Trac Installation Guide for 1.0

Trac は Python で書かれており、データベースとして SQLite, PostgreSQL, MySQL のどれかが必要です。 Trac は HTML レンダリングのために Genshi テンプレートシステムを使用します。

バージョン 0.12 以降、 Trac はローカライズされているため、自分が普段使っている言語に翻訳されているかもしれません。 Trac のインタフェースで別の言語を使用したい場合は、任意のパッケージである Babel をインストールする必要があります。Installing Trac セクションのローカライゼーションのサポートの手順を参照してください。 Babel がない場合、通常通り、デフォルトの英語バージョンのみ使用することができます。

新しい言語での翻訳の作成や、すでにある言語での翻訳のエンハンスをコントリビュートした場合は TracL10N を見てみてください。 (訳注: 日本語の翻訳はすでにコントリビュートされています)

Trac のインストールとセットアップに対する一般的な手順と必要な条件を以下に示します。 Trac を特定のシステムにインストールする手順は Trac Project サイトの TracInstallPlatforms にありますが、 まず最初に以下の一般的な手順を読み通して タスクの関係を確実に理解してください。

SilverFrost 目次

Trac Installation Guide for 1.0 1
依存関係 2
必須の依存関係 2
SQLite の場合 2
PostgreSQL の場合 2
MySQL の場合 2
任意の依存関係 2
バージョン管理システム 2
Subversion 2
その他のバージョン管理システム 3
Web サーバ 3
その他の Python パッケージ 3
Trac のインストール 3
easy_install を使用したインストール 3
pip を使用したインストール 4
ソースからのインストール 4
高度なオプション 4
プロジェクト Environment の作成 5
Trac の利用 5
スタンドアロンサーバの起動 5
Web サーバ上での Trac の起動 6
Trac の cgi-bin ディレクトリを生成する 6
静的なリソースをマッピングする 6
例: Apache と ScriptAlias 7
プラグインキャッシュの設定 7
認証の構成 8
admin ユーザへの管理者権限の付与 8
インストールを終えて 8
SVN のチェンジセットを Trac のチケットに自動リンクする方法 8
Trac を使用する 8

依存関係

必須の依存関係

Trac をインストールするためには以下のソフトウェアパッケージがインストールされていなければなりません:

- Python, 2.5 以上 3.0 未満 (Python 2.4 のサポートは、このリリース（訳注: 1.0）で打ち切られました)
- setuptools, 0.6 以上, ディストリビュート を使用してもかまいません
- Genshi, 0.6 以上(リリースはされていませんが 0.7dev でも正しく動作します)

また、データベースと、それに対応する Python のバインディングが必要です。 データベースは SQLite, PostgreSQL, MySQL のいずれかが使用できます。

SQLite の場合

Python 2.5 、 2.6 、 2.7 には、SQLite データベースのライブラリがPython の標準ディストリビューションに(`sqlite3` モジュールとして)同梱されています。

もちろん、最新の Pysqlite をダウンロードすることも可能です。 google code から Windows インストーラやソースからビルドするための tar.gz アーカイブがダウンロードできます。

```
$ tar xvfz <version>.tar.gz
$ cd <version>
$ python setup.py build_static install
```

上記の手順で最新の SQLite コードがダウンロードされバインディングがビルドされます。

SQLite 2.x のサポートは終了しました。

既知のバグとして、 PySqlite バージョン 2.5.2-4 では、 Trac のデータベースを 0.11.x から 0.12. にアップグレードすることができません。 2.5.5 以降か 2.5.1 と それ以前のバージョンを使用して下さい。詳細については、 本家チケット 9434 を参照して下さい。

PySqlite にさらに情報があります。

PostgreSQL の場合

下記いずれかの データベースと、データベースに対応した Python バインディングをインストールする必要があります:

- PostgreSQL, バージョン 8.0 以降
- psycopg2

詳しくは DatabaseBackend を参照してください。

MySQL の場合

Trac は以下のガイドラインで、 MySQL でも良好に動作するようになりました。

- MySQL, 5.0 以降
- MySQLdb, 1.2.2 以降

非常に 重要なことが記載されているので、データベースを作成する前に、 MySqlDb のページを注意深く読んで下さい。

任意の依存関係

バージョン管理システム

Subversion

- Subversion 1.5.x または 1.6.x と 対応する Python バインディング。1.0 から 1.2.4 や 1.3.2 、 1.4.2 などの古いバージョンも今のところ動作します。トラブルシューティングの情報が TracSubversion のページに記載されていますので、確認してください

様々なプラットフォーム向けに SWIG バインディング が用意されています。(このリストから Windows パッケージ用のコンパイル済みの SWIG バインディングを探してください。TracSubversion には Algazam がよいです。私も python 2.6 の基、使用しています。)

Note: Trac は PySVN のような新しい ctype 形式のバインディングでは 動作しません。

重要な Note: Subversion を使用するなら Trac を 同じマシン にインストールする必要があります。リモートリポジトリは現在 サポートされていません。

その他のバージョン管理システム

Subversion 以外のバージョン管理システムのサポートはサードパーティから提供されます。 PluginList や VersionControlSystem を参照して下さい。

Web サーバ

Trac にはサーバ機能が組み込まれているので、 Web サーバは必須ではありません。このページに下にある スタンドアロンサーバの起動 セクションを参照してください。

Trac は下記の環境において動作します。

Apache との組み合わせで
  - mod_wsgi, TracModWSGI と http://code.google.com/p/modwsgi/wiki/IntegrationWithTrac を参照
  - mod_python 3.3.1, 非奨励 (TracModPython 参照)
- FastCGI 対応 Web サーバ (TracFastCgi 参照)
- AJP 対応 Web サーバ (TracOnWindowsIisAjp 参照)
- CGI 対応 Web サーバ (TracCgi 参照)。’’’Trac を CGI スクリプトとして使用することは 全く推奨されていませんので’’’、上記に挙げた方法を選択して下さい

その他の Python パッケージ

- Babel, 0.9.5 以上 ローカライズの機能を使用する場合は必要(リリースはされていませんが 1.0dev でも正しく動作します)
- docutils, 0.3.9 以上 WikiRestructuredText の機能を使用する場合は必要
- Pygments シンタックスハイライティング の機能を使用する場合は必要 SilverCity や Enscript も、今のところ使用できますが、サポートを打ち切る予定なので、 Pygments を使用してください
- pytz , タイムゾーンの完全なリストを取得する場合に必要 pytz がない場合、 Trac は内部で定義している短いタイムゾーンの実装にフォールバックします

Attention: これらの依存関係は様々なバージョンで必ずしも置き換えできるとは限らないので、上記のバージョン番号に注意してください。 Trac を動かす上で問題が発生した場合は メーリングリスト や IRC チャネル で質問をする前にすべての依存関係を再度確認してください。

これらのパッケージのドキュメンテーションを参照して、それらが最も上手にインストールできる方法を探してください。また プラットフォーム特有の説明 の多くに、これらの依存関係のインストール方法が記述されています。しかしながら プラットフォーム特有の説明 の情報はあなたがインストールしている Trac より古い バージョンについての説明があることを覚えておいてください (なんと Trac 0.8 に関する説明をしているページもあります)。

Trac のインストール

easy_install を使用したインストール

Trac をインストールする方法のひとつに setuptools の利用があります。 setuptools を使用すると、 Trac を Subversion リポジトリからインストールすることもできます。

例:

- Trac 1.0 をインストールします:

```
easy_install Trac==1.0
```

  (まだ有効になっていない場合)
- 開発中の最新バージョン 1.0dev をインストールします:

```
easy_install Trac==dev
```

この場合、ローカライズに対応していないバージョンである可能性があります。
リリースされているバージョンを使用するか、ソースからインストールするようにしてください

pip を使用したインストール

'pip' は easy_install のリプレースであり、とても簡単に素早く Python パッケージをインストールすることができます。 Trac を 5 分程度でインストール、起動できます:

pip によってインストールされる場所を /opt/user/trac とした場合の例です:

-

```
pip -E /opt/user/trac install trac psycopg2
```

または

-

```
pip -E /opt/user/trac install trac mysql-python
```

PostgreSQL (libpq-dev) や MySQL (libmysqlclient-dev) へのバインディングも自動でビルドできるように、 pip が OS 固有のヘッダファイルを確実に利用できるようにして下さい。

また pip は (Genshi, Pygments などの) 依存関係を解決し、 pypi.python.org から最新のパッケージをダウンロードして、 /opt/user/trac の配下にインストールするところまで自動化されています。

すべてのコマンド (tracd, trac-admin) は /opt/user/trac/bin の配下にインストールされます。 mod_python (PythonHandler ディレクティブを使用する場合) や mod_wsgi (WSGIDaemonProcess ディレクティブを使用する場合) においても活用することができます。

加えて、 Trac プラグインのうちのいくつか (ここ で一覧を見ることができます) も pip からインストールすることが可能です。

ソースからのインストール

もちろん、ソースディレクトリのトップに格納されている setup.py を使用してインストールすることもできます。

(Trac-1.0.tar.gz など)リリースパッケージの .tar.gz や .zip ファイル、または直接リポジトリからソースを取得することが出来ます(詳細は Trac:SubversionRepository を参照してください)。

```
$ python ./setup.py install
```

このステップを実行するためには root 権限 (または root 権限と同等の権限) が必要です。

この操作で Python のソースコードがバイトコンパイルされ、 .egg ファイルかディレクトリが Python インストールの site-packages ディレクトリにインストールされます。 .egg には htdocs や templates のようなソースファイル以外に標準の Trac が必要とするすべてのリソースが含まれています。

このスクリプトは tracd スタンドアロンサーバと一緒に、 プロジェクト Environment を作成し維持するための trac-admin コマンドラインツールをインストールします。

ソースからインストールする場合や、新しい言語で Trac を国際化するためには Babel をインストールしておく必要があります。この場合も install を実行するだけです (Babel がインストールされていない状態で、すでに Trac をインストールしてしまった場合でも、 install をやり直すことで Babel サポートを有効化できます):

```
$ python ./setup.py install
```

また、 bdist_egg を実行すると dist ディレクトリに作成される .egg ファイルをインストール先にコピーしたり、 (bdist_wininst の実行によって) Windows インストーラを作成しても構いません。

高度なオプション

Trac のインストール場所を変えるなどの高度なインストールオプションを知りたければ以下を実行してください:

```
easy_install --help
```

詳細な情報は Python モジュールをインストールする を参照してください。

特にあなたは以下に興味を持つかもしれません:

```
easy_install --prefix=/path/to/installdir
```

Mac OS X に Trac をインストールする場合:

```
easy_install --prefix=/usr/local --install-dir=/Library/Python/2.5/site-packages
```

Note: Mac OS X 10.6 上で easy_install http://svn.edgewall.org/repos/trac/trunk を使用する場合は、オプションを指定しなくても /usr/local および /Library/Python/2.6/site-packages にインストールされます。

上記の例は、 tracd と trac-admin コマンドを /usr/local/bin に、 Trac のライブラリと依存ファイルを /Library/Python/2.5/site-packages にインストールします。これらのパスは Apple での Pyhton サードパーティアプリケーションの標準ロケーションです。(訳注:
つまり、上記と違うパスにインストールしたい場合のみ、オプションの指定が必要になります)

プロジェクト Environment の作成

Trac Environment は Trac が Wiki ページ、チケット、レポート、設定などの情報を保存するバックエンドストレージです。基本的に人間が読み込み可能な 構成ファイル と他の様々なファイルやディレクトリで構成されます。

新しい Environment は trac-admin を使用して作成します:

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

trac-admin は、プロジェクトの名前や データベース接続文字列 など、 Environment を新規作成するために必要な情報を入力するためのプロンプトを表示します。これらの設定項目について特に変更が必要ない場合は、単に <Enter> を押下すると、デフォルト値が使用されます。

データベース接続文字列のデフォルトは SQLite が使用されます。 SQLite がインストールされている場合は、他の設定は不要です。 他の データベースバックエンド を使用する場合は、あらかじめデータベースが使用可能な状態にしておかねばなりません。

0.12 以降で Trac は、新しい Environment の作成時に ソースコードリポジトリ を尋ねないようになりました。リポジトリを後で 追加する までの間、バージョン管理のサポートは無効化されます。

また、 ここで指定した値は conf/trac.ini 設定ファイルを直接編集することで後から変更できます。

最後に、 Web のフロントエンドを実行しているユーザアカウントは、 Environment のディレクトリと、その中のすべてのファイルに対する書き込み権限が必要です。 trac-admin ... initenv の実行を該当するユーザで実行した場合は、この作業は不要ですが、そうでない場合、ただしユーザに権限を付与する作業が必要になります。たとえば Linux で apache ユーザ apache グループで Web サーバを起動する場合は:

```
# chown -R apache.apache /path/to/myproject
```

Warning: アカウント名とプロジェクトパスには ASCII 文字のみを使用して下さい。 unicode 文字はサポートしていません。

Trac の利用

スタンドアロンサーバの起動

Trac 環境を作成した後に、スタンドアロンサーバ tracd を実行することで簡単に Web インタフェースを試すことができます。

```
$ tracd --port 8000 /path/to/myproject
```

ブラウザを起動して、 http://localhost:8000/ にアクセスしてください。 tracd が認識しているすべての Environment の簡単な一覧が表示されます。作成した Environment へのリンクにアクセスすることで Trac が動作中であることを確認できます。 Trac でプロジェクトをひとつだけ管理したい場合、以下のように起動することで、スタンドアロンサーバは Environment 一覧の表示をスキップして、直接 Environment を表示します:

```
$ tracd -s --port 8000 /path/to/myproject
```

Web サーバ上での Trac の起動

Trac には "実際の" Web サーバへ接続するために、いくつかの選択肢があります:

- FastCGI
- mod_wsgi
- mod_python (mod_python は現在活発なメンテナンスが行われていないため、推奨されません)
- CGI (最適なパフォーマンスに遠く及ばないため、使用すべきではありません)

Trac では AJP もサポートしており、IIS と接続したい場合に選択肢の一つとなるかもしれません。その他の選択肢もあります: nginx, uwsgi, Isapi-wsgi 等。

Trac の cgi-bin ディレクトリを生成する

Trac を FastCGI などで正しく機能させるには、 FastCGI であれば trac.fcgi ファイル、 mod_wsgi であれば trac.wsgi ファイルが必要となります。これらのファイルは適切な Python コードをロードする Python スクリプトです。 trac-admin コマンドの deploy オプションを使用することで生成できます。

若干の「卵が先か鶏が先か」問題があります。 trac-admin コマンドが機能するためには Environment が必要なのですが、 deploy には既に存在するディレクトリは使用できません。これに起因して、 Environment は depoly するディレクトリのサブディレクトリを使用することができません。この制限を回避するには次のようにします:

```
mkdir -p /usr/share/trac/projects/my-project
trac-admin /usr/share/trac/projects/my-project initenv
trac-admin /usr/share/trac/projects/my-project deploy /tmp/deploy
mv /tmp/deploy/* /usr/share/trac
```

静的なリソースをマッピングする

特に設定することなく Trac はスタイルシートや画像のような静的なリソースを扱うことができます。tracd は唯一基本的な動作環境ですが、 Web サーバがそれら静的リソースを直接供給するように設定することより、はるかに最適になります。(CGI でセットアップした場合、極めて望ましくないし、著しい性能悪化の原因となります)

Apache のような Web サーバはリソースに対して "Alias" を設定することで仮想の URL を与え、サーバのファイルシステムのレイアウトとは異なる位置にマップすることができます。また、Trac自身によるこれらの要求に対する処理を避けて、直接フ

静的なリソースに対する主要な URL パスとして /chrome/common と /chrome/site があります。プラグインを使用している場合、各々の静的リソースとして /chrome/<plugin> のようなパスが追加されていることがありますが、主要なパスに対して追加を行えるだけなので、 /chrome に対して Alias を設定しても、プラグインが提供する静的リソースに対してアクセスできる訳ではありません。(訳注: このような場合 trac-admin の deploy で作成された htdocs ディレクトリに対して Alias を使用してください)

Note: ファイルシステムにおける静的なリソースを取得するためには、最初に trac-admin の <environment> deploy コマンドを使用して、 Trac 関連のリソースを拡張する必要があります。

```
deploy <directory>

   Extract static resources from Trac and all plugins
```

ターゲットの <directory> は、以下のように htdocs ディレクトリに含まれます:

- site/ - Environment 内の htdocs/ ディレクトリのコピーです
- common/ - Trac 自身が持つ静的なリソース

- <plugins>/ - Environment で使用可能なプラグインによって管理される個々のリソースディレクトリ

例: Apache と ScriptAlias

以下のような環境を仮定すると:

```
$ trac-admin /var/trac/env deploy /path/to/trac/htdocs/common
```

Apache の設定ファイル内において ScriptAlias や WSGIScriptAlias の記述(Trac
アプリケーションへの他のすべてのリクエストに対するマップの記述) より上に 以下のブロックを追記します。 パスは環境に合わせ変更してください:

```
Alias /trac/chrome/common /path/to/trac/htdocs/common
Alias /trac/chrome/site /path/to/trac/htdocs/site

<Directory "/path/to/www/trac/htdocs">
 Order allow,deny
 Allow from all
</Directory>
```

もし mod_python を使用している場合、この設定を追加した方が良いかもしれません(使用していない場合は、このエイリアスは無視されます):

```
<Location "/trac/chrome/common/">
 SetHandler None
</Location>
```

Note: trac.*cgi スクリプトを /trac にマップしている場合、 /trac/chrome/common
を加えたパスに対するリクエストは、静的リソースで処理するように割り込みを加えます。

同様に、静的なリソースを project の htdocs ディレクトリで使用している場合 (テーマの中で /trac/chrome/site を参照しているなど)、
これらのリソースを供給するよう Apache を設定することが出来ます。(再度、.*cgi スクリプトの ScriptAlias や WSGIScriptAlias より上に
以下のブロックを記述します。インストール状況に合わせて、ファイル名やロケーションは適宜変更してください):

```
Alias /trac/chrome/site /path/to/projectenv/htdocs

<Directory "/path/to/projectenv/htdocs">
 Order allow,deny
 Allow from all
</Directory>
```

一方で /trac/chrome/common をエイリアスとするような場合、 Trac では [trac htdocs_location]
を設定することで、それらリソースに対して直接リンクを生成することができます:

```
[trac]
htdocs_location = http://static.example.org/trac-common/
```

Note: これは 静的なリソースを供給する専用のドメインを簡単に設定します。(cookie-less)

当然、Web サーバのドキュメントルートにディレクトリをコピーする(又は、リンクする)等して、特定の URL で Web サーバに接続できるよう Trac
の htdocs/common ディレクトリを作成する必要があります。

```
$ ln -s /path/to/trac/htdocs/common /var/www/static.example.org/trac-common
```

プラグインキャッシュの設定

Python
プラグインの中にはキャッシュディレクトリを必要とするものがあります。デフォルトではキャッシュディレクトリは、現在のユーザのホームディレクトリに置かれ
Trac を Web サーバで動作させている場合、ホームディレクトリを持たない専用ユーザであることが多く
(強く推奨します)、プラグインの起動が妨げられることがあります。キャッシュディレクトリの場所を変更するには、環境変数 PYTHON_EGG_CACHE
を設定してください。環境変数を設定する方法の詳細は使用しているサーバのドキュメントから参照してください。

認証の構成

Trac は HTTP 認証を使用します。.../login の URL ("ログイン"(英語版では "login") ボタンの仮想パス)
がリクエストされた際に、認証を要求するように Web サーバを設定する必要があります。Trac は認証情報を得た後自動的に REMOTE_USER
変数を獲得します。そのため、すべてのユーザの管理は Web サーバ の設定で行います。設定方法等の詳細な情報は、利用している Web
サーバのドキュメントを参照してください。

認証のためのユーザアカウントを追加、削除、設定する方法は Trac を起動する方法により異なります。

以下に記すセクションを適宜参照してください:

- スタンドアロンサーバ tracd を使用する場合は TracStandalone#UsingAuthentication
- Apache Web サーバ と mod_wsgi に代表される mod_python や mod_fcgi, mod_fastcgi といったフロントエンドを使用する場合は TracModWSGI#ConfiguringAuthentication
- Apache 以外の FCGI をサポートしている Web サーバ (Cherokee, Lighttpd, LiteSpeed, nginx) を使用する場合 TracFastCgi

admin ユーザへの管理者権限の付与

admin ユーザに管理者権限を付与します:

```
$ trac-admin /path/to/myproject permission add admin TRAC_ADMIN
```

このユーザは Trac プロジェクトに管理者としてアクセスするため、メニュー内に "管理"(英語版では "Admin") が表示されます。

インストールを終えて

SVN のチェンジセットを Trac のチケットに自動リンクする方法

変更をリポジトリにコミットした時に、チェンジセットへのリンクをチケットのコメントに自動で追加するように SVN
を設定することができます。コミットメッセージには以下に示すいずれかの書式が含まれていなければなりません:

- Refs #123 - このチェンジセットへのリンクをチケット #123 に追加します
- Fixes #123 - このチェンジセットへのリンクをチケット #123 に追加し、チケットを fixed でクローズします

この機能を使用するためには post-commit フックを TracRepositoryAdmin に記載したリポジトリにインストールし、 commit updater
コンポーネントを有効にせねばなりません。コンポーネントの有効化は、 trac.ini ファイルの [components] セクションに下記記述を追加するか、
"プラグイン" (英語版では "Plugins") 管理パネルから設定します。

```
tracopt.ticket.commit_updater.* = enabled
```

詳細な情報は "プラグイン" 管理パネルの CommitTicketUpdater コンポーネントにあるドキュメントを参照してください。

Trac を使用する

一度 Trac サイトを稼働させれば、チケットを作成したり、タイムラインを見たり、 (設定されていれば)
バージョン管理のリポジトリを閲覧したりできるはずです。

anonymous (ログインしていない)
でアクセスするユーザは、デフォルトではほとんど機能を使用することができません。特に、リソースに対して読み取りのみのアクセスになります。すべての機能を
アクセス許可 を与える必要があります。

Enjoy!

The Trac Team

See also: TracInstallPlatforms, TracGuide, TracUpgrade, TracPermissions