Wikiprint Book

Title: Trac と mod_wsgi

Subject: SilverFrost - TracModWSGI

Version: 2

Date: 12/20/25 13:46:33

SilverFrost 目次

Trac と mod_wsgi

重要な Note: バージョン 1.6 または 2.4 以降の *mod_wsgi* を使用してください。 2.X ブランチの 2.4 以前のバージョンでは、 Apache の特定の設定下において WSGI の file_wrapper 拡張の扱いに問題があります。この拡張は Trac では添付ファイルおよび、スタイルシートなどの静的メディアファイルを提供する箇所で使われています。この問題の影響を受けてしまった場合、添付ファイルは空 HTML ページではスタイルシートがロードされないため描画が正しく行われなくなります。詳細は mod_wsgi のチケット #100 と #132 を確認してください。

mod_wsgi は WSGI 互換の Python アプリケーションを Apache 上で直接起動させることができる Apache のモジュールです。 mod_wsgi アダプターは完全に C 言語で書かれており、 mod_python や CGI に比べて非常によいパフォーマンスを提供します。

Trac は mod_wsgi のトップレベルに application を定義するスクリプトだけで動作します。このスクリプトは単独の Python ファイルで、通常は .wsgi という拡張子で保存されます。このファイルを生成するためには trac-admin <env> deploy <dir> コマンドを使用します。コマンドを実行すると必要となるパスが自動的に設定されます。

```
import os

os.environ['TRAC_ENV'] = '/usr/local/trac/mysite'
os.environ['PYTHON_EGG_CACHE'] = '/usr/local/trac/mysite/eggs'

import trac.web.main
application = trac.web.main.dispatch_request
```

環境変数 TRAC_ENV は通常通り Trac environment のディレクトリを指定します（複数の Trac environment を含むディレクトリであれば TRAC_ENV_PARENT_DIR を使うこともできます)。 PYTHON_EGG_CACHE は Python eggs を一時的に展開するのに使用するディレクトリを指定します。

重要な Note: 複数の .wsgi ファイルを使用する場合（それぞれのファイルに別個の Trac environment を設定するケースなど）は、 os.environ['TRAC_ENV'] には Trac environment のパスを 設定しない でください。この方法を使うと、別の Trac environment の設定が Trac にロードされてしまうことがあります。（以前にロードした Trac environment のパスが使われてしまいます。）この問題は .wsgi ファイルの内容を下記の通り変更することで回避できます:

```
import os

os.environ['PYTHON_EGG_CACHE'] = '/usr/local/trac/mysite/eggs'

import trac.web.main
def application(environ, start_response):
 environ['trac.env_path'] = '/usr/local/trac/mysite'
 return trac.web.main.dispatch_request(environ, start_response)
```

分かりやすくするために、このファイルの拡張子は .wsgi とすべきです。 Apache にアクセス権を開放できるのであれば、このファイルは自分が所有権を持つディレクトリに置くこともできます。この .wsgi ファイルは TracAdmin のコマンド deploy を使用することで作成することができます。

Trac と egg ファイルをインストールしたパスが通常と異なる場合、それらのパスを以下の要領で wsgi スクリプトの先頭に記述する必要があります:

```
import site
site.addsitedir('/usr/local/trac/lib/python2.4/site-packages')
```

パスはインストールした Trac のライブラリに位置に一致するように変更してください。

wsgi-script の準備ができたら、以下のように httpd.conf に設定を追加してください。

```
WSGIScriptAlias /trac /usr/local/trac/mysite/apache/mysite.wsgi

<Directory /usr/local/trac/mysite/apache>
   WSGIApplicationGroup %{GLOBAL}
   Order deny,allow
   Allow from all
```

```
</Directory>
```

Trac environment のサブディレクトリにスクリプトがある場合、 Apache がスクリプトを起動する為には、スクリプトが含まれるディレクトリまで完全に Apache がアクセスできなければなりません。 WSGIApplicationGroup ディレクティブを使用すると、常に mod_wsgi が作成した最初の Python インタプリタ内で Trac が起動することが保証されます; これは Trac で使用している Subversion の Python バインディングがサブインタプリタでは動作しないことがあるため必要になります。リクエストがハングし、 Apache がクラッシュしたような結果が返ります。この設定を行った後は Apache を再起動しないと反映されません。

Apache, mod_wsgi, Python 本体 (Trac とその依存ライブラリを除く) の設定をテストしたい場合、簡単な wsgi アプリケーションを使用するとリクエストが処理されているか確認することができます (以下に示す内容だけを持つ .wsgi スクリプトを使用してください):

```
def application(environ, start_response):
       start_response('200 OK',[('Content-type','text/html')])
       return ['<html><body>Hello World!</body></html>']
```

mod_wsgi の インストール例 に Trac の情報が掲載されています。

トラブルシューティングの Tips は mod_python のトラブルシューティング セクションも参考になります。 Apache に関連する問題の多くは似通っていて、多くの場合 mod_wsgi を使用する アプリケーション側の問題 です。

Note: 私の環境では mod_wsgi 2.5 と Python 2.6.1 を使うと Internal Server Error が発生しました (Apache 2.2.11 および Trac 0.11.2.1)。 Python 2.6.2 にアップグレードすると解決しました。 (ここ に書いてある通り)
-- Graham Shanks

mod_wsgi 使用におけるTrac での Apache 基本認証

上記の mod_wsgi のドキュメントには、 Apache の設定例 a) trac をバーチャルホストでサブドメインを作成して動かす例と b) Trac の認証として、Apache の基本認証を設定する例が記載されています。

例えば、 trac を http://trac.my-proj.my-site.org としてホストし、/home/trac-for-my-proj フォルダから起動する場合で、 the-env を作成するために、 trac-admin the-env initenv コマンドを使用し、 the-deploy フォルダを作成するために、 trac-admin the-env deploy the-deploy コマンドを使用した場合です:

htpasswd ファイルを作成します:

```
cd /home/trac-for-my-proj/the-env
htpasswd -c htpasswd firstuser
### and add more users to it as needed:
htpasswd htpasswd seconduser
```

(セキュリティ面より、このファイルはドキュメントルートにおきます)

以下の設定を含んだファイルを作成します。例 /etc/apache2/sites-enabled/trac.my-proj.my-site.org.conf (ubuntu):

```
<Directory /home/trac-for-my-proj/the-deploy/cgi-bin/trac.wsgi>
 WSGIApplicationGroup %{GLOBAL}
 Order deny,allow
 Allow from all
</Directory>

<VirtualHost *:80>
 ServerName trac.my-proj.my-site.org
 DocumentRoot /home/trac-for-my-proj/the-env/htdocs/
 WSGIScriptAlias / /home/trac-for-my-proj/the-deploy/cgi-bin/trac.wsgi
 <Location '/'>
   AuthType Basic
   AuthName "Trac"
   AuthUserFile /home/trac-for-my-proj/the-env/htpasswd
   Require valid-user
```

```
 </Location>
</VirtualHost>
```

(サブドメインが適切に動くようにするには、 /etc/hosts ファイルの変更や、ホストサーバの DNS の A レコードにサブドメインを追加する必要があります)

Trac と PostgreSQL

mod_wsgi アダプタを使用し、 Trac のインスタンスを複数ホストしている場合に、 PostgreSQL (もしかすると MySQL も?) をデータベースバックエンドとして使用していると、大量のデータベース接続が生成され (PostgreSQL のプロセスも大量に発生し) てしまいます。

さしあたり動く解決方法として、 Trac が持つコネクションプールを無効化する方法があります。これは trac.db.postgres_backend の PostgreSQLConnection クラスで定義されている poolable を False に設定することで適用できます。

この方法を適用するために、 Trac のソースを変更する必要はありません。以下に示す行を trac.wsgi に追加してください:

```
import trac.db.postgres_backend
trac.db.postgres_backend.PostgreSQLConnection.poolable = False
```

この設定で Trac ページを生成した後にコネクションを捨てるようになり、データベースへの接続数は最小に保たれます。

SSPI および 'Require Group' 使用時に Trac を動かす方法

Trac を Win32 上の Apache で起動し、 SSPI 設定して 'Require group' オプションを構成している場合、 'SSPIOmitDomain' オプションはおそらく動作しません。 Trac にユーザ名が認識されない場合は、 'user' が 'DOMAIN / user' のように見えている可能性があります。

このような場合、以下のように WSGI スクリプトを修正すると解決すると思います:

```
import os
import trac.web.main

os.environ['TRAC_ENV'] = '/usr/local/trac/mysite'
os.environ['PYTHON_EGG_CACHE'] = '/usr/local/trac/mysite/eggs'

def application(environ, start_response):
   if "\\" in environ['REMOTE_USER']:
       environ['REMOTE_USER'] = environ['REMOTE_USER'].split("\\", 1)[1]
   return trac.web.main.dispatch_request(environ, start_response)
```

See also: TracGuide, TracInstall, FastCGI, ModPython, TracNginxRecipe