Upgrade Instructions

Table of Contents

Instructions

Typically, there are seven steps involved in upgrading to a newer version of Trac:

1. Bring your server off-line

It is not a good idea to update a running server: the server processes may have parts of the current packages cached in memory, and updating the code will likely trigger internal errors.

2. Update the Trac Code

Get the new version as described in TracInstall, or your operating system specific procedure.

If you already have a 0.12 version of Trac installed via `easy_install`, it might be easiest to also use `easy_install` to upgrade your Trac installation:

```
# easy_install --upgrade Trac==1.0
```

If you do a manual (not operating system-specific) upgrade, you should also stop any running Trac servers before the installation. Doing "hot" upgrades is not advised, especially on Windows (■#7265).

You may also want to remove the pre-existing Trac code by deleting the `trac` directory from the Python `lib/site-packages` directory, or remove Trac `.egg` files from former versions. The location of the site-packages directory depends on the operating system and the location in which Python was installed. However, the following locations are typical:

- on Linux: `/usr/lib/python2.X/site-packages`
- on Windows: `C:\Python2.X\lib\site-packages`
- on MacOSX: `/Library/Python/2.X/site-packages`

You may also want to remove the Trac `cgi-bin`, `htdocs`, `templates` and `wiki-default` directories that are commonly found in a directory called `share/trac`. (The exact location depends on your platform.)

This cleanup is not mandatory, but makes it easier to troubleshoot issues later on, as you won't waste your time looking at code or templates from a previous release that are not being used anymore... As usual, make a backup before actually deleting things.

3. Upgrade the Trac Environment

Environment upgrades are not necessary for minor version releases unless otherwise noted.

After restarting, Trac should show the instances which need a manual upgrade via the automated upgrade scripts to ease the pain. These scripts are run via trac-admin:

```
trac-admin /path/to/projenv upgrade
```

This command will do nothing if the environment is already up-to-date.

Note that a backup of your database will be performed automatically prior to the upgrade. This feature is relatively new for the PostgreSQL or MySQL database backends, so if it fails, you will have to backup the database manually. Then, to perform the actual upgrade, run:

```
trac-admin /path/to/projenv upgrade --no-backup
```

4. Update the Trac Documentation

Every Trac environment includes a copy of the Trac documentation for the installed version. As you probably want to keep the included documentation in sync with the installed version of Trac, trac-admin provides a command to upgrade the documentation:

```
trac-admin /path/to/projenv wiki upgrade
```

Note that this procedure will leave your `WikiStart` page intact.

5. Refresh static resources

If you have set up a web server to give out static resources directly (accessed using the `/chrome/` URL) then you will need to refresh them using the same command:

```
trac-admin /path/to/env deploy /deploy/path
```

this will extract static resources and CGI scripts (`trac.wsgi`, etc) from new Trac version and its plugins into `/deploy/path`.

Some web browsers (IE, Opera) cache CSS and Javascript files aggressively, so you may need to instruct your users to manually erase the contents of their browser's cache, a forced refreshed (<F5>) should be enough.

6. Steps specific to a given Trac version

Upgrading from Trac 0.12 to Trac 1.0

Python 2.4 no longer supported

The minimum supported version of python is now 2.5

Subversion components not enabled by default for new installations

The Trac components for Subversion support are no longer enabled by default. To enable the svn support, you need to make sure the `tracopt.versioncontrol.svn` components are enabled, for example by setting the following in the TracIni:

```
[components]
tracopt.versioncontrol.svn.* = enabled
```

The upgrade procedure should take care of this and change the TracIni appropriately, unless you already had the svn components explicitly disabled.

Attachments migrated to new location

Another step in the automatic upgrade will change the way the attachments are stored. If you're a bit paranoid, you might want to take a backup of the `attachments` directory before upgrading (but if you are, you already did a full copy of the environment, no?). In case the `attachments` directory contains some files which are *not* attachments, the last step of the migration to the new layout will fail: the deletion of the now unused `attachments` directory can't be done if there are still files and folders in it. You may ignore this error, but better go have a look to these files, move them elsewhere and remove the `attachments` directory manually to cleanup the environment. The attachments themselves are now all located in your environment below the `files/attachments` directory.

Behavior of `[ticket] default_owner` changed

Prior to 1.0, the owner field of new tickets always defaulted to `[ticket] default_owner` when the value was not empty. If the value was empty, the owner field defaulted to to the Component's owner. In 1.0 and later, the `default_owner` must be set to `< default >` to make new tickets default to the Component's owner. This change allows the `default_owner` to be set to an empty value if no default owner is desired.

Upgrading from Trac 0.11 to Trac 0.12

Python 2.3 no longer supported

The minimum supported version of python is now 2.4

SQLite v3.x required

SQLite v2.x is no longer supported. If you still use a Trac database of this format, you'll need to convert it to SQLite v3.x first. See ■PySqlite#UpgradingSQLitefrom2.xto3.x for details.

PySqlite? 2 required

PySqlite? 1.1.x is no longer supported. Please install 2.5.5 or later if possible (see Trac database upgrade below).

Multiple Repository Support

The latest version includes support for multiple repositories. If you plan to add more repositories to your Trac instance, please refer to TracRepositoryAdmin#Migration.

This may be of interest to users with only one repository, since there's now a way to avoid the potentially costly resync check at every request.

Resynchronize the Trac Environment Against the Source Code Repository

Each Trac environment must be resynchronized against the source code repository in order to avoid errors such as "■No changeset ??? in the repository" while browsing the source through the Trac interface:

```
trac-admin /path/to/projenv repository resync '*'
```

Improved repository synchronization

In addition to supporting multiple repositories, there is now a more efficient method for synchronizing Trac and your repositories.

While you can keep the same synchronization as in 0.11 adding the post-commit hook as outlined in TracRepositoryAdmin#Synchronization and TracRepositoryAdmin#ExplicitSync will allow more efficient synchronization and is more or less required for multiple repositories.

Note that if you were using the `trac-post-commit-hook`, *you're strongly advised to upgrade it* to the new hook documented in the above references and here, as the old hook will not work with anything else than the default repository and even for this case, it won't trigger the appropriate notifications.

Authz permission checking

The authz permission checking has been migrated to a fine-grained permission policy. If you use authz permissions (aka `[trac] authz_file` and `authz_module_name`), you must add `AuthzSourcePolicy` in front of your permission policies in `[trac] permission_policies`. You must also remove `BROWSER_VIEW`, `CHANGESET_VIEW`, `FILE_VIEW` and `LOG_VIEW` from your global permissions (with `trac-admin $ENV permission remove` or the "Permissions" admin panel).

Microsecond timestamps

All timestamps in database tables (except the `session` table) have been changed from "seconds since epoch" to "microseconds since epoch" values. This change should be transparent to most users, except for custom reports. If any of your reports use date/time columns in calculations (e.g. to pass them to `datetime()`), you must divide the values retrieved from the database by 1'000'000. Similarly, if a report provides a calculated value to be displayed as a date/time (i.e. with a column named "time", "datetime", "changetime", "date", "created" or "modified"), you must provide a microsecond timestamp, that is, multiply your previous calculation with 1'000'000.

Upgrading from Trac 0.10 to Trac 0.11

Site Templates and Styles

The templating engine has changed in 0.11 to Genshi, please look at TracInterfaceCustomization for more information.

If you are using custom CSS styles or modified templates in the `templates` directory of the TracEnvironment, you will need to convert them to the Genshi way of doing things. To continue to use your style sheet, follow the instructions at TracInterfaceCustomization#SiteAppearance.

Trac Macros, Plugins

The Trac macros will need to be adapted, as the old-style wiki-macros are not supported anymore (due to the drop of ■ClearSilver and the HDF); they need to be converted to the new-style macros, see WikiMacros. When they are converted to the new style, they need to be placed into the plugins directory instead and not wiki-macros, which is no longer scanned for macros or plugins.

For FCGI/WSGI/CGI users

For those who run Trac under the CGI environment, run this command in order to obtain the trac.*gi file:

```
trac-admin /path/to/env deploy /deploy/directory/path
```

This will create a deploy directory with the following two subdirectories: `cgi-bin` and `htdocs`. Then update your Apache configuration file `httpd.conf` with this new `trac.cgi` location and `htdocs` location.

Web Admin plugin integrated

If you had the webadmin plugin installed, you can uninstall it as it is part of the Trac code base since 0.11.

7. Restart the Web Server

If you are not running CGI, reload the new Trac code by restarting your web server.

Known Issues

Things you should pay attention to, while upgrading.

Customized Templates

Trac supports customization of its Genshi templates by placing copies of the templates in the `<env>/templates` folder of your environment or in a common location specified in the [inherit] templates_dir configuration setting. If you choose to do so, be wary that you will need to repeat your changes manually on a copy of the new templates when you upgrade to a new release of Trac (even a minor one), as the templates will likely evolve. So keep a diff around ;-)

The preferred way to perform TracInterfaceCustomization is to write a custom plugin doing an appropriate `ITemplateStreamFilter` transformation, as this is more robust in case of changes: we usually won't modify element `id`s or change CSS `class`es, and if we have to do so, this will be documented in the TracDev/ApiChanges? pages.

ZipImportError

Due to internal caching of zipped packages, whenever the content of the packages change on disk, the in-memory zip index will no longer match and you'll get irrecoverable ZipImportError errors. Better anticipate and bring your server down for maintenance before upgrading. See ■#7014 for details.

Wiki Upgrade

`trac-admin` will not delete or remove default wiki pages that were present in a previous version but are no longer in the new version.

Trac database upgrade

A known issue in some versions of PySqlite? (2.5.2-2.5.4) prevents the trac-admin upgrade script from successfully upgrading the database format. It is advised to use either a newer or older version of the sqlite python bindings to avoid this error. For more details see ticket ■#9434.

parent dir

If you use a trac parent env configuration and one of the plugins in one child does not work, none of the children work.

Related topics

Upgrading Python

Upgrading Python to a newer version will require reinstallation of Python packages: Trac of course; also ■easy_install, if you've been using that. Assuming you're using Subversion, you'll also need to upgrade the Python bindings for svn.

Windows and Python 2.6

If you've been using CollabNet's Subversion package, you may need to uninstall that in favor of ■Alagazam, which has the Python bindings readily available (see TracSubversion?). The good news is, that works with no tweaking.

Changing Database Backend

The ■TracMigratePlugin on ■trac-hacks.org has been written to assist in migrating between SQLite, MySQL and PostgreSQL databases.

Upgrading from older versions of Trac

For upgrades from versions older than Trac 0.10, refer first to ■wiki:0.10/TracUpgrade#SpecificVersions.

See also: TracGuide, TracInstall