stored in the database in one of several formats:

- CRYPT - deprecated.
- CRYPT_FULL_SALT - deprecated.
- PLAIN - deprecated.
- MD5 - This is default and recommended approach. See MD5 topic on Wikipedia[1] for more details.

See $g_login_methods for more details about how to configure MantisBT to use one of the above authentication techniques.

HTTP_AUTH

When MantisBT is configured to use basic auth, it automatically detects the logged in user and checks if they are already registered in MantisBT, if not, then a new account is automatically created for the username.

BASIC_AUTH

TODO

Open Id

MantisBT can be configured to accept open ids in addition to other supported login schemes. The open id support allows new and existng users to log into MantisBT using any open id identity provider, this includes but is not limited to the following:

- My Open Id
- Google
- Yahoo
- AOL
- Verisign
- Blogspot

- Flickr
- WordPress.com
- Live Journal

A user may use more than one Open Id to login into the same MantisBT account as long as these ids have an associated email address that matches the user's account.

When a user logs in via an Open Id the following steps occur:

- If the open id was used to login into MantisBT before, then the internal user id is retrieved from the open id and the user is logged in.
- If the open id was associated with a user that was deleted, then the user is re-created (if signup is enabled).
- If the open id is associated with a disabled user, then user is not allowed to login.
- If none of the above, then all matching enabled users with the same email address are retrieved. A user with the same exact user name / email gets higher priority, if not, then the user with the highest access level is used.
- If no matches are found based on the email and signup is enabled, then a new user is created.
- If none of the above, then the user is redirected back to the login page.

The MantisBT Open Id implementation is based on the service provided by https://rpxnow.com.[2]. Hence, administrators will need to register for a free account, add their site and update the MantisBT configuration accordingly.

Note that the Open Id support requires the PHP curl extension to be available.

LDAP

Functionality is provided by using the php-ldap module (/usr/lib/php4/ldap.so). An extra login method is defined within core/user_API.php inside of function is_password_match $f_username, $p_test_password, $p_password).This has a simple, non encrypted (yet) test of the LDAP directory for that user by asking for an entry with uid=username and password=test_password, if this exists, it is presumed that the user should be granted access.

Configuration basics

the LDIF format that was tested is as follows:

```
dn: uid=tests,
                        dc=test, dc=com, dc=au
                        department: testdep
                        organizationname: Testing Organization
                        cn: Test Smith
                        assignedgroup: users
                        givename: Test
                        sn: Smith
                        mail: tests@example.com.au
                        uid: testsuser
                        Password: password
                        objectclass: testPerson
```

The password may be in clear, taken from the /etc/passwd or /etc/shadow file, or simply encrypted and added using current LDAP tools.There are some specialized software for replicating passwd to LDAP and inversely (eg. http://freshmeat.net/projects/cpu/).

Also setup the LDAP parameters explained in the Authentication[4] section. Don't forget to change your $g_login_method to LDAP.

Creating new accounts

There is still a bit of problem when you want to create a new user to MantisBT using LDAP, you must create the LDIF entry to LDAP, and also sign up for a new account, if both of these line up correctly, authentication will proceed. Email is queried from the LDAP database if the authentication is set to use LDAP instead of the user record in the database entry.

Microsoft Active Directory

TODO

Notes

1. http://en.wikipedia.org/wiki/MD5
2. https://rpxnow.com
3. http://freshmeat.net/projects/cpu/
4. manual.configuration.authentication.html

Chapter 9. Project Management

This section covers the project management features of MantisBT. This includes features like change log, roadmap, time tracking, reporting and others.

Change Log

MantisBT doesn't just track the status of issues, it also relates issues to versions. Each project can have several versions, which are marked with attributes like released and obsolete. Users typically report issues against released issues and developers typically fix issues in not released versions. With every new release comes question like: what's new? what has been fixed? Customers wonder if the new release is of interest to them and whether they should take an upgrade. Well, the change log is specifically tailored to answer these kind of questions.

In order for an issue to show up in the change log, it has to satisfy certain criteria. The criteria is that the issue has to be resolved with a 'fixed' resolution and has to have the 'fixed_in_version' field set. Users sometimes wonder why resolved or closed issues don't show up in the change log, and the answer is that the 'fixed_in_version' field is not set. Without the 'fixed_in_version', it is not possible for MantisBT to include the issues in the appropriate section of the changelog. Note that it is possible to set the 'fixed_in_version' for multiple issues using the 'Update Fixed in Version' group action on the View Issues page (just below the issues list). This option is only available when the selected project is not 'All Projects'. Once a version is marked as obsolete, it is now longer included in the change log.

MantisBT also provides the ability to customize the criteria used for an issue to be included in the change log. For example, for installations that use a custom set of resolutions, it is possible to select multiple resolutions as valid candidates for the change log. This can be done using custom functions (see custom functions documentation for more details). The custom function below overrides the MantisBT default behavior to include issues with both FIXED and IMPLEMENTED (a custom resolution) resolutions in the change log.

```
# --------------------
# Checks the provided bug and determines whether it should be included in the changelog
# or not.
# returns true: to include, false: to exclude.
function custom_function_override_changelog_include_issue( $p_issue_id ) {
    $t_issue = bug_get( $p_issue_id );

    return ( ( $t_issue->resolution == FIXED || $t_issue->resolution == IMPLEMENTED ) &&
        ( $t_issue->status >= config_get( 'bug_resolved_status_threshold' ) ) );
}
```

MantisBT also provides the ability to customize the details to include from the issue and in what format. This can be done using the following custom function.

```
# --------------------
# Prints one entry in the changelog.
function custom_function_override_changelog_print_issue( $p_issue_id, $p_issue_level = 0 )
    $t_bug = bug_get( $p_issue_id );

    if ( $t_bug->category_id ) {
        $t_category_name = category_get_name( $t_bug->category_id );
    } else {
        $t_category_name = '';
    }

    $t_category = is_blank( $t_category_name ) ? '' : '<b>[' . $t_category_name . ']</b> ';
    echo str_pad( '', $p_issue_level * 6, ' ' ), '- ', string_get_bug_view_link( $p_issue_id
```

```
    if ( $t_bug->handler_id != 0 ) {
        echo ' (', prepare_user_name( $t_bug->handler_id ), ')';
    }

    echo ' - ', get_enum_element( 'status', $t_bug->status ), '.<br />';
}
```

By combining both customization features, it is also possible to do more advanced customization scenarios. For example, users can add a 'ChangelogSummary' custom field and include all issues that have such field in the change log. Through customizing what information being included for a qualifying issue, users can also include the 'ChangelogSummary' text rather than the native summary field.

In some cases, users know that they fixed an issue and that the fix will be included in the next release, however, they don't know yet the name of the release. In such case, the recommended approach is to always have a version defined that corresponds to the next release, which is typicalled called 'Next Release'. Once the release is cut and has a concrete name, then 'Next Release' can be renamed to the appropriate name and a new 'Next Release' can then be created. For teams that manage releases from multiple branches for the same project, then more than one next release can be possible. For example, 'Next Dev Release' and 'Next Stable Release'.

Another common requirement is to be able to link to the change log of a specific project from the project's main website. There is a variety of ways to do that:

- To link to the changelog of version "ver1" of project "myproject":

  `http://www.example.com/mantisbt/changelog_page.php?project=myproject&version=ver1`

- To link to the changelog of all non-obsolete versions of project 'myproject':

  `http://www.example.com/mantisbt/changelog_page.php?project=myproject`

- To link to the changelog of project with id 1. The project id can be figured out by going to the management page for the project and getting the value of project_id field form the URL.

  `http://www.example.com/mantisbt/changelog_page.php?project_id=1`

- To link to the changelog of version with id 1. The version id is unique accross all projects and hence in this case it is not necessary to include the project id/name. The version id can be figured out by going to the manage project page and editing the required version. The version_id will be included in the URL.

  `http://www.example.com/mantisbt/changelog_page.php?version_id=1`

Another approach is to go to the project page and from there users can get to multiple other locations relating to the project include the change log. This can be done by a URL like the following:

`http://www.example.com/mantisbt/project_page.php?project_id=1`

It is possible to customize the access level required for viewing the change log page. This can be done using the $g_view_changelog_threshold configuration option.

Roadmap

One of the very important scenarios in project management is where the project managers (or team leads) triage the issues to set their priorities, target version, and possibly assign the issues to specific developers or take other actions on the issue. By setting the target version of an issue to a version that is not yet released, the issue shows up on the project roadmap, providing user with information about when to expect the issues to be resolved. The roadmap page has a section for each release showing information like planned issues, issues done and percentage of issues completed. Issues that are fixed in a specific version, but didn't have the target_version field set, will not show up in the roadmap. This allows the ability to control the issues that are significant enough to show in the roadmap, while all resolved fields can be found in the change log. Note that it is possible to set the 'target_version' for multiple issues using the 'Update Target Version' group action that is available through the View Issues page (below the issues list). This option is only available when the current project is not 'All Projects'. Although it is not a typical scenario, it is worth mentioning that once a version is marked as obsolete, it is not included in the roadmap.

Note that the roadmap only includes future versions, once a version is marked as released, it no longer is included in the roadmap. For information about such releases, the change log feature should be used. For an issue to be shown on the roadmap, it has to have the target version set. It does not matter whether the feature is resolved or not. Resolved features will be decorated with a strikethrough and will be counted as done.

MantisBT provides the ability to customize the criteria for issues to show up on the roadmap. The default criteria is that the issue has to belong to a version that is not yet released and that the issues is not a duplicate. However, such criteria can be customized by using custom functions as below.

```
# --------------------
# Checks the provided bug and determines whether it should be included in the roadmap or no
# returns true: to include, false: to exclude.
function custom_function_override_roadmap_include_issue( $p_issue_id ) {
    return ( true );
}
```

It is also possible to customize the details included about an issues and the presentation of such details. This can be done through the following custom function:

```
# --------------------
# Prints one entry in the roadmap.
function custom_function_override_roadmap_print_issue( $p_issue_id, $p_issue_level = 0 ) {
    $t_bug = bug_get( $p_issue_id );

    if ( bug_is_resolved( $p_issue_id ) ) {
        $t_strike_start = '<strike>';
        $t_strike_end = '</strike>';
    } else {
        $t_strike_start = $t_strike_end = '';
    }

    if ( $t_bug->category_id ) {
        $t_category_name = category_get_name( $t_bug->category_id );
    } else {
        $t_category_name = '';
    }

    $t_category = is_blank( $t_category_name ) ? '' : '<b>[' . $t_category_name . ']</b> ';

    echo str_pad( '', $p_issue_level * 6, ' ' ), '- ', $t_strike_start, string_get_bug_view_

    if ( $t_bug->handler_id != 0 ) {
        echo ' (', prepare_user_name( $t_bug->handler_id ), ')';
```

```
    }

    echo ' - ', get_enum_element( 'status', $t_bug->status ), $t_strike_end, '.<br />';
}
```

Some teams manage different branches for each of their projects (e.g. development and maintenance branches). As part of triaging the issue, they may decide that an issue should be targetted to multiple branches. Hence, frequently the request comes up to be able to target a single issue to multiple releases. The current MantisBT approach is that an issues represents an implementation or a fix for an issue on a specific branch. Since sometimes applying and verifying a fix to the two branches does not happen at the same time and in some cases the approach for fixing an issue is different based on the branch. Hence, the way to manage such scenario is to have the main issue for the initial fix and have related issues which capture the work relating to applying the fix to other branches. The issues for porting the fix can contain any discussions relating to progress, reflect the appropriate status and can go through the standard workflow process independent of the original issues.

Another common requirement is to be able to link to the roadmap of a specific project from the project's main website. There is a variety of ways to do that:

- To link to the roadmap of version "ver1" of project "myproject":

  ```
  http://www.example.com/mantisbt/roadmap_page.php?project=myproject&version=ver1
  ```

- To link to the roadmap of all non-obsolete versions of project 'myproject':

  ```
  http://www.example.com/mantisbt/roadmap_page.php?project=myproject
  ```

- To link to the roadmap of project with id 1. The project id can be figured out by going to the management page for the project and getting the value of project_id field form the URL.

  ```
  http://www.example.com/mantisbt/roadmap_page.php?project_id=1
  ```

- To link to the roadmap of version with id 1. The version id is unique accross all projects and hence in this case it is not necessary to include the project id/name. The version id can be figured out by going to the manage project page and editing the required version. The version_id will be included in the URL.

  ```
  http://www.example.com/mantisbt/roadmap_page.php?version_id=1
  ```

Another approach is to go to the project page and from there users can get to multiple other locations relating to the project include the roadmap. This can be done by a URL like the following:

```
http://www.example.com/mantisbt/project_page.php?project_id=1
```

The access level required to view and modify the roadmap can be configured through $g_roadmap_view_threshold and $g_roadmap_update_threshold respectively. Modifying the roadmap is the ability to set the target versions for issues. Users who have such access can set the target versions while reporting new issues or by updating existing issues.

Time Tracking

Graphs

Assigned to me: TODO

Release Delta: TODO

Category: TODO

Severity: TODO

Severity / Status: TODO

Daily Delta: TODO

Reported by Me: TODO

Summary Page

By Status: TODO

By Severity: TODO

By Category: TODO

Time Stats for Resolved Issues (days): TODO

Developer Status: TODO

Reporter by Resolution: TODO

Developer by Resolution: TODO

By Date: TODO

Most Active: TODO

Longest Open: TODO

By Resolution: TODO

By Priority: TODO

Reporter Status: TODO

Reporter Effectiveness: TODO

Chapter 10. Contributing to MantisBT

Talent and Time

One of the greatest ways to contribute to MantisBT is to contribute your talent and time. For MantisBT to keep growing we need such support in all areas related to the software development cycle. This includes: business analysts, developers, web designers, graphics designers, technical writers, globalization developers, translators, testers, super users, packagers and active users. If you would like to contribute in any of these capacities please contact us through the "Contact Us" page.

Recommend MantisBT to Others

It feels great when we get feedback from the user community about how MantisBT boosted their productivity, and benefited their organization. A lot of the feedback I get is via email, some on mailing lists, and some on forums. I would encourage such users to blog about it, tell their friends about MantisBT, and recommend MantisBT to other organizations. MantisBT is driven by it's community, the greater the community, the greater the ideas, the greater of a product it becomes.

Blog about MantisBT

If you have a blog, then blog about MantisBT, review it's features and help us spread the word. A lot of users also like to blog about how they customized MantisBT to fit their needs or to integrate with other tools that they use in their work environment.

Integrate with MantisBT

If you have a product that can be integrates with MantisBT to provide a value for MantisBT users, that would be a great place to contribute and benefit both your project's community and the MantisBT community. A great examples in this area are integrations with content management systems (e.g. *Nuke, Xoops), project management (PHPProjekt), and TestLink for Test Management. MantisBT can easily be integrated with projects in any programming language whether it is hosted on the same webserver or anywhere else in the world. This can be achieved through it's SOAP API and MantisConnect client libraries. MantisConnect comes with client libraries and samples in languages like PHP, .NET, Java and Cocoa.

Registered in MantisBT Users Directory

Join the Users Directory[1] by clicking Submit. While entering your company or project details you will be able to write up a testimonial about MantisBT which will then be included in the Testimonials page[2].

Donate Money

Use the PayPal donate buttons that you will see around the website (e.g. bottom left of the home page). You can donate using your credit card even if you don't have a PayPal account. Donations is a very easy way for the MantisBT community to give back to MantisBT. The donations can be as small as US$1 or as big as you or your organization want it to be.

Sponsor MantisBT

MantisBT Sponsors are organizations that have realized the benefit brought to their business by using MantisBT and would like to protect such investment and always push it forward via sponsorship. Please take a moment to visit our sponsors page and use the "Contact Us" page for further information.

Notes

1. http://www.mantisbt.org/directory.php
2. http://www.mantisbt.org/testimonials.php

Acknowledgements

Acknowledgements

We would like to thank all the past and present contributors to the project:

- name
- name
- name
- name