The Bugzilla Guide - 3.0.3 Release

The Bugzilla Team

The Bugzilla Guide - 3.0.3 Release

by The Bugzilla Team

Published 2008-01-08

This is the documentation for Bugzilla, a bug-tracking system from mozilla.org. Bugzilla is an enterprise-class piece of software that tracks millions of bugs and issues for hundreds of organizations around the world.

The most current version of this document can always be found on the Bugzilla Documentation Page (http://www.bugzilla.org/docs/).

Table of Contents

1. About This Guide 1

1.1. Copyright Information 1
1.2. Disclaimer 1
1.3. New Versions 1
1.4. Credits 2
1.5. Document Conventions 2

2. Installing Bugzilla 4

2.1. Installation 4
2.1.1. Perl 4
2.1.2. Database Engine 5
2.1.2.1. MySQL 5
2.1.2.2. PostgreSQL 5
2.1.3. Web Server 5
2.1.4. Bugzilla 5
2.1.5. Perl Modules 6
2.1.5.1. DBD::mysql 8
2.1.5.2. Template Toolkit (2.12) 8
2.1.5.3. GD (1.20) 8
2.1.5.4. Chart::Base (1.0) 8
2.1.5.5. GD::Graph (any) 8
2.1.5.6. GD::Text (any) 8
2.1.5.7. XML::Twig (any) 9
2.1.5.8. SOAP::Lite (any) 9
2.1.5.9. PatchReader (0.9.4) 9
2.1.6. Mail Transfer Agent (MTA) 9
2.1.7. Installing Bugzilla on mod_perl 9
2.2. Configuration 10
2.2.1. localconfig 10
2.2.2. Database Server 10
2.2.2.1. Bugzilla Database Schema 11
2.2.2.2. MySQL 11
2.2.2.3. PostgreSQL 12
2.2.3. checksetup.pl 13
2.2.4. Web server 13
2.2.4.1. Bugzilla using Apache 13
2.2.4.2. Microsoft Internet Information Services 15
2.2.5. Bugzilla 16
2.3. Optional Additional Configuration 17
2.3.1. Bug Graphs 17
2.3.2. Dependency Charts 17
2.3.3. The Whining Cron 18
2.3.4. Whining 18
2.3.5. Patch Viewer 18
2.3.6. LDAP Authentication 19
2.3.7. Serving Alternate Formats with the right MIME type 20

2.4. Multiple Bugzilla databases with a single installation 20
2.5. OS-Specific Installation Notes 21
2.5.1. Microsoft Windows 21
2.5.1.1. Win32 Perl 21
2.5.1.2. Perl Modules on Win32 22
2.5.1.3. Code changes required to run on Win32 22
2.5.1.4. Serving the web pages 22
2.5.1.5. Sending Email 23
2.5.2. Mac OS X 23
2.5.2.1. Sendmail 23
2.5.2.2. Libraries & Perl Modules on Mac OS X 23
2.5.3. Linux Distributions 24
2.6. UNIX (non-root) Installation Notes 24
2.6.1. Introduction 24
2.6.2. MySQL 24
2.6.2.1. Running MySQL as Non-Root 25
2.6.3. Perl 26
2.6.4. Perl Modules 26
2.6.4.1. The Independant Method 26
2.6.4.2. The Mixed Method 27
2.6.5. HTTP Server 28
2.6.5.1. Running Apache as Non-Root 28
2.6.6. Bugzilla 29
2.6.6.1. suexec or shared hosting 29
3. Administering Bugzilla 31
3.1. Bugzilla Configuration 31
3.2. User Administration 33
3.2.1. Creating the Default User 33
3.2.2. Managing Other Users 34
3.2.2.1. Searching for existing users 34
3.2.2.2. Creating new users 34
3.2.2.3. Modifying Users 34
3.2.2.4. Deleting Users 36
3.2.2.5. Impersonating Users 36
3.3. Classifications 36
3.4. Products 37
3.4.1. Assigning Group Controls to Products 38
3.5. Components 39
3.6. Versions 39
3.7. Milestones 40
3.8. Flags 40
3.8.1. A Simple Example 40
3.8.2. About Flags 41
3.8.2.1. Values 41
3.8.3. Using flag requests 41
3.8.4. Two Types of Flags 41
3.8.4.1. Attachment Flags 41

3.8.4.2. Bug Flags 42
3.8.5. Administering Flags....42
3.8.5.1. Creating a Flag 42
3.8.5.2. Deleting a Flag 44
3.8.5.3. Editing a Flag 45
3.9. Keywords....45
3.10. Custom Fields....45
3.10.1. Adding Custom Fields 45
3.10.2. Editing Custom Fields 46
3.10.3. Deleting Custom Fields 46
3.11. Legal Values 46
3.11.1. Viewing/Editing legal values 46
3.11.2. Deleting legal values....46
3.12. Voting 47
3.13. Quips 47
3.14. Groups and Group Security....48
3.14.1. Creating Groups....48
3.14.2. Assigning Users to Groups 49
3.14.3. Assigning Group Controls to Products 49
3.14.4. Common Applications of Group Controls....49
3.14.4.1. General User Access With Security Group....49
3.14.4.2. General User Access With A Security Product....49
3.14.4.3. Product Isolation With Common Group 50
3.15. Checking and Maintaining Database Integrity....50
3.16. Upgrading to New Releases 51
3.16.1. Version Definitions 51
3.16.2. Upgrading - Notifications 51
3.16.3. Upgrading - Methods and Procedure....52
3.16.3.1. Upgrading using CVS 52
3.16.3.2. Upgrading using the tarball....53
3.16.3.3. Upgrading using patches....54
3.16.4. Completing Your Upgrade....54
4. Bugzilla Security 56
4.1. Operating System 56
4.1.1. TCP/IP Ports 56
4.1.2. System User Accounts....56
4.1.3. The `chroot` Jail....56
4.2. MySQL....56
4.2.1. The MySQL System Account....57
4.2.2. The MySQL “root” and “anonymous” Users 57
4.2.3. Network Access 57
4.3. Web server....58
4.3.1. Disabling Remote Access to Bugzilla Configuration Files 58
4.4. Bugzilla 59
4.4.1. Prevent users injecting malicious Javascript....59

5. Using Bugzilla **60**
5.1. Introduction 60
5.2. Create a Bugzilla Account 60
5.3. Anatomy of a Bug 61
5.4. Life Cycle of a Bug 62
5.5. Searching for Bugs 63
5.5.1. Boolean Charts 64
5.5.1.1. Pronoun Substitution 64
5.5.1.2. Negation 64
5.5.1.3. Multiple Charts 65
5.5.2. Quicksearch 65
5.5.3. Case Sensitivity in Searches 65
5.5.4. Bug Lists 65
5.5.5. Adding/removing tags to/from bugs 66
5.6. Filing Bugs 66
5.6.1. Reporting a New Bug 66
5.6.2. Clone an Existing Bug 67
5.7. Attachments 67
5.7.1. Patch Viewer 68
5.7.1.1. Viewing Patches in Patch Viewer 68
5.7.1.2. Seeing the Difference Between Two Patches 68
5.7.1.3. Getting More Context in a Patch 69
5.7.1.4. Collapsing and Expanding Sections of a Patch 69
5.7.1.5. Linking to a Section of a Patch 69
5.7.1.6. Going to Bonsai and LXR 69
5.7.1.7. Creating a Unified Diff 69
5.8. Hints and Tips 69
5.8.1. Autolinkification 69
5.8.2. Comments 70
5.8.3. Server-Side Comment Wrapping 70
5.8.4. Dependency Tree 70
5.9. Time Tracking Information 70
5.10. User Preferences 71
5.10.1. General Preferences 71
5.10.2. Email Preferences 71
5.10.3. Saved Searches 72
5.10.4. Name and Password 73
5.10.5. Permissions 73
5.11. Reports and Charts 74
5.11.1. Reports 74
5.11.2. Charts 74
5.11.2.1. Creating Charts 75
5.11.2.2. Creating New Data Sets 75
5.12. Flags 75
5.13. Whining 76
5.13.1. The Event 77
5.13.2. Whining Schedule 77
5.13.3. Whining Searches 77

5.13.4. Saving Your Changes 78
6. Customizing Bugzilla 79
6.1. Custom Skins 79
6.2. Template Customization 79
6.2.1. Template Directory Structure 79
6.2.2. Choosing a Customization Method 79
6.2.3. How To Edit Templates 80
6.2.4. Template Formats and Types 81
6.2.5. Particular Templates 81
6.2.6. Configuring Bugzilla to Detect the User's Language 83
6.3. The Bugzilla Extension Mechanism 83
6.4. Customizing Who Can Change What 86
6.5. Integrating Bugzilla with Third-Party Tools 87
6.5.1. Bonsai 87
6.5.2. CVS 87
6.5.3. Perforce SCM 88
6.5.4. Subversion 88
6.5.5. Tinderbox/Tinderbox2 88
A. The Bugzilla FAQ 89
B. Troubleshooting 103
B.1. General Advice 103
B.2. The Apache webserver is not serving Bugzilla pages 103
B.3. I installed a Perl module, but `checksetup.pl` claims it's not installed! 103
B.4. DBD::Sponge::db prepare failed 104
B.5. cannot chdir(/var/spool/mqueue) 104
B.6. Everybody is constantly being forced to relogin 104
B.7. Some users are constantly being forced to relogin 105
B.8. `index.cgi` doesn't show up unless specified in the URL 106
B.9. checksetup.pl reports "Client does not support authentication protocol requested by server..." 106
C. Contrib 107
C.1. Command-line Search Interface 107
C.2. Command-line 'Send Unsent Bug-mail' tool 107
D. Manual Installation of Perl Modules 108
D.1. Instructions 108
D.2. Download Locations 108
D.3. Optional Modules 110
E. GNU Free Documentation License 112
0. Preamble 112
1. Applicability and Definition 112
2. Verbatim Copying 113
3. Copying in Quantity 113
4. Modifications 114
5. Combining Documents 115
6. Collections of Documents 115
7. Aggregation with Independent Works 115
8. Translation 116

9. Termination 116
10. Future Revisions of this License 116
How to use this License for your documents 116
Glossary 118

List of Figures

5-1. Lifecycle of a Bugzilla Bug 62

List of Examples

4-1. Assigning the MySQL “root” User a Password 57
4-2. Disabling the MySQL “anonymous” User 57
4-3. Disabling Networking in MySQL 57
B-1. Examples of urlbase/cookiepath pairs for sharing login cookies 105
B-2. Examples of urlbase/cookiepath pairs to restrict the login cookie 105

Chapter 1. About This Guide

1.1. Copyright Information

This document is copyright (c) 2000-2008 by the various Bugzilla contributors who wrote it.

> Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in Appendix E.

If you have any questions regarding this document, its copyright, or publishing this document in non-electronic form, please contact the Bugzilla Team.

1.2. Disclaimer

No liability for the contents of this document can be accepted. Follow the instructions herein at your own risk. This document may contain errors and inaccuracies that may damage your system, cause your partner to leave you, your boss to fire you, your cats to pee on your furniture and clothing, and global thermonuclear war. Proceed with caution.

Naming of particular products or brands should not be seen as endorsements, with the exception of the term "GNU/Linux". We wholeheartedly endorse the use of GNU/Linux; it is an extremely versatile, stable, and robust operating system that offers an ideal operating environment for Bugzilla.

Although the Bugzilla development team has taken great care to ensure that all exploitable bugs have been fixed, security holes surely exist in any piece of code. Great care should be taken both in the installation and usage of this software. The Bugzilla development team members assume no liability for your use of Bugzilla. You have the source code, and are responsible for auditing it yourself to ensure your security needs are met.

1.3. New Versions

This is the 3.0.3 version of The Bugzilla Guide. It is so named to match the current version of Bugzilla.

The latest version of this guide can always be found at http://www.bugzilla.org, or checked out via CVS by following the Mozilla CVS (http://www.mozilla.org/cvs.html) instructions and check out the `mozilla/webtools/bugzilla/docs/` subtree. However, you should read the version which came with the Bugzilla release you are using.

The Bugzilla Guide, or a section of it, is also available in the following languages: French (http://www.traduc.org/docs/guides/lecture/bugzilla/), German (http://bugzilla-de.sourceforge.net/docs/html/), Japanese (http://www.bugzilla.jp/docs/2.18/). Note that these may be outdated or not up to date.

In addition, there are Bugzilla template localization projects in the following languages. They may have translated documentation available: Arabic (http://sourceforge.net/projects/bugzilla-ar/), Belarusian (http://sourceforge.net/projects/bugzilla-be/), Bulgarian (http://openfmi.net/projects/mozilla-bg/), Brazilian Portuguese (http://sourceforge.net/projects/bugzilla-br/), Chinese (http://sourceforge.net/projects/bugzilla-cn/), French (http://sourceforge.net/projects/bugzilla-fr/), German (http://germzilla.ganderbay.net/), Italian (http://sourceforge.net/projects/bugzilla-it/), Japanese (http://www.bugzilla.jp/about/jp.html), Korean

(http://sourceforge.net/projects/bugzilla-kr/), Russian (http://sourceforge.net/projects/bugzilla-ru/) and Spanish (http://sourceforge.net/projects/bugzilla-es/).

If you would like to volunteer to translate the Guide into additional languages, please contact Dave Miller (mailto:justdave@bugzilla.org).

1.4. Credits

The people listed below have made enormous contributions to the creation of this Guide, through their writing, dedicated hacking efforts, numerous e-mail and IRC support sessions, and overall excellent contribution to the Bugzilla community:

Matthew P. Barnson <`mbarnson@sisna.com`>

for the Herculean task of pulling together the Bugzilla Guide and shepherding it to 2.14.

Terry Weissman <`terry@mozilla.org`>

for initially writing Bugzilla and creating the README upon which the UNIX installation documentation is largely based.

Tara Hernandez <`tara@tequilarists.org`>

for keeping Bugzilla development going strong after Terry left mozilla.org and for running landfill.

Dave Lawrence <`dkl@redhat.com`>

for providing insight into the key differences between Red Hat's customized Bugzilla.

Dawn Endico <`endico@mozilla.org`>

for being a hacker extraordinaire and putting up with Matthew's incessant questions and arguments on irc.mozilla.org in #mozwebtools

Jacob Steenhagen <`jake@bugzilla.org`>

for taking over documentation during the 2.17 development period.

Dave Miller <`justdave@bugzilla.org`>

for taking over as project lead when Tara stepped down and continually pushing for the documentation to be the best it can be.

Thanks also go to the following people for significant contributions to this documentation: Kevin Brannen, Vlad Dascalu, Ben FrantzDale, Eric Hanson, Zach Lipton, Gervase Markham, Andrew Pearson, Joe Robins, Spencer Smith, Ron Teitelbaum, Shane Travis, Martin Wulffeld.

Also, thanks are due to the members of the mozilla.support.bugzilla (news://news.mozilla.org/mozilla.support.bugzilla) newsgroup (and its predecessor, netscape.public.mozilla.webtools). Without your discussions, insight, suggestions, and patches, this could never have happened.

1.5. Document Conventions

This document uses the following conventions:

Descriptions	Appearance
Caution	**Caution** Don't run with scissors!
Hint or Tip	**Tip:** For best results...
Note	**Note:** Dear John...
Warning	**Warning** Read this or the cat gets it.
File or directory name	`filename`
Command to be typed	**command**
Application name	application
Normal user's prompt under bash shell	bash$
Root user's prompt under bash shell	bash#
Normal user's prompt under tcsh shell	tcsh$
Environment variables	VARIABLE
Term found in the glossary	*Bugzilla*
Code example	`<para> Beginning and end of paragraph</para>`

This documentation is maintained in DocBook 4.1.2 XML format. Changes are best submitted as plain text or XML diffs, attached to a bug filed in the Bugzilla Documentation (https://bugzilla.mozilla.org/enter_bug.cgi?product=Bugzilla&component=Documentation) component.

Chapter 2. Installing Bugzilla

2.1. Installation

> **Note:** If you just want to *use* Bugzilla, you do not need to install it. None of this chapter is relevant to you. Ask your Bugzilla administrator for the URL to access it over the web.

The Bugzilla server software is usually installed on Linux or Solaris. If you are installing on another OS, check Section 2.5 before you start your installation to see if there are any special instructions.

As an alternative to following these instructions, you may wish to try Arne Schirmacher's unofficial and unsupported Bugzilla Installer (http://www.softwaretesting.de/article/view/33/1/8/), which installs Bugzilla and all its prerequisites on Linux or Solaris systems.

This guide assumes that you have administrative access to the Bugzilla machine. It not possible to install and run Bugzilla itself without administrative access except in the very unlikely event that every single prerequisite is already installed.

> **Warning**
>
> The installation process may make your machine insecure for short periods of time. Make sure there is a firewall between you and the Internet.

You are strongly recommended to make a backup of your system before installing Bugzilla (and at regular intervals thereafter :-).

In outline, the installation proceeds as follows:

1. Install Perl (5.8.0 or above for non-Windows platforms; 5.8.1 for Windows)
2. Install a Database Engine
3. Install a Webserver
4. Install Bugzilla
5. Install Perl modules
6. Install a Mail Transfer Agent (Sendmail 8.7 or above, or an MTA that is Sendmail-compatible with at least this version)
7. Configure all of the above.

2.1.1. Perl

Installed Version Test: `perl -v`

Any machine that doesn't have Perl on it is a sad machine indeed. If you don't have it and your OS doesn't provide official packages, visit http://www.perl.com. Although Bugzilla runs with Perl 5.8.0, it's a good idea to be using the latest stable version.

2.1.2. Database Engine

From Bugzilla 2.20, support is included for using both the MySQL and PostgreSQL database servers. You only require one of these systems to make use of Bugzilla.

2.1.2.1. MySQL

Installed Version Test: `mysql -V`

If you don't have it and your OS doesn't provide official packages, visit http://www.mysql.com. You need MySQL version 4.1.2 or higher.

> **Note:** Many of the binary versions of MySQL store their data files in `/var`. On some Unix systems, this is part of a smaller root partition, and may not have room for your bug database. To change the data directory, you have to build MySQL from source yourself, and set it as an option to `configure`.

If you install from something other than a packaging/installation system, such as .rpm (Redhat Package), .deb (Debian Package), .exe (Windows Executable), or .msi (Microsoft Installer), make sure the MySQL server is started when the machine boots.

2.1.2.2. PostgreSQL

Installed Version Test: `psql -V`

If you don't have it and your OS doesn't provide official packages, visit http://www.postgresql.org/. You need PostgreSQL version 8.00.0000 or higher.

If you install from something other than a packaging/installation system, such as .rpm (Redhat Package), .deb (Debian Package), .exe (Windows Executable), or .msi (Microsoft Installer), make sure the PostgreSQL server is started when the machine boots.

2.1.3. Web Server

Installed Version Test: view the default welcome page at http://<your-machine>/

You have freedom of choice here, pretty much any web server that is capable of running *CGI* scripts will work. However, we strongly recommend using the Apache web server (either 1.3.x or 2.x), and the installation instructions usually assume you are using it. If you have got Bugzilla working using another webserver, please share your experiences with us by filing a bug in Bugzilla Documentation (https://bugzilla.mozilla.org/enter_bug.cgi?product=Bugzilla&component=Documentation).

If you don't have Apache and your OS doesn't provide official packages, visit http://httpd.apache.org/.

2.1.4. Bugzilla

Download a Bugzilla tarball (or check it out from CVS) and place it in a suitable directory, accessible by the default web server user (probably "apache" or "www"). Good locations are either directly in the main web space for your web server or perhaps in `/usr/local` with a symbolic link from the web space.

> **Caution**
>
> The default Bugzilla distribution is NOT designed to be placed in a `cgi-bin` directory. This includes any directory which is configured using the `ScriptAlias` directive of Apache.

Once all the files are in a web accessible directory, make that directory writable by your webserver's user. This is a temporary step until you run the `checksetup.pl` script, which locks down your installation.

2.1.5. Perl Modules

Bugzilla's installation process is based on a script called `checksetup.pl`. The first thing it checks is whether you have appropriate versions of all the required Perl modules. The aim of this section is to pass this check. When it passes, proceed to Section 2.2.

At this point, you need to `su` to root. You should remain as root until the end of the install. To check you have the required modules, run:

```
bash# ./checksetup.pl --check-modules
```

`checksetup.pl` will print out a list of the required and optional Perl modules, together with the versions (if any) installed on your machine. The list of required modules is reasonably long; however, you may already have several of them installed.

There is a meta-module called Bundle::Bugzilla, which installs all the other modules with a single command. You should use this if you are running Perl 5.6.1 or above.

The preferred way of installing Perl modules is via CPAN on Unix, or PPM on Windows (see Section 2.5.1.2). These instructions assume you are using CPAN; if for some reason you need to install the Perl modules manually, see Appendix D.

```
bash# perl -MCPAN -e 'install "<modulename>"'
```

If you using Bundle::Bugzilla, invoke the magic CPAN command on it. Otherwise, you need to work down the list of modules that `checksetup.pl` says are required, in the order given, invoking the command on each.

> **Tip:** Many people complain that Perl modules will not install for them. Most times, the error messages complain that they are missing a file in "@INC". Virtually every time, this error is due to permissions being set too restrictively for you to compile Perl modules or not having the necessary Perl development libraries installed on your system. Consult your local UNIX systems administrator for help solving these permissions issues; if you *are* the local UNIX sysadmin, please consult the newsgroup/mailing list for further assistance or hire someone to help you out.

> **Note:** If you are using a package-based system, and attempting to install the Perl modules from CPAN, you may need to install the "development" packages for MySQL and GD before attempting to install the related Perl

modules. The names of these packages will vary depending on the specific distribution you are using, but are often called `<packagename>-devel`.

Here is a complete list of modules and their minimum versions. Some modules have special installation notes, which follow.

Required Perl modules:

1. CGI 2.93 or CGI 3.11 if using mod_perl
2. Date::Format (2.21)
3. DBI (1.41)
4. DBD::mysql (2.9003) if using MySQL
5. DBD::Pg (1.45) if using PostgreSQL
6. File::Spec (0.84)
7. Template (2.12)
8. Email::Send (2.00)
9. Email::MIME::Modifier (any)

Optional Perl modules:

1. GD (1.20) for bug charting
2. Template::Plugin::GD::Image (1.20) for Graphical Reports
3. Chart::Base (1.0) for bug charting
4. GD::Graph (any) for bug charting
5. GD::Text (any) for bug charting
6. XML::Twig (any) for bug import/export
7. MIME::Parser (5.406) for bug import/export
8. LWP::UserAgent (any) for Automatic Update Notifications
9. PatchReader (0.9.4) for pretty HTML view of patches
10. Image::Magick (any) for converting BMP image attachments to PNG
11. Net::LDAP (any) for LDAP Authentication
12. SOAP::Lite (any) for the web service interface
13. HTML::Parser (3.40) for More HTML in Product/Group Descriptions
14. HTML::Scrubber (any) for More HTML in Product/Group Descriptions
15. Email::MIME::Attachment::Stripper (any) for Inbound Email
16. Email::Reply (any) for Inbound Email
17. mod_perl2 (1.999022) for mod_perl
18. CGI (2.93) for mod_perl

2.1.5.1. DBD::mysql

The installation process will ask you a few questions about the desired compilation target and your MySQL installation. For most of the questions the provided default will be adequate, but when asked if your desired target is the MySQL or mSQL packages, you should select the MySQL-related ones. Later you will be asked if you wish to provide backwards compatibility with the older MySQL packages; you should answer YES to this question. The default is NO.

A host of 'localhost' should be fine. A testing user of 'test', with a null password, should have sufficient access to run tests on the 'test' database which MySQL creates upon installation.

2.1.5.2. Template Toolkit (2.12)

When you install Template Toolkit, you'll get asked various questions about features to enable. The defaults are fine, except that it is recommended you use the high speed XS Stash of the Template Toolkit, in order to achieve best performance.

2.1.5.3. GD (1.20)

The GD module is only required if you want graphical reports.

> **Note:** The Perl GD module requires some other libraries that may or may not be installed on your system, including `libpng` and `libgd`. The full requirements are listed in the Perl GD module README. If compiling GD fails, it's probably because you're missing a required library.

> **Tip:** The version of the GD module you need is very closely tied to the `libgd` version installed on your system. If you have a version 1.x of `libgd` the 2.x versions of the GD module won't work for you.

2.1.5.4. Chart::Base (1.0)

The Chart::Base module is only required if you want graphical reports. Note that earlier versions that 0.99c used GIFs, which are no longer supported by the latest versions of GD.

2.1.5.5. GD::Graph (any)

The GD::Graph module is only required if you want graphical reports.

2.1.5.6. GD::Text (any)

The GD::Text module is only required if you want graphical reports.

2.1.5.7. XML::Twig (any)

The XML::Twig module is only required if you want to import XML bugs using the `importxml.pl` script. This is required to use Bugzilla's "move bugs" feature; you may also want to use it for migrating from another bug database.

2.1.5.8. SOAP::Lite (any)

Installing SOAP::Lite enables your Bugzilla installation to be accessible at a standardized Web Service interface (SOAP/XML-RPC) by third-party applications via HTTP(S).

2.1.5.9. PatchReader (0.9.4)

The PatchReader module is only required if you want to use Patch Viewer, a Bugzilla feature to show code patches in your web browser in a more readable form.

2.1.6. Mail Transfer Agent (MTA)

Bugzilla is dependent on the availability of an e-mail system for its user authentication and for other tasks.

> **Note:** This is not entirely true. It is possible to completely disable email sending, or to have Bugzilla store email messages in a file instead of sending them. However, this is mainly intended for testing, as disabling or diverting email on a production machine would mean that users could miss important events (such as bug changes or the creation of new accounts).
>
> For more information, see the “mail_delivery_method” parameter in Section 3.1.

On Linux, any Sendmail-compatible MTA (Mail Transfer Agent) will suffice. Sendmail, Postfix, qmail and Exim are examples of common MTAs. Sendmail is the original Unix MTA, but the others are easier to configure, and therefore many people replace Sendmail with Postfix or Exim. They are drop-in replacements, so Bugzilla will not distinguish between them.

If you are using Sendmail, version 8.7 or higher is required. If you are using a Sendmail-compatible MTA, it must be congruent with at least version 8.7 of Sendmail.

Consult the manual for the specific MTA you choose for detailed installation instructions. Each of these programs will have their own configuration files where you must configure certain parameters to ensure that the mail is delivered properly. They are implemented as services, and you should ensure that the MTA is in the auto-start list of services for the machine.

If a simple mail sent with the command-line 'mail' program succeeds, then Bugzilla should also be fine.

2.1.7. Installing Bugzilla on mod_perl

It is now possible to run the Bugzilla software under `mod_perl` on Apache. `mod_perl` has some additional requirements to that of running Bugzilla under `mod_cgi` (the standard and previous way).

Bugzilla requires `mod_perl` to be installed, which can be obtained from http://perl.apache.org - Bugzilla requires version 1.999022 (AKA 2.0.0-RC5) to be installed.

Bugzilla also requires a more up-to-date version of the CGI perl module to be installed, version 3.11 as opposed to 2.93

2.2. Configuration

Warning

Poorly-configured MySQL and Bugzilla installations have given attackers full access to systems in the past. Please take the security parts of these guidelines seriously, even for Bugzilla machines hidden away behind your firewall. Be certain to read Chapter 4 for some important security tips.

2.2.1. localconfig

You should now run `checksetup.pl` again, this time without the `--check-modules` switch.

```
bash# ./checksetup.pl
```

This time, `checksetup.pl` should tell you that all the correct modules are installed and will display a message about, and write out a file called, `localconfig`. This file contains the default settings for a number of Bugzilla parameters.

Load this file in your editor. The only value you *need* to change is $db_pass, the password for the user you will create for your database. Pick a strong password (for simplicity, it should not contain single quote characters) and put it here.

You may need to change the value of *webservergroup* if your web server does not run in the "apache" group. On Debian, for example, Apache runs in the "www-data" group. If you are going to run Bugzilla on a machine where you do not have root access (such as on a shared web hosting account), you will need to leave *webservergroup* empty, ignoring the warnings that `checksetup.pl` will subsequently display every time it is run.

Caution

If you are using suexec, you should use your own primary group for *webservergroup* rather than leaving it empty, and see the additional directions in the suexec section Section 2.6.6.1.

The other options in the `localconfig` file are documented by their accompanying comments. If you have a slightly non-standard MySQL setup, you may wish to change one or more of the other "$db_*" parameters.

You may also wish to change the names of the priorities, severities, operating systems and platforms for your installation. However, you can always change these after installation has finished; if you then re-run `checksetup.pl`, the changes will get picked up.

2.2.2. Database Server

This section deals with configuring your database server for use with Bugzilla. Currently, MySQL (Section 2.2.2.2) and PostgreSQL (Section 2.2.2.3) are available.

2.2.2.1. Bugzilla Database Schema

The Bugzilla database schema is available at Ravenbrook (http://www.ravenbrook.com/project/p4dti/tool/cgi/bugzilla-schema/). This very valuable tool can generate a written description of the Bugzilla database schema for any version of Bugzilla. It can also generate a diff between two versions to help someone see what has changed.

2.2.2.2. MySQL

Caution

MySQL's default configuration is very insecure. Section 4.2 has some good information for improving your installation's security.

2.2.2.2.1. Allow large attachments

By default, MySQL will only accept packets up to 64Kb in size. If you want to have attachments larger than this, you will need to modify your `/etc/my.cnf` as below.

```
[mysqld]
# Allow packets up to 1M
max_allowed_packet=1M
```

There is also a parameter in Bugzilla called 'maxattachmentsize' (default = 1000 Kb) that controls the maximum allowable attachment size. Attachments larger than *either* the 'max_allowed_packet' or 'maxattachmentsize' value will not be accepted by Bugzilla.

> **Note:** This does not affect Big Files, attachments that are stored directly on disk instead of in the database. Their maximum size is controlled using the 'maxlocalattachment' parameter.

2.2.2.2.2. Allow small words in full-text indexes

By default, words must be at least four characters in length in order to be indexed by MySQL's full-text indexes. This causes a lot of Bugzilla specific words to be missed, including "cc", "ftp" and "uri".

MySQL can be configured to index those words by setting the ft_min_word_len param to the minimum size of the words to index. This can be done by modifying the `/etc/my.cnf` according to the example below:

```
[mysqld]
# Allow small words in full-text indexes
ft_min_word_len=2
```

Rebuilding the indexes can be done based on documentation found at http://www.mysql.com/doc/en/Fulltext_Fine-tuning.html.

2.2.2.2.3. Add a user to MySQL

You need to add a new MySQL user for Bugzilla to use. (It's not safe to have Bugzilla use the MySQL root account.) The following instructions assume the defaults in `localconfig`; if you changed those, you need to modify the SQL command appropriately. You will need the *`$db_pass`* password you set in `localconfig` in Section 2.2.1.

We use an SQL **GRANT** command to create a "bugs" user. This also restricts the "bugs"user to operations within a database called "bugs", and only allows the account to connect from "localhost". Modify it to reflect your setup if you will be connecting from another machine or as a different user.

Run the `mysql` command-line client and enter:

```
mysql> GRANT SELECT, INSERT,
         UPDATE, DELETE, INDEX, ALTER, CREATE, LOCK TABLES,
         CREATE TEMPORARY TABLES, DROP, REFERENCES ON bugs.*
         TO bugs@localhost IDENTIFIED BY '$db_pass';
         mysql> FLUSH PRIVILEGES;
```

2.2.2.2.4. Permit attachments table to grow beyond 4GB

By default, MySQL will limit the size of a table to 4GB. This limit is present even if the underlying filesystem has no such limit. To set a higher limit, follow these instructions.

After you have completed the rest of the installation (or at least the database setup parts), you should run the `MySQL` command-line client and enter the following, replacing `$bugs_db` with your Bugzilla database name (*bugs* by default):

```
mysql> use $bugs_db
mysql> ALTER TABLE attachments
AVG_ROW_LENGTH=1000000, MAX_ROWS=20000;
```

The above command will change the limit to 20GB. Mysql will have to make a temporary copy of your entire table to do this. Ideally, you should do this when your attachments table is still small.

> **Note:** This does not affect Big Files, attachments that are stored directly on disk instead of in the database.

2.2.2.3. PostgreSQL

2.2.2.3.1. Add a User to PostgreSQL

You need to add a new user to PostgreSQL for the Bugzilla application to use when accessing the database. The following instructions assume the defaults in `localconfig`; if you changed those, you need to modify the commands appropriately. You will need the *`$db_pass`* password you set in `localconfig` in Section 2.2.1.

On most systems, to create the user in PostgreSQL, you will need to login as the root user, and then

```
bash# su - postgres
```

As the postgres user, you then need to create a new user:

```
bash$ createuser -U postgres -dAP bugs
```

When asked for a password, provide the password which will be set as *$db_pass* in `localconfig`. The created user will have the ability to create databases and will not be able to create new users.

2.2.2.3.2. Configure PostgreSQL

Now, you will need to edit `pg_hba.conf` which is usually located in `/var/lib/pgsql/data/`. In this file, you will need to add a new line to it as follows:

```
host all bugs 127.0.0.1 255.255.255.255 md5
```

This means that for TCP/IP (host) connections, allow connections from '127.0.0.1' to 'all' databases on this server from the 'bugs' user, and use password authentication (md5) for that user.

Now, you will need to restart PostgreSQL, but you will need to fully stop and start the server rather than just restarting due to the possibility of a change to `postgresql.conf`. After the server has restarted, you will need to edit `localconfig`, finding the `$db_driver` variable and setting it to `Pg` and changing the password in `$db_pass` to the one you picked previously, while setting up the account.

2.2.3. checksetup.pl

Next, rerun `checksetup.pl`. It reconfirms that all the modules are present, and notices the altered localconfig file, which it assumes you have edited to your satisfaction. It compiles the UI templates, connects to the database using the 'bugs' user you created and the password you defined, and creates the 'bugs' database and the tables therein.

After that, it asks for details of an administrator account. Bugzilla can have multiple administrators - you can create more later - but it needs one to start off with. Enter the email address of an administrator, his or her full name, and a suitable Bugzilla password.

`checksetup.pl` will then finish. You may rerun `checksetup.pl` at any time if you wish.

2.2.4. Web server

Configure your web server according to the instructions in the appropriate section. (If it makes a difference in your choice, the Bugzilla Team recommends Apache.) To check whether your web server is correctly configured, try to access `testagent.cgi` from your web server. If "OK" is displayed, then your configuration is successful. Regardless of which web server you are using, however, ensure that sensitive information is not remotely available by properly applying the access controls in Section 4.3.1. You can run `testserver.pl` to check if your web server serves Bugzilla files as expected.

2.2.4.1. Bugzilla using Apache

You have two options for running Bugzilla under Apache - mod_cgi (the default) and mod_perl (new in Bugzilla 2.23)

2.2.4.1.1. Apache httpd with mod_cgi

To configure your Apache web server to work with Bugzilla while using mod_cgi, do the following:

1. Load `httpd.conf` in your editor. In Fedora and Red Hat Linux, this file is found in `/etc/httpd/conf`.
2. Apache uses `<Directory>` directives to permit fine-grained permission setting. Add the following lines to a directive that applies to the location of your Bugzilla installation. (If such a section does not exist, you'll want to add one.) In this example, Bugzilla has been installed at `/var/www/html/bugzilla`.

   ```
   <Directory /var/www/html/bugzilla>
   AddHandler cgi-script .cgi
   Options +Indexes +ExecCGI
   DirectoryIndex index.cgi
   AllowOverride Limit
   </Directory>
   ```

   These instructions: allow apache to run .cgi files found within the bugzilla directory; instructs the server to look for a file called `index.cgi` if someone only types the directory name into the browser; and allows Bugzilla's `.htaccess` files to override global permissions.

   > **Note:** It is possible to make these changes globally, or to the directive controlling Bugzilla's parent directory (e.g. `<Directory /var/www/html/>`). Such changes would also apply to the Bugzilla directory... but they would also apply to many other places where they may or may not be appropriate. In most cases, including this one, it is better to be as restrictive as possible when granting extra access.

3. `checksetup.pl` can set tighter permissions on Bugzilla's files and directories if it knows what group the web-server runs as. Find the `Group` line in `httpd.conf`, place the value found there in the *`$webservergroup`* variable in `localconfig`, then rerun `checksetup.pl`.
4. Optional: If Bugzilla does not actually reside in the webspace directory, but instead has been symbolically linked there, you will need to add the following to the `Options` line of the Bugzilla `<Directory>` directive (the same one as in the step above):

   ```
   +FollowSymLinks
   ```

   Without this directive, Apache will not follow symbolic links to places outside its own directory structure, and you will be unable to run Bugzilla.

2.2.4.1.2. Apache httpd with mod_perl

Some configuration is required to make Bugzilla work with Apache and mod_perl

1. Load `httpd.conf` in your editor. In Fedora and Red Hat Linux, this file is found in `/etc/httpd/conf`.

2. Add the following information to your httpd.conf file, substituting where appropriate with your own local paths.

   Note: This should be used instead of the <Directory> block shown above. This should also be above any other `mod_perl` directives within the `httpd.conf` and must be specified in the order as below.

   Warning

   You should also ensure that you have disabled `KeepAlive` support in your Apache install when utilizing Bugzilla under mod_perl

```
PerlSwitches -I/var/www/html/bugzilla -w -T
PerlConfigRequire /var/www/html/bugzilla/mod_perl.pl
```

3. `checksetup.pl` can set tighter permissions on Bugzilla's files and directories if it knows what group the web-server runs as. Find the `Group` line in `httpd.conf`, place the value found there in the *`$webservergroup`* variable in `localconfig`, then rerun `checksetup.pl`.

On restarting Apache, Bugzilla should now be running within the mod_perl environment. Please ensure you have run checksetup.pl to set permissions before you restart Apache.

Note: Please bear the following points in mind when looking at using Bugzilla under mod_perl:

- mod_perl support in Bugzilla can take up a HUGE amount of RAM. You could be looking at 30MB per httpd child, easily. Basically, you just need a lot of RAM. The more RAM you can get, the better. mod_perl is basically trading RAM for speed. At least 2GB total system RAM is recommended for running Bugzilla under mod_perl.
- Under mod_perl, you have to restart Apache if you make any manual change to any Bugzilla file. You can't just reload--you have to actually *restart* the server (as in make sure it stops and starts again). You *can* change localconfig and the params file manually, if you want, because those are re-read every time you load a page.
- You must run in Apache's Prefork MPM (this is the default). The Worker MPM may not work--we haven't tested Bugzilla's mod_perl support under threads. (And, in fact, we're fairly sure it *won't* work.)
- Bugzilla generally expects to be the only mod_perl application running on your entire server. It may or may not work if there are other applications also running under mod_perl. It does try its best to play nice with other mod_perl applications, but it still may have conflicts.
- It is recommended that you have one Bugzilla instance running under mod_perl on your server. Bugzilla has not been tested with more than one instance running.

2.2.4.2. Microsoft Internet Information Services

If you are running Bugzilla on Windows and choose to use Microsoft's Internet Information Services or Personal Web Server you will need to perform a number of other configuration steps as explained below. You may also want to refer

to the following Microsoft Knowledge Base articles: 245225 (http://support.microsoft.com/default.aspx?scid=kb;en-us;245225) “HOW TO: Configure and Test a PERL Script with IIS 4.0, 5.0, and 5.1” (for Internet Information Services) and 231998 (http://support.microsoft.com/default.aspx?scid=kb;en-us;231998) “HOW TO: FP2000: How to Use Perl with Microsoft Personal Web Server on Windows 95/98” (for Personal Web Server).

You will need to create a virtual directory for the Bugzilla install. Put the Bugzilla files in a directory that is named something *other* than what you want your end-users accessing. That is, if you want your users to access your Bugzilla installation through “http://<yourdomainname>/Bugzilla”, then do *not* put your Bugzilla files in a directory named “Bugzilla”. Instead, place them in a different location, and then use the IIS Administration tool to create a Virtual Directory named "Bugzilla" that acts as an alias for the actual location of the files. When creating that virtual directory, make sure you add the “Execute (such as ISAPI applications or CGI)” access permission.

You will also need to tell IIS how to handle Bugzilla’s .cgi files. Using the IIS Administration tool again, open up the properties for the new virtual directory and select the Configuration option to access the Script Mappings. Create an entry mapping .cgi to:

```
<full path to perl.exe >\perl.exe -x<full path to Bugzilla> -wT "%s" %s
```

For example:

```
c:\perl\bin\perl.exe -xc:\bugzilla -wT "%s" %s
```

> **Note:** The ActiveState install may have already created an entry for .pl files that is limited to “GET,HEAD,POST”. If so, this mapping should be *removed* as Bugzilla’s .pl files are not designed to be run via a webserver.

IIS will also need to know that the index.cgi should be treated as a default document. On the Documents tab page of the virtual directory properties, you need to add index.cgi as a default document type. If you wish, you may remove the other default document types for this particular virtual directory, since Bugzilla doesn’t use any of them.

Also, and this can’t be stressed enough, make sure that files such as `localconfig` and your `data` directory are secured as described in Section 4.3.1.

2.2.5. Bugzilla

Your Bugzilla should now be working. Access `http://<your-bugzilla-server>/` - you should see the Bugzilla front page. If not, consult the Troubleshooting section, Appendix B.

> **Note:** The URL above may be incorrect if you installed Bugzilla into a subdirectory or used a symbolic link from your web site root to the Bugzilla directory.

Log in with the administrator account you defined in the last `checksetup.pl` run. You should go through the parameters on the Edit Parameters page (see link in the footer) and see if there are any you wish to change. They key parameters are documented in Section 3.1; you should certainly alter **maintainer** and **urlbase**; you may also want to alter **cookiepath** or **requirelogin**.

This would also be a good time to revisit the `localconfig` file and make sure that the names of the priorities, severities, platforms and operating systems are those you wish to use when you start creating bugs. Remember to rerun `checksetup.pl` if you change it.

Bugzilla has several optional features which require extra configuration. You can read about those in Section 2.3.

2.3. Optional Additional Configuration

Bugzilla has a number of optional features. This section describes how to configure or enable them.

2.3.1. Bug Graphs

If you have installed the necessary Perl modules you can start collecting statistics for the nifty Bugzilla graphs.

```
bash# crontab -e
```

This should bring up the crontab file in your editor. Add a cron entry like this to run `collectstats.pl` daily at 5 after midnight:

```
5 0 * * * cd <your-bugzilla-directory> ; ./collectstats.pl
```

After two days have passed you'll be able to view bug graphs from the Reports page.

When upgrading Bugzilla, this format may change. To create new status data, (re)move old data and run the following commands:

```
bash$
cd <your-bugzilla-directory>
bash$
./collectstats.pl --regenerate
```

Note: Windows does not have 'cron', but it does have the Task Scheduler, which performs the same duties. There are also third-party tools that can be used to implement cron, such as nncron (http://www.nncron.ru/).

2.3.2. Dependency Charts

As well as the text-based dependency trees, Bugzilla also supports a graphical view of dependency relationships, using a package called 'dot'. Exactly how this works is controlled by the 'webdotbase' parameter, which can have one of three values:

1. A complete file path to the command 'dot' (part of GraphViz (http://www.graphviz.org/)) will generate the graphs locally
2. A URL prefix pointing to an installation of the webdot package will generate the graphs remotely

3. A blank value will disable dependency graphing.

The easiest way to get this working is to install GraphViz (http://www.graphviz.org/). If you do that, you need to enable server-side image maps (http://httpd.apache.org/docs/mod/mod_imap.html) in Apache. Alternatively, you could set up a webdot server, or use the AT&T public webdot server. This is the default for the webdotbase param, but it's often overloaded and slow. Note that AT&T's server won't work if Bugzilla is only accessible using HARTS. *Editor's note: What the heck is HARTS? Google doesn't know...*

2.3.3. The Whining Cron

What good are bugs if they're not annoying? To help make them more so you can set up Bugzilla's automatic whining system to complain at engineers which leave their bugs in the NEW or REOPENED state without triaging them.

This can be done by adding the following command as a daily crontab entry, in the same manner as explained above for bug graphs. This example runs it at 12.55am.

```
55 0 * * * cd <your-bugzilla-directory> ; ./whineatnews.pl
```

> **Note:** Windows does not have 'cron', but it does have the Task Scheduler, which performs the same duties. There are also third-party tools that can be used to implement cron, such as nncron (http://www.nncron.ru/).

2.3.4. Whining

As of Bugzilla 2.20, users can configure Bugzilla to regularly annoy them at regular intervals, by having Bugzilla execute saved searches at certain times and emailing the results to the user. This is known as "Whining". The process of configuring Whining is described in Section 5.13, but for it to work a Perl script must be executed at regular intervals.

This can be done by adding the following command as a daily crontab entry, in the same manner as explained above for bug graphs. This example runs it every 15 minutes.

```
*/15 * * * * cd <your-bugzilla-directory> ; ./whine.pl
```

> **Note:** Whines can be executed as often as every 15 minutes, so if you specify longer intervals between executions of whine.pl, some users may not be whined at as often as they would expect. Depending on the person, this can either be a very Good Thing or a very Bad Thing.

> **Note:** Windows does not have 'cron', but it does have the Task Scheduler, which performs the same duties. There are also third-party tools that can be used to implement cron, such as nncron (http://www.nncron.ru/).

2.3.5. Patch Viewer

Patch Viewer is the engine behind Bugzilla's graphical display of code patches. You can integrate this with copies of the `cvs`, `lxr` and `bonsai` tools if you have them, by giving the locations of your installation of these tools in `editparams.cgi`.

Patch Viewer also optionally will use the `cvs`, `diff` and `interdiff` command-line utilities if they exist on the system. Interdiff can be obtained from http://cyberelk.net/tim/patchutils/. If these programs are not in the system path, you can configure their locations in `localconfig`.

2.3.6. LDAP Authentication

LDAP authentication is a module for Bugzilla's plugin authentication architecture.

The existing authentication scheme for Bugzilla uses email addresses as the primary user ID, and a password to authenticate that user. All places within Bugzilla where you need to deal with user ID (e.g assigning a bug) use the email address. The LDAP authentication builds on top of this scheme, rather than replacing it. The initial log in is done with a username and password for the LDAP directory. Bugzilla tries to bind to LDAP using those credentials, and if successful, try to map this account to a Bugzilla account. If a LDAP mail attribute is defined, the value of this attribute is used, otherwise emailsuffix parameter is appended to LDAP username to form a full email address. If an account for this address already exists in your Bugzilla system, it will log in to that account. If no account for that email address exists, one is created at the time of login. (In this case, Bugzilla will attempt to use the "displayName" or "cn" attribute to determine the user's full name.) After authentication, all other user-related tasks are still handled by email address, not LDAP username. You still assign bugs by email address, query on users by email address, etc.

Caution

Because the Bugzilla account is not created until the first time a user logs in, a user who has not yet logged is unknown to Bugzilla. This means they cannot be used as an assignee or QA contact (default or otherwise), added to any cc list, or any other such operation. One possible workaround is the `bugzilla_ldapsync.rb` script in the *`contrib`* directory. Another possible solution is fixing bug 201069 (http://bugzilla.mozilla.org/show_bug.cgi?id=201069).

Parameters required to use LDAP Authentication:

user_verify_class

This parameter should be set to "LDAP" *only* if you will be using an LDAP directory for authentication. If you set this param to "LDAP" but fail to set up the other parameters listed below you will not be able to log back in to Bugzilla one you log out. If this happens to you, you will need to manually edit `data/params` and set user_verify_class to "DB".

LDAPserver

This parameter should be set to the name (and optionally the port) of your LDAP server. If no port is specified, it assumes the default LDAP port of 389.

Ex. "ldap.company.com" or "ldap.company.com:3268"

You can also specify a LDAP URI, so as to use other protocols, such as LDAPS or LDAPI. If port was not specified in the URI, the default is either 389 or 636 for 'LDAP' and 'LDAPS' schemes respectively.

Ex. “ldap://ldap.company.com”, “ldaps://ldap.company.com” or “ldapi://%2fvar%2flib%2fldap_sock”

LDAPbinddn [Optional]

Some LDAP servers will not allow an anonymous bind to search the directory. If this is the case with your configuration you should set the LDAPbinddn parameter to the user account Bugzilla should use instead of the anonymous bind.

Ex. “cn=default,cn=user:password”

LDAPBaseDN

The LDAPBaseDN parameter should be set to the location in your LDAP tree that you would like to search for email addresses. Your uids should be unique under the DN specified here.

Ex. “ou=People,o=Company”

LDAPuidattribute

The LDAPuidattribute parameter should be set to the attribute which contains the unique UID of your users. The value retrieved from this attribute will be used when attempting to bind as the user to confirm their password.

Ex. “uid”

LDAPmailattribute

The LDAPmailattribute parameter should be the name of the attribute which contains the email address your users will enter into the Bugzilla login boxes.

Ex. “mail”

2.3.7. Serving Alternate Formats with the right MIME type

Some Bugzilla pages have alternate formats, other than just plain HTML. In particular, a few Bugzilla pages can output their contents as either XUL (a special Mozilla format, that looks like a program GUI) or RDF (a type of structured XML that can be read by various programs).

In order for your users to see these pages correctly, Apache must send them with the right MIME type. To do this, add the following lines to your Apache configuration, either in the `<VirtualHost>` section for your Bugzilla, or in the `<Directory>` section for your Bugzilla:

```
AddType application/vnd.mozilla.xul+xml .xul
AddType application/rdf+xml .rdf
```

2.4. Multiple Bugzilla databases with a single installation

The previous instructions referred to a standard installation, with one unique Bugzilla database. However, you may want to host several distinct installations, without having several copies of the code. This is possible by using the PROJECT environment variable. When accessed, Bugzilla checks for the existence of this variable, and if present, uses its value to check for an alternative configuration file named `localconfig.<PROJECT>` in the same location as the default one (`localconfig`). It also checks for customized templates in a directory named `<PROJECT>` in the same location as the default one (`template/<langcode>`). By default this is `template/en/default` so PROJECT's templates would be located at `template/en/PROJECT`.

To set up an alternate installation, just export PROJECT=foo before running **checksetup.pl** for the first time. It will result in a file called `localconfig.foo` instead of `localconfig`. Edit this file as described above, with reference to a new database, and re-run **checksetup.pl** to populate it. That's all.

Now you have to configure the web server to pass this environment variable when accessed via an alternate URL, such as virtual host for instance. The following is an example of how you could do it in Apache, other Webservers may differ.

```
<VirtualHost 212.85.153.228:80>
    ServerName foo.bar.baz
    SetEnv PROJECT foo
    Alias /bugzilla /var/www/bugzilla
</VirtualHost>
```

Don't forget to also export this variable before accessing Bugzilla by other means, such as cron tasks for instance.

2.5. OS-Specific Installation Notes

Many aspects of the Bugzilla installation can be affected by the operating system you choose to install it on. Sometimes it can be made easier and others more difficult. This section will attempt to help you understand both the difficulties of running on specific operating systems and the utilities available to make it easier.

If you have anything to add or notes for an operating system not covered, please file a bug in Bugzilla Documentation (https://bugzilla.mozilla.org/enter_bug.cgi?product=Bugzilla&component=Documentation).

2.5.1. Microsoft Windows

Making Bugzilla work on Windows is more difficult than making it work on Unix. For that reason, we still recommend doing so on a Unix based system such as GNU/Linux. That said, if you do want to get Bugzilla running on Windows, you will need to make the following adjustments. A detailed step-by-step installation guide for Windows (http://www.bugzilla.org/docs/win32install.html) is also available if you need more help with your installation.

2.5.1.1. Win32 Perl

Perl for Windows can be obtained from ActiveState (http://www.activestate.com/). You should be able to find a compiled binary at http://aspn.activestate.com/ASPN/Downloads/ActivePerl/. The following instructions assume that you are using version 5.8.1 of ActiveState.

Note: These instructions are for 32-bit versions of Windows. If you are using a 64-bit version of Windows, you will need to install 32-bit Perl in order to install the 32-bit modules as described below.

2.5.1.2. Perl Modules on Win32

Bugzilla on Windows requires the same perl modules found in Section 2.1.5. The main difference is that windows uses *PPM* instead of CPAN.

```
C:\perl> ppm install <module name>
```

The best source for the Windows PPM modules needed for Bugzilla is probably the Bugzilla Test Server (aka 'Landfill'), so you should add the Landfill package repository as follows:

```
ppm repo add landfill http://www.landfill.bugzilla.org/ppm/
```

Note: In versions prior to 5.8.8 build 819 of PPM the command is

```
ppm repository add landfill http://www.landfill.bugzilla.org/ppm/
```

Note: The PPM repository stores modules in 'packages' that may have a slightly different name than the module. If retrieving these modules from there, you will need to pay attention to the information provided when you run **checksetup.pl** as it will tell you what package you'll need to install.

Tip: If you are behind a corporate firewall, you will need to let the ActiveState PPM utility know how to get through it to access the repositories by setting the HTTP_proxy system environmental variable. For more information on setting that variable, see the ActiveState documentation.

2.5.1.3. Code changes required to run on Win32

Bugzilla on Win32 is supported out of the box from version 2.20; this means that no code changes are required to get Bugzilla running.

2.5.1.4. Serving the web pages

As is the case on Unix based systems, any web server should be able to handle Bugzilla; however, the Bugzilla Team still recommends Apache whenever asked. No matter what web server you choose, be sure to pay attention to the security notes in Section 4.3.1. More information on configuring specific web servers can be found in Section 2.2.4.

> **Note:** If using Apache on windows, you can set the ScriptInterpreterSource (http://httpd.apache.org/docs-2.0/mod/core.html#scriptinterpretersource) directive in your Apache config to avoid having to modify the first line of every script to contain your path to Perl instead of `/usr/bin/perl`. When setting `ScriptInterpreterSource`, do not forget to specify the **-T** flag to enable the taint mode. For example: **C:\Perl\bin\perl.exe -T**.

2.5.1.5. Sending Email

To enable Bugzilla to send email on Windows, the server running the Bugzilla code must be able to connect to, or act as, an SMTP server.

2.5.2. Mac OS X

Making Bugzilla work on Mac OS X requires the following adjustments.

2.5.2.1. Sendmail

In Mac OS X 10.3 and later, Postfix (http://www.postfix.org/) is used as the built-in email server. Postfix provides an executable that mimics sendmail enough to fool Bugzilla, as long as Bugzilla can find it.

As of version 2.20, Bugzilla will be able to find the fake sendmail executable without any assistance. However, you will have to turn on the sendmailnow parameter before you do anything that would result in email being sent. For more information, see the description of the sendmailnow parameter in Section 3.1.

2.5.2.2. Libraries & Perl Modules on Mac OS X

Apple did not include the GD library with Mac OS X. Bugzilla needs this for bug graphs.

You can install it using a program called Fink, which is similar in nature to the CPAN installer, but installs common GNU utilities. Fink is available from http://sourceforge.net/projects/fink/.

Follow the instructions for setting up Fink. Once it's installed, you'll want to use it to install the `gd2` package.

It will prompt you for a number of dependencies, type 'y' and hit enter to install all of the dependencies and then watch it work. You will then be able to use *CPAN* to install the GD Perl module.

> **Note:** To prevent creating conflicts with the software that Apple installs by default, Fink creates its own directory tree at `/sw` where it installs most of the software that it installs. This means your libraries and headers will be at `/sw/lib` and `/sw/include` instead of `/usr/lib` and `/usr/include`. When the Perl module config script asks where your `libgd` is, be sure to tell it `/sw/lib`.

Also available via Fink is `expat`. After using fink to install the expat package you will be able to install XML::Parser using CPAN. There is one caveat. Unlike recent versions of the GD module, XML::Parser doesn't prompt for the location of the required libraries. When using CPAN, you will need to use the following command sequence:

```
# perl -MCPAN -e'look XML::Parser'        ❶
# perl Makefile.PL EXPATLIBPATH=/sw/lib EXPATINCPATH=/sw/include
# make; make test; make install           ❷
# exit                                    ❸
```

❶❸ The look command will download the module and spawn a new shell with the extracted files as the current working directory. The exit command will return you to your original shell.

❷ You should watch the output from these make commands, especially "make test" as errors may prevent XML::Parser from functioning correctly with Bugzilla.

2.5.3. Linux Distributions

Many Linux distributions include Bugzilla and its dependencies in their native package management systems. Installing Bugzilla with root access on any Linux system should be as simple as finding the Bugzilla package in the package management application and installing it using the normal command syntax. Several distributions also perform the proper web server configuration automatically on installation.

Please consult the documentation of your Linux distribution for instructions on how to install packages, or for specific instructions on installing Bugzilla with native package management tools. There is also a Bugzilla Wiki Page (http://wiki.mozilla.org/Bugzilla:Linux_Distro_Installation) for distro-specific installation notes.

2.6. UNIX (non-root) Installation Notes

2.6.1. Introduction

If you are running a *NIX OS as non-root, either due to lack of access (web hosts, for example) or for security reasons, this will detail how to install Bugzilla on such a setup. It is recommended that you read through the Section 2.1 first to get an idea on the installation steps required. (These notes will reference to steps in that guide.)

2.6.2. MySQL

You may have MySQL installed as root. If you're setting up an account with a web host, a MySQL account needs to be set up for you. From there, you can create the bugs account, or use the account given to you.

Warning

You may have problems trying to set up **GRANT** permissions to the database. If you're using a web host, chances are that you have a separate database which is already locked down (or one big database with limited/no access to the other areas), but you may want to ask your system administrator what the security settings are set to, and/or run the **GRANT** command for you.

Also, you will probably not be able to change the MySQL root user password (for obvious reasons), so skip that step.

2.6.2.1. Running MySQL as Non-Root

2.6.2.1.1. The Custom Configuration Method

Create a file .my.cnf in your home directory (using /home/foo in this example) as follows....

```
[mysqld]
datadir=/home/foo/mymysql
socket=/home/foo/mymysql/thesock
port=8081

[mysql]
socket=/home/foo/mymysql/thesock
port=8081

[mysql.server]
user=mysql
basedir=/var/lib

[safe_mysqld]
err-log=/home/foo/mymysql/the.log
pid-file=/home/foo/mymysql/the.pid
```

2.6.2.1.2. The Custom Built Method

You can install MySQL as a not-root, if you really need to. Build it with PREFIX set to `/home/foo/mysql`, or use pre-installed executables, specifying that you want to put all of the data files in `/home/foo/mysql/data`. If there is another MySQL server running on the system that you do not own, use the -P option to specify a TCP port that is not in use.

2.6.2.1.3. Starting the Server

After your mysqld program is built and any .my.cnf file is in place, you must initialize the databases (ONCE).

```
bash$
mysql_install_db
```

Then start the daemon with

```
bash$
safe_mysql &
```

After you start mysqld the first time, you then connect to it as "root" and **GRANT** permissions to other users. (Again, the MySQL root account has nothing to do with the *NIX root account.)

Note: You will need to start the daemons yourself. You can either ask your system administrator to add them to system startup files, or add a crontab entry that runs a script to check on these daemons and restart them if needed.

Warning

Do NOT run daemons or other services on a server without first consulting your system administrator! Daemons use up system resources and running one may be in violation of your terms of service for any machine on which you are a user!

2.6.3. Perl

On the extremely rare chance that you don't have Perl on the machine, you will have to build the sources yourself. The following commands should get your system installed with your own personal version of Perl:

```
bash$
wget http://perl.com/CPAN/src/stable.tar.gz
bash$
tar zvxf stable.tar.gz
bash$
cd perl-5.8.1 (or whatever the version of Perl is called)
bash$
sh Configure -de -Dprefix=/home/foo/perl
bash$
make && make test && make install
```

Once you have Perl installed into a directory (probably in `~/perl/bin`), you'll have to change the locations on the scripts, which is detailed later on this page.

2.6.4. Perl Modules

Installing the Perl modules as a non-root user is probably the hardest part of the process. There are two different methods: a completely independant Perl with its own modules, or personal modules using the current (root installed) version of Perl. The independant method takes up quite a bit of disk space, but is less complex, while the mixed method only uses as much space as the modules themselves, but takes more work to setup.

2.6.4.1. The Independant Method

The independant method requires that you install your own personal version of Perl, as detailed in the previous section. Once installed, you can start the CPAN shell with the following command:

```
bash$
/home/foo/perl/bin/perl -MCPAN -e 'shell'
```

And then:

```
cpan>
install Bundle::Bugzilla
```

With this method, module installation will usually go a lot smoother, but if you have any hang-ups, you can consult the next section.

2.6.4.2. The Mixed Method

First, you'll need to configure CPAN to install modules in your home directory. The CPAN FAQ says the following on this issue:

```
5)  I am not root, how can I install a module in a personal directory?

    You will most probably like something like this:

      o conf makepl_arg "LIB=~/myperl/lib \
                         INSTALLMAN1DIR=~/myperl/man/man1 \
                         INSTALLMAN3DIR=~/myperl/man/man3"
    install Sybase::Sybperl

    You can make this setting permanent like all "o conf" settings with "o conf commit".

    You will have to add ~/myperl/man to the MANPATH environment variable and also tell your Perl pr
    look into ~/myperl/lib, e.g. by including

      use lib "$ENV{HOME}/myperl/lib";

    or setting the PERL5LIB environment variable.

    Another thing you should bear in mind is that the UNINST parameter should never be set if you ar
```

So, you will need to create a Perl directory in your home directory, as well as the `lib`, `man`, `man/man1`, and `man/man3` directories in that Perl directory. Set the MANPATH variable and PERL5LIB variable, so that the installation of the

modules goes smoother. (Setting UNINST=0 in your "make install" options, on the CPAN first-time configuration, is also a good idea.)

After that, go into the CPAN shell:

```
bash$
perl -MCPAN -e 'shell'
```

From there, you will need to type in the above "o conf" command and commit the changes. Then you can run through the installation:

```
cpan>
install Bundle::Bugzilla
```

Most of the module installation process should go smoothly. However, you may have some problems with Template. When you first start, you will want to try to install Template with the XS Stash options on. If this doesn't work, it may spit out C compiler error messages and croak back to the CPAN shell prompt. So, redo the install, and turn it off. (In fact, say no to all of the Template questions.) It may also start failing on a few of the tests. If the total tests passed is a reasonable figure (90+%), force the install with the following command:

```
cpan>
force install Template
```

You may also want to install the other optional modules:

```
cpan>
install GD
cpan>
install Chart::Base
cpan>
install MIME::Parser
```

2.6.5. HTTP Server

Ideally, this also needs to be installed as root and run under a special webserver account. As long as the web server will allow the running of *.cgi files outside of a cgi-bin, and a way of denying web access to certain files (such as a .htaccess file), you should be good in this department.

2.6.5.1. Running Apache as Non-Root

You can run Apache as a non-root user, but the port will need to be set to one above 1024. If you type **httpd -V**, you will get a list of the variables that your system copy of httpd uses. One of those, namely HTTPD_ROOT, tells you where that installation looks for its config information.

From there, you can copy the config files to your own home directory to start editing. When you edit those and then use the -d option to override the HTTPD_ROOT compiled into the web server, you get control of your own customized web server.

> **Note:** You will need to start the daemons yourself. You can either ask your system administrator to add them to system startup files, or add a crontab entry that runs a script to check on these daemons and restart them if needed.

Warning

Do NOT run daemons or other services on a server without first consulting your system administrator! Daemons use up system resources and running one may be in violation of your terms of service for any machine on which you are a user!

2.6.6. Bugzilla

If you had to install Perl modules as a non-root user (Section 2.6.4) or to non-standard directories, you will need to change the scripts, setting the correct location of the Perl modules:

```
perl -pi -e
        's@use strict\;@use strict\; use lib \"/home/foo/perl/lib\"\;@'
        *cgi *pl Bug.pm processmail syncshadowdb
```

Change `/home/foo/perl/lib` to your personal Perl library directory. You can probably skip this step if you are using the independant method of Perl module installation.

When you run **./checksetup.pl** to create the `localconfig` file, it will list the Perl modules it finds. If one is missing, go back and double-check the module installation from the CPAN shell, then delete the `localconfig` file and try again.

Warning

The one option in `localconfig` you might have problems with is the web server group. If you can't successfully browse to the `index.cgi` (like a Forbidden error), you may have to relax your permissions, and blank out the web server group. Of course, this may pose as a security risk. Having a properly jailed shell and/or limited access to shell accounts may lessen the security risk, but use at your own risk.

2.6.6.1. suexec or shared hosting

If you are running on a system that uses suexec (most shared hosting environments do this), you will need to set the *webservergroup* value in `localconfig` to match *your* primary group, rather than the one the web server runs under. You will need to run the following shell commands after running **./checksetup.pl**, every time you run it (or modify `checksetup.pl` to do them for you via the system() command).

```
for i in docs graphs images js skins; do find $i -type d -exec chmod o+rx {} \; ; done
for i in jpg gif css js png html rdf xul; do find . -name \*.$i -exec chmod o+r {} \; ; done
find . -name .htaccess -exec chmod o+r {} \;
```

Pay particular attention to the number of semicolons and dots. They are all important. A future version of Bugzilla will hopefully be able to do this for you out of the box.

Chapter 3. Administering Bugzilla

3.1. Bugzilla Configuration

Bugzilla is configured by changing various parameters, accessed from the "Edit parameters" link in the page footer. Here are some of the key parameters on that page. You should run down this list and set them appropriately after installing Bugzilla.

maintainer

The maintainer parameter is the email address of the person responsible for maintaining this Bugzilla installation. The address need not be that of a valid Bugzilla account.

urlbase

This parameter defines the fully qualified domain name and web server path to your Bugzilla installation.

For example, if your Bugzilla query page is `http://www.foo.com/bugzilla/query.cgi`, set your “urlbase” to `http://www.foo.com/bugzilla/`.

makeproductgroups

This dictates whether or not to automatically create groups when new products are created.

useentrygroupdefault

Bugzilla products can have a group associated with them, so that certain users can only see bugs in certain products. When this parameter is set to “on”, this causes the initial group controls on newly created products to place all newly-created bugs in the group having the same name as the product immediately. After a product is initially created, the group controls can be further adjusted without interference by this mechanism.

mail_delivery_method

This is used to specify how email is sent, or if it is sent at all. There are several options included for different MTAs, along with two additional options that disable email sending. "testfile" does not send mail, but instead saves it in `data/mailer.testfile` for later review. "none" disables email sending entirely.

shadowdb

You run into an interesting problem when Bugzilla reaches a high level of continuous activity. MySQL supports only table-level write locking. What this means is that if someone needs to make a change to a bug, they will lock the entire table until the operation is complete. Locking for write also blocks reads until the write is complete. Note that more recent versions of mysql support row level locking using different table types. These types are slower than the standard type, and Bugzilla does not yet take advantage of features such as transactions which would justify this speed decrease. The Bugzilla team are, however, happy to hear about any experiences with row level locking and Bugzilla.

The “shadowdb” parameter was designed to get around this limitation. While only a single user is allowed to write to a table at a time, reads can continue unimpeded on a read-only shadow copy of the database. Although

your database size will double, a shadow database can cause an enormous performance improvement when implemented on extremely high-traffic Bugzilla databases.

As a guide, on reasonably old hardware, mozilla.org began needing “shadowdb” when they reached around 40,000 Bugzilla users with several hundred Bugzilla bug changes and comments per day.

The value of the parameter defines the name of the shadow bug database. You will need to set the host and port settings from the params page, and set up replication in your database server so that updates reach this readonly mirror. Consult your database documentation for more detail.

shutdownhtml

If you need to shut down Bugzilla to perform administration, enter some descriptive text (with embedded HTML codes, if you’d like) into this box. Anyone who tries to use Bugzilla (including admins) will receive a page displaying this text. Users can neither log in nor log out while shutdownhtml is enabled.

> **Note:** Although regular log-in capability is disabled while ’shutdownhtml’ is enabled, safeguards are in place to protect the unfortunate admin who loses connection to Bugzilla. Should this happen to you, go directly to the `editparams.cgi` (by typing the URL in manually, if necessary). Doing this will prompt you to log in, and your name/password will be accepted here (but nowhere else).

movebugs

This option is an undocumented feature to allow moving bugs between separate Bugzilla installations. You will need to understand the source code in order to use this feature. Please consult `movebugs.pl` in your Bugzilla source tree for further documentation, such as it is.

useqacontact

This allows you to define an email address for each component, in addition to that of the default assignee, who will be sent carbon copies of incoming bugs.

globalwatcher

This allows to define specific users that will receive notification each time a new bug in entered, or when an existing bug changes, according to the normal groupset permissions. It may be useful for sending notifications to a mailing-list, for instance.

usestatuswhiteboard

This defines whether you wish to have a free-form, overwritable field associated with each bug. The advantage of the Status Whiteboard is that it can be deleted or modified with ease, and provides an easily-searchable field for indexing some bugs that have some trait in common.

whinedays

Set this to the number of days you want to let bugs go in the NEW or REOPENED state before notifying people they have untouched new bugs. If you do not plan to use this feature, simply do not set up the whining cron job described in the installation instructions, or set this value to "0" (never whine).

commenton*

All these fields allow you to dictate what changes can pass without comment, and which must have a comment from the person who changed them. Often, administrators will allow users to add themselves to the CC list, accept bugs, or change the Status Whiteboard without adding a comment as to their reasons for the change, yet require that most other changes come with an explanation.

Set the "commenton" options according to your site policy. It is a wise idea to require comments when users resolve, reassign, or reopen bugs at the very least.

> **Note:** It is generally far better to require a developer comment when resolving bugs than not. Few things are more annoying to bug database users than having a developer mark a bug "fixed" without any comment as to what the fix was (or even that it was truly fixed!)

supportwatchers

Turning on this option allows users to ask to receive copies of bug mail sent to another user. Watching a user with different group permissions is not a way to 'get around' the system; copied emails are still subject to the normal groupset permissions of a bug, and "watchers" will only be copied on emails from bugs they would normally be allowed to view.

noresolveonopenblockers

This option will prevent users from resolving bugs as FIXED if they have unresolved dependencies. Only the FIXED resolution is affected. Users will be still able to resolve bugs to resolutions other than FIXED if they have unresolved dependent bugs.

sendmailnow

When Bugzilla is using Sendmail older than 8.12, turning this option off will improve performance by not waiting for Sendmail to actually send mail. If Sendmail 8.12 or later is being used, there is nothing to gain by turning this off. If another MTA is being used, such as Postfix, then this option *must* be turned on (even if you are using the fake sendmail executable that Postfix provides).

3.2. User Administration

3.2.1. Creating the Default User

When you first run checksetup.pl after installing Bugzilla, it will prompt you for the administrative username (email address) and password for this "super user". If for some reason you delete the "super user" account, re-running checksetup.pl will again prompt you for this username and password.

> **Tip:** If you wish to add more administrative users, add them to the "admin" group and, optionally, edit the tweakparams, editusers, creategroups, editcomponents, and editkeywords groups to add the entire admin group to those groups (which is the case by default).

3.2.2. Managing Other Users

3.2.2.1. Searching for existing users

If you have “editusers” privileges or if you are allowed to grant privileges for some groups, the “Users” link will appear in the Administration page.

The first screen is a search form to search for existing user accounts. You can run searches based either on the user ID, real name or login name (i.e. the email address, or just the first part of the email address if the "emailsuffix" parameter is set). The search can be conducted in different ways using the listbox to the right of the text entry box. You can match by case-insensitive substring (the default), regular expression, a *reverse* regular expression match (which finds every user name which does NOT match the regular expression), or the exact string if you know exactly who you are looking for. The search can be restricted to users who are in a specific group. By default, the restriction is turned off.

The search returns a list of users matching your criteria. User properties can be edited by clicking the login name. The Account History of a user can be viewed by clicking the "View" link in the Account History column. The Account History displays changes that have been made to the user account, the time of the change and the user who made the change. For example, the Account History page will display details of when a user was added or removed from a group.

3.2.2.2. Creating new users

3.2.2.2.1. Self-registration

By default, users can create their own user accounts by clicking the “New Account” link at the bottom of each page (assuming they aren’t logged in as someone else already). If you want to disable this self-registration, or if you want to restrict who can create his own user account, you have to edit the “createemailregexp” parameter in the “Configuration” page, see Section 3.1.

3.2.2.2.2. Accounts created by an administrator

Users with “editusers” privileges, such as administrators, can create user accounts for other users:

1. After logging in, click the "Users" link at the footer of the query page, and then click "Add a new user".
2. Fill out the form presented. This page is self-explanatory. When done, click "Submit".

> **Note:** Adding a user this way will *not* send an email informing them of their username and password. While useful for creating dummy accounts (watchers which shuttle mail to another system, for instance, or email addresses which are a mailing list), in general it is preferable to log out and use the “New Account” button to create users, as it will pre-populate all the required fields and also notify the user of her account name and password.

3.2.2.3. Modifying Users

Once you have found your user, you can change the following fields:

- *Login Name*: This is generally the user's full email address. However, if you have are using the "emailsuffix" parameter, this may just be the user's login name. Note that users can now change their login names themselves (to any valid email address).
- *Real Name*: The user's real name. Note that Bugzilla does not require this to create an account.
- *Password*: You can change the user's password here. Users can automatically request a new password, so you shouldn't need to do this often. If you want to disable an account, see Disable Text below.
- *Bugmail Disabled*: Mark this checkbox to disable bugmail and whinemail completely for this account. This checkbox replaces the data/nomail file which existed in older versions of Bugzilla.
- *Disable Text*: If you type anything in this box, including just a space, the user is prevented from logging in, or making any changes to bugs via the web interface. The HTML you type in this box is presented to the user when they attempt to perform these actions, and should explain why the account was disabled.

  Users with disabled accounts will continue to receive mail from Bugzilla; furthermore, they will not be able to log in themselves to change their own preferences and stop it. If you want an account (disabled or active) to stop receiving mail, simply check the "Bugmail Disabled" checkbox above.

  Note: Even users whose accounts have been disabled can still submit bugs via the e-mail gateway, if one exists. The e-mail gateway should *not* be enabled for secure installations of Bugzilla.

  Warning

  Don't disable all the administrator accounts!

- *<groupname>*: If you have created some groups, e.g. "securitysensitive", then checkboxes will appear here to allow you to add users to, or remove them from, these groups.
- *canconfirm*: This field is only used if you have enabled the "unconfirmed" status. If you enable this for a user, that user can then move bugs from "Unconfirmed" to a "Confirmed" status (e.g.: "New" status).
- *creategroups*: This option will allow a user to create and destroy groups in Bugzilla.
- *editbugs*: Unless a user has this bit set, they can only edit those bugs for which they are the assignee or the reporter. Even if this option is unchecked, users can still add comments to bugs.
- *editcomponents*: This flag allows a user to create new products and components, as well as modify and destroy those that have no bugs associated with them. If a product or component has bugs associated with it, those bugs must be moved to a different product or component before Bugzilla will allow them to be destroyed.
- *editkeywords*: If you use Bugzilla's keyword functionality, enabling this feature allows a user to create and destroy keywords. As always, the keywords for existing bugs containing the keyword the user wishes to destroy must be changed before Bugzilla will allow it to die.
- *editusers*: This flag allows a user to do what you're doing right now: edit other users. This will allow those with the right to do so to remove administrator privileges from other users or grant them to themselves. Enable with care.

- *tweakparams*: This flag allows a user to change Bugzilla's Params (using `editparams.cgi`.)
- *<productname>*: This allows an administrator to specify the products in which a user can see bugs. If you turn on the "makeproductgroups" parameter in the Group Security Panel in the Parameters page, then Bugzilla creates one group per product (at the time you create the product), and this group has exactly the same name as the product itself. Note that for products that already exist when the parameter is turned on, the corresponding group will not be created. The user must still have the "editbugs" privilege to edit bugs in these products.

3.2.2.4. Deleting Users

If the "allowuserdeletion" parameter is turned on, see Section 3.1, then you can also delete user accounts. Note that this is most of the time not the best thing to do. If only a warning in a yellow box is displayed, then the deletion is safe. If a warning is also displayed in a red box, then you should NOT try to delete the user account, else you will get referential integrity problems in your database, which can lead to unexpected behavior, such as bugs not appearing in bug lists anymore, or data displaying incorrectly. You have been warned!

3.2.2.5. Impersonating Users

There may be times when an administrator would like to do something as another user. The **sudo** feature may be used to do this.

> **Note:** To use the sudo feature, you must be in the *bz_sudoers* group. By default, all administrators are in this group.

If you have access to this feature, you may start a session by going to the Edit Users page, Searching for a user and clicking on their login. You should see a link below their login name titled "Impersonate this user". Click on the link. This will take you to a page where you will see a description of the feature and instructions for using it. After reading the text, simply enter the login of the user you would like to impersonate, provide a short message explaining why you are doing this, and press the button.

As long as you are using this feature, everything you do will be done as if you were logged in as the user you are impersonating.

> **Warning**
>
> The user you are impersonating will not be told about what you are doing. If you do anything that results in mail being sent, that mail will appear to be from the user you are impersonating. You should be extremely careful while using this feature.

3.3. Classifications

Classifications tend to be used in order to group several related products into one distinct entity.

The classifications layer is disabled by default; it can be turned on or off using the useclassification parameter, in the *Bug Fields* section of the edit parameters screen.

Access to the administration of classifications is controlled using the *editclassifications* system group, which defines a privilege for creating, destroying, and editing classifications.

When activated, classifications will introduce an additional step when filling bugs (dedicated to classification selection), and they will also appear in the advanced search form.

3.4. Products

Products typically represent real-world shipping products. Products can be given Classifications. For example, if a company makes computer games, they could have a classification of "Games", and a separate product for each game. This company might also have a “Common” product for units of technology used in multiple games, and perhaps a few special products that represent items that are not actually shipping products (for example, "Website", or "Administration").

Many of Bugzilla’s settings are configurable on a per-product basis. The number of “votes” available to users is set per-product, as is the number of votes required to move a bug automatically from the UNCONFIRMED status to the NEW status.

When creating or editing products the following options are available:

Product

The name of the product

Description

A brief description of the product

URL describing milestones for this product

If there is reference URL, provide it here

Default milestone

Select the default milestone for this product.

Closed for bug entry

Select this box to prevent new bugs from being entered against this product.

Maximum votes per person

Maximum votes a user is allowed to give for this product

Maximum votes a person can put on a single bug

Maximum votes a user is allowed to give for this product in a single bug

Confirmation threshold

Number of votes needed to automatically remove any bug against this product from the UNCONFIRMED state

Version

Specify which version of the product bugs will be entered against.

Create chart datasets for this product

Select to make chart datasets available for this product.

When editing a product there is also a link to edit Group Access Controls.

To create a new product:

1. Select “products” from the footer.
2. Select the “Add” link in the bottom right.
3. Enter the name of the product and a description. The Description field may contain HTML.
4. When the product is created, Bugzilla will give a message stating that a component must be created before any bugs can be entered against the new product. Follow the link to create a new component. See Components for more information.

3.4.1. Assigning Group Controls to Products

On the “Product Edit” page, there is a link called “Edit Group Access Controls”. The settings on this page control the relationship of the groups to the product being edited.

Groups may be applicable, default, and mandatory for each product. Groups can also control access to bugs for a given product, or be used to make bugs for a product totally read-only unless the group restrictions are met.

If any group has *Entry* selected, then the product will restrict bug entry to only those users who are members of all the groups with *Entry* selected.

If any group has *Canedit* selected, then the product will be read-only for any users who are not members of all of the groups with *Canedit* selected. ONLY users who are members of all the *Canedit* groups will be able to edit. This is an additional restriction that further limits what can be edited by a user.

The following settings let you choose privileges on a *per-product basis*. This is a convenient way to give privileges to some users for some products only, without having to give them global privileges which would affect all products.

Any group having *editcomponents* selected allows users who are in this group to edit all aspects of this product, including components, milestones and versions.

Any group having *canconfirm* selected allows users who are in this group to confirm bugs in this product.

Any group having *editbugs* selected allows users who are in this group to edit all fields of bugs in this product.

The *MemberControl* and *OtherControl* fields indicate which bugs will be placed in this group according to the following definitions.

For each group, it is possible to specify if membership in that group is:

1. Required for bug entry.
2. Not applicable to this product(NA), a possible restriction for a member of the group to place on a bug in this product(Shown), a default restriction for a member of the group to place on a bug in this product(Default), or a mandatory restriction to be placed on bugs in this product(Mandatory).

3. Not applicable by non-members to this product(NA), a possible restriction for a non-member of the group to place on a bug in this product(Shown), a default restriction for a non-member of the group to place on a bug in this product(Default), or a mandatory restriction to be placed on bugs in this product when entered by a non-member(Mandatory).
4. Required in order to make *any* change to bugs in this product *including comments.*

These controls are often described in this order, so a product that requires a user to be a member of group "foo" to enter a bug and then requires that the bug stay restricted to group "foo" at all times and that only members of group "foo" can edit the bug even if they otherwise could see the bug would have its controls summarized by...

```
foo: ENTRY, MANDATORY/MANDATORY, CANEDIT
```

3.5. Components

Components are subsections of a Product. E.g. the computer game you are designing may have a "UI" component, an "API" component, a "Sound System" component, and a "Plugins" component, each overseen by a different programmer. It often makes sense to divide Components in Bugzilla according to the natural divisions of responsibility within your Product or company.

Each component has a default assignee and (if you turned it on in the parameters), a QA Contact. The default assignee should be the primary person who fixes bugs in that component. The QA Contact should be the person who will ensure these bugs are completely fixed. The Assignee, QA Contact, and Reporter will get email when new bugs are created in this Component and when these bugs change. Default Assignee and Default QA Contact fields only dictate the *default assignments*; these can be changed on bug submission, or at any later point in a bug's life.

To create a new Component:

1. Select the “Edit components” link from the “Edit product” page
2. Select the “Add” link in the bottom right.
3. Fill out the “Component” field, a short “Description”, the “Default Assignee”, “Default CC List” and “Default QA Contact” (if enabled). The “Component Description” field may contain a limited subset of HTML tags. The “Default Assignee” field must be a login name already existing in the Bugzilla database.

3.6. Versions

Versions are the revisions of the product, such as "Flinders 3.1", "Flinders 95", and "Flinders 2000". Version is not a multi-select field; the usual practice is to select the earliest version known to have the bug.

To create and edit Versions:

1. From the "Edit product" screen, select "Edit Versions"
2. You will notice that the product already has the default version "undefined". Click the "Add" link in the bottom right.

3. Enter the name of the Version. This field takes text only. Then click the "Add" button.

3.7. Milestones

Milestones are "targets" that you plan to get a bug fixed by. For example, you have a bug that you plan to fix for your 3.0 release, it would be assigned the milestone of 3.0.

> **Note:** Milestone options will only appear for a Product if you turned on the "usetargetmilestone" Param in the "Edit Parameters" screen.

To create new Milestones, set Default Milestones, and set Milestone URL:

1. Select "Edit milestones" from the "Edit product" page.
2. Select "Add" in the bottom right corner. text
3. Enter the name of the Milestone in the "Milestone" field. You can optionally set the "sortkey", which is a positive or negative number (-32768 to 32767) that defines where in the list this particular milestone appears. This is because milestones often do not occur in alphanumeric order For example, "Future" might be after "Release 1.2". Select "Add".
4. From the Edit product screen, you can enter the URL of a page which gives information about your milestones and what they mean.

3.8. Flags

Flags are a way to attach a specific status to a bug or attachment, either “+” or “-”. The meaning of these symbols depends on the text the flag itself, but contextually they could mean pass/fail, accept/reject, approved/denied, or even a simple yes/no. If your site allows requestable flags, then users may set a flag to “?” as a request to another user that they look at the bug/attachment, and set the flag to its correct status.

3.8.1. A Simple Example

A developer might want to ask their manager, “Should we fix this bug before we release version 2.0?” They might want to do this for a *lot* of bugs, so it would be nice to streamline the process...

In Bugzilla, it would work this way:

1. The Bugzilla administrator creates a flag type called “blocking2.0” that shows up on all bugs in your product.

   It shows up on the “Show Bug” screen as the text “blocking2.0” with a drop-down box next to it. The drop-down box contains four values: an empty space, “?”, “-”, and “+”.

2. The developer sets the flag to “?”.
3. The manager sees the `blocking2.0` flag with a “?” value.

4. If the manager thinks the feature should go into the product before version 2.0 can be released, he sets the flag to "+". Otherwise, he sets it to "-".
5. Now, every Bugzilla user who looks at the bug knows whether or not the bug needs to be fixed before release of version 2.0.

3.8.2. About Flags

3.8.2.1. Values

Flags can have three values:

?

A user is requesting that a status be set. (Think of it as 'A question is being asked'.)

-

The status has been set negatively. (The question has been answered "no".)

+

The status has been set positively. (The question has been answered "yes".)

Actually, there's a fourth value a flag can have -- "unset" -- which shows up as a blank space. This just means that nobody has expressed an opinion (or asked someone else to express an opinion) about this bug or attachment.

3.8.3. Using flag requests

If a flag has been defined as 'requestable', and a user has enough privileges to request it (see below), the user can set the flag's status to "?". This status indicates that someone (a.k.a. "the requester") is asking someone else to set the flag to either "+" or "-".

If a flag has been defined as 'specifically requestable', a text box will appear next to the flag into which the requester may enter a Bugzilla username. That named person (a.k.a. "the requestee") will receive an email notifying them of the request, and pointing them to the bug/attachment in question.

If a flag has *not* been defined as 'specifically requestable', then no such text-box will appear. A request to set this flag cannot be made of any specific individual, but must be asked "to the wind". A requester may "ask the wind" on any flag simply by leaving the text-box blank.

3.8.4. Two Types of Flags

Flags can go in two places: on an attachment, or on a bug.

3.8.4.1. Attachment Flags

Attachment flags are used to ask a question about a specific attachment on a bug.

Many Bugzilla installations use this to request that one developer "review" another developer's code before they check it in. They attach the code to a bug report, and then set a flag on that attachment called "review" to `review?boss@domain.com`. boss@domain.com is then notified by email that he has to check out that attachment and approve it or deny it.

For a Bugzilla user, attachment flags show up in three places:

1. On the list of attachments in the "Show Bug" screen, you can see the current state of any flags that have been set to ?, +, or -. You can see who asked about the flag (the requester), and who is being asked (the requestee).
2. When you "Edit" an attachment, you can see any settable flag, along with any flags that have already been set. This "Edit Attachment" screen is where you set flags to ?, -, +, or unset them.
3. Requests are listed in the "Request Queue", which is accessible from the "My Requests" link (if you are logged in) or "Requests" link (if you are logged out) visible in the footer of all pages.

3.8.4.2. Bug Flags

Bug flags are used to set a status on the bug itself. You can see Bug Flags in the "Show Bug" and "Requests" screens, as described above.

Only users with enough privileges (see below) may set flags on bugs. This doesn't necessarily include the assignee, reporter, or users with the `editbugs` permission.

3.8.5. Administering Flags

If you have the "editcomponents" permission, you will have "Edit: ... | Flags | ..." in your page footer. Clicking on that link will bring you to the "Administer Flag Types" page. Here, you can select whether you want to create (or edit) a Bug flag, or an Attachment flag.

No matter which you choose, the interface is the same, so we'll just go over it once.

3.8.5.1. Creating a Flag

When you click on the "Create a Flag Type for..." link, you will be presented with a form. Here is what the fields in the form mean:

3.8.5.1.1. Name

This is the name of the flag. This will be displayed to Bugzilla users who are looking at or setting the flag. The name may contain any valid Unicode characters except commas and spaces.

3.8.5.1.2. Description

The description describes the flag in more detail. It is visible in a tooltip when hovering over a flag either in the "Show Bug" or "Edit Attachment" pages. This field can be as long as you like, and can contain any character you want.

3.8.5.1.3. Category

Default behaviour for a newly-created flag is to appear on products and all components, which is why "__Any__:__Any__" is already entered in the "Inclusions" box. If this is not your desired behaviour, you must either set some exclusions (for products on which you don't want the flag to appear), or you must remove "__Any__:__Any__" from the Inclusions box and define products/components specifically for this flag.

To create an Inclusion, select a Product from the top drop-down box. You may also select a specific component from the bottom drop-down box. (Setting "__Any__" for Product translates to, "all the products in this Bugzilla". Selecting "__Any__" in the Component field means "all components in the selected product.") Selections made, press "Include", and your Product/Component pairing will show up in the "Inclusions" box on the right.

To create an Exclusion, the process is the same; select a Product from the top drop-down box, select a specific component if you want one, and press "Exclude". The Product/Component pairing will show up in the "Exclusions" box on the right.

This flag *will* and *can* be set for any products/components that appearing in the "Inclusions" box (or which fall under the appropriate "__Any__"). This flag *will not* appear (and therefore cannot be set) on any products appearing in the "Exclusions" box. *IMPORTANT: Exclusions override inclusions.*

You may select a Product without selecting a specific Component, but you can't select a Component without a Product, or to select a Component that does not belong to the named Product. If you do so, Bugzilla will display an error message, even if all your products have a component by that name.

Example: Let's say you have a product called "Jet Plane" that has thousands of components. You want to be able to ask if a problem should be fixed in the next model of plane you release. We'll call the flag "fixInNext". But, there's one component in "Jet Plane," called "Pilot." It doesn't make sense to release a new pilot, so you don't want to have the flag show up in that component. So, you include "Jet Plane:__Any__" and you exclude "Jet Plane:Pilot".

3.8.5.1.4. Sort Key

Flags normally show up in alphabetical order. If you want them to show up in a different order, you can use this key set the order on each flag. Flags with a lower sort key will appear before flags with a higher sort key. Flags that have the same sort key will be sorted alphabetically, but they will still be after flags with a lower sort key, and before flags with a higher sort key.

Example: I have AFlag (Sort Key 100), BFlag (Sort Key 10), CFlag (Sort Key 10), and DFlag (Sort Key 1). These show up in the order: DFlag, BFlag, CFlag, AFlag.

3.8.5.1.5. Active

Sometimes, you might want to keep old flag information in the Bugzilla database, but stop users from setting any new flags of this type. To do this, uncheck "active". Deactivated flags will still show up in the UI if they are ?, +, or -, but they may only be cleared (unset), and cannot be changed to a new value. Once a deactivated flag is cleared, it will completely disappear from a bug/attachment, and cannot be set again.

3.8.5.1.6. Requestable

New flags are, by default, “requestable”, meaning that they offer users the “?” option, as well as “+” and “-”. To remove the ? option, uncheck “requestable”.

3.8.5.1.7. Specifically Requestable

By default this box is checked for new flags, meaning that users may make flag requests of specific individuals. Unchecking this box will remove the text box next to a flag; if it is still requestable, then requests may only be made “to the wind.” Removing this after specific requests have been made will not remove those requests; that data will stay in the database (though it will no longer appear to the user).

3.8.5.1.8. Multiplicable

Any flag with “Multiplicable” set (default for new flags is ’on’) may be set more than once. After being set once, an unset flag of the same type will appear below it with “addl.” (short for “additional”) before the name. There is no limit to the number of times a Multiplicable flags may be set on the same bug/attachment.

3.8.5.1.9. CC List

If you want certain users to be notified every time this flag is set to ?, -, +, or unset, add them here. This is a comma-separated list of email addresses that need not be restricted to Bugzilla usernames.

3.8.5.1.10. Grant Group

When this field is set to some given group, only users in the group can set the flag to “+” and “-”. This field does not affect who can request or cancel the flag. For that, see the “Request Group” field below. If this field is left blank, all users can set or delete this flag. This field is useful for restricting which users can approve or reject requests.

3.8.5.1.11. Request Group

When this field is set to some given group, only users in the group can request or cancel this flag. Note that this field has no effect if the “grant group” field is empty. You can set the value of this field to a different group, but both fields have to be set to a group for this field to have an effect.

3.8.5.2. Deleting a Flag

When you are at the “Administer Flag Types” screen, you will be presented with a list of Bug flags and a list of Attachment Flags.

To delete a flag, click on the “Delete” link next to the flag description.

Warning

Once you delete a flag, it is *gone* from your Bugzilla. All the data for that flag will be deleted. Everywhere that flag was set, it will disappear, and you cannot get that data back. If you want to keep flag data, but don't want anybody to set any new flags or change current flags, unset "active" in the flag Edit form.

3.8.5.3. Editing a Flag

To edit a flag's properties, just click on the "Edit" link next to the flag's description. That will take you to the same form described in the "Creating a Flag" section.

3.9. Keywords

The administrator can define keywords which can be used to tag and categorise bugs. For example, the keyword "regression" is commonly used. A company might have a policy stating all regressions must be fixed by the next release - this keyword can make tracking those bugs much easier.

Keywords are global, rather than per-product. If the administrator changes a keyword currently applied to any bugs, the keyword cache must be rebuilt using the Section 3.15 script. Currently keywords can not be marked obsolete to prevent future usage.

Keywords can be created, edited or deleted by clicking the "Keywords" link in the footer. There are two fields for each keyword - the keyword itself and a brief description. Once created, keywords can be selected and applied to individual bugs in that bug's "Details" section.

3.10. Custom Fields

One of the most requested features was the ability to add your own custom fields to bugs, based on your needs. With the release of Bugzilla 3.0, this dream finally comes true. Administrators can manage these fields using the "Custom Fields" link in the footer of pages. The first thing they will see is the list of existing custom fields (which is empty by default).

3.10.1. Adding Custom Fields

The "Add a new custom field" link permits you to add a new field which can be either a free text box or a drop down menu. More field types will be available in future releases.

The following attributes must be set for each new custom field:

- *Name:* the name of the field, used internally. This name MUST begin with "cf_". If you omit this string, it will be automatically added to the name you entered. This way, all custom fields added to Bugzilla begin with "cf_", avoiding any conflict with default fields.

- *Description:* the string which is used as a label for this custom field. That is the string that users will see, and so should be short and explicit.
- *Type:* as mentioned above, only two types are implemented so far. Free text boxes let you type any string, while drop down menus only let you choose one value in the list provided. The list of legal values for this field can be created and edited as soon as this custom field is added to the DB. See Section 3.11.1 for information about editing legal values.
- *Sortkey:* this integer determines in which order custom fields are displayed in the UI, especially when viewing a bug. Fields with lower values are displayed first.
- *Can be set on bug creation:* this boolean determines whether this field can be set on bug creation or not. If not, then you have to create the bug first before being able to set this field. Else you can set its value at the same time you file a bug, see Section 5.6 about filing bugs.
- *Displayed in bugmail for new bugs:* this boolean determines whether the value set on this field should appear in bugmail when the bug is filed. This attribute has no effect if the field cannot be set on bug creation.
- *Is obsolete:* this boolean determines whether or not this field should be displayed at all. Obsolete custom fields are hidden.

3.10.2. Editing Custom Fields

As soon as a custom field is created, its name and type cannot be changed. If this field is a drop down menu, its legal values can be set as described in Section 3.11.1. All other attributes can be edited as described above.

3.10.3. Deleting Custom Fields

At this point, it is not possible to delete custom fields from your web browser. If you don't want to make one available anymore, mark it as obsolete. This way, you will preserve your DB referential integrity.

3.11. Legal Values

Since Bugzilla 2.20 RC1, legal values for Operating Systems, platforms, bug priorities and severities can be edited from the User Interface directly. This means that it is no longer required to manually edit `localconfig`. Starting with Bugzilla 2.23.3, you can also customize the list of valid resolutions from the same interface.

3.11.1. Viewing/Editing legal values

Editing legal values requires "admin" privileges. A link named "Field Values" is visible in your footer and clicking on it displays the list of fields whose values can be edited.

You can add as many legal values as you want, and each value must be unique (on a per field basis). The sortkey is important to display these values in the desired order.

3.11.2. Deleting legal values

You can also delete legal values, but only if the two following conditions are respected:

1. The value is not used by default for the field.
2. No bug is currently using this value.

If any of these conditions is not respected, the value cannot be deleted. The only way to delete these values is to reassign bugs to another value and to set another value as default for the field.

3.12. Voting

Voting allows users to be given a pot of votes which they can allocate to bugs, to indicate that they'd like them fixed. This allows developers to gauge user need for a particular enhancement or bugfix. By allowing bugs with a certain number of votes to automatically move from "UNCONFIRMED" to "NEW", users of the bug system can help high-priority bugs garner attention so they don't sit for a long time awaiting triage.

To modify Voting settings:

1. Navigate to the "Edit product" screen for the Product you wish to modify
2. *Maximum Votes per person*: Setting this field to "0" disables voting.
3. *Maximum Votes a person can put on a single bug*: It should probably be some number lower than the "Maximum votes per person". Don't set this field to "0" if "Maximum votes per person" is non-zero; that doesn't make any sense.
4. *Number of votes a bug in this product needs to automatically get out of the UNCONFIRMED state*: Setting this field to "0" disables the automatic move of bugs from UNCONFIRMED to NEW.
5. Once you have adjusted the values to your preference, click "Update".

3.13. Quips

Quips are small text messages that can be configured to appear next to search results. A Bugzilla installation can have its own specific quips. Whenever a quip needs to be displayed, a random selection is made from the pool of already existing quips.

Quips are controlled by the *enablequips* parameter. It has several possible values: on, approved, frozen or off. In order to enable quips approval you need to set this parameter to "approved". In this way, users are free to submit quips for addition but an administrator must explicitly approve them before they are actually used.

In order to see the user interface for the quips, it is enough to click on a quip when it is displayed together with the search results. Or it can be seen directly in the browser by visiting the quips.cgi URL (prefixed with the usual web location of the Bugzilla installation). Once the quip interface is displayed, it is enough to click the "view and edit the whole quip list" in order to see the administration page. A page with all the quips available in the database will be displayed.

Next to each tip there is a checkbox, under the "Approved" column. Quips who have this checkbox checked are already approved and will appear next to the search results. The ones that have it unchecked are still preserved in the database but they will not appear on search results pages. User submitted quips have initially the checkbox unchecked.

Also, there is a delete link next to each quip, which can be used in order to permanently delete a quip.

3.14. Groups and Group Security

Groups allow the administrator to isolate bugs or products that should only be seen by certain people. The association between products and groups is controlled from the product edit page under “Edit Group Controls.”

If the makeproductgroups param is on, a new group will be automatically created for every new product. It is primarily available for backward compatibility with older sites.

Note that group permissions are such that you need to be a member of *all* the groups a bug is in, for whatever reason, to see that bug. Similarly, you must be a member of *all* of the entry groups for a product to add bugs to a product and you must be a member of *all* of the canedit groups for a product in order to make *any* change to bugs in that product.

> **Note:** By default, bugs can also be seen by the Assignee, the Reporter, and by everyone on the CC List, regardless of whether or not the bug would typically be viewable by them. Visibility to the Reporter and CC List can be overridden (on a per-bug basis) by bringing up the bug, finding the section that starts with “Users in the roles selected below...” and un-checking the box next to either ’Reporter’ or ’CC List’ (or both).

3.14.1. Creating Groups

To create Groups:

1. Select the “groups” link in the footer.
2. Take a moment to understand the instructions on the “Edit Groups” screen, then select the “Add Group” link.
3. Fill out the “Group”, “Description”, and “User RegExp” fields. “User RegExp” allows you to automatically place all users who fulfill the Regular Expression into the new group. When you have finished, click “Add”.

   Users whose email addresses match the regular expression will automatically be members of the group as long as their email addresses continue to match the regular expression.

   > **Note:** This is a change from 2.16 where the regular expression resulted in a user acquiring permanent membership in a group. To remove a user from a group the user was in due to a regular expression in version 2.16 or earlier, the user must be explicitly removed from the group. This can easily be done by pressing buttons named ’Remove Memberships’ or ’Remove Memberships included in regular expression’ under the table.

Warning

If specifying a domain in the regexp, make sure you end the regexp with a $. Otherwise, when granting access to "@mycompany\.com", you will allow access to 'badperson@mycompany.com.cracker.net'. You need to use '@mycompany\.com$' as the regexp.

4. If you plan to use this group to directly control access to bugs, check the "use for bugs" box. Groups not used for bugs are still useful because other groups can include the group as a whole.
5. After you add your new group, edit the new group. On the edit page, you can specify other groups that should be included in this group and which groups should be permitted to add and delete users from this group.

3.14.2. Assigning Users to Groups

Users can become a member of a group in several ways.

1. The user can be explicitly placed in the group by editing the user's own profile
2. The group can include another group of which the user is a member.
3. The user's email address can match a regular expression that the group specifies to automatically grant membership to the group.

3.14.3. Assigning Group Controls to Products

For information on assigning group controls to products, see Products.

3.14.4. Common Applications of Group Controls

3.14.4.1. General User Access With Security Group

To permit any user to file bugs in each product (A, B, C...) and to permit any user to submit those bugs into a security group....

```
Product A...
security: SHOWN/SHOWN
Product B...
security: SHOWN/SHOWN
Product C...
security: SHOWN/SHOWN
```

3.14.4.2. General User Access With A Security Product

To permit any user to file bugs in a Security product while keeping those bugs from becoming visible to anyone outside the securityworkers group unless a member of the securityworkers group removes that restriction....

```
Product Security...
securityworkers: DEFAULT/MANDATORY
```

3.14.4.3. Product Isolation With Common Group

To permit users of product A to access the bugs for product A, users of product B to access product B, and support staff to access both, 3 groups are needed

1. Support: Contains members of the support staff.
2. AccessA: Contains users of product A and the Support group.
3. AccessB: Contains users of product B and the Support group.

Once these 3 groups are defined, the products group controls can be set to..

```
Product A...
AccessA: ENTRY, MANDATORY/MANDATORY
Product B...
AccessB: ENTRY, MANDATORY/MANDATORY
```

Optionally, the support group could be permitted to make bugs inaccessible to the users and could be permitted to publish bugs relevant to all users in a common product that is read-only to anyone outside the support group. That configuration could be...

```
Product A...
AccessA: ENTRY, MANDATORY/MANDATORY
Support: SHOWN/NA
Product B...
AccessB: ENTRY, MANDATORY/MANDATORY
Support: SHOWN/NA
Product Common...
Support: ENTRY, DEFAULT/MANDATORY, CANEDIT
```

3.15. Checking and Maintaining Database Integrity

Over time it is possible for the Bugzilla database to become corrupt or to have anomalies. This could happen through normal usage of Bugzilla, manual database administration outside of the Bugzilla user interface, or from some other

unexpected event. Bugzilla includes a "Sanity Check" script that can perform several basic database checks, and repair certain problems or inconsistencies.

To run the "Sanity Check" script, log in as an Administrator and click the "Sanity Check" link in the footer. Any problems that are found will be displayed in red letters. If the script is capable of fixing a problem, it will present a link to initiate the fix. If the script can not fix the problem it will require manual database administration or recovery.

The "Sanity Check" script should be run on a regular basis as a matter of best practice.

Warning

The "Sanity Check" script is no substitute for a competent database administrator. It is only designed to check and repair basic database problems.

3.16. Upgrading to New Releases

Upgrading Bugzilla is something we all want to do from time to time, be it to get new features or pick up the latest security fix. How easy it is to update depends on a few factors:

- If the new version is a revision or a new point release
- How many local changes (if any) have been made

3.16.1. Version Definitions

Bugzilla displays the version you are using at the top of the home page `index.cgi`. It looks something like '2.20.3', '2.22.1' or '3.0rc1'. The first number in this series is the Major Version. This does not change very often; Bugzilla was 1.x.x when it was first created, and went to 2.x.x when it was re-written in perl in Sept 1998. The major version 3.x.x, released in early 2007, is pretty far from what the 2.x.x series looked like, both about its UI and its code.

The second number in the version is called the 'minor number', and a release that changes the minor number is called a 'point release'. An even number in this position (2.18, 2.20, 2.22, 3.0, 3.2, etc.) represents a stable version, while an odd number (2.19, 2.21, 2.23, etc.) represents a development version. In the past, stable point releases were feature-based, coming when certain enhancements had been completed, or the Bugzilla development team felt that enough progress had been made overall. As of version 2.18, however, Bugzilla has moved to a time-based release schedule; current plans are to create a stable point release every 6 months or so after 2.18 is deployed.

The third number in the Bugzilla version represents a bugfix version. Bugfix Revisions are released only to address security vulnerabilities and, for a limited period, bug fixes. Once enough of these bugfixes have accumulated (or a new security vulnerability is identified and closed), a bugfix release is made. As an example, 2.20.3 was a bugfix release, and improved on 2.20.2.

Note: When reading version numbers, everything separated by a point ('.') should be read as a single number. It is *not* the same as decimal. 2.22 is newer than 2.8 because minor version 22 is greater than minor version 8. The now unsupported release 2.16.11 was newer than 2.16.9 (because bugfix 11 is greater than bugfix 9. This is confusing to some people who aren't used to dealing with software.

3.16.2. Upgrading - Notifications

Bugzilla 3.0 introduces the ability to automatically notify administrators when new releases are available, based on the `upgrade_notification` parameter, see Section 3.1. Administrators will see these notifications when they access the `index.cgi` page, i.e. generally when logging in. Bugzilla will check once per day for new releases, unless the parameter is set to "disabled". If you are behind a proxy, you may have to set the `proxy_url` parameter accordingly. If the proxy requires authentication, use the `http://user:pass@proxy_url/` syntax.

3.16.3. Upgrading - Methods and Procedure

There are three different ways to upgrade your installation.

1. Using CVS (Section 3.16.3.1)
2. Downloading a new tarball (Section 3.16.3.2)
3. Applying the relevant patches (Section 3.16.3.3)

Each of these options has its own pros and cons; the one that's right for you depends on how long it has been since you last installed, the degree to which you have customized your installation, and/or your network configuration. (Some discussion of the various methods of updating compared with degree and methods of local customization can be found in Section 6.2.2.)

The larger the jump you are trying to make, the more difficult it is going to be to upgrade if you have made local customizations. Upgrading from 2.22 to 2.22.1 should be fairly painless even if you are heavily customized, but going from 2.18 to 3.0 is going to mean a fair bit of work re-writing your local changes to use the new files, logic, templates, etc. If you have done no local changes at all, however, then upgrading should be approximately the same amount of work regardless of how long it has been since your version was released.

Warning

Upgrading is a one-way process. You should backup your database and current Bugzilla directory before attempting the upgrade. If you wish to revert to the old Bugzilla version for any reason, you will have to restore from these backups.

The examples in the following sections are written as though the user were updating to version 2.22.1, but the procedures are the same regardless of whether one is updating to a new point release or simply trying to obtain a new bugfix release. Also, in the examples the user's Bugzilla installation is found at `/var/www/html/bugzilla`. If that is not the same as the location of your Bugzilla installation, simply substitute the proper paths where appropriate.

3.16.3.1. Upgrading using CVS

Every release of Bugzilla, whether it is a point release or a bugfix, is tagged in CVS. Also, every tarball that has been distributed since version 2.12 has been created in such a way that it can be used with CVS once it is unpacked. Doing so, however, requires that you are able to access cvs-mirror.mozilla.org on port 2401, which may not be an option or a possibility for some users, especially those behind a highly restrictive firewall.

Tip: If you can, updating using CVS is probably the most painless method, especially if you have a lot of local changes.

The following shows the sequence of commands needed to update a Bugzilla installation via CVS, and a typical series of results.

```
bash$ cd /var/www/html/bugzilla
bash$ cvs login
Logging in to :pserver:anonymous@cvs-mirror.mozilla.org:2401/cvsroot
CVS password: ('anonymous', or just leave it blank)
bash$ cvs -q update -r BUGZILLA-2_22_1 -dP
P checksetup.pl
P collectstats.pl
P docs/rel_notes.txt
P template/en/default/list/quips.html.tmpl
(etc.)
```

Caution

If a line in the output from **cvs update** begins with a `C`, then that represents a file with local changes that CVS was unable to properly merge. You need to resolve these conflicts manually before Bugzilla (or at least the portion using that file) will be usable.

3.16.3.2. Upgrading using the tarball

If you are unable (or unwilling) to use CVS, another option that's always available is to obtain the latest tarball from the Download Page (http://www.bugzilla.org/download/) and create a new Bugzilla installation from that.

This sequence of commands shows how to get the tarball from the command-line; it is also possible to download it from the site directly in a web browser. If you go that route, save the file to the `/var/www/html` directory (or its equivalent, if you use something else) and omit the first three lines of the example.

```
bash$ cd /var/www/html
bash$ wget http://ftp.mozilla.org/pub/mozilla.org/webtools/bugzilla-2.22.1.tar.gz
(Output omitted)
bash$ tar xzvf bugzilla-2.22.1.tar.gz
bugzilla-2.22.1/
bugzilla-2.22.1/.cvsignore
(Output truncated)
bash$ cd bugzilla-2.22.1
bash$ cp ../bugzilla/localconfig* .
bash$ cp -r ../bugzilla/data .
bash$ cd ..
bash$ mv bugzilla bugzilla.old
bash$ mv bugzilla-2.22.1 bugzilla
```

> **Warning**
>
> The **cp** commands both end with periods which is a very important detail, it tells the shell that the destination directory is the current working directory.

This upgrade method will give you a clean install of Bugzilla with the same version as the tarball. That's fine if you don't have any local customizations that you want to maintain, but if you do then you will need to reapply them by hand to the appropriate files.

It's worth noting that since 2.12, the Bugzilla tarballs come CVS-ready, so if you decide at a later date that you'd rather use CVS as an upgrade method, your code will already be set up for it.

3.16.3.3. Upgrading using patches

If you are doing a bugfix upgrade -- that is, one where only the last number of the revision changes, such as from 2.22 to 2.22.1 -- then you have the option of obtaining and applying a patch file from the Download Page (http://www.bugzilla.org/download/). This file is made available by the Bugzilla Development Team (http://www.bugzilla.org/developers/profiles.html), and is a collection of all the bug fixes and security patches that have been made since the last bugfix release. If you are planning to upgrade via patches, it is safer to grab this developer-made patch file than to read the patch notes and apply all (or even just some of) the patches oneself, as sometimes patches on bugs get changed before they get checked in.

As above, this example starts with obtaining the file via the command line. If you have already downloaded it, you can omit the first two commands.

```
bash$ cd /var/www/html/bugzilla
bash$ wget http://ftp.mozilla.org/pub/mozilla.org/webtools/bugzilla-2.22-to-2.22.1.diff.gz
(Output omitted)
bash$ gunzip bugzilla-2.22-to-2.22.1.diff.gz
bash$ patch -p1 < bugzilla-2.22-to-2.22.1.diff
patching file checksetup.pl
patching file collectstats.pl
(etc.)
```

> **Warning**
>
> Be aware that upgrading from a patch file does not change the entries in your CVS directory. This could make it more difficult to upgrade using CVS (Section 3.16.3.1) in the future.

3.16.4. Completing Your Upgrade

Regardless of which upgrade method you choose, you will need to run **./checksetup.pl** before your Bugzilla upgrade will be complete.

```
bash$ cd bugzilla
bash$ ./checksetup.pl
```

Warning

The period at the beginning of the command **./checksetup.pl** is important and can not be omitted.

If you have done a lot of local modifications, it wouldn't hurt to run the Bugzilla Testing suite. This is not a required step, but it isn't going to hurt anything, and might help point out some areas that could be improved. (More information on the test suite can be had by following this link to the appropriate section in the Developers' Guide (http://www.bugzilla.org/docs/developer.html#testsuite).)

Chapter 4. Bugzilla Security

While some of the items in this chapter are related to the operating system Bugzilla is running on or some of the support software required to run Bugzilla, it is all related to protecting your data. This is not intended to be a comprehensive guide to securing Linux, Apache, MySQL, or any other piece of software mentioned. There is no substitute for active administration and monitoring of a machine. The key to good security is actually right in the middle of the word: *U R It*.

While programmers in general always strive to write secure code, accidents can and do happen. The best approach to security is to always assume that the program you are working with isn't 100% secure and restrict its access to other parts of your machine as much as possible.

4.1. Operating System

4.1.1. TCP/IP Ports

The TCP/IP standard defines more than 65,000 ports for sending and receiving traffic. Of those, Bugzilla needs exactly one to operate (different configurations and options may require up to 3). You should audit your server and make sure that you aren't listening on any ports you don't need to be. It's also highly recommended that the server Bugzilla resides on, along with any other machines you administer, be placed behind some kind of firewall.

4.1.2. System User Accounts

Many *daemons*, such as Apache's `httpd` or MySQL's `mysqld`, run as either "root" or "nobody". This is even worse on Windows machines where the majority of *services* run as "SYSTEM". While running as "root" or "SYSTEM" introduces obvious security concerns, the problems introduced by running everything as "nobody" may not be so obvious. Basically, if you run every daemon as "nobody" and one of them gets compromised it can compromise every other daemon running as "nobody" on your machine. For this reason, it is recommended that you create a user account for each daemon.

> **Note:** You will need to set the `webservergroup` option in `localconfig` to the group your webserver runs as. This will allow `./checksetup.pl` to set file permissions on Unix systems so that nothing is world-writable.

4.1.3. The `chroot` Jail

If your system supports it, you may wish to consider running Bugzilla inside of a `chroot` jail. This option provides unprecedented security by restricting anything running inside the jail from accessing any information outside of it. If you wish to use this option, please consult the documentation that came with your system.

4.2. MySQL

4.2.1. The MySQL System Account

As mentioned in Section 4.1.2, the MySQL daemon should run as a non-privileged, unique user. Be sure to consult the MySQL documentation or the documentation that came with your system for instructions.

4.2.2. The MySQL “root” and “anonymous” Users

By default, MySQL comes with a “root” user with a blank password and an “anonymous” user, also with a blank password. In order to protect your data, the “root” user should be given a password and the anonymous user should be disabled.

Example 4-1. Assigning the MySQL “root” User a Password

```
bash$ mysql mysql
mysql> UPDATE user SET password = password('new_password') WHERE user = 'root';
mysql> FLUSH PRIVILEGES;
```

Example 4-2. Disabling the MySQL “anonymous” User

```
bash$ mysql -u root -p mysql          ❶
Enter Password: new_password
mysql> DELETE FROM user WHERE user = '';
mysql> FLUSH PRIVILEGES;
```

❶ This command assumes that you have already completed Example 4-1.

4.2.3. Network Access

If MySQL and your webserver both run on the same machine and you have no other reason to access MySQL remotely, then you should disable the network access. This, along with the suggestion in Section 4.1.1, will help protect your system from any remote vulnerabilities in MySQL.

Example 4-3. Disabling Networking in MySQL

Simply enter the following in `/etc/my.cnf`:

```
[mysqld]
# Prevent network access to MySQL.
skip-networking
```

4.3. Web server

4.3.1. Disabling Remote Access to Bugzilla Configuration Files

There are many files that are placed in the Bugzilla directory area that should not be accessible from the web. Because of the way Bugzilla is currently layed out, the list of what should and should not be accessible is rather complicated. A quick way is to run `testserver.pl` to check if your web server serves Bugzilla files as expected. If not, you may want to follow the few steps below.

> **Tip:** Bugzilla ships with the ability to create *`.htaccess`* files that enforce these rules. Instructions for enabling these directives in Apache can be found in Section 2.2.4.1

- In the main Bugzilla directory, you should:
  - Block: `*.pl`, `*localconfig*`

- In `data`:
  - Block everything
  - But allow: `duplicates.rdf`

- In `data/webdot`:
  - If you use a remote webdot server:
    - Block everything
    - But allow `*.dot` only for the remote webdot server

  - Otherwise, if you use a local GraphViz:
    - Block everything
    - But allow: `*.png`, `*.gif`, `*.jpg`, `*.map`

  - And if you don't use any dot:
    - Block everything

- In `Bugzilla`:
  - Block everything

- In `template`:
  - Block everything

Be sure to test that data that should not be accessed remotely is properly blocked. Of particular interest is the localconfig file which contains your database password. Also, be aware that many editors create temporary and backup files in the working directory and that those should also not be accessible. For more

information, see bug 186383 (http://bugzilla.mozilla.org/show_bug.cgi?id=186383) or Bugtraq ID 6501 (http://online.securityfocus.com/bid/6501). To test, simply run `testserver.pl`, as said above.

Tip: Be sure to check Section 2.2.4 for instructions specific to the web server you use.

4.4. Bugzilla

4.4.1. Prevent users injecting malicious Javascript

If you installed Bugzilla version 2.22 or later from scratch, then the *utf8* parameter is switched on by default. This makes Bugzilla explicitly set the character encoding, following a CERT advisory (http://www.cert.org/tech_tips/malicious_code_mitigation.html#3) recommending exactly this. The following therefore does not apply to you; just keep *utf8* turned on.

If you've upgraded from an older version, then it may be possible for a Bugzilla user to take advantage of character set encoding ambiguities to inject HTML into Bugzilla comments. This could include malicious scripts. This is because due to internationalization concerns, we are unable to turn the *utf8* parameter on by default for upgraded installations. Turning it on manually will prevent this problem.

Chapter 5. Using Bugzilla

5.1. Introduction

This section contains information for end-users of Bugzilla. There is a Bugzilla test installation, called Landfill (http://landfill.bugzilla.org/), which you are welcome to play with (if it's up). However, not all of the Bugzilla installations there will necessarily have all Bugzilla features enabled, and different installations run different versions, so some things may not quite work as this document describes.

5.2. Create a Bugzilla Account

If you want to use Bugzilla, first you need to create an account. Consult with the administrator responsible for your installation of Bugzilla for the URL you should use to access it. If you're test-driving Bugzilla, use this URL: http://landfill.bugzilla.org/bugzilla-3.0-branch/.

1. On the home page `index.cgi`, click the "Open a new Bugzilla account" link, or the "New Account" link available in the footer of pages. Now enter your email address, then click the "Send" button.

   Note: If none of these links is available, this means that the administrator of the installation has disabled self-registration. This means that only an administrator can create accounts for other users. One reason could be that this installation is private.

   Note: Also, if only some users are allowed to create an account on the installation, you may see these links but your registration may fail if your email address doesn't match the ones accepted by the installation. This is another way to restrict who can access and edit bugs in this installation.

2. Within moments, and if your registration is accepted, you should receive an email to the address you provided, which contains your login name (generally the same as the email address), and two URLs with a token (a random string generated by the installation) to confirm, respectively cancel, your registration. This is a way to prevent users from abusing the generation of user accounts, for instance by entering inexistent email addresses, or email addresses which do not belong to them.
3. By default, you have 3 days to confirm your registration. Past this timeframe, the token is invalidated and the registration is automatically canceled. You can also cancel this registration sooner by using the appropriate URL in the email you got.
4. If you confirm your registration, Bugzilla will ask you your real name (optional, but recommended) and your password, which must be between 3 and 16 characters long.
5. Now all you need to do is to click the "Log In" link in the footer at the bottom of the page in your browser, enter your email address and password you just chose into the login form, and click the "Log in" button.

You are now logged in. Bugzilla uses cookies to remember you are logged in so, unless you have cookies disabled or your IP address changes, you should not have to log in again during your session.

5.3. Anatomy of a Bug

The core of Bugzilla is the screen which displays a particular bug. It's a good place to explain some Bugzilla concepts. Bug 1 on Landfill (http://landfill.bugzilla.org/bugzilla-3.0-branch/show_bug.cgi?id=1) is a good example. Note that the labels for most fields are hyperlinks; clicking them will take you to context-sensitive help on that particular field. Fields marked * may not be present on every installation of Bugzilla.

1. *Product and Component*: Bugs are divided up by Product and Component, with a Product having one or more Components in it. For example, bugzilla.mozilla.org's "Bugzilla" Product is composed of several Components:

   Administration: Administration of a Bugzilla installation.
   Bugzilla-General: Anything that doesn't fit in the other components, or spans multiple components.
   Creating/Changing Bugs: Creating, changing, and viewing bugs.
   Documentation: The Bugzilla documentation, including The Bugzilla Guide.
   Email: Anything to do with email sent by Bugzilla.
   Installation: The installation process of Bugzilla.
   Query/Buglist: Anything to do with searching for bugs and viewing the buglists.
   Reporting/Charting: Getting reports from Bugzilla.
   User Accounts: Anything about managing a user account from the user's perspective. Saved queries, creating accounts, changing
   User Interface: General issues having to do with the user interface cosmetics (not functionality) including cosmetic issues, HTM

2. *Status and Resolution:* These define exactly what state the bug is in - from not even being confirmed as a bug, through to being fixed and the fix confirmed by Quality Assurance. The different possible values for Status and Resolution on your installation should be documented in the context-sensitive help for those items.
3. *Assigned To:* The person responsible for fixing the bug.
4. **QA Contact:* The person responsible for quality assurance on this bug.
5. **URL:* A URL associated with the bug, if any.
6. *Summary:* A one-sentence summary of the problem.
7. **Status Whiteboard:* (a.k.a. Whiteboard) A free-form text area for adding short notes and tags to a bug.
8. **Keywords:* The administrator can define keywords which you can use to tag and categorise bugs - e.g. The Mozilla Project has keywords like crash and regression.
9. *Platform and OS:* These indicate the computing environment where the bug was found.
10. *Version:* The "Version" field is usually used for versions of a product which have been released, and is set to indicate which versions of a Component have the particular problem the bug report is about.
11. *Priority:* The bug assignee uses this field to prioritize his or her bugs. It's a good idea not to change this on other people's bugs.
12. *Severity:* This indicates how severe the problem is - from blocker ("application unusable") to trivial ("minor cosmetic issue"). You can also use this field to indicate whether a bug is an enhancement request.
13. **Target:* (a.k.a. Target Milestone) A future version by which the bug is to be fixed. e.g. The Bugzilla Project's milestones for future Bugzilla versions are 2.18, 2.20, 3.0, etc. Milestones are not restricted to numbers, thought

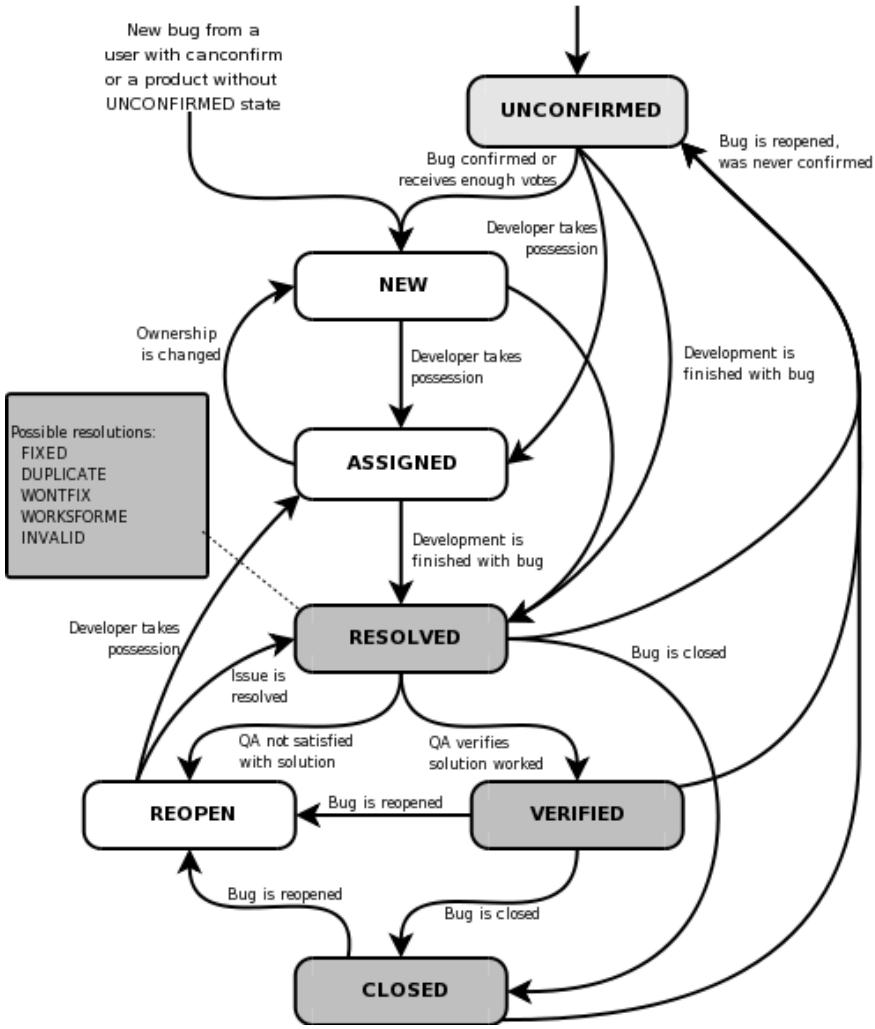
- you can use any text strings, such as dates.

14. *Reporter:* The person who filed the bug.
15. *CC list:* A list of people who get mail when the bug changes.
16. **Time Tracking:* This form can be used for time tracking. To use this feature, you have to be blessed group membership specified by the "timetrackinggroup" parameter.
    Orig. Est.: This field shows the original estimated time.
    Current Est.: This field shows the current estimated time. This number is calculated from "Hours Worked" and "Hours Left".
    Hours Worked: This field shows the number of hours worked.
    Hours Left: This field shows the "Current Est." - "Hours Worked". This value + "Hours Worked" will become the new Current Es
    %Complete: This field shows what percentage of the task is complete.
    Gain: This field shows the number of hours that the bug is ahead of the "Orig. Est.".
    Deadline: This field shows the deadline for this bug.
17. *Attachments:* You can attach files (e.g. testcases or patches) to bugs. If there are any attachments, they are listed in this section. Attachments are normally stored in the Bugzilla database, unless they are marked as Big Files, which are stored directly on disk.
18. **Dependencies:* If this bug cannot be fixed unless other bugs are fixed (depends on), or this bug stops other bugs being fixed (blocks), their numbers are recorded here.
19. **Votes:* Whether this bug has any votes.
20. *Additional Comments:* You can add your two cents to the bug discussion here, if you have something worthwhile to say.

5.4. Life Cycle of a Bug

The life cycle, also known as work flow, of a bug is currently hardcoded into Bugzilla. Figure 5-1 contains a graphical representation of this life cycle. If you wish to customize this image for your site, the diagram file (../images/bzLifecycle.xml) is available in Dia's (http://www.gnome.org/projects/dia) native XML format.

Figure 5-1. Lifecycle of a Bugzilla Bug



5.5. Searching for Bugs

The Bugzilla Search page is the interface where you can find any bug report, comment, or patch currently in the Bugzilla system. You can play with it here: http://landfill.bugzilla.org/bugzilla-3.0-branch/query.cgi.

The Search page has controls for selecting different possible values for all of the fields in a bug, as described above. For some fields, multiple values can be selected. In those cases, Bugzilla returns bugs where the content of the field matches any one of the selected values. If none is selected, then the field can take any value.

Once you've run a search, you can save it as a Saved Search, which appears in the page footer. On the Saved Searches tab of your User Preferences page (the Prefs link in Bugzilla's footer), members of the group defined in the queryshare-group parameter can share such Saved Searches with user groups so that other users may use them. At the same place, you can see Saved Searches other users are sharing, and have them show up in your personal Bugzilla footer along with your own Saved Searches. If somebody is sharing a Search with a group she or he is allowed to assign users to,

the sharer may opt to have the Search show up in the group's direct members' footers by default.

5.5.1. Boolean Charts

Highly advanced querying is done using Boolean Charts.

The boolean charts further restrict the set of results returned by a query. It is possible to search for bugs based on elaborate combinations of criteria.

The simplest boolean searches have only one term. These searches permit the selected left *field* to be compared using a selectable *operator* to a specified *value.* Using the "And," "Or," and "Add Another Boolean Chart" buttons, additional terms can be included in the query, further altering the list of bugs returned by the query.

There are three fields in each row of a boolean search.

- *Field:* the items being searched
- *Operator:* the comparison operator
- *Value:* the value to which the field is being compared

5.5.1.1. Pronoun Substitution

Sometimes, a query needs to compare a field containing a user's ID (such as ReportedBy) with a user's ID (such as the user running the query or the user to whom each bug is assigned). When the operator is either "equals" or "notequals", the value can be "%reporter%", "%assignee%", "%qacontact%", or "%user%." The user pronoun refers to the user who is executing the query or, in the case of whining reports, the user who will be the recipient of the report. The reporter, assignee, and qacontact pronouns refer to the corresponding fields in the bug.

5.5.1.2. Negation

At first glance, negation seems redundant. Rather than searching for

```
NOT("summary" "contains the string" "foo"),
```

one could search for

```
("summary" "does not contain the string" "foo").
```

However, the search

```
("CC" "does not contain the string" "@mozilla.org")
```

would find every bug where anyone on the CC list did not contain "@mozilla.org" while

```
NOT("CC" "contains the string" "@mozilla.org")
```

would find every bug where there was nobody on the CC list who did contain the string. Similarly, the use of negation also permits complex expressions to be built using terms OR'd together and then negated. Negation permits queries such as

```
NOT(("product" "equals" "update") OR ("component" "equals" "Documentation"))
```

to find bugs that are neither in the update product or in the documentation component or

```
NOT(("commenter" "equals" "%assignee%") OR ("component" "equals" "Documentation"))
```

to find non-documentation bugs on which the assignee has never commented.

5.5.1.3. Multiple Charts

The terms within a single row of a boolean chart are all constraints on a single piece of data. If you are looking for a bug that has two different people cc'd on it, then you need to use two boolean charts. A search for

("cc" "contains the string" "foo@") AND ("cc" "contains the string" "@mozilla.org")

would return only bugs with "foo@mozilla.org" on the cc list. If you wanted bugs where there is someone on the cc list containing "foo@" and someone else containing "@mozilla.org", then you would need two boolean charts.

First chart: ("cc" "contains the string" "foo@")

Second chart: ("cc" "contains the string" "@mozilla.org")

The bugs listed will be only the bugs where ALL the charts are true.

5.5.2. Quicksearch

Quicksearch is a single-text-box query tool which uses metacharacters to indicate what is to be searched. For example, typing "`foo|bar`" into Quicksearch would search for "foo" or "bar" in the summary and status whiteboard of a bug; adding "`:BazProduct`" would search only in that product. You can use it to find a bug by its number or its alias, too.

You'll find the Quicksearch box in Bugzilla's footer area. On Bugzilla's front page, there is an additional Help (../../page.cgi?id=quicksearch.html) link which details how to use it.

5.5.3. Case Sensitivity in Searches

Bugzilla queries are case-insensitive and accent-insensitive, when used with MySQL databases. When using Bugzilla with PostgreSQL, however, some queries are case-sensitive. This is due to the way PostgreSQL handles case and accent sensitivity.

5.5.4. Bug Lists

If you run a search, a list of matching bugs will be returned.

The format of the list is configurable. For example, it can be sorted by clicking the column headings. Other useful features can be accessed using the links at the bottom of the list:

Long Format: this gives you a large page with a non-editable summary of the fields of each bug.
XML: get the buglist in the XML format.
CSV: get the buglist as comma-separated values, for import into e.g. a spreadsheet.
Feed: get the buglist as an Atom feed. Copy this link into your favorite feed reader. If you are using Firefox, you can also save the list
iCalendar: Get the buglist as an iCalendar file. Each bug is represented as a to-do item in the imported calendar.
Change Columns: change the bug attributes which appear in the list.
Change several bugs at once: If your account is sufficiently empowered, and more than one bug appear in the bug list, this link is disp
Send mail to bug assignees: If more than one bug appear in the bug list and there are at least two distinct bug assignees, this links is di
Edit Search: If you didn't get exactly the results you were looking for, you can return to the Query page through this link and make sn

Remember Search As: You can give a search a name and remember it; a link will appear in your page footer giving you quick access t

If you would like to access the bug list from another program it is often useful to have the list returned in something other than HTML. By adding the ctype=type parameter into the bug list URL you can specify several alternate formats. Besides the types described above, the following formats are also supported: ECMAScript, also known as JavaScript (ctype=js), and Resource Description Framework RDF/XML (ctype=rdf).

5.5.5. Adding/removing tags to/from bugs

You can add and remove tags from individual bugs, which let you find and manage them more easily. Creating a new tag automatically generates a saved search - whose name is the name of the tag - which lists bugs with this tag. This saved search will be displayed in the footer of pages by default, as all other saved searches. The main difference between tags and normal saved searches is that saved searches, as described in the previous section, are stored in the form of a list of matching criteria, while the saved search generated by tags is a list of bug numbers. Consequently, you can easily edit this list by either adding or removing tags from bugs. To enable this feature, you have to turn on the “Enable tags for bugs” user preference, see Section 5.10. This feature is disabled by default.

This feature is useful when you want to keep track of several bugs, but for different reasons. Instead of adding yourself to the CC list of all these bugs and mixing all these reasons, you can now store these bugs in separate lists, e.g. “Keep in mind”, “Interesting bugs”, or “Triage”. One big advantage of this way to manage bugs is that you can easily add or remove bugs one by one, which is not possible to do with saved searches without having to edit the search criteria again.

5.6. Filing Bugs

5.6.1. Reporting a New Bug

Years of bug writing experience has been distilled for your reading pleasure into the Bug Writing Guidelines (http://landfill.bugzilla.org/bugzilla-3.0-branch/page.cgi?id=bug-writing.html). While some of the advice is Mozilla-specific, the basic principles of reporting Reproducible, Specific bugs, isolating the Product you are using, the Version of the Product, the Component which failed, the Hardware Platform, and Operating System you were using at the time of the failure go a long way toward ensuring accurate, responsible fixes for the bug that bit you.

The procedure for filing a bug is as follows:

1. Click the “New” link available in the footer of pages, or the “Enter a new bug report” link displayed on the home page of the Bugzilla installation.

   Note: If you want to file a test bug to see how Bugzilla works, you can do it on one of our test installations on Landfill (http://landfill.bugzilla.org/bugzilla-3.0-branch/).

2. You first have to select the product in which you found a bug.

3. You now see a form where you can specify the component (part of the product which is affected by the bug you discovered; if you have no idea, just select “General” if such a component exists), the version of the program you were using, the Operating System and platform your program is running on and the severity of the bug (if the bug you found crashes the program, it’s probably a major or a critical bug; if it’s a typo somewhere, that’s something pretty minor; if it’s something you would like to see implemented, then that’s an enhancement).
4. You now have to give a short but descriptive summary of the bug you found. “My program is crashing all the time” is a very poor summary and doesn’t help developers at all. Try something more meaningful or your bug will probably be ignored due to a lack of precision. The next step is to give a very detailed list of steps to reproduce the problem you encountered. Try to limit these steps to a minimum set required to reproduce the problem. This will make the life of developers easier, and the probability that they consider your bug in a reasonable timeframe will be much higher.

   Note: Try to make sure that everything in the summary is also in the first comment. Summaries are often updated and this will ensure your original information is easily accessible.

5. As you file the bug, you can also attach a document (testcase, patch, or screenshot of the problem).
6. Depending on the Bugzilla installation you are using and the product in which you are filing the bug, you can also request developers to consider your bug in different ways (such as requesting review for the patch you just attached, requesting your bug to block the next release of the product, and many other product specific requests).
7. Now is a good time to read your bug report again. Remove all misspellings, otherwise your bug may not be found by developers running queries for some specific words, and so your bug would not get any attention. Also make sure you didn’t forget any important information developers should know in order to reproduce the problem, and make sure your description of the problem is explicit and clear enough. When you think your bug report is ready to go, the last step is to click the “Commit” button to add your report into the database.

You do not need to put "any" or similar strings in the URL field. If there is no specific URL associated with the bug, leave this field blank.

If you feel a bug you filed was incorrectly marked as a DUPLICATE of another, please question it in your bug, not the bug it was duped to. Feel free to CC the person who duped it if they are not already CCed.

5.6.2. Clone an Existing Bug

Starting with version 2.20, Bugzilla has a feature that allows you to clone an existing bug. The newly created bug will inherit most settings from the old bug. This allows you to track more easily similar concerns in a new bug. To use this, go to the bug that you want to clone, then click the “Clone This Bug” link on the bug page. This will take you to the “Enter Bug” page that is filled with the values that the old bug has. You can change those values and/or texts if needed.

5.7. Attachments

You should use attachments, rather than comments, for large chunks of ASCII data, such as trace, debugging output files, or log files. That way, it doesn't bloat the bug for everyone who wants to read it, and cause people to receive fat, useless mails.

You should make sure to trim screenshots. There's no need to show the whole screen if you are pointing out a single-pixel problem.

Bugzilla stores and uses a Content-Type for each attachment (e.g. text/html). To download an attachment as a different Content-Type (e.g. application/xhtml+xml), you can override this using a 'content_type' parameter on the URL, e.g. `&content_type=text/plain`.

If you have a really large attachment, something that does not need to be recorded forever (as most attachments are), or something that is too big for your database, you can mark your attachment as a "Big File", assuming the administrator of the installation has enabled this feature. Big Files are stored directly on disk instead of in the database. The maximum size of a "Big File" is normally larger than the maximum size of a regular attachment. Independently of the storage system used, an administrator can delete these attachments at any time. Nevertheless, if these files are stored in the database, the "allow_attachment_deletion" parameter (which is turned off by default) must be enabled in order to delete them.

Also, if the administrator turned on the "allow_attach_url" parameter, you can enter the URL pointing to the attachment instead of uploading the attachment itself. For example, this is useful if you want to point to an external application, a website or a very large file. Note that there is no guarantee that the source file will always be available, nor that its content will remain unchanged.

5.7.1. Patch Viewer

Viewing and reviewing patches in Bugzilla is often difficult due to lack of context, improper format and the inherent readability issues that raw patches present. Patch Viewer is an enhancement to Bugzilla designed to fix that by offering increased context, linking to sections, and integrating with Bonsai, LXR and CVS.

Patch viewer allows you to:

View patches in color, with side-by-side view rather than trying to interpret the contents of the patch.
See the difference between two patches.
Get more context in a patch.
Collapse and expand sections of a patch for easy reading.
Link to a particular section of a patch for discussion or review
Go to Bonsai or LXR to see more context, blame, and cross-references for the part of the patch you are looking at
Create a rawtext unified format diff out of any patch, no matter what format it came from

5.7.1.1. Viewing Patches in Patch Viewer

The main way to view a patch in patch viewer is to click on the "Diff" link next to a patch in the Attachments list on a bug. You may also do this within the edit window by clicking the "View Attachment As Diff" button in the Edit Attachment screen.

5.7.1.2. Seeing the Difference Between Two Patches

To see the difference between two patches, you must first view the newer patch in Patch Viewer. Then select the older patch from the dropdown at the top of the page ("Differences between [dropdown] and this patch") and click the "Diff" button. This will show you what is new or changed in the newer patch.

5.7.1.3. Getting More Context in a Patch

To get more context in a patch, you put a number in the textbox at the top of Patch Viewer ("Patch / File / [textbox]") and hit enter. This will give you that many lines of context before and after each change. Alternatively, you can click on the "File" link there and it will show each change in the full context of the file. This feature only works against files that were diffed using "cvs diff".

5.7.1.4. Collapsing and Expanding Sections of a Patch

To view only a certain set of files in a patch (for example, if a patch is absolutely huge and you want to only review part of it at a time), you can click the "(+)" and "(-)" links next to each file (to expand it or collapse it). If you want to collapse all files or expand all files, you can click the "Collapse All" and "Expand All" links at the top of the page.

5.7.1.5. Linking to a Section of a Patch

To link to a section of a patch (for example, if you want to be able to give someone a URL to show them which part you are talking about) you simply click the "Link Here" link on the section header. The resulting URL can be copied and used in discussion.

5.7.1.6. Going to Bonsai and LXR

To go to Bonsai to get blame for the lines you are interested in, you can click the "Lines XX-YY" link on the section header you are interested in. This works even if the patch is against an old version of the file, since Bonsai stores all versions of the file.

To go to LXR, you click on the filename on the file header (unfortunately, since LXR only does the most recent version, line numbers are likely to rot).

5.7.1.7. Creating a Unified Diff

If the patch is not in a format that you like, you can turn it into a unified diff format by clicking the "Raw Unified" link at the top of the page.

5.8. Hints and Tips

This section distills some Bugzilla tips and best practices that have been developed.

5.8.1. Autolinkification

Bugzilla comments are plain text - so typing <U> will produce less-than, U, greater-than rather than underlined text. However, Bugzilla will automatically make hyperlinks out of certain sorts of text in comments. For example, the text "http://www.bugzilla.org" will be turned into a link: http://www.bugzilla.org. Other strings which get linkified in the obvious manner are:

bug 12345
comment 7
bug 23456, comment 53
attachment 4321
mailto:george@example.com
george@example.com
ftp://ftp.mozilla.org
Most other sorts of URL

A corollary here is that if you type a bug number in a comment, you should put the word "bug" before it, so it gets autolinkified for the convenience of others.

5.8.2. Comments

If you are changing the fields on a bug, only comment if either you have something pertinent to say, or Bugzilla requires it. Otherwise, you may spam people unnecessarily with bug mail. To take an example: a user can set up their account to filter out messages where someone just adds themselves to the CC field of a bug (which happens a lot.) If you come along, add yourself to the CC field, and add a comment saying "Adding self to CC", then that person gets a pointless piece of mail they would otherwise have avoided.

Don’t use sigs in comments. Signing your name ("Bill") is acceptable, if you do it out of habit, but full mail/news-style four line ASCII art creations are not.

5.8.3. Server-Side Comment Wrapping

Bugzilla stores comments unwrapped and wraps them at display time. This ensures proper wrapping in all browsers. Lines beginning with the ">" character are assumed to be quotes, and are not wrapped.

5.8.4. Dependency Tree

On the “Dependency tree” page linked from each bug page, you can see the dependency relationship from the bug as a tree structure.

You can change how much depth to show, and you can hide resolved bugs from this page. You can also collaps/expand dependencies for each bug on the tree view, using the [-]/[+] buttons that appear before its summary. This option is not available for terminal bugs in the tree (that don’t have further dependencies).

5.9. Time Tracking Information

Users who belong to the group specified by the “timetrackinggroup” parameter have access to time-related fields. Developers can see deadlines and estimated times to fix bugs, and can provide time spent on these bugs.

At any time, a summary of the time spent by developers on bugs is accessible either from bug lists when clicking the “Time Summary” button or from individual bugs when clicking the “Summarize time” link in the time tracking table. The `summarize_time.cgi` page lets you view this information either per developer or per bug, and can be split on a month basis to have greater details on how time is spent by developers.

As soon as a bug is marked as RESOLVED, the remaining time expected to fix the bug is set to zero. This lets QA people set it again for their own usage, and it will be set to zero again when the bug will be marked as CLOSED.

5.10. User Preferences

Once logged in, you can customize various aspects of Bugzilla via the "Preferences" link in the page footer. The preferences are split into five tabs:

5.10.1. General Preferences

This tab allows you to change several default settings of Bugzilla.

- Bugzilla’s general appearance (skin) - select which skin to use. Bugzilla supports adding custom skins.
- Field separator character for CSV files - Select between a comma and semi-colon for exported CSV bug lists.
- Automatically add me to the CC list of bugs I change - set default behavior of CC list. Options include "Always", "Never", and "Only if I have no role on them".
- Language used in email - select which language email will be sent in, from the list of available languages.
- After changing a bug - This controls what page is displayed after changes to a bug are submitted. The options include to show the bug just modified, to show the next bug in your list, or to do nothing.
- Enable tags for bugs - turn bug tagging on or off.
- When viewing a bug, show comments in this order - controls the order of comments. Options include "Oldest to Newest", "Newest to Oldest" and "Newest to Oldest, but keep the bug description at the top".
- Show a quip at the top of each bug list - controls whether a quip will be shown on the Bug list page.
- Zoom textareas large when in use (requires JavaScript) - enable or disable the automatic expanding of text areas when text is being entered into them.

5.10.2. Email Preferences

This tab allows you to enable or disable email notification on specific events.

In general, users have almost complete control over how much (or how little) email Bugzilla sends them. If you want to receive the maximum amount of email possible, click the “Enable All Mail” button. If you don’t want to receive any email from Bugzilla at all, click the “Disable All Mail” button.

> **Note:** A Bugzilla administrator can stop a user from receiving bugmail by clicking the “Bugmail Disabled” checkbox when editing the user account. This is a drastic step best taken only for disabled accounts, as it overrides the user’s individual mail preferences.

There are two global options -- “Email me when someone asks me to set a flag” and “Email me when someone sets a flag I asked for”. These define how you want to receive bugmail with regards to flags. Their use is quite straightforward; enable the checkboxes if you want Bugzilla to send you mail under either of the above conditions.

If you’d like to set your bugmail to something besides ’Completely ON’ and ’Completely OFF’, the “Field/recipient specific options” table allows you to do just that. The rows of the table define events that can happen to a bug -- things like attachments being added, new comments being made, the priority changing, etc. The columns in the table define your relationship with the bug:

- Reporter - Where you are the person who initially reported the bug. Your name/account appears in the “Reporter:” field.
- Assignee - Where you are the person who has been designated as the one responsible for the bug. Your name/account appears in the “Assigned To:” field of the bug.
- QA Contact - You are one of the designated QA Contacts for the bug. Your account appears in the “QA Contact:” text-box of the bug.
- CC - You are on the list CC List for the bug. Your account appears in the “CC:” text box of the bug.
- Voter - You have placed one or more votes for the bug. Your account appears only if someone clicks on the “Show votes for this bug” link on the bug.

> **Note:** Some columns may not be visible for your installation, depending on your site’s configuration.

To fine-tune your bugmail, decide the events for which you want to receive bugmail; then decide if you want to receive it all the time (enable the checkbox for every column), or only when you have a certain relationship with a bug (enable the checkbox only for those columns). For example: if you didn’t want to receive mail when someone added themselves to the CC list, you could uncheck all the boxes in the “CC Field Changes” line. As another example, if you never wanted to receive email on bugs you reported unless the bug was resolved, you would un-check all boxes in the “Reporter” column except for the one on the “The bug is resolved or verified” row.

> **Note:** Bugzilla adds the “X-Bugzilla-Reason” header to all bugmail it sends, describing the recipient’s relationship (AssignedTo, Reporter, QAContact, CC, or Voter) to the bug. This header can be used to do further client-side filtering.

Bugzilla has a feature called “Users Watching”. When you enter one or more comma-delineated user accounts (usually email addresses) into the text entry box, you will receive a copy of all the bugmail those users are sent (security settings permitting). This powerful functionality enables seamless transitions as developers change projects or users go on holiday.

> **Note:** The ability to watch other users may not be available in all Bugzilla installations. If you don’t see this feature, and feel that you need it, speak to your administrator.

Each user listed in the “Users watching you” field has you listed in their “Users to watch” list and can get bugmail according to your relationship to the bug and their “Field/recipient specific options” setting.

5.10.3. Saved Searches

On this tab you can view and run any Saved Searches that you have created, and also any Saved Searches that other members of the group defined in the "querysharegroup" parameter have shared. Saved Searches can be added to the page footer from this screen. If somebody is sharing a Search with a group she or he is allowed to assign users to, the sharer may opt to have the Search show up in the footer of the group's direct members by default.

5.10.4. Name and Password

On this tab, you can change your basic account information, including your password, email address and real name. For security reasons, in order to change anything on this page you must type your *current* password into the "Password" field at the top of the page. If you attempt to change your email address, a confirmation email is sent to both the old and new addresses, with a link to use to confirm the change. This helps to prevent account hijacking.

5.10.5. Permissions

This is a purely informative page which outlines your current permissions on this installation of Bugzilla.

A complete list of permissions is below. Only users with *editusers* privileges can change the permissions of other users.

admin

Indicates user is an Administrator.

bz_canusewhineatothers

Indicates user can configure whine reports for other users.

bz_canusewhines

Indicates user can configure whine reports for self.

bz_sudoers

Indicates user can perform actions as other users.

bz_sudo_protect

Indicates user can not be impersonated by other users.

canconfirm

Indicates user can confirm a bug or mark it a duplicate.

creategroups

Indicates user can create and destroy groups.

editbugs

Indicates user can edit all bug fields.

editclassifications

Indicates user can create, destroy, and edit classifications.

editcomponents

Indicates user can create, destroy, and edit components.

editkeywords

Indicates user can create, destroy, and edit keywords.

editusers

Indicates user can edit or disable users.

tweakparams

Indicates user can change Parameters.

Note: For more information on how permissions work in Bugzilla (i.e. who can change what), see Section 6.4.

5.11. Reports and Charts

As well as the standard buglist, Bugzilla has two more ways of viewing sets of bugs. These are the reports (which give different views of the current state of the database) and charts (which plot the changes in particular sets of bugs over time.)

5.11.1. Reports

A report is a view of the current state of the bug database.

You can run either an HTML-table-based report, or a graphical line/pie/bar-chart-based one. The two have different pages to define them, but are close cousins - once you've defined and viewed a report, you can switch between any of the different views of the data at will.

Both report types are based on the idea of defining a set of bugs using the standard search interface, and then choosing some aspect of that set to plot on the horizontal and/or vertical axes. You can also get a form of 3-dimensional report by choosing to have multiple images or tables.

So, for example, you could use the search form to choose "all bugs in the WorldControl product", and then plot their severity against their component to see which component had had the largest number of bad bugs reported against it.

Once you've defined your parameters and hit "Generate Report", you can switch between HTML, CSV, Bar, Line and Pie. (Note: Pie is only available if you didn't define a vertical axis, as pie charts don't have one.) The other controls are fairly self-explanatory; you can change the size of the image if you find text is overwriting other text, or the bars are too thin to see.

5.11.2. Charts

A chart is a view of the state of the bug database over time.

Bugzilla currently has two charting systems - Old Charts and New Charts. Old Charts have been part of Bugzilla for a long time; they chart each status and resolution for each product, and that's all. They are deprecated, and going away soon - we won't say any more about them. New Charts are the future - they allow you to chart anything you can define as a search.

> **Note:** Both charting forms require the administrator to set up the data-gathering script. If you can't see any charts, ask them whether they have done so.

An individual line on a chart is called a data set. All data sets are organised into categories and subcategories. The data sets that Bugzilla defines automatically use the Product name as a Category and Component names as Subcategories, but there is no need for you to follow that naming scheme with your own charts if you don't want to.

Data sets may be public or private. Everyone sees public data sets in the list, but only their creator sees private data sets. Only administrators can make data sets public. No two data sets, even two private ones, can have the same set of category, subcategory and name. So if you are creating private data sets, one idea is to have the Category be your username.

5.11.2.1. Creating Charts

You create a chart by selecting a number of data sets from the list, and pressing Add To List for each. In the List Of Data Sets To Plot, you can define the label that data set will have in the chart's legend, and also ask Bugzilla to Sum a number of data sets (e.g. you could Sum data sets representing RESOLVED, VERIFIED and CLOSED in a particular product to get a data set representing all the resolved bugs in that product.)

If you've erroneously added a data set to the list, select it using the checkbox and click Remove. Once you add more than one data set, a "Grand Total" line automatically appears at the bottom of the list. If you don't want this, simply remove it as you would remove any other line.

You may also choose to plot only over a certain date range, and to cumulate the results - that is, to plot each one using the previous one as a baseline, so the top line gives a sum of all the data sets. It's easier to try than to explain :-)

Once a data set is in the list, one can also perform certain actions on it. For example, one can edit the data set's parameters (name, frequency etc.) if it's one you created or if you are an administrator.

Once you are happy, click Chart This List to see the chart.

5.11.2.2. Creating New Data Sets

You may also create new data sets of your own. To do this, click the "create a new data set" link on the Create Chart page. This takes you to a search-like interface where you can define the search that Bugzilla will plot. At the bottom of the page, you choose the category, sub-category and name of your new data set.

If you have sufficient permissions, you can make the data set public, and reduce the frequency of data collection to less than the default seven days.

5.12. Flags

A flag is a kind of status that can be set on bugs or attachments to indicate that the bugs/attachments are in a certain state. Each installation can define its own set of flags that can be set on bugs or attachments.

If your installation has defined a flag, you can set or unset that flag, and if your administrator has enabled requesting of flags, you can submit a request for another user to set the flag.

To set a flag, select either "+" or "-" from the drop-down menu next to the name of the flag in the "Flags" list. The meaning of these values are flag-specific and thus cannot be described in this documentation, but by way of example, setting a flag named "review" to "+" may indicate that the bug/attachment has passed review, while setting it to "-" may indicate that the bug/attachment has failed review.

To unset a flag, click its drop-down menu and select the blank value. Note that marking an attachment as obsolete automatically cancels all pending requests for the attachment.

If your administrator has enabled requests for a flag, request a flag by selecting "?" from the drop-down menu and then entering the username of the user you want to set the flag in the text field next to the menu.

A set flag appears in bug reports and on "edit attachment" pages with the abbreviated username of the user who set the flag prepended to the flag name. For example, if Jack sets a "review" flag to "+", it appears as Jack: review [+]

A requested flag appears with the user who requested the flag prepended to the flag name and the user who has been requested to set the flag appended to the flag name within parentheses. For example, if Jack asks Jill for review, it appears as Jack: review [?] (Jill).

You can browse through open requests made of you and by you by selecting 'My Requests' from the footer. You can also look at open requests limited by other requesters, requestees, products, components, and flag names from this page. Note that you can use '-' for requestee to specify flags with 'no requestee' set.

5.13. Whining

Whining is a feature in Bugzilla that can regularly annoy users at specified times. Using this feature, users can execute saved searches at specific times (i.e. the 15th of the month at midnight) or at regular intervals (i.e. every 15 minutes on Sundays). The results of the searches are sent to the user, either as a single email or as one email per bug, along with some descriptive text.

> **Warning**
>
> Throughout this section it will be assumed that all users are members of the bz_canusewhines group, membership in which is required in order to use the Whining system. You can easily make all users members of the bz_canusewhines group by setting the User RegExp to ".*" (without the quotes).
>
> Also worth noting is the bz_canusewhineatothers group. Members of this group can create whines for any user or group in Bugzilla using a extended form of the whining interface. Features only available to members of the bz_canusewhineatothers group will be noted in the appropriate places.

Note: For whining to work, a special Perl script must be executed at regular intervals. More information on this is available in Section 2.3.4.

Note: This section does not cover the whineatnews.pl script. See Section 2.3.3 for more information on The Whining Cron.

5.13.1. The Event

The whining system defines an "Event" as one or more queries being executed at regular intervals, with the results of said queries (if there are any) being emailed to the user. Events are created by clicking on the "Add new event" button.

Once a new event is created, the first thing to set is the "Email subject line". The contents of this field will be used in the subject line of every email generated by this event. In addition to setting a subject, space is provided to enter some descriptive text that will be included at the top of each message (to help you in understanding why you received the email in the first place).

The next step is to specify when the Event is to be run (the Schedule) and what searches are to be performed (the Searches).

5.13.2. Whining Schedule

Each whining event is associated with zero or more schedules. A schedule is used to specify when the query (specified below) is to be run. A new event starts out with no schedules (which means it will never run, as it is not scheduled to run). To add a schedule, press the "Add a new schedule" button.

Each schedule includes an interval, which you use to tell Bugzilla when the event should be run. An event can be run on certain days of the week, certain days of the month, during weekdays (defined as Monday through Friday), or every day.

Warning

Be careful if you set your event to run on the 29th, 30th, or 31st of the month, as your event may not run exactly when expected. If you want your event to run on the last day of the month, select "Last day of the month" as the interval.

Once you have specified the day(s) on which the event is to be run, you should now specify the time at which the event is to be run. You can have the event run at a certain hour on the specified day(s), or every hour, half-hour, or quarter-hour on the specified day(s).

If a single schedule does not execute an event as many times as you would want, you can create another schedule for the same event. For example, if you want to run an event on days whose numbers are divisible by seven, you would need to add four schedules to the event, setting the schedules to run on the 7th, 14th, 21st, and 28th (one day per schedule) at whatever time (or times) you choose.

Note: If you are a member of the bz_canusewhineatothers group, then you will be presented with another option: "Mail to". Using this you can control who will receive the emails generated by this event. You can choose to send the emails to a single user (identified by email address) or a single group (identified by group name). To send to multiple users or groups, create a new schedule for each additional user/group.

5.13.3. Whining Searches

Each whining event is associated with zero or more searches. A search is any saved search to be run as part of the specified schedule (see above). You start out without any searches associated with the event (which means that the event will not run, as there will never be any results to return). To add a search, press the "Include search" button.

The first field to examine in your newly added search is the Sort field. Searches are run, and results included, in the order specified by the Sort field. Searches with smaller Sort values will run before searches with bigger Sort values.

The next field to examine is the Search field. This is where you choose the actual search that is to be run. Instead of defining search parameters here, you are asked to choose from the list of saved searches (the same list that appears at the bottom of every Bugzilla page). You are only allowed to choose from searches that you have saved yourself (the default saved search, "My Bugs", is not a valid choice). If you do not have any saved searches, you can take this opportunity to create one (see Section 5.5.4).

> **Note:** When running queries, the whining system acts as if you are the user executing the query. This means that the whining system will ignore bugs that match your query, but that you can not access.

Once you have chosen the saved search to be executed, give the query a descriptive title. This title will appear in the email, above the results of the query. If you choose "One message per bug", the query title will appear at the top of each email that contains a bug matching your query.

Finally, decide if the results of the query should be sent in a single email, or if each bug should appear in its own email.

> **Warning**
>
> Think carefully before checking the "One message per bug" box. If you create a query that matches thousands of bugs, you will receive thousands of emails!

5.13.4. Saving Your Changes

Once you have defined at least one schedule, and created at least one query, go ahead and "Update/Commit". This will save your Event and make it available for immediate execution.

> **Note:** If you ever feel like deleting your event, you may do so using the "Remove Event" button in the upper-right corner of each Event. You can also modify an existing event, so long as you "Update/Commit" after completing your modifications.

Chapter 6. Customizing Bugzilla

6.1. Custom Skins

Bugzilla allows you to have multiple skins. These are custom CSS and possibly also custom images for Bugzilla. To create a new custom skin, you have two choices:

- Make a single CSS file, and put it in the `skins/contrib` directory.
- Make a directory that contains all the same CSS file names as `skins/standard/`, and put your directory in `skins/contrib/`.

After you put the file or the directory there, make sure to run checksetup.pl so that it can reset the file permissions correctly.

After you have installed the new skin, it will show up as an option in the user's General Preferences. If you would like to force a particular skin on all users, just select it in the Default Preferences and then uncheck "Enabled" on the preference.

6.2. Template Customization

Administrators can configure the look and feel of Bugzilla without having to edit Perl files or face the nightmare of massive merge conflicts when they upgrade to a newer version in the future.

Templatization also makes localized versions of Bugzilla possible, for the first time. It's possible to have Bugzilla's UI language determined by the user's browser. More information is available in Section 6.2.6.

6.2.1. Template Directory Structure

The template directory structure starts with top level directory named `template`, which contains a directory for each installed localization. The next level defines the language used in the templates. Bugzilla comes with English templates, so the directory name is `en`, and we will discuss `template/en` throughout the documentation. Below `template/en` is the `default` directory, which contains all the standard templates shipped with Bugzilla.

Warning

A directory `data/templates` also exists; this is where Template Toolkit puts the compiled versions of the templates from either the default or custom directories. *Do not* directly edit the files in this directory, or all your changes will be lost the next time Template Toolkit recompiles the templates.

6.2.2. Choosing a Customization Method

If you want to edit Bugzilla's templates, the first decision you must make is how you want to go about doing so. There are two choices, and which you use depends mainly on the scope of your modifications, and the method you plan to use to upgrade Bugzilla.

The first method of making customizations is to directly edit the templates found in `template/en/default`. This is probably the best way to go about it if you are going to be upgrading Bugzilla through CVS, because if you then execute a **cvs update**, any changes you have made will be merged automagically with the updated versions.

> **Note:** If you use this method, and CVS conflicts occur during an update, the conflicted templates (and possibly other parts of your installation) will not work until they are resolved.

The second method is to copy the templates to be modified into a mirrored directory structure under `template/en/custom`. Templates in this directory structure automatically override any identically-named and identically-located templates in the `default` directory.

> **Note:** The `custom` directory does not exist at first and must be created if you want to use it.

The second method of customization should be used if you use the overwriting method of upgrade, because otherwise your changes will be lost. This method may also be better if you are using the CVS method of upgrading and are going to make major changes, because it is guaranteed that the contents of this directory will not be touched during an upgrade, and you can then decide whether to continue using your own templates, or make the effort to merge your changes into the new versions by hand.

Using this method, your installation may break if incompatible changes are made to the template interface. Such changes should be documented in the release notes, provided you are using a stable release of Bugzilla. If you use using unstable code, you will need to deal with this one yourself, although if possible the changes will be mentioned before they occur in the deprecations section of the previous stable release's release notes.

> **Note:** Regardless of which method you choose, it is recommended that you run **./checksetup.pl** after creating or editing any templates in the `template/en/default` directory, and after editing any templates in the `custom` directory.

> **Warning**
>
> It is *required* that you run **./checksetup.pl** after creating a new template in the `custom` directory. Failure to do so will raise an incomprehensible error message.

6.2.3. How To Edit Templates

> **Note:** If you are making template changes that you intend on submitting back for inclusion in standard Bugzilla, you should read the relevant sections of the Developers' Guide (http://www.bugzilla.org/docs/developer.html).

The syntax of the Template Toolkit language is beyond the scope of this guide. It's reasonably easy to pick up by looking at the current templates; or, you can read the manual, available on the Template Toolkit home page (http://www.template-toolkit.org).

One thing you should take particular care about is the need to properly HTML filter data that has been passed into the template. This means that if the data can possibly contain special HTML characters such as <, and the data was not intended to be HTML, they need to be converted to entity form, i.e. <. You use the 'html' filter in the Template Toolkit to do this. If you forget, you may open up your installation to cross-site scripting attacks.

Also note that Bugzilla adds a few filters of its own, that are not in standard Template Toolkit. In particular, the 'url_quote' filter can convert characters that are illegal or have special meaning in URLs, such as &, to the encoded form, i.e. %26. This actually encodes most characters (but not the common ones such as letters and numbers and so on), including the HTML-special characters, so there's never a need to HTML filter afterwards.

Editing templates is a good way of doing a "poor man's custom fields". For example, if you don't use the Status Whiteboard, but want to have a free-form text entry box for "Build Identifier", then you can just edit the templates to change the field labels. It's still be called status_whiteboard internally, but your users don't need to know that.

6.2.4. Template Formats and Types

Some CGI's have the ability to use more than one template. For example, `buglist.cgi` can output itself as RDF, or as two formats of HTML (complex and simple). The mechanism that provides this feature is extensible.

Bugzilla can support different types of output, which again can have multiple formats. In order to request a certain type, you can append the &ctype=<contenttype> (such as rdf or html) to the `<cginame>.cgi` URL. If you would like to retrieve a certain format, you can use the &format=<format> (such as simple or complex) in the URL.

To see if a CGI supports multiple output formats and types, grep the CGI for "get_format". If it's not present, adding multiple format/type support isn't too hard - see how it's done in other CGIs, e.g. config.cgi.

To make a new format template for a CGI which supports this, open a current template for that CGI and take note of the INTERFACE comment (if present.) This comment defines what variables are passed into this template. If there isn't one, I'm afraid you'll have to read the template and the code to find out what information you get.

Write your template in whatever markup or text style is appropriate.

You now need to decide what content type you want your template served as. The content types are defined in the `Bugzilla/Constants.pm` file in the `contenttypes` constant. If your content type is not there, add it. Remember the three- or four-letter tag assigned to your content type. This tag will be part of the template filename.

> **Note:** After adding or changing a content type, it's suitable to edit `Bugzilla/Constants.pm` in order to reflect the changes. Also, the file should be kept up to date after an upgrade if content types have been customized in the past.

Save the template as `<stubname>-<formatname>.<contenttypetag>.tmpl`. Try out the template by calling the CGI as `<cginame>.cgi?format=<formatname>&ctype=<type>`.

6.2.5. Particular Templates

There are a few templates you may be particularly interested in customizing for your installation.

index.html.tmpl: This is the Bugzilla front page.

global/header.html.tmpl: This defines the header that goes on all Bugzilla pages. The header includes the banner, which is what appears to users and is probably what you want to edit instead. However the header also includes the HTML HEAD section, so you could for example add a stylesheet or META tag by editing the header.

global/banner.html.tmpl: This contains the “banner”, the part of the header that appears at the top of all Bugzilla pages. The default banner is reasonably barren, so you’ll probably want to customize this to give your installation a distinctive look and feel. It is recommended you preserve the Bugzilla version number in some form so the version you are running can be determined, and users know what docs to read.

global/footer.html.tmpl: This defines the footer that goes on all Bugzilla pages. Editing this is another way to quickly get a distinctive look and feel for your Bugzilla installation.

global/variables.none.tmpl: This defines a list of terms that may be changed in order to “brand” the Bugzilla instance In this way, terms like “bugs” can be replaced with “issues” across the whole Bugzilla installation. The name “Bugzilla” and other words can be customized as well.

list/table.html.tmpl: This template controls the appearance of the bug lists created by Bugzilla. Editing this template allows per-column control of the width and title of a column, the maximum display length of each entry, and the wrap behaviour of long entries. For long bug lists, Bugzilla inserts a ’break’ every 100 bugs by default; this behaviour is also controlled by this template, and that value can be modified here.

bug/create/user-message.html.tmpl: This is a message that appears near the top of the bug reporting page. By modifying this, you can tell your users how they should report bugs.

bug/process/midair.html.tmpl: This is the page used if two people submit simultaneous changes to the same bug. The second person to submit their changes will get this page to tell them what the first person did, and ask if they wish to overwrite those changes or go back and revisit the bug. The default title and header on this page read "Mid-air collision detected!" If you work in the aviation industry, or other environment where this might be found offensive (yes, we have true stories of this happening) you’ll want to change this to something more appropriate for your environment.

bug/create/create.html.tmpl and **bug/create/comment.txt.tmpl**: You may not wish to go to the effort of creating custom fields in Bugzilla, yet you want to make sure that each bug report contains a number of pieces of important information for which there is not a special field. The bug entry system has been designed in an extensible fashion to enable you to add arbitrary HTML widgets, such as drop-down lists or textboxes, to the bug entry page and have their values appear formatted in the initial comment. A hidden field that indicates the format should be added inside the form in order to make the template functional. Its value should be the suffix of the template filename. For example, if the file is called `create-cust.html.tmpl`, then

```
<input type="hidden" name="format" value="cust">
```

should be used inside the form.

An example of this is the mozilla.org guided bug submission form (http://landfill.bugzilla.org/bugzilla-tip/enter_bug.cgi?product=WorldControl&format=guided). The code for this comes with the Bugzilla distribution as an example for you to copy. It can be found in the files `create-guided.html.tmpl` and `comment-guided.html.tmpl`.

So to use this feature, create a custom template for `enter_bug.cgi`. The default template, on which you could base it, is `custom/bug/create/create.html.tmpl`. Call it `create-<formatname>.html.tmpl`, and in it, add

widgets for each piece of information you'd like collected - such as a build number, or set of steps to reproduce.

Then, create a template like `custom/bug/create/comment.txt.tmpl`, and call it `comment-<formatname>.txt.tmpl`. This template should reference the form fields you have created using the syntax `[% form.<fieldname> %]`. When a bug report is submitted, the initial comment attached to the bug report will be formatted according to the layout of this template.

For example, if your custom enter_bug template had a field

```
<input type="text" name="buildid" size="30">
```

and then your comment.txt.tmpl had

```
BuildID: [% form.buildid %]
```

then something like

```
BuildID: 20020303
```

would appear in the initial comment.

6.2.6. Configuring Bugzilla to Detect the User's Language

Bugzilla honours the user's Accept: HTTP header. You can install templates in other languages, and Bugzilla will pick the most appropriate according to a priority order defined by you. Many language templates can be obtained from http://www.bugzilla.org/download.html#localizations. Instructions for submitting new languages are also available from that location.

After untarring the localizations (or creating your own) in the `BUGZILLA_ROOT/template` directory, you must update the `languages` parameter to contain any localizations you'd like to permit. You may also wish to set the `defaultlanguage` parameter to something other than "en" if you don't want English to be the default language.

6.3. The Bugzilla Extension Mechanism

Warning

Custom extensions require Template Toolkit version 2.12 or above, or the application of a patch. See bug 239112 (http://bugzilla.mozilla.org/show_bug.cgi?id=239112) for details.

Extensions are a way for extensions to Bugzilla to insert code into the standard Bugzilla templates and source files without modifying these files themselves. The extension mechanism defines a consistent API for extending the standard templates and source files in a way that cleanly separates standard code from extension code. Hooks reduce merge conflicts and make it easier to write extensions that work across multiple versions of Bugzilla, making upgrading a Bugzilla installation with installed extensions easier. Furthermore, they make it easy to install and remove extensions as each extension is nothing more than a simple directory structure.

There are two main types of hooks: code hooks and template hooks. Code hooks allow extensions to invoke code at specific points in various source files, while template hooks allow extensions to add elements to the Bugzilla user interface.

A hook is just a named place in a standard source or template file where extension source code or template files for that hook get processed. Each extension has a corresponding directory in the Bugzilla directory tree (`BUGZILLA_ROOT/extensions/extension_name`). Hooking an extension source file or template to a hook is as simple as putting the extension file into extension's template or code directory. When Bugzilla processes the source file or template and reaches the hook, it will process all extension files in the hook's directory. The hooks themselves can be added into any source file or standard template upon request by extension authors.

To use hooks to extend Bugzilla, first make sure there is a hook at the appropriate place within the source file or template you want to extend. The exact appearance of a hook depends on if the hook is a code hook or a template hook.

Code hooks appear in Bugzilla source files as a single method call in the format `Bugzilla::Hook->process("name");`. For instance, `enter_bug.cgi` may invoke the hook `"enter_bug-entrydefaultvars"`. Thus, a source file at `BUGZILLA_ROOT/extensions/EXTENSION_NAME/code/enter_bug-entrydefaultvars.pl` will be automatically invoked when the code hook is reached.

Template hooks appear in the standard Bugzilla templates as a single directive in the format `[% Hook.process("name") %]`, where `name` is the unique name of the hook.

If you aren't sure what you want to extend or just want to browse the available hooks, either use your favorite multi-file search tool (e.g. **grep**) to search the standard templates for occurrences of `Hook.process` or the source files for occurrences of `Bugzilla::Hook::process`.

If there is no hook at the appropriate place within the Bugzilla source file or template you want to extend, file a bug requesting one (http://bugzilla.mozilla.org/enter_bug.cgi?product=Bugzilla&component=User%20Interface), specifying:

the source or template file for which you are requesting a hook;
where in the file you would like the hook to be placed (line number/position for latest version of the file in CVS or description of loca
the purpose of the hook;
a link to information about your extension, if any.

The Bugzilla reviewers will promptly review each hook request, name the hook, add it to the template or source file, and check the new version of the template into CVS.

You may optionally attach a patch to the bug which implements the hook and check it in yourself after receiving approval from a Bugzilla reviewer. The developers may suggest changes to the location of the hook based on their analysis of your needs or so the hook can satisfy the needs of multiple extensions, but the process of getting hooks approved and checked in is not as stringent as the process for general changes to Bugzilla, and any extension, whether released or still in development, can have hooks added to meet their needs.

After making sure the hook you need exists (or getting it added if not), add your extension to the directory within the Bugzilla extensions tree corresponding to the hook.

That's it! Now, when the source file or template containing the hook is processed, your extension file will be processed at the point where the hook appears.

For example, let's say you have an extension named Projman that adds project management capabilities to Bugzilla. Projman has an administration interface `edit-projects.cgi`, and you want to add a link to it into the navigation bar at the bottom of every Bugzilla page for those users who are authorized to administer projects.

The navigation bar is generated by the template file `useful-links.html.tmpl`, which is located in the `global/` subdirectory on the standard Bugzilla template path `BUGZILLA_ROOT/template/en/default/`. Looking in `useful-links.html.tmpl`, you find the following hook at the end of the list of standard Bugzilla administration links:

```
...
    [% ', <a href="editkeywords.cgi">keywords</a>'
                                            IF user.groups.editkeywords %]
    [% Hook.process("edit") %]
...
```

The corresponding extension file for this hook is `BUGZILLA_ROOT/extensions/projman/template/en/hook/global/useful-l`
You then create that template file and add the following constant:

```
...[% ', <a href="edit-projects.cgi">projects</a>' IF user.groups.projman_admins %]
```

Voila! The link now appears after the other administration links in the navigation bar for users in the `projman_admins` group.

Now, let us say your extension adds a custom "project_manager" field to enter_bug.cgi. You want to modify the CGI script to set the default project manager to be productname@company.com. Looking at `enter_bug.cgi`, you see the enter_bug-entrydefaultvars hook near the bottom of the file before the default form values are set. The corresponding extension source file for this hook is located at `BUGZILLA_ROOT/extensions/projman/code/enter_bug-entrydefaultvars.pl`. You then create that file and add the following:

```
$default{'project_manager'} = $product.'@company.com';
```

This code will be invoked whenever enter_bug.cgi is executed. Assuming that the rest of the customization was completed (e.g. the custom field was added to the enter_bug template and the required hooks were used in process_bug.cgi), the new field will now have this default value.

Notes:

- If your extension includes entirely new templates in addition to extensions of standard templates, it should store those new templates in its `BUGZILLA_ROOT/extensions/template/en/` directory. Extension template directories, like the `default/` and `custom/` directories, are part of the template search path, so putting templates there enables them to be found by the template processor.

  The template processor looks for templates first in the `custom/` directory (i.e. templates added by the specific installation), then in the `extensions/` directory (i.e. templates added by extensions), and finally in the `default/` directory (i.e. the standard Bugzilla templates). Thus, installation-specific templates override both default and extension templates.

- If you are looking to customize Bugzilla, you can also take advantage of template hooks. To do so, create a directory in `BUGZILLA_ROOT/template/en/custom/hook/` that corresponds to the hook you wish to use, then place your customization templates into those directories. For example, if you wanted to use the hook "end" in `global/useful-links.html.tmpl`, you would create the directory `BUGZILLA_ROOT/template/en/custom/hook/ global/useful-links.html.tmpl/end/` and add your customization template to this directory.

Obviously this method of customizing Bugzilla only lets you add code to the standard source files and templates; you cannot change the existing code. Nevertheless, for those customizations that only add code, this method can reduce conflicts when merging changes, making upgrading your customized Bugzilla installation easier.

6.4. Customizing Who Can Change What

Warning

This feature should be considered experimental; the Bugzilla code you will be changing is not stable, and could change or move between versions. Be aware that if you make modifications as outlined here, you may have to re-make them or port them if Bugzilla changes internally between versions, and you upgrade.

Companies often have rules about which employees, or classes of employees, are allowed to change certain things in the bug system. For example, only the bug's designated QA Contact may be allowed to VERIFY the bug. Bugzilla has been designed to make it easy for you to write your own custom rules to define who is allowed to make what sorts of value transition.

By default, assignees, QA owners and users with *editbugs* privileges can edit all fields of bugs, except group restrictions (unless they are members of the groups they are trying to change). Bug reporters also have the ability to edit some fields, but in a more restrictive manner. Other users, without *editbugs* privileges, can not edit bugs, except to comment and add themselves to the CC list.

For maximum flexibility, customizing this means editing Bugzilla's Perl code. This gives the administrator complete control over exactly who is allowed to do what. The relevant method is called `check_can_change_field()`, and is found in `Bug.pm` in your Bugzilla/ directory. If you open that file and search for "sub check_can_change_field", you'll find it.

This function has been carefully commented to allow you to see exactly how it works, and give you an idea of how to make changes to it. Certain marked sections should not be changed - these are the "plumbing" which makes the rest of the function work. In between those sections, you'll find snippets of code like:

```
# Allow the assignee to change anything.
if ($ownerid eq $whoid) {
    return 1;
}
```

It's fairly obvious what this piece of code does.

So, how does one go about changing this function? Well, simple changes can be made just by removing pieces - for example, if you wanted to prevent any user adding a comment to a bug, just remove the lines marked "Allow anyone to change comments." If you don't want the Reporter to have any special rights on bugs they have filed, just remove the entire section that deals with the Reporter.

More complex customizations are not much harder. Basically, you add a check in the right place in the function, i.e. after all the variables you are using have been set up. So, don't look at $ownerid before $ownerid has been obtained from the database. You can either add a positive check, which returns 1 (allow) if certain conditions are true, or a negative check, which returns 0 (deny.) E.g.:

```
if ($field eq "qacontact") {
```

```
        if (Bugzilla->user->groups("quality_assurance")) {
            return 1;
        }
        else {
            return 0;
        }
    }
```

This says that only users in the group "quality_assurance" can change the QA Contact field of a bug.

Getting more weird:

```
    if (($field eq "priority") &&
        (Bugzilla->user->email =~ /.*\@example\.com$/))
    {
        if ($oldvalue eq "P1") {
            return 1;
        }
        else {
            return 0;
        }
    }
```

This says that if the user is trying to change the priority field, and their email address is @example.com, they can only do so if the old value of the field was "P1". Not very useful, but illustrative.

Warning

If you are modifying `process_bug.cgi` in any way, do not change the code that is bounded by DO_NOT_CHANGE blocks. Doing so could compromise security, or cause your installation to stop working entirely.

For a list of possible field names, look at the bugs table in the database. If you need help writing custom rules for your organization, ask in the newsgroup.

6.5. Integrating Bugzilla with Third-Party Tools

6.5.1. Bonsai

Bonsai is a web-based tool for managing CVS, the Concurrent Versioning System . Using Bonsai, administrators can control open/closed status of trees, query a fast relational database back-end for change, branch, and comment information, and view changes made since the last time the tree was closed. Bonsai also integrates with Tinderbox, the Mozilla automated build management system.

6.5.2. CVS

CVS integration is best accomplished, at this point, using the Bugzilla Email Gateway.

Follow the instructions in this Guide for enabling Bugzilla e-mail integration. Ensure that your check-in script sends an email to your Bugzilla e-mail gateway with the subject of “[Bug XXXX]”, and you can have CVS check-in comments append to your Bugzilla bug. If you want to have the bug be closed automatically, you’ll have to modify the `contrib/bugzilla_email_append.pl` script.

There is also a CVSZilla project, based upon somewhat dated Bugzilla code, to integrate CVS and Bugzilla through CVS’ ability to email. Check it out at: http://www.cvszilla.org/.

Another system capable of CVS integration with Bugzilla is Scmbug. This system provides generic integration of Source code Configuration Management with Bugtracking. Check it out at: http://freshmeat.net/projects/scmbug/.

6.5.3. Perforce SCM

You can find the project page for Bugzilla and Teamtrack Perforce integration (p4dti) at: http://www.ravenbrook.com/project/p4dti/ . “p4dti” is now an officially supported product from Perforce, and you can find the "Perforce Public Depot" p4dti page at http://public.perforce.com/public/perforce/p4dti/index.html .

Integration of Perforce with Bugzilla, once patches are applied, is seamless. Perforce replication information will appear below the comments of each bug. Be certain you have a matching set of patches for the Bugzilla version you are installing. p4dti is designed to support multiple defect trackers, and maintains its own documentation for it. Please consult the pages linked above for further information.

6.5.4. Subversion

Subversion is a free/open-source version control system, designed to overcome various limitations of CVS. Integration of Subversion with Bugzilla is possible using Scmbug, a system providing generic integration of Source Code Configuration Management with Bugtracking. Scmbug is available at http://freshmeat.net/projects/scmbug/.

6.5.5. Tinderbox/Tinderbox2

Tinderbox is a continuous-build system which can integrate with Bugzilla - see http://www.mozilla.org/projects/tinderbox for details of Tinderbox, and http://tinderbox.mozilla.org/showbuilds.cgi to see it in action.

Appendix A. The Bugzilla FAQ

This FAQ includes questions not covered elsewhere in the Guide.

1. General Questions

1.1. Can I try out Bugzilla somewhere?

If you want to take a test ride, there are test installations at http://landfill.bugzilla.org/, ready to play with directly from your browser.

1.2. What license is Bugzilla distributed under?

Bugzilla is covered by the Mozilla Public License. See details at http://www.mozilla.org/MPL/.

1.3. How do I get commercial support for Bugzilla?

http://www.bugzilla.org/support/consulting.html is a list of companies and individuals who have asked us to list them as consultants for Bugzilla.

There are several experienced Bugzilla hackers on the mailing list/newsgroup who are willing to make themselves available for generous compensation. Try sending a message to the mailing list asking for a volunteer.

1.4. What major companies or projects are currently using Bugzilla for bug-tracking?

There are *dozens* of major companies with public Bugzilla sites to track bugs in their products. We have a fairly complete list available on our website at http://bugzilla.org/installation-list/. If you have an installation of Bugzilla and would like to be added to the list, whether it's a public install or not, simply e-mail Gerv <`gerv@mozilla.org`>.

1.5. Who maintains Bugzilla?

A core team (http://www.bugzilla.org/developers/profiles.html), led by Dave Miller (justdave@bugzilla.org).

1.6. How does Bugzilla stack up against other bug-tracking databases?

We can't find any head-to-head comparisons of Bugzilla against other defect-tracking software. If you know of one, please get in touch. In the experience of Matthew Barnson (the original author of this FAQ), though, Bugzilla offers superior performance on commodity hardware, better price (free!), more developer-friendly features (such as stored queries, email integration, and platform independence), improved scalability, greater flexibility, and superior ease-of-use when compared to commercial bug-tracking software.

If you happen to be a vendor for commercial bug-tracking software, and would like to submit a list of advantages your product has over Bugzilla, simply send it to <`documentation@bugzilla.org`> and we'd be happy to include the comparison in our documentation.

1.7. Why doesn't Bugzilla offer this or that feature or compatibility with this other tracking software?

It may be that the support has not been built yet, or that you have not yet found it. While Bugzilla makes strides in usability, customizability, scalability, and user interface with each release, that doesn't mean it can't still use improvement!

The best way to make an enhancement request is to file a bug at bugzilla.mozilla.org (https://bugzilla.mozilla.org/enter_bug.cgi?product=Bugzilla) and set the Severity to 'enhancement'. Your 'request for enhancement' (RFE) will start out in the UNCONFIRMED state, and will stay there until someone with the ability to CONFIRM the bug reviews it. If that person feels it to be a good request that fits in with Bugzilla's overall direction, the status will be changed to NEW; if not, they will probably explain why and set the bug to RESOLVED/WONTFIX. If someone else has made the same (or almost the same) request before, your request will be marked RESOLVED/DUPLICATE, and a pointer to the previous RFE will be added.

Even if your RFE gets approved, that doesn't mean it's going to make it right into the next release; there are a limited number of developers, and a whole lot of RFEs... some of which are *quite* complex. If you're a code-hacking sort of person, you can help the project along by making a patch yourself that supports the functionality you require. If you have never contributed anything to Bugzilla before, please be sure to read the Developers' Guide (http://www.bugzilla.org/docs/developer.html) and Contributors' Guide (http://www.bugzilla.org/docs/contributor.html) before going ahead.

1.8. What databases does Bugzilla run on?

MySQL is the default database for Bugzilla. It was originally chosen because it is free, easy to install, and was available for the hardware Netscape intended to run it on.

As of Bugzilla 2.22, complete support for PostgreSQL is included. With this release using PostgreSQL with Bugzilla should be as stable as using MySQL. If you experience any problems with PostgreSQL compatibility, they will be taken as seriously as if you were running MySQL.

There are plans to include an Oracle driver for Bugzilla 3.1.2. Track progress at Bug 189947 (https://bugzilla.mozilla.org/show_bug.cgi?id=189947).

Sybase support was worked on for a time. However, several complicating factors have prevented Sybase support from being realized. There are currently no plans to revive it.

Bug 237862 (https://bugzilla.mozilla.org/show_bug.cgi?id=237862) is a good bug to read through if you'd like to see what progress is being made on general database compatibility.

1.9. My perl is located at `/usr/local/bin/perl` and not `/usr/bin/perl`. Is there an easy to change that in all the files that have this hard-coded?

The easiest way to get around this is to create a link from one to the other: **ln -s /usr/local/bin/perl /usr/bin/perl**. If that's not an option for you, the following bit of perl magic will change all the shebang lines (that is to say, the line at the top of each file that starts with '#!' and contains the path) to something else:

```
perl -pi -e 's@#\!/usr/bin/perl@#\!/usr/local/bin/perl@' *cgi *pl
```

Sadly, this command-line won't work on Windows unless you also have Cygwin. However, MySQL comes with a binary called **replace** which can do the job:

```
C:\mysql\bin\replace "#!/usr/bin/perl" "#!C:\perl\bin\perl" -- *.cgi *.pl
```

> **Note:** If your perl path is something else again, just follow the above examples and replace `/usr/local/bin/perl` with your own perl path.

Once you've modified all your files, you'll also need to modify the `t/002goodperl.t` test, as it tests that all shebang lines are equal to `/usr/bin/perl`. (For more information on the test suite, please check out the appropriate section in the Developers' Guide (http://www.bugzilla.org/docs/developer.html#testsuite).) Having done this, run the test itself:

```
perl runtests.pl 2 --verbose
```

to ensure that you've modified all the relevant files.

If using Apache on Windows, you can avoid the whole problem by setting the ScriptInterpreterSource (http://httpd.apache.org/docs-2.0/mod/core.html#scriptinterpretersource) directive to 'Registry'. (If using Apache 2 or higher, set it to 'Registry-Strict'.) ScriptInterperterSource requires a registry entry "HKEY_CLASSES_ROOT\.cgi\Shell\ExecCGI\Command" to associate .cgi files with your perl executable. If one does not already exist, create it with a default value of "<full path to perl> -T", e.g. "C:\Perl\bin\perl.exe -T".

1.10. Is there an easy way to change the Bugzilla cookie name?

At present, no.

1.11. How can Bugzilla be made to work under SELinux?

As a web application, Bugzilla simply requires its root directory to have the httpd context applied for it to work properly under SELinux. This should happen automatically on distributions that use SELinux and that package Bugzilla (if it is installed with the native package management tools). Information on how to view and change SELinux file contexts can be found at the SELinux FAQ (http://docs.fedoraproject.org/selinux-faq-fc5/).

2. Managerial Questions

2.1. Is it possible to delete bug reports?

Yes. You have to turn on the 'allowbugdeletion' parameter, which is off by default. Note that you cannot delete bug reports one by one. You have to move them in a product or component, e.g. named "Trash", and then delete this product or component. The reason we make it hard is that you generally don't want to do that.

2.2. Is Bugzilla web-based, or do you have to have specific software or a specific operating system on your machine?

It is web and e-mail based.

2.3. Does Bugzilla allow us to define our own priorities and levels? Do we have complete freedom to change the labels of fields and format of them, and the choice of acceptable values?

Yes. However, modifying some fields, notably those related to bug progression states, also require adjusting the program logic to compensate for the change.

As of Bugzilla 3.0 custom fields can be created via the "Custom Fields" admin page.

2.4. Does Bugzilla provide any reporting features, metrics, graphs, etc? You know, the type of stuff that management likes to see. :)

Yes. Look at https://bugzilla.mozilla.org/report.cgi for samples of what Bugzilla can do in reporting and graphing. Fuller documentation is provided in Section 5.11.

If you can not get the reports you want from the included reporting scripts, it is possible to hook up a professional reporting package such as Crystal Reports using ODBC. If you choose to do this, beware that giving direct access to the database does contain some security implications. Even if you give read-only access to the bugs database it will bypass the secure bugs features of Bugzilla.

2.5. Is there email notification? If so, what do you see when you get an email?

Email notification is user-configurable. By default, the bug id and summary of the bug report accompany each email notification, along with a list of the changes made.

2.6. Do users have to have any particular type of email application?

Bugzilla email is sent in plain text, the most compatible mail format on the planet.

> **Note:** If you decide to use the bugzilla_email integration features to allow Bugzilla to record responses to mail with the associated bug, you may need to caution your users to set their mailer to "respond to messages in the format in which they were sent". For security reasons Bugzilla ignores HTML tags in comments, and if a user sends HTML-based email into Bugzilla the resulting comment looks downright awful.

2.7. Does Bugzilla allow data to be imported and exported? If I had outsiders write up a bug report using a MS Word bug template, could that template be imported into "matching" fields? If I wanted to take the results of a query and export that data to MS Excel, could I do that?

Bugzilla can output buglists as HTML (the default), CSV or RDF. The link for CSV can be found at the bottom of the buglist in HTML format. This CSV format can easily be imported into MS Excel or other spreadsheet applications.

To use the RDF format of the buglist it is necessary to append a `&ctype=rdf` to the URL. RDF is meant to be machine readable and thus it is assumed that the URL would be generated programmatically so there is no user visible link to this format.

Currently the only script included with Bugzilla that can import data is `importxml.pl` which is intended to be used for importing the data generated by the XML ctype of `show_bug.cgi` in association with bug moving. Any other use is left as an exercise for the user.

There are also scripts included in the `contrib/` directory for using e-mail to import information into Bugzilla, but these scripts are not currently supported and included for educational purposes.

2.8. Has anyone converted Bugzilla to another language to be used in other countries? Is it localizable?

Yes. For more information including available translated templates, see http://www.bugzilla.org/download.html#localizations. Some admin interfaces have been

templatized (for easy localization) but many of them are still available in English only. Also, there may be issues with the charset not being declared. See bug 126226 (https://bugzilla.mozilla.org/show_bug.cgi?id=126266) for more information.

2.9. Can a user create and save reports? Can they do this in Word format? Excel format?

Yes. No. Yes (using the CSV format).

2.10. Are there any backup features provided?

You should use the backup options supplied by your database platform. Vendor documentation for backing up a MySQL database can be found at http://www.mysql.com/doc/B/a/Backup.html. PostgreSQL backup documentation can be found at http://www.postgresql.org/docs/8.0/static/backup.html.

2.11. What type of human resources are needed to be on staff to install and maintain Bugzilla? Specifically, what type of skills does the person need to have? I need to find out what types of individuals would we need to hire and how much would that cost if we were to go with Bugzilla vs. buying an "out-of-the-box" solution.

If Bugzilla is set up correctly from the start, continuing maintenance needs are minimal and can be done easily using the web interface.

Commercial Bug-tracking software typically costs somewhere upwards of $20,000 or more for 5-10 floating licenses. Bugzilla consultation is available from skilled members of the newsgroup. Simple questions are answered there and then.

2.12. What time frame are we looking at if we decide to hire people to install and maintain the Bugzilla? Is this something that takes hours or days to install and a couple of hours per week to maintain and customize, or is this a multi-week install process, plus a full time job for 1 person, 2 people, etc?

It all depends on your level of commitment. Someone with much Bugzilla experience can get you up and running in less than a day, and your Bugzilla install can run untended for years. If your Bugzilla strategy is critical to your business workflow, hire somebody to who has reasonable Perl skills, and a familiarity with the operating system on which Bugzilla will be running, and have them handle your process management, bug-tracking maintenance, and local customization.

2.13. Is there any licensing fee or other fees for using Bugzilla? Any out-of-pocket cost other than the bodies needed as identified above?

No. Bugzilla, Perl, the Template Toolkit, and all other support software needed to make Bugzilla work can be downloaded for free. MySQL and PostgreSQL -- the databases supported by Bugzilla -- are also open-source. MySQL asks that if you find their product valuable, you purchase a support contract from them that suits your needs.

2.14. We don't like referring to problems as 'bugs'. Can we change that?

Yes! As of Bugzilla 2.18, it is a simple matter to change the word 'bug' into whatever word/phrase is used by your organization. See the documentation on Customization for more details, specifically Section 6.2.5.

3. Administrative Questions

3.1. Does Bugzilla provide record locking when there is simultaneous access to the same bug? Does the second person get a notice that the bug is in use or how are they notified?

Bugzilla does not lock records. It provides mid-air collision detection -- which means that it warns a user when a commit is about to conflict with commits recently made by another user, and offers the second user a choice of options to deal with the conflict.

3.2. Can users be on the system while a backup is in progress?

Refer to your database platform documentation for details on how to do hot backups. Vendor documentation for backing up a MySQL database can be found at http://www.mysql.com/doc/B/a/Backup.html. PostgreSQL backup documentation can be found at http://www.postgresql.org/docs/8.0/static/backup.html.

3.3. How can I update the code and the database using CVS?

1. Make a backup of both your Bugzilla directory and the database. For the Bugzilla directory this is as easy as doing **cp -rp bugzilla bugzilla.bak**. For the database, there's a number of options - see the MySQL docs and pick the one that fits you best (the easiest is to just make a physical copy of the database on the disk, but you have to have the database server shut down to do that without risking dataloss).
2. Make the Bugzilla directory your current directory.
3. Use **cvs -q update -AdP** if you want to update to the tip or **cvs -q update -dP -rTAGNAME** if you want a specific version (in that case you'll have to replace TAGNAME with a CVS tag name such as BUGZILLA-2_16_5).

   If you've made no local changes, this should be very clean. If you have made local changes, then watch the cvs output for C results. If you get any lines that start with a C it means there were conflicts between your local changes and what's in CVS. You'll need to fix those manually before continuing.

4. After resolving any conflicts that the cvs update operation generated, running **./checksetup.pl** will take care of updating the database for you as well as any other changes required for the new version to operate.

> **Warning**
>
> Once you run checksetup.pl, the only way to go back is to restore the database backups. You can't "downgrade" the system cleanly under most circumstances.

See also the instructions in Section 3.16.3.1.

3.4. How do I make it so that bugs can have an UNCONFIRMED status?

To use the UNCONFIRMED status, you must have the 'usevotes' parameter set to "On". You must then visit the `editproducts.cgi` page and set the " Number of votes a bug in this product needs to automatically get out of the UNCONFIRMED state" to be a non-zero number. (You will have to do this for each product that wants to use the UNCONFIRMED state.) If you do not actually want users to be able to vote for bugs entered against this product, leave the "Maximum votes per person" value at '0'.

There is work being done to decouple the UNCONFIRMED state from the 'usevotes' parameter for future versions of Bugzilla. Follow the discussion and progress at bug 162060 (https://bugzilla.mozilla.org/show_bug.cgi?id=162060).

3.5. How do I move a Bugzilla installation from one machine to another?

Reference your database vendor's documentation for information on backing up and restoring your Bugzilla database on to a different server. Vendor documentation for backing up a MySQL database can be found at http://dev.mysql.com/doc/mysql/en/mysqldump.html. PostgreSQL backup documentation can be found at http://www.postgresql.org/docs/8.0/static/backup.html.

On your new machine, follow the instructions found in Chapter 2 as far as setting up the physical environment of the new machine with perl, webserver, modules, etc. Having done that, you can either: copy your entire Bugzilla directory from the old machine to a new one (if you want to keep your existing code and modifications), or download a newer version (if you are planning to upgrade at the same time). Even if you are upgrading to clean code, you will still want to bring over the `localconfig` file, and the `data` directory from the old machine, as they contain configuration information that you probably won't want to re-create.

> **Note:** If the hostname or port number of your database server changed as part of the move, you'll need to update the appropriate variables in localconfig before taking the next step.

Once you have your code in place, and your database has been restored from the backup you made in step 1, run **checksetup.pl**. This will upgrade your database (if necessary), rebuild your templates, etc.

3.6. How do I make a new Bugzilla administrator? The previous administrator is gone...

Run **checksetup.pl** with `--make-admin` option. Its usage is `--make-admin=user@example.org`. The user account must be exist in the Bugzilla database.

4. Bugzilla Security

4.1. How do I completely disable MySQL security if it's giving me problems? (I've followed the instructions in the installation section of this guide...)

You can run MySQL like this: **mysqld --skip-grant-tables**. However, doing so disables all MySQL security. This is a bad idea. Please consult Section 4.2 of this guide and the MySQL documentation for better solutions.

4.2. Are there any security problems with Bugzilla?

The Bugzilla code has undergone a reasonably complete security audit, and user-facing CGIs run under Perl's taint mode. However, it is recommended that you closely examine permissions on your Bugzilla installation, and follow the recommended security guidelines found in The Bugzilla Guide.

5. Bugzilla Email

5.1. I have a user who doesn't want to receive any more email from Bugzilla. How do I stop it entirely for this user?

The user can stop Bugzilla from sending any mail by unchecking all boxes on the 'Edit prefs' -> 'Email settings' page. (As of 2.18,this is made easier by the addition of a 'Disable All Mail' button.) Alternately, you can add their email address to the `data/nomail` file (one email address per line). This will override their personal preferences, and they will never be sent mail again.

5.2. I'm evaluating/testing Bugzilla, and don't want it to send email to anyone but me. How do I do it?

To disable email, set the `mail_delivery_method` parameter to `none` (2.20 and later), or

```
$enableSendMail
```

parameter to '0' in either `BugMail.pm` (2.18 and later) or `processmail` (up to 2.16.x).

> **Note:** Up to 2.16.x, changing
>
> ```
> $enableSendMail
> ```
>
> will only affect bugmail; email related to password changes, email address changes, bug imports, flag changes, etc. will still be sent out. As of the final release of 2.18, however, the above step will disable *all* mail sent from Bugzilla for any purpose.

To have bugmail (and only bugmail) redirected to you instead of its intended recipients, leave

```
$enableSendMail
```

alone; instead, edit the "newchangedmail" parameter as follows:

- Replace "To:" with "X-Real-To:"
- Replace "Cc:" with "X-Real-CC:"
- Add a "To: %lt;your_email_address>"

5.3. I want whineatnews.pl to whine at something other than new and reopened bugs. How do I do it?

For older versions of Bugzilla, you may be able to apply Klaas Freitag's patch for "whineatassigned", which can be found in bug 6679 (https://bugzilla.mozilla.org/show_bug.cgi?id=6679). Note that this patch was made in 2000, so it may take some work to apply cleanly to any releases of Bugzilla newer than that, but you can use it as a starting point.

An updated (and much-expanded) version of this functionality is due to be released as part of Bugzilla 2.20; see bug 185090 (https://bugzilla.mozilla.org/show_bug.cgi?id=185090) for the discussion, and for more up-to-date patches if you just can't wait.

5.4. How do I set up the email interface to submit or change bugs via email?

Bugzilla 3.0 and later offers the ability submit or change bugs via email, using the `email_in.pl` script within the root directory of the Bugzilla installation. More information on the script can be found in docs/html/api/email_in.html (api/email_in.html).

5.5. Email takes FOREVER to reach me from Bugzilla -- it's extremely slow. What gives?

If you are using sendmail, try enabling `sendmailnow` in `editparams.cgi`. For earlier versions of sendmail, one could achieve significant performance improvement in the UI (at the cost of delaying the sending of mail) by setting this parameter to `off`. Sites with sendmail version 8.12 (or higher) should leave this `on`, as they will not see any performance benefit.

If you are using an alternate *MTA*, make sure the options given in `Bugzilla/BugMail.pm` and any other place where sendmail is called are correct for your MTA.

5.6. How come email from Bugzilla changes never reaches me?

Double-check that you have not turned off email in your user preferences. Confirm that Bugzilla is able to send email by visiting the "Log In" link of your Bugzilla installation and clicking the "Submit Request" button after entering your email address.

If you never receive mail from Bugzilla, chances are you do not have sendmail in "/usr/lib/sendmail". Ensure sendmail lives in, or is symlinked to, "/usr/lib/sendmail".

If you are using an MTA other than sendmail the `sendmailnow` param must be set to `on` or no mail will be sent.

6. Bugzilla Database

6.1. I think my database might be corrupted, or contain invalid entries. What do I do?

Run the "sanity check" utility (`sanitycheck.cgi`) from your web browser to see! If it finishes without errors, you're *probably* OK. If it doesn't come back OK (i.e. any red letters), there are certain things Bugzilla can recover from and certain things it can't. If it can't auto-recover, I hope you're familiar with mysqladmin commands or have installed another way to manage your database. Sanity Check, although it is a good basic check on your database integrity, by no means is a substitute for competent database administration and avoiding deletion of data. It is not exhaustive, and was created to do a basic check for the most common problems in Bugzilla databases.

6.2. I want to manually edit some entries in my database. How?

There is no facility in Bugzilla itself to do this. It's also generally not a smart thing to do if you don't know exactly what you're doing. If you understand SQL, though, you can use the **mysql** or **psql** command line utilities to manually insert, delete and modify table information. There are also more intuitive GUI clients available for both MySQL and PostgreSQL. For MySQL, we recommend phpMyAdmin (http://www.phpmyadmin.net/).

Remember, backups are your friend. Everyone makes mistakes, and it's nice to have a safety net in case you mess something up.

6.3. I think I've set up MySQL permissions correctly, but Bugzilla still can't connect.

Try running MySQL from its binary: **mysqld --skip-grant-tables**. This will allow you to completely rule out grant tables as the cause of your frustration. If this Bugzilla is able to connect at this point then you need to check that you have granted proper permission to the user password combo defined in `localconfig`.

> **Warning**
>
> Running MySQL with this command line option is very insecure and should only be done when not connected to the external network as a troubleshooting step. Please do not run your production database in this mode.

You may also be suffering from a client version mismatch:

MySQL 4.1 and up uses an authentication protocol based on a password hashing algorithm that is incompatible with that used by older clients. If you upgrade the server to 4.1, attempts to connect to it with an older client may fail with the following message:

```
shell> mysql
            Client does not support authentication protocol requested
            by server; consider upgrading MySQL client
```

To solve this problem, you should use one of the following approaches:

- Upgrade all client programs to use a 4.1.1 or newer client library.
- When connecting to the server with a pre-4.1 client program, use an account that still has a pre-4.1-style password.
- Reset the password to pre-4.1 style for each user that needs to use a pre-4.1 client program. This can be done using the SET PASSWORD statement and the OLD_PASSWORD() function:

```
mysql> SET PASSWORD FOR
    -> ' some_user '@' some_host ' = OLD_PASSWORD(' newpwd ');
```

6.4. How do I synchronize bug information among multiple different Bugzilla databases?

Well, you can synchronize or you can move bugs. Synchronization will only work one way -- you can create a read-only copy of the database at one site, and have it regularly updated at intervals from the main database.

MySQL has some synchronization features built-in to the latest releases. It would be great if someone looked into the possibilities there and provided a report to the newsgroup on how to effectively synchronize two Bugzilla installations.

If you simply need to transfer bugs from one Bugzilla to another, checkout the "move.pl" script in the Bugzilla distribution.

7. Can Bugzilla run on a Windows server?

7.1. What is the easiest way to run Bugzilla on Win32 (Win98+/NT/2K)?

Making Bugzilla work easily with Windows was one of the major goals of the 2.18 milestone. If the necessary components are in place (perl, a webserver, an MTA, etc.) then installation of Bugzilla on a Windows box should be no more difficult than on any other platform. As with any installation, we recommend that you carefully and completely follow the installation instructions in Section 2.5.1.

While doing so, don't forget to check out the very excellent guide to Installing Bugzilla on Microsoft Windows (http://www.bugzilla.org/docs/win32install.html) written by Byron Jones. Thanks, Byron!

7.2. Is there a "Bundle::Bugzilla" equivalent for Win32?

Not currently. Bundle::Bugzilla enormously simplifies Bugzilla installation on UNIX systems. If someone can volunteer to create a suitable PPM bundle for Win32, it would be appreciated.

7.3. CGI's are failing with a "something.cgi is not a valid Windows NT application" error. Why?

Depending on what Web server you are using, you will have to configure the Web server to treat *.cgi files as CGI scripts. In IIS, you do this by adding *.cgi to the App Mappings with the <path>\perl.exe %s %s as the executable.

Microsoft has some advice on this matter, as well:

> "Set application mappings. In the ISM, map the extension for the script file(s) to the executable for the script interpreter. For example, you might map the extension .py to Python.exe, the executable for the Python script interpreter. Note For the ActiveState Perl script interpreter, the extension '.pl' is associated with PerlIS.dll by default. If you want to change the association of .pl to perl.exe, you need to change the application mapping. In the mapping, you must add two percent (%) characters to the end of the pathname for perl.exe, as shown in this example: **c:\perl\bin\perl.exe %s %s**"

7.4. I'm having trouble with the perl modules for NT not being able to talk to the database.

Your modules may be outdated or inaccurate. Try:

1. Hitting http://www.activestate.com/ActivePerl
2. Download ActivePerl
3. Go to your prompt
4. Type 'ppm'
5. `PPM>` **install DBI DBD-mysql GD**

I reckon TimeDate comes with the activeperl. You can check the ActiveState site for packages for installation through PPM. http://www.activestate.com/Packages/.

8. Bugzilla Usage

8.1. How do I change my user name (email address) in Bugzilla?

You can change your email address from the Name and Password section in Preferences. You will be emailed at both the old and new addresses for confirmation. 'Administrative Policies' must have the 'allowemailchange' parameter set to "On".

8.2. The query page is very confusing. Isn't there a simpler way to query?

The interface was simplified by a UI designer for 2.16. Further suggestions for improvement are welcome, but we won't sacrifice power for simplicity.

As of 2.18, there is also a 'simpler' search available. At the top of the search page are two links; "Advanced Search" will take you to the familiar full-power/full-complexity search page. The "Find a Specific Bug" link will take you to a much-simplified page where you can pick a product and status (open,closed, or both), then enter words that appear in the bug you want to find. This search will scour the 'Summary' and 'Comment' fields, and return a list of bugs sorted so that the bugs with the most hits/matches are nearer to the top.

> **Note:** Matches in the Summary will 'trump' matches in comments, and bugs with summary-matches will be placed higher in the buglist -- even if a lower-ranked bug has more matches in the comments section.

Bugzilla uses a cookie to remember which version of the page you visited last, and brings that page up when you next do a search. The default page for new users (or after an upgrade) is the 'simple' search.

8.3. I'm confused by the behavior of the "Accept" button in the Show Bug form. Why doesn't it assign the bug to me when I accept it?

The current behavior is acceptable to bugzilla.mozilla.org and most users. If you want to change this behavior, though, you have your choice of patches:

Bug 35195 (https://bugzilla.mozilla.org/show_bug.cgi?id=35195) seeks to add an "...and accept the bug" checkbox to the UI. It
Bug 37613 (https://bugzilla.mozilla.org/show_bug.cgi?id=37613) also provides two patches (against Bugzilla 2.12): one to add

These patches are all somewhat dated now, and cannot be applied directly, but they are simple enough to provide a guide on how Bugzilla can be customized and updated to suit your needs.

8.4. I can't upload anything into the database via the "Create Attachment" link. What am I doing wrong?

The most likely cause is a very old browser or a browser that is incompatible with file upload via POST. Download the latest version of your favourite browser to handle uploads correctly.

8.5. How do I change a keyword in Bugzilla, once some bugs are using it?

In the Bugzilla administrator UI, edit the keyword and it will let you replace the old keyword name with a new one. This will cause a problem with the keyword cache; run **sanitycheck.cgi** to fix it.

8.6. Why can't I close bugs from the "Change Several Bugs at Once" page?

Simple answer; you can.

The logic behind the page checks every bug in the list to determine legal state changes, and then only shows you controls to do things that could apply to *every* bug on the list. The reason for this is that if you try to do

something illegal to a bug, the whole process will grind to a halt, and all changes after the failed one will *also* fail. Since that isn't a good outcome, the page doesn't even present you with the option.

In practical terms, that means that in order to mark multiple bugs as CLOSED, then every bug on the page has to be either RESOLVED or VERIFIED already; if this is not the case, then the option to close the bugs will not appear on the page.

The rationale is that if you pick one of the bugs that's not VERIFIED and try to CLOSE it, the bug change will fail miserably (thus killing any changes in the list after it while doing the bulk change) so it doesn't even give you the choice.

9. Bugzilla Hacking

9.1. What kind of style should I use for templatization?

Gerv and Myk suggest a 2-space indent, with embedded code sections on their own line, in line with outer tags. Like this:

```
<fred>
[% IF foo %]
  <bar>
  [% FOREACH x = barney %]
    <tr>
      <td>
        [% x %]
      </td>
    <tr>
  [% END %]
[% END %]
</fred>
```

Myk also recommends you turn on PRE_CHOMP in the template initialization to prevent bloating of HTML with unnecessary whitespace.

Please note that many have differing opinions on this subject, and the existing templates in Bugzilla espouse both this and a 4-space style. Either is acceptable; the above is preferred.

9.2. What bugs are in Bugzilla right now?

Try this link (https://bugzilla.mozilla.org/buglist.cgi?bug_status=NEW&bug_status=ASSIGNED&bug_status=REOPENED&pro
to view current bugs or requests for enhancement for Bugzilla.

You can view bugs marked for 3.2 release here (https://bugzilla.mozilla.org/buglist.cgi?product=Bugzilla&target_milestone=Bugzi
nextver;). This list includes bugs for the 3.2 release that have already been fixed and checked into CVS. Please consult the Bugzilla Project Page (http://www.bugzilla.org/) for details on how to check current sources out of CVS so you can have these bug fixes early!

9.3. How can I change the default priority to a null value? For instance, have the default priority be "---" instead of "P2"?

This is well-documented in bug 49862 (https://bugzilla.mozilla.org/show_bug.cgi?id=49862). Ultimately, it's as easy as adding the "---" priority field to your localconfig file in the appropriate area, re-running checksetup.pl, and then changing the default priority in your browser using **editparams.cgi**.

9.4. What's the best way to submit patches? What guidelines should I follow?

1. Enter a bug into bugzilla.mozilla.org for the "Bugzilla (https://bugzilla.mozilla.org/enter_bug.cgi?product=Bugzilla)" product.
2. Upload your patch as a unified diff (having used "diff -u" against the *current sources* checked out of CVS), or new source file by clicking "Create a new attachment" link on the bug page you've just created, and include any descriptions of database changes you may make, into the bug ID you submitted in step #1. Be sure and click the "Patch" checkbox to indicate the text you are sending is a patch!
3. Announce your patch and the associated URL (https://bugzilla.mozilla.org/show_bug.cgi?id=XXXXXX) for discussion in the newsgroup (mozilla.support.bugzilla). You'll get a really good, fairly immediate reaction to the implications of your patch, which will also give us an idea how well-received the change would be.
4. If it passes muster with minimal modification, the person to whom the bug is assigned in Bugzilla is responsible for seeing the patch is checked into CVS.
5. Bask in the glory of the fact that you helped write the most successful open-source bug-tracking software on the planet :)

Appendix B. Troubleshooting

This section gives solutions to common Bugzilla installation problems. If none of the section headings seems to match your problem, read the general advice.

B.1. General Advice

If you can't get `checksetup.pl` to run to completion, it normally explains what's wrong and how to fix it. If you can't work it out, or if it's being uncommunicative, post the errors in the mozilla.support.bugzilla (news://news.mozilla.org/mozilla.support.bugzilla) newsgroup.

If you have made it all the way through Section 2.1 (Installation) and Section 2.2 (Configuration) but accessing the Bugzilla URL doesn't work, the first thing to do is to check your webserver error log. For Apache, this is often located at `/etc/logs/httpd/error_log`. The error messages you see may be self-explanatory enough to enable you to diagnose and fix the problem. If not, see below for some commonly-encountered errors. If that doesn't help, post the errors to the newsgroup.

Bugzilla can also log all user-based errors (and many code-based errors) that occur, without polluting the web server error log. To enable Bugzilla error logging, create a file that Bugzilla can write to, named `errorlog`, in the Bugzilla `data` directory. Errors will be logged as they occur, and will include the type of the error, the IP address and username (if available) of the user who triggered the error, and the values of all environment variables; if a form was being submitted, the data in the form will also be included. To disable error logging, delete or rename the `errorlog` file.

B.2. The Apache webserver is not serving Bugzilla pages

After you have run **checksetup.pl** twice, run **testserver.pl http://yoursite.yourdomain/yoururl** to confirm that your webserver is configured properly for Bugzilla.

```
bash$ ./testserver.pl http://landfill.bugzilla.org/bugzilla-tip
TEST-OK Webserver is running under group id in $webservergroup.
TEST-OK Got ant picture.
TEST-OK Webserver is executing CGIs.
TEST-OK Webserver is preventing fetch of http://landfill.bugzilla.org/bugzilla-tip/localconfig.
```

B.3. I installed a Perl module, but `checksetup.pl` claims it's not installed!

This could be caused by one of two things:

1. You have two versions of Perl on your machine. You are installing modules into one, and Bugzilla is using the other. Rerun the CPAN commands (or manual compile) using the full path to Perl from the top of `checksetup.pl`. This will make sure you are installing the modules in the right place.

2. The permissions on your library directories are set incorrectly. They must, at the very least, be readable by the webserver user or group. It is recommended that they be world readable.

B.4. DBD::Sponge::db prepare failed

The following error message may appear due to a bug in DBD::mysql (over which the Bugzilla team have no control):

```
DBD::Sponge::db prepare failed: Cannot determine NUM_OF_FIELDS at D:/Perl/site/lib/DBD/mysql.pm lin
 SV = NULL(0x0) at 0x20fc444
 REFCNT = 1
 FLAGS = (PADBUSY,PADMY)
```

To fix this, go to `<path-to-perl>/lib/DBD/sponge.pm` in your Perl installation and replace

```
my $numFields;
if ($attribs->{'NUM_OF_FIELDS'}) {
    $numFields = $attribs->{'NUM_OF_FIELDS'};
} elsif ($attribs->{'NAME'}) {
    $numFields = @{$attribs->{NAME}};
```

with

```
my $numFields;
if ($attribs->{'NUM_OF_FIELDS'}) {
    $numFields = $attribs->{'NUM_OF_FIELDS'};
} elsif ($attribs->{'NAMES'}) {
    $numFields = @{$attribs->{NAMES}};
```

(note the S added to NAME.)

B.5. cannot chdir(/var/spool/mqueue)

If you are installing Bugzilla on SuSE Linux, or some other distributions with "paranoid" security options, it is possible that the checksetup.pl script may fail with the error:

```
cannot chdir(/var/spool/mqueue): Permission denied
```

This is because your `/var/spool/mqueue` directory has a mode of `drwx------`. Type **chmod 755 `/var/spool/mqueue`** as root to fix this problem. This will allow any process running on your machine the ability to *read* the `/var/spool/mqueue` directory.

B.6. Everybody is constantly being forced to relogin

The most-likely cause is that the "cookiepath" parameter is not set correctly in the Bugzilla configuration. You can change this (if you're a Bugzilla administrator) from the editparams.cgi page via the web.

The value of the cookiepath parameter should be the actual directory containing your Bugzilla installation, *as seen by the end-user's web browser*. Leading and trailing slashes are mandatory. You can also set the cookiepath to any directory which is a parent of the Bugzilla directory (such as '/', the root directory). But you can't put something that isn't at least a partial match or it won't work. What you're actually doing is restricting the end-user's browser to sending the cookies back only to that directory.

How do you know if you want your specific Bugzilla directory or the whole site?

If you have only one Bugzilla running on the server, and you don't mind having other applications on the same server with it being able to see the cookies (you might be doing this on purpose if you have other things on your site that share authentication with Bugzilla), then you'll want to have the cookiepath set to "/", or to a sufficiently-high enough directory that all of the involved apps can see the cookies.

Example B-1. Examples of urlbase/cookiepath pairs for sharing login cookies

```
urlbase is http://bugzilla.mozilla.org/
cookiepath is /

urlbase is http://tools.mysite.tld/bugzilla/
    but you have http://tools.mysite.tld/someotherapp/ which shares
    authentication with your Bugzilla
cookiepath is /
```

On the other hand, if you have more than one Bugzilla running on the server (some people do - we do on landfill) then you need to have the cookiepath restricted enough so that the different Bugzillas don't confuse their cookies with one another.

Example B-2. Examples of urlbase/cookiepath pairs to restrict the login cookie

```
urlbase is http://landfill.bugzilla.org/bugzilla-tip/
cookiepath is /bugzilla-tip/

urlbase is http://landfill.bugzilla.org/bugzilla-2.16-branch/
cookiepath is /bugzilla-2.16-branch/
```

If you had cookiepath set to "/" at any point in the past and need to set it to something more restrictive (i.e. "/bugzilla/"), you can safely do this without requiring users to delete their Bugzilla-related cookies in their browser (this is true starting with Bugzilla 2.18 and Bugzilla 2.16.5).

B.7. Some users are constantly being forced to relogin

First, make sure cookies are enabled in the user's browser.

If that doesn't fix the problem, it may be that the user's ISP implements a rotating proxy server. This causes the user's effective IP address (the address which the Bugzilla server perceives him coming from) to change periodically. Since Bugzilla cookies are tied to a specific IP address, each time the effective address changes, the user will have to log in again.

If you are using 2.18 (or later), there is a parameter called “loginnetmask”, which you can use to set the number of bits of the user’s IP address to require to be matched when authenticating the cookies. If you set this to something less than 32, then the user will be given a checkbox for “Restrict this login to my IP address” on the login screen, which defaults to checked. If they leave the box checked, Bugzilla will behave the same as it did before, requiring an exact match on their IP address to remain logged in. If they uncheck the box, then only the left side of their IP address (up to the number of bits you specified in the parameter) has to match to remain logged in.

B.8. `index.cgi` doesn’t show up unless specified in the URL

You probably need to set up your web server in such a way that it will serve the index.cgi page as an index page.

If you are using Apache, you can do this by adding `index.cgi` to the end of the `DirectoryIndex` line as mentioned in Section 2.2.4.1.

B.9. checksetup.pl reports "Client does not support authentication protocol requested by server..."

This error is occurring because you are using the new password encryption that comes with MySQL 4.1, while your `DBD::mysql` module was compiled against an older version of MySQL. If you recompile `DBD::mysql` against the current MySQL libraries (or just obtain a newer version of this module) then the error may go away.

If that does not fix the problem, or if you cannot recompile the existing module (e.g. you’re running Windows) and/or don’t want to replace it (e.g. you want to keep using a packaged version), then a workaround is available from the MySQL docs: http://dev.mysql.com/doc/mysql/en/Old_client.html

Appendix C. Contrib

There are a number of unofficial Bugzilla add-ons in the `$BUGZILLA_ROOT/contrib/` directory. This section documents them.

C.1. Command-line Search Interface

There are a suite of Unix utilities for searching Bugzilla from the command line. They live in the `contrib/cmdline` directory. There are three files - `query.conf`, `buglist` and `bugs`.

> **Warning**
>
> These files pre-date the templatization work done as part of the 2.16 release, and have not been updated.

`query.conf` contains the mapping from options to field names and comparison types. Quoted option names are "grepped" for, so it should be easy to edit this file. Comments (#) have no effect; you must make sure these lines do not contain any quoted "option".

`buglist` is a shell script that submits a Bugzilla query and writes the resulting HTML page to stdout. It supports both short options, (such as "-Afoo" or "-Rbar") and long options (such as "--assignedto=foo" or "--reporter=bar"). If the first character of an option is not "-", it is treated as if it were prefixed with "--default=".

The column list is taken from the COLUMNLIST environment variable. This is equivalent to the "Change Columns" option that is available when you list bugs in buglist.cgi. If you have already used Bugzilla, grep for COLUMNLIST in your cookies file to see your current COLUMNLIST setting.

`bugs` is a simple shell script which calls `buglist` and extracts the bug numbers from the output. Adding the prefix "http://bugzilla.mozilla.org/buglist.cgi?bug_id=" turns the bug list into a working link if any bugs are found. Counting bugs is easy. Pipe the results through **sed -e 's/,/ /g' | wc | awk '{printf $2 "\n"}'**

Akkana Peck says she has good results piping `buglist` output through **w3m -T text/html -dump**

C.2. Command-line 'Send Unsent Bug-mail' tool

Within the `contrib` directory exists a utility with the descriptive (if compact) name of `sendunsentbugmail.pl`. The purpose of this script is, simply, to send out any bug-related mail that should have been sent by now, but for one reason or another has not.

To accomplish this task, `sendunsentbugmail.pl` uses the same mechanism as the `sanitycheck.cgi` script; it scans through the entire database looking for bugs with changes that were made more than 30 minutes ago, but where there is no record of anyone related to that bug having been sent mail. Having compiled a list, it then uses the standard rules to determine who gets mail, and sends it out.

As the script runs, it indicates the bug for which it is currently sending mail; when it has finished, it gives a numerical count of how many mails were sent and how many people were excluded. (Individual user names are not recorded or displayed.) If the script produces no output, that means no unsent mail was detected.

Usage: move the sendunsentbugmail.pl script up into the main directory, ensure it has execute permission, and run it from the command line (or from a cron job) with no parameters.

Appendix D. Manual Installation of Perl Modules

D.1. Instructions

If you need to install Perl modules manually, here's how it's done. Download the module using the link given in the next section, and then apply this magic incantation, as root:

```
bash# tar -xzvf <module>.tar.gz
bash# cd <module>
bash# perl Makefile.PL
bash# make
bash# make test
bash# make install
```

> **Note:** In order to compile source code under Windows you will need to obtain a 'make' utility. The **nmake** utility provided with Microsoft Visual C++ may be used. As an alternative, there is a utility called **dmake** available from CPAN which is written entirely in Perl. The majority of the links given below, however, are to pre-compiled versions of the modules, which can be installed on Windows simply by issuing the following command once you have downloaded the PPD file (which may be packaged within a ZIP file):
>
> ```
> > ppm install <filename.ppd>
> ```

D.2. Download Locations

> **Note:** Running Bugzilla on Windows requires the use of ActiveState Perl 5.8.1 or higher. Some modules already exist in the core distribution of ActiveState Perl so no PPM link is given. (This is noted where it occurs.)

CGI:

CPAN Download Page: http://search.cpan.org/dist/CGI.pm/
PPM Download Link: Part of core distribution.
Documentation: http://perldoc.perl.org/CGI.html

Data-Dumper:

CPAN Download Page: http://search.cpan.org/dist/Data-Dumper/
PPM Download Page: Part of core distribution.
Documentation: http://search.cpan.org/dist/Data-Dumper/Dumper.pm

Date::Format (part of TimeDate):

CPAN Download Page: http://search.cpan.org/dist/TimeDate/
PPM Download Link: http://landfill.bugzilla.org/ppm/TimeDate.ppd
Documentation: http://search.cpan.org/dist/TimeDate/lib/Date/Format.pm

DBI:

CPAN Download Page: http://search.cpan.org/dist/DBI/
PPM Download Link: http://landfill.bugzilla.org/ppm/DBI.ppd
Documentation: http://dbi.perl.org/docs/

DBD::mysql:

CPAN Download Page: http://search.cpan.org/dist/DBD-mysql/
PPM Download Link: http://landfill.bugzilla.org/ppm/DBD-mysql.ppd
Documentation: http://search.cpan.org/dist/DBD-mysql/lib/DBD/mysql.pm

DBD::Pg:

CPAN Download Page: http://search.cpan.org/dist/DBD-Pg/
PPM Download Link: http://theoryx5.uwinnipeg.ca/ppms/x86/DBD-Pg.tar.gz
Documentation: http://search.cpan.org/dist/DBD-Pg/Pg.pm

File::Spec:

CPAN Download Page: http://search.cpan.org/dist/File-Spec/
PPM Download Page: Part of core distribution.
Documentation: http://perldoc.perl.org/File/Spec.html

Template-Toolkit:

CPAN Download Page: http://search.cpan.org/dist/Template-Toolkit/
PPM Download Link: http://landfill.bugzilla.org/ppm/Template-Toolkit.ppd
Documentation: http://www.template-toolkit.org/docs.html

GD:

CPAN Download Page: http://search.cpan.org/dist/GD/
PPM Download Link: http://landfill.bugzilla.org/ppm/GD.ppd
Documentation: http://search.cpan.org/dist/GD/GD.pm

Template::Plugin::GD:

CPAN Download Page: http://search.cpan.org/dist/Template-GD/
PPM Download Link: (Just install Template-Toolkit using the instructions below)
Documentation: http://www.template-toolkit.org/docs/aqua/Modules/index.html

MIME::Parser (part of MIME-tools):

CPAN Download Page: http://search.cpan.org/dist/MIME-tools/
PPM Download Link: http://ppm.activestate.com/PPMPackages/zips/8xx-builds-only/Windows/MIME-tools-5.411a.zip
Documentation: http://search.cpan.org/dist/MIME-tools/lib/MIME/Parser.pm

D.3. Optional Modules

Chart::Base:

CPAN Download Page: http://search.cpan.org/dist/Chart/
PPM Download Page: http://landfill.bugzilla.org/ppm/Chart.ppd
Documentation: http://search.cpan.org/dist/Chart/Chart.pod

GD::Graph:

CPAN Download Page: http://search.cpan.org/dist/GDGraph/
PPM Download Link: http://landfill.bugzilla.org/ppm/GDGraph.ppd

Documentation: http://search.cpan.org/dist/GDGraph/Graph.pm

GD::Text::Align (part of GD::Text::Util):

CPAN Download Page: http://search.cpan.org/dist/GDTextUtil/
PPM Download Page: http://landfill.bugzilla.org/ppm/GDTextUtil.ppd
Documentation: http://search.cpan.org/dist/GDTextUtil/Text/Align.pm

XML::Twig:

CPAN Download Page: http://search.cpan.org/dist/XML-Twig/
PPM Download Link: http://ppm.activestate.com/PPMPackages/zips/8xx-builds-only/Windows/XML-Twig-3.22.zip
Documentation: http://standards.ieee.org/resources/spasystem/twig/twig_stable.html

PatchReader:

CPAN Download Page: http://search.cpan.org/author/JKEISER/PatchReader/
PPM Download Link: http://landfill.bugzilla.org/ppm/PatchReader.ppd
Documentation: http://www.johnkeiser.com/mozilla/Patch_Viewer.html

Image::Magick:

CPAN Download Page: http://search.cpan.org/dist/PerlMagick/
PPM Download Link: Included in Windows binary package.
Documentation: http://www.imagemagick.org/script/resources.php

Appendix E. GNU Free Documentation License

Version 1.1, March 2000

Copyright (C) 2000 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. Preamble

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. Applicability and Definition

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available draw-

ing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. Verbatim Copying

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. Copying in Quantity

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. Modifications

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as

invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. Combining Documents

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. Collections of Documents

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. Aggregation with Independent Works

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document,

provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. Translation

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. Termination

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. Future Revisions of this License

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with

> the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Glossary

0-9, high ascii

.htaccess

Apache web server, and other NCSA-compliant web servers, observe the convention of using files in directories called `.htaccess` to restrict access to certain files. In Bugzilla, they are used to keep secret files which would otherwise compromise your installation - e.g. the `localconfig` file contains the password to your database. curious.

A

Apache

In this context, Apache is the web server most commonly used for serving up Bugzilla pages. Contrary to popular belief, the apache web server has nothing to do with the ancient and noble Native American tribe, but instead derived its name from the fact that it was "a patchy" version of the original NCSA world-wide-web server.

Useful Directives when configuring Bugzilla

`AddHandler (http://httpd.apache.org/docs-2.0/mod/core.html#addhandler)`

Tell Apache that it's OK to run CGI scripts.

`AllowOverride (http://httpd.apache.org/docs-2.0/mod/core.html#allowoverride)`
`Options (http://httpd.apache.org/docs-2.0/mod/core.html#options)`

These directives are used to tell Apache many things about the directory they apply to. For Bugzilla's purposes, we need them to allow script execution and `.htaccess` overrides.

`DirectoryIndex (http://httpd.apache.org/docs-2.0/mod/mod_dir.html#directoryindex)`

Used to tell Apache what files are indexes. If you can not add `index.cgi` to the list of valid files, you'll need to set `$index_html` to 1 in `localconfig` so **./checksetup.pl** will create an `index.html` that redirects to `index.cgi`.

`ScriptInterpreterSource (http://httpd.apache.org/docs-2.0/mod/core.html#scriptinterpretersource)`

Used when running Apache on windows so the shebang line doesn't have to be changed in every Bugzilla script.

For more information about how to configure Apache for Bugzilla, see Section 2.2.4.1.

B

Bug

A “bug” in Bugzilla refers to an issue entered into the database which has an associated number, assignments, comments, etc. Some also refer to a “tickets” or “issues”; in the context of Bugzilla, they are synonymous.

Bug Number

Each Bugzilla bug is assigned a number that uniquely identifies that bug. The bug associated with a bug number can be pulled up via a query, or easily from the very front page by typing the number in the "Find" box.

Bugzilla

Bugzilla is the world-leading free software bug tracking system.

C

Common Gateway Interface

CGI is an acronym for Common Gateway Interface. This is a standard for interfacing an external application with a web server. Bugzilla is an example of a CGI application.

Component

A Component is a subsection of a Product. It should be a narrow category, tailored to your organization. All Products must contain at least one Component (and, as a matter of fact, creating a Product with no Components will create an error in Bugzilla).

Comprehensive Perl Archive Network

CPAN stands for the “Comprehensive Perl Archive Network”. CPAN maintains a large number of extremely useful *Perl* modules - encapsulated chunks of code for performing a particular task.

`contrib`

The `contrib` directory is a location to put scripts that have been contributed to Bugzilla but are not a part of the official distribution. These scripts are written by third parties and may be in languages other than perl. For those

that are in perl, there may be additional modules or other requirements than those of the official distribution.

> **Note:** Scripts in the `contrib` directory are not officially supported by the Bugzilla team and may break in between versions.

D

daemon

A daemon is a computer program which runs in the background. In general, most daemons are started at boot time via System V init scripts, or through RC scripts on BSD-based systems. *mysqld*, the MySQL server, and *apache*, a web server, are generally run as daemons.

DOS Attack

A DOS, or Denial of Service attack, is when a user attempts to deny access to a web server by repeatedly accessing a page or sending malformed requests to a webserver. A D-DOS, or Distributed Denial of Service attack, is when these requests come from multiple sources at the same time. Unfortunately, these are much more difficult to defend against.

G

Groups

The word "Groups" has a very special meaning to Bugzilla. Bugzilla's main security mechanism comes by placing users in groups, and assigning those groups certain privileges to view bugs in particular *Products* in the *Bugzilla* database.

J

JavaScript

JavaScript is cool, we should talk about it.

M

Message Transport Agent

A Message Transport Agent is used to control the flow of email on a system. The Email::Send (http://search.cpan.org/dist/Email-Send/lib/Email/Send.pm) Perl module, which Bugzilla uses to send email, can be configured to use many different underlying implementations for actually sending the mail using the `mail_delivery_method` parameter. Implementations other than `sendmail` require that the `sendmailnow` param be set to `on`.

MySQL

MySQL is currently the required *RDBMS* for Bugzilla. MySQL can be downloaded from http://www.mysql.com. While you should familiarize yourself with all of the documentation, some high points are:

Backup (http://www.mysql.com/doc/en/Backup.html)

Methods for backing up your Bugzilla database.

Option Files (http://www.mysql.com/doc/en/Option_files.html)

Information about how to configure MySQL using `my.cnf`.

Privilege System (http://www.mysql.com/doc/en/Privilege_system.html)

Much more detailed information about the suggestions in Section 4.2.

P

Perl Package Manager

http://aspn.activestate.com/ASPN/Downloads/ActivePerl/PPM/

Product

A Product is a broad category of types of bugs, normally representing a single piece of software or entity. In general, there are several Components to a Product. A Product may define a group (used for security) for all bugs entered into its Components.

Perl

First written by Larry Wall, Perl is a remarkable program language. It has the benefits of the flexibility of an interpreted scripting language (such as shell script), combined with the speed and power of a compiled language, such as C. *Bugzilla* is maintained in Perl.

Q

QA

"QA", "Q/A", and "Q.A." are short for "Quality Assurance". In most large software development organizations, there is a team devoted to ensuring the product meets minimum standards before shipping. This team will also generally want to track the progress of bugs over their life cycle, thus the need for the "QA Contact" field in a bug.

R

Relational DataBase Management System

A relational database management system is a database system that stores information in tables that are related to each other.

Regular Expression

A regular expression is an expression used for pattern matching. Documentation (http://perldoc.com/perl5.6/pod/perlre.html#Regular-Expressions)

S

Service

In Windows NT environment, a boot-time background application is referred to as a service. These are generally managed through the control panel while logged in as an account with "Administrator" level capabilities. For more information, consult your Windows manual or the MSKB.

SGML

SGML stands for “Standard Generalized Markup Language”. Created in the 1980’s to provide an extensible means to maintain documentation based upon content instead of presentation, SGML has withstood the test of time as a robust, powerful language. *XML* is the “baby brother” of SGML; any valid XML document it, by definition, a valid SGML document. The document you are reading is written and maintained in SGML, and is also valid XML if you modify the Document Type Definition.

T

Target Milestone

Target Milestones are Product goals. They are configurable on a per-Product basis. Most software development houses have a concept of “milestones” where the people funding a project expect certain functionality on certain dates. Bugzilla facilitates meeting these milestones by giving you the ability to declare by which milestone a bug will be fixed, or an enhancement will be implemented.

Tool Command Language

TCL is an open source scripting language available for Windows, Macintosh, and Unix based systems. Bugzilla 1.0 was written in TCL but never released. The first release of Bugzilla was 2.0, which was when it was ported to perl.

Z

Zarro Boogs Found

This is just a goofy way of saying that there were no bugs found matching your query. When asked to explain this message, Terry had the following to say:

> I’ve been asked to explain this ... way back when, when Netscape released version 4.0 of its browser, we had a release party. Naturally, there had been a big push to try and fix every known bug before the release. Naturally, that hadn’t actually happened. (This is not unique to Netscape or to 4.0; the same thing has happened with every software project I’ve ever seen.) Anyway, at the release party, T-shirts were handed out that said something like "Netscape 4.0: Zarro Boogs". Just like the software, the T-shirt had no known bugs. Uh-huh.
>
> So, when you query for a list of bugs, and it gets no results, you can think of this as a friendly reminder. Of *course* there are bugs matching your query, they just aren’t in the bugsystem yet...
>
> —Terry Weissman