The Trac Environment

Error: Macro TracGuideToc(None) failed

```
'NoneType' object has no attribute 'find'
```

Contents

Trac uses a directory structure and a database for storing project data. The directory is referred to as the **environment**.

Trac supports ?SQLite, ?PostgreSQL and ?MySQL databases. With PostgreSQL and MySQL you have to create the database before running `trac-admin initenv`.

Creating an Environment

A new Trac environment is created using the initenv command:

```
$ trac-admin /path/to/myproject initenv
```

`trac-admin` will ask you for the name of the project and the database connection string.

Useful Tips

- Place your environment's directory on a filesystem which supports sub-second timestamps, as Trac monitors the timestamp of its configuration files and changes happening on a filesystem with too coarse-grained timestamp resolution may go undetected in Trac < 1.0.2. This is also true for the location of authentication files when using TracStandalone.

- The user under which the web server runs will require file system write permission to the environment directory and all the files inside. Please remember to set the appropriate permissions. The same applies to the source code repository, although the user under which Trac runs will only require write access to a Subversion repository created with the BDB file system; for other repository types, check the corresponding plugin's documentation.

- `initenv` does not create a version control repository for the specified path. If you wish to specify a default repository using the optional arguments to `initenv` you must create the repository first, otherwise you will see a message when initializing the environment: *Warning: couldn't index the default repository*.

- Non-ascii environment paths are not supported.

- TracPlugins located in a shared plugins directory that is defined in an inherited configuration are not enabled by default, in contrast to plugins in the environment `plugins` directory. Hence, if they need to create extra tables, for example, the tables will not be created during environment creation and you'll need to upgrade the environment. Alternatively you can avoid the need to upgrade the environment by explicitly enabling the plugin in the inherited configuration, or in a configuration file using the `--config` option. See TracAdmin#FullCommandReference for more information.

Caveat: don't confuse the *Trac environment directory* with the *source code repository directory*.

This is a common beginners' mistake. It happens that the structure for a Trac environment is loosely modeled after the Subversion repository directory structure, but those are two disjoint entities and they are not and *must not* be located at the same place.

Database Connection Strings

You will need to specify a database connection string at the time the environment is created. The default is SQLite, which is sufficient for most projects. The SQLite database file is stored in the environment directory, and can easily be backed up together with the rest of the environment.

Note that if the username or password of the connection string (if applicable) contains the `:`, `/` or `@` characters, they need to be ?URL encoded.

```
$ python -c "import urllib; print urllib.quote('password@:/123', '')"
password%40%3A%2F123
```

SQLite Connection String

The connection string for an SQLite database is:

```
sqlite:db/trac.db
```

where `db/trac.db` is the path to the database file within the Trac environment.

See ?DatabaseBackend#SQLite for more information.

PostgreSQL Connection String

The connection string for PostgreSQL is a bit more complex. For example, to connect to a PostgreSQL database named `trac` on `localhost` for user `johndoe` and password `letmein`, use:

```
postgres://johndoe:letmein@localhost/trac
```

If PostgreSQL is running on a non-standard port, for example 9342, use:

```
postgres://johndoe:letmein@localhost:9342/trac
```

On UNIX, you might want to select a UNIX socket for the transport, either the default socket as defined by the PGHOST environment variable:

```
postgres://user:password@/database
```

or a specific one:

```
postgres://user:password@/database?host=/path/to/socket/dir
```

See the ?PostgreSQL documentation for detailed instructions on how to administer ?PostgreSQL. Generally, the following is sufficient to create a database user named `tracuser` and a database named `trac`:

```
$ createuser -U postgres -E -P tracuser
$ createdb -U postgres -O tracuser -E UTF8 trac
```

When running `createuser` you will be prompted for the password for the `tracuser`. This new user will not be a superuser, will not be allowed to create other databases and will not be allowed to create other roles. These privileges are not needed to run a Trac instance. If no password is desired for the user, simply remove the `-P` and `-E` options from the `createuser` command. Also note that the database should be created as UTF8. LATIN1 encoding causes errors, because of Trac's use of unicode.

Under some default configurations (Debian), run the `createuser` and `createdb` scripts as the `postgres` user:

```
$ sudo su - postgres -c 'createuser -U postgres -S -D -R -E -P tracuser'
$ sudo su - postgres -c 'createdb -U postgres -O tracuser -E UTF8 trac'
```

Trac uses the `public` schema by default, but you can specify a different schema in the connection string:

```
postgres://user:pass@server/database?schema=yourschemaname
```

MySQL Connection String

The format of the MySQL connection string is similar to PostgreSQL, with the `postgres` scheme being replaced by `mysql`. For example, to connect to a MySQL database on `localhost` named `trac` for user `johndoe` with password `letmein`:

```
mysql://johndoe:letmein@localhost:3306/trac
```

Source Code Repository

A single environment can be connected to more than one repository. However, by default Trac is not connected to any source code repository, and the *Browse Source* navigation item will not be displayed.

There are several ways to connect repositories to an environment, see TracRepositoryAdmin. A single repository can be specified when the environment is created by passing the optional arguments `repository_type` and `repository_dir` to the `initenv` command.

Directory Structure

An environment consists of the following files and directories:

- `README` - Brief description of the environment.
- `VERSION` - Environment version identifier.
- `files`

    - ♦ `attachments` - Attachments to wiki pages and tickets.
- `conf`
    - ♦ `trac.ini` - Main configuration file.
- `db`
    - ♦ `trac.db` - The SQLite database, if you are using SQLite.
- `htdocs` - Directory containing web resources, which can be referenced in templates using the path `/chrome/site/...`.
- `log` - Default directory for log files when `file` logging is enabled and a relative path is given.
- `plugins` - Environment-specific plugins.
- `templates` - Custom Genshi environment-specific templates.
    - ♦ `site.html` - Method to customize the site header, footer, and style.

See also: TracAdmin, TracBackup, TracIni