

Wikiprint Book

Title: Trac Environment

Subject: SilverFrost - TracEnvironment

Version: 3

Date: 06/04/26 06:07:33

## SilverFrost 目次

Trac Environment	3
Environment の作成	3
Tips	3
データベースに接続するための文字列	3
SQLite の接続文字列	3
PostgreSQL の接続文字列	3
MySQL の接続文字列	4
ソースコードリポジトリ	4
ディレクトリ構造	5

## Trac Environment

Trac はプロジェクトのデータを保存するために、ディレクトリ構造とデータベースを使用します。このディレクトリを "Environment" と呼びます。

### Environment の作成

新しい Trac Environment を作成するには、[trac-admin の initenv](#) を使用します:

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

trac-admin はプロジェクトの名前とデータベースに接続するための文字列 (後で説明します) を尋ねてきます。

### Tips

- Environment のディレクトリ配下は、Web サーバの実行ユーザが書き込み可能でなければなりません。適切なパーミッションの付与を忘れないようにしてください。同じことがソースコードリポジトリにも当てはまりますが、Subversion リポジトリの場合は、ファイルシステムに Berkeley DB を使用していない場合は、Trac は読み取りパーミッションしか必要としません; Subversion以外のリポジトリについては、各々のプラグインを確認してください。
- initenv で Subversion リポジトリを使用するように指定されても、trac-admin コマンドは指定されたりポジトリのパスに対して svnadmin create を実行したりしません。新しい Trac environment を作るときに、Subversion リポジトリも新規で作る場合は、svnadmin create は trac-admin initenv よりも前に実行しておいてください。Environment の初期化中にリポジトリが見つからない場合、"Warning: couldn't index the repository" というメッセージが表示されてしまいます。
- 非 ASCII 文字を含むパスはサポートされていません。
- プロジェクト名に空白文字が混じっていると認証で問題が生じることがありますので注意してください (See [#7163](#))。
- [グローバル設定ファイル](#) に定義した [共有するプラグインのディレクトリ](#) に配置した [TracPlugins](#) は Environment の作成時はロードされないようになりました。そのため、データベースのテーブルなどを追加で作成する必要がある場合は [Environment をアップグレード](#) せねばなりません。

### データベースに接続するための文字列

バージョン 0.9 以降、Trac は [SQLite](#) と [PostgreSQL](#) データベースバックエンドの両方をサポートします。[MySQL](#) のサポートは 0.10 で加えられました。デフォルトでは SQLite を使用します。(ほとんどのプロジェクトは SQLite で十分です)。データベース ファイルは Environment ディレクトリに保存されますので、Environment の残りと共に容易に [バックアップ](#) することができます。

#### SQLite の接続文字列

SQLite データベースに接続するための文字列は以下の通りです:

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

db/trac.db の箇所は Trac Environment のデータベースファイルへのパスを指します。

#### PostgreSQL の接続文字列

PostgreSQL や MySQL を代わりに使用する場合、異なる接続用文字列を使用しなければなりません。例えば PostgreSQL に接続するとき、ユーザ名 johndoe でパスワード letmein で 同じマシンの trac と呼ばれるデータベースに接続するには以下のように指定します:

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

"/" 及び "@" はパスワードの一部として使用出来ないので注意してください

PostgreSQL がデフォルト以外のポート番号 (例えば、9432) で起動しているときはこのようにします:

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

UNIX ホストでは、UNIX ソケットで接続するように設定できます。この場合、環境変数 PGHOST に定義されたデフォルトソケットを使用します:

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

ソケットを特定する場合はこうです:

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

PostgreSQL を使用するとき、`trac-admin initenv` を実行する前に データベースを作成しなければいけません。

[PostgreSQL](#) の詳細設定の方法については [PostgreSQL ドキュメント](#) を参照してください。 下記は `tracuser` という名のデータベースユーザ及び `trac` という名のデータベースを作成します。

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

`createuser` を実行する時、`'tracuser'` のパスワードの入力を促されます。この新しいユーザはスーパーユーザではないので、他のデータベースを作ったり、他の role (訳注: PostgreSQL でのユーザ) を作る権限を与えられていません。これらの権限は `trac` のインスタンスを実行する為には必要ではありません。ユーザにパスワードを付与したくない場合、`createuser` コマンドから `-P` と `-E` オプションを取り除いてください。また、データベースが UTF8 で作成する必要があることに注意してください。LATIN1 のエンコードが原因のエラーを引き起こします。SQL\_ASCII でも同様です。

デフォルト設定 (debian) の下では、`postgres` ユーザとして `createuser` と `createdb` スクリプトを実行してください。例えば:

```
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 はデフォルトで `public` スキーマを使用しますが、明示的に違うスキーマを指定することができます:

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

## MySQL の接続文字列

MySQL を代わりに使用したい場合、違う接続文字列を使用します。例えば、同じマシンにある `trac` という MySQL データベースに、`johndoe` というユーザでパスワード `letmein` で接続する場合の MySQL の接続文字列は次の通りです:

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

## ソースコードリポジトリ

0.12 以降のバージョンでは、一つの Environment から複数のリポジトリに接続できるようになりました。Environment からリポジトリに接続する方法は沢山あるので、詳細は [TracRepositoryAdmin](#) を参照してください。左記のページではリポジトリごとに設定する項目 (`type`, `url`, `description` など) について詳しく解説されています。

Trac 0.12 の `trac-admin` コマンドでは、接続するリポジトリの指定が必須ではなくなりました。デフォルトでは Trac はソースコードリポジトリへの接続を作成せず、メインナビゲーションのリポジトリブラウザ (英語版では Browse Source) も表示されなくなります。ロードされている `trac.versioncontrol.*` コンポーネントを完全に無効化するには下記設定を行ってください

```
[components]
trac.versioncontrol.* = disabled
```

バージョン管理システムによっては、リポジトリへのパスだけではなく、リポジトリ内の `scope` を設定することもできます。Trac はそのスコープ以下に限定したファイルとチェンジセットに関連する情報を表示します。Trac のバックエンドに Subversion を使う場合は、この機能を利用できます; 他のリポジトリシステムについては、対応するプラグインのドキュメントで確認して下さい。

デフォルトリポジトリのみで構成される Subversion リポジトリの設定の一例です:

```
[trac]
repository_type = svn
repository_dir = /path/to/your/repository
```

スコープを絞った Subversion リポジトリの設定の一例です:

```
[trac]
repository_type = svn
repository_dir = /path/to/your/repository/scope/within/repos
```

## ディレクトリ構造

プロジェクト Environment のディレクトリは通常、以下に示すファイルとディレクトリから成り立ちます。

- README - Environment について記述したドキュメント。
- VERSION - Environment のバージョン識別情報。
- attachments - 全ての添付ファイルはここに保存されます。
- conf
  - trac.ini - メインとなる設定ファイル。詳細は [TracIni](#) に記述しています。
- db
  - trac.db - SQLite データベース (SQLite を使用している場合)
- htdocs - Web のリソースを格納するディレクトリ。URL `/htdocs/site/...` を使用した Genshi テンプレートから参照する。 (0.11 以降)
- log - ログファイルのデフォルトディレクトリ。ログ機能が有効に設定され相対パスが与えられた場合に使用する。
- plugins - Environment に固有の [プラグイン](#) (Python egg または単独のファイル、[0.10](#) 以降)
- templates - カスタム (プロジェクトに固有の) Genshi テンプレート (0.11 以降)
  - site.html - カスタムヘッダ, フッタ, スタイルシート。 [TracInterfaceCustomization#SiteAppearance](#) に記載

要注意: Trac Environment のディレクトリとソースコードリポジトリのディレクトリを一緒にしないで下さい。

初心者がよくやる誤りです。上記のディレクトリ構造は Subversion リポジトリのディレクトリ構造をざっくりと真似ているだけです。全く関連はありませんので、2つは同じ場所においては いけません。

---

See also: [TracAdmin](#), [TracBackup](#), [TracIni](#), [TracGuide](#)