# CGI として Trac をインストールする

Trac を CGI 経由で使用するのは、他の方法である mod_python, FastCGI や Windows 上の IIS/AJP よりも極めて遅いことに注意して下さい。

CGI スクリプトは、 Web サーバがリクエストを受信した時にアプリケーションを起動するためのエントリポイントになります。下記のスクリプトで trac.cgi を生成できます:

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

trac.cgi は、 Web ブラウザから実行可能なパスの通った cgi-bin フォルダに置きます。このコマンドは与えられたパスの htdocs に ■■■■■■ ファイルをコピーします。

## Apache web-server 設定

Apache HTTPD を使用している場合、 Trac で CGI を動作させる設定方法は 2 通りあります:

1. ScriptAlias を使用し、 trac.cgi スクリプトを URL にマップする。(推奨)
2. CGI プログラムが実行可能な、 Web サーバ上のディレクトリ (通常は cgi-bin が使用される) に trac.cgi ファイルをコピーする。シンボリックリンクでも代用できますが、その場合 cgi-bin ディレクトリでは FollowSymLinks オプションを有効にして下さい。

http://yourhost.example.org/trac で Trac を実行するには、インストールしたマシンのApache の設定ファイルに、 trac.cgi パスに変更した上で、ScriptAlias ディレクティブを追加します。

```
ScriptAlias /trac /path/to/www/trac/cgi-bin/trac.cgi
```

このディレクティブを使用するには *mod_alias* モジュールを有効にしておく必要があります。

Trac のプロジェクトが 1 つの場合、 TRAC_ENV 環境変数を使用してプロジェクトへのパスを指定する必要があります:

```
<Location "/trac">
 SetEnv TRAC_ENV "/path/to/projectenv"
</Location>
```

もしくは、複数のプロジェクトを扱うために、それらの親ディレクトリを TRAC_ENV_PARENT_DIR を使用して設定することができます。

```
<Location "/trac">
 SetEnv TRAC_ENV_PARENT_DIR "/path/to/project/parent/dir"
</Location>
```

Note: *SetEnv* ディレクティブを使用するには *mod_env* モジュールをインストールし有効にしておく必要があります。 trac.cgi に TRAC_ENV を設定することで代用できます。 "try" と "from trac.web ..." の間に以下のコードを加えてください:

```
    import os
   os.environ['TRAC_ENV'] = "/path/to/projectenv"
```

もしくは、 TRAC_ENV_PARENT_DIR を設定します:

```
    import os
   os.environ['TRAC_ENV_PARENT_DIR'] = "/path/to/project/parent/dir"
```

Apache suEXEC 機能を使用している場合、 ApacheSuexec を参照して下さい。

システムによっては、trac.cgi ファイルの shebang 行を編集して、実際に Python がインストールされているパスを指すように修正する必要があるかもしれません。 Windows システム上では、 Windowsが .cgi ファイルを実行できるように設定する必要があるでしょう。 (エクスプローラ -> ツール -> フォルダオプション -> ファイルの種類 -> CGI を Python と関連付ける)

## 静的なリソースをマッピングする

このままでも Trac はスタイルシートや画像ファイルなどの静的なリソースを扱えますが、 CGI のセットアップとしては 妥当な設定とは言えません。 Web サーバ自身がはるかに効率良くに直接扱うことができるドキュメントまで、 CGI スクリプトとして呼び出してしまうという結果になるからです。

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。すでに CGI スクリプトに対して ScriptAlias を定義していますので、ファイルシステム上の静的リソースを含んだコンテンツをマッピングすることで CGI スクリプトの要求を回避します。

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)

CGI スクリプトの ScriptAlias を追記したブロックより 上に 以下のブロックを追記します。ファイル名とロケーションは適宜変更してください:

```
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 を使用しているならば、この設定も追加したいかもしれません ( mod_python を使用していないなら、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、その一部である /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。

例えば、Trac が /cgi-bin/trac.cgi にマッピングされているとしたら、 Alias の URL は cgi-bin/trac.cgi/chrome/common となります。

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 /chrome/site を参照しているなど)、プロジェクトディレクトリの htdocs に Apache の設定を行うことが出来ます (再度、 CGI スクリプトの ScriptAlias を追記したブロックより上に、以下のブロックを追記します。ファイル名と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>
```

あるいは /trac/chrome/site をハックする場合は、 trac.ini に htdocs_location のオプションを設定することができます。

```
[trac]
htdocs_location = http://yourhost.example.org/trac-htdocs
```

Trac は HTML ページに静的リソースを組み込むときに、この URL を使用するようになります。もちろん、それでも、特定の URL が指定されたときに例えば、 Web サーバーのドキュメントルートにディレクトリをコピーするなり、シンボリックリンクを張るなりして、Web サーバ経由で、 Trac が htdocs ディレクトリを利用できるようにしておく必要があります:

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

Note: htdocs ディレクトリの内容を取得するには、最初に TracAdmin の deploy コマンドを使用して、対応する Trac リソースを展開する必要があります。

```
deploy <directory>
```

```
   Extract static resources from Trac and all plugins
```

## 認証を追加する

Apache で認証を追加する最も単純な方法はパスワードファイルを作ることです。 htpasswd
プログラムを使用してパスワードファイルを作成します:

```
$ htpasswd -c /somewhere/trac.htpasswd admin
New password: <type password>
Re-type new password: <type password again>
Adding password for user admin
```

一番最初のユーザ以外は "-c" オプションは必要ありません:

```
$ htpasswd /somewhere/trac.htpasswd john
New password: <type password>
Re-type new password: <type password again>
Adding password for user john
```

*htpasswd* についての詳細は man を見てください。

ユーザを作成した後、 TracPermissions の記述通りユーザに権限を設定することができます。

Apache の設定ファイルの中にパスワードファイル名を記述し、認証を有効にする必要があります:

```
<Location "/trac/login">
 AuthType Basic
 AuthName "Trac"
 AuthUserFile /somewhere/trac.htpasswd
 Require valid-user
</Location>
```

複数のプロジェクトを持っている場合でも、パスワードファイルはプロジェクトで共通なものを使用することができます:

```
<LocationMatch "/trac/[^/]+/login">
 AuthType Basic
 AuthName "Trac"
 AuthUserFile /somewhere/trac.htpasswd
 Require valid-user
</LocationMatch>
```

より堅固なセキュリティのために、 SSL を有効にするか、少なくとも "基本認証" の代わりに "ダイジェスト認証"
を使用することを推奨します。より詳しい情報については Apache HTTPD documentation を参照して下さい。例えば、Dian 4.0r1 (etch)
システムの上では、Apache 構成における関連セクションは下記のように記述できます。

```
<Location "/trac/login">
   LoadModule auth_digest_module /usr/lib/apache2/modules/mod_auth_digest.so
   AuthType Digest
   AuthName "trac"
   AuthDigestDomain /trac
   AuthUserFile /somewhere/trac.htpasswd
   Require valid-user
</Location>
```

.htpasswd ファイルは htpasswd の代わりに htdigest を使って、以下の通りに作成する必要があります:

```
# htdigest /somewhere/trac.htpasswd trac admin
```

上記の "trac" パラメータは 上の AuthName と同じ値にします (apache-docs の "Realm")。

---

See also: TracGuide, TracInstall, TracModWSGI, TracFastCgi, TracModPython