Wikiprint Book

Title: リポジトリ管理

Subject: SilverFrost - TracRepositoryAdmin

Version: 2

Date: 03/22/26 05:49:33

SilverFrost 目次

リポジトリ管理

クイックスタート

- リポジトリの管理は、 "リポジトリ" (英語版では "Repository") 管理パネル、 trac-admin または、 trac.ini の [repositories] セクションで管理します
- 各リポジトリの post-commit フックに trac-admin $ENV changeset added $REPO $REV を実行するよう設定して下さい。さらに、リポジトリのリビジョンのプロパティが変更されたときのために、 post-revprop-change フックに trac-admin $ENV changeset modified $REPO $REV を追加してください
- [trac] repository_sync_per_request オプションを empty 値に設定し、リクエスト毎の同期を行なわないようにします
- Subversion のフックを実行するユーザが Trac environment に対して書き込み権限を持っていることを確認して下さい。もしくは、一時的に権限を高める sudo のようなツールを使って下さい

リポジトリを指定する

バージョン 0.12 から Trac は 1 つの Trac Environment に対して 1 つ以上のリポジトリを扱えるようになりました。 0.12 以前の trac.ini の [trac] セクションの repository_dir と repository_type オプションでリポジトリを指定する方法もサポートしていますが、 2 つの新しいメカニズムにより、 Trac Environment にリポジトリを追加することができます。

リポジトリのエイリアスを定義することも可能です。エイリアスは実際のリポジトリへの "ポインタ" となります。これはリポジトリの名前変更を行なったときに古い名前へのリンク切れを防ぐのに便利です。

各リポジトリに関連したいくつかの属性があり、リポジトリのロケーション、名前、リポジトリブラウザでどのように表示されるかを定義できます。下記の属性がサ

属性	概要
alias	リポジトリは 実在するリポジトリへのエイリアスとなる alias 属性を 1 つ持っています。すべての TracLinks はエイリアスされたリポジトリを解決するために エイリアスを参照します。複数の間接参照はサポートされていません。従って、エイリアスは常に実在するリポジトリをポイントしなければなりません。alias 属性と dir 属性は相反するものです。
description	description 属性で指定されているテキストは、リポジトリブラウザ内のそのリポジトリのトップレベルのエントリの下部に表示されます。 WikiFormatting をサポートします。
dir	dir 属性はファイルシステム内におけるリポジトリのロケーションを指定します。これは以前に設定していた [trac] repository_dir と一致するものです。 alias 属性と dir 属性は相反する設定です。
hidden	true に設定すると、リポジトリブラウザのインデックスのページから見えなくなります。それでもなお、リポジトリをブラウズすることはできますし、リポジトリを参照するリンクは有効です。
type	type 属性はリポジトリが使用しているバージョン管理システムのタイプを設定します。 Trac は Subversion を当初からサポートしており、プラグインを使用して他のシステムをサポートします。もし type 属性が指定されていなかったら、 [trac] repository_type オプションの値が使用されます。
url	url 属性はリポジトリからチェックアウトするときに使用するルートのURLを指定します。指定された場合、 "リポジトリ URL" リンクがリポジトリブラウザのナビゲーションコンテキストに追加されます。 URL はツールにコピーすることができるので、ワーキングコピーを作るときに使用できます。

リポジトリの name 属性とalias 属性または dir 属性は必須項目です。他の項目はオプションです。

リポジトリを追加した後、そのリポジトリのキャッシュは trac-admin $ENV repository resync コマンドで再同期されなければなりません。

repository resync <repos>

Trac とリポジトリを再同期する。

ファイル (trac.ini) を使用した設定

リポジトリとリポジトリの属性は trac.ini の [repositories] セクションで設定することができます。すべてのキーは {name}.{attribute} という規則で構成されていて、キーに対応する値は、等号 (=) で区切られています。デフォルトのリポジトリの名前は empty になります。

trac.ini でリポジトリを設定する主たる利点は、グローバルな設定から継承できることです (TracIni の グローバルな設定 参照) 欠点は、 trac.ini をパースするのに使用されている ConfigParser クラスの制限事項として、リポジトリ名が常に小文字しか受け付けないということです。

下記の例では project, lib という 2 つのSubversion リポジトリが定義し、デフォルトのリポジトリとして project にエイリアスが設定されています。これは以前に Trac Environment が単一のリポジトリ (project リポジトリ) だったものを、複数のリポジトリに変換した際の典型的な使用例です。エイリアスは設定変更前から存在するリンクが project

リポジトリを解決できることを保証します。

```
[repositories]
project.dir = /var/repos/project
project.description = This is the ''main'' project repository.
project.type = svn
project.url = http://example.com/svn/project
project.hidden = true

lib.dir = /var/repos/lib
lib.description = This is the secondary library code.
lib.type = svn
lib.url = http://example.com/svn/lib

.alias = project
```

Note: `name.alias = target` の場合、 `name` を `target` リポジトリへのエイリアスにします。他の方法がありません。

データベースを使用した設定

リポジトリは（trac.ini ファイルではなく）データベース内でも設定することができます。 "バージョンコントロール"（英語版では "Version Control"）配下の "リポジトリ" 管理パネル、もしくは `trac-admin $ENV repository` コマンドを使用します。

管理パネルは Trac Environment
内に定義されているすべてのリポジトリのリストを表示します。リポジトリ、エイリアスの追加、リポジトリの属性の編集、リポジトリの削除ができます。
`trac.ini` に定義されているリポジトリもまた表示されますが、編集はできません。

以下の trac-admin コマンドはコマンドラインからリポジトリを操作する際に使用します。

`repository add <repos> <dir> [type]`

`<dir>` にあるリポジトリ `<repos>` を Trac に追加し、オプションとして、リポジトリのタイプを指定します。

`repository alias <name> <target>`

リポジトリ `<target>` のエイリアス `<name>` を設定します。

`repository remove <repos>`

リポジトリ `<repos>` を削除します。

`repository set <repos> <key> <value>`

リポジトリ `<repos>` の属性 `<key>` と `<value>` を設定します。

Note: デフォルトリポジトリはの名前は空文字列です。したがって、 `trac-admin`
をシェルから起動させたときは、おそらくクオートする必要があるでしょう。代替手段として、例えば `trac-admin`
を対話モードで起動しているときは、代わりに `"(default)"` を使用することができます。

リポジトリの同期

Trac 0.12 以前では、HTTP リクエストが発生するたびに Trac
が持つキャッシュとリポジトリの同期処理を行なっていました。このアプローチは効果的ではなく、複数のリポジトリを扱う上ではもはや実用的ではありません。こ
post-commit フックを利用した明示的な同期処理が追加されました。

リポジトリの変更を監視する拡張ポイントを定義するインタフェース（IRepositoryChangeListener）
が追加されました。このインタフェースの呼び出しは、チェンジセットが追加または修正されたときの post-commit
フックをトリガーとしています。コミット時に何かしらのアクションを実行するプラグインで使用することができます。

Mercurial リポジトリ

このドキュメントを書いている時点では、 Mercurial 導入時の同期作業やフックは必要ではありません - 詳細は 本家チケット 9485
を参照して下さい。

明示的な同期

これは推奨するリポジトリの同期方法です。 trac.ini の `[trac] repository_sync_per_request` オプションを empty
値に設定する必要があります。そして、各リポジトリの post-commit フック内で `trac-admin`

を呼び出すようにします。さらに、リポジトリでリビジョンのメタデータを変更することが許されているならば、 post-revprop-change フックにも同様に trac-admin を呼び出すように設定します。

changeset added <repos> <rev> […]

Trac に 1 つ以上のチェンジセットがリポジトリに発生したことを知らせる。

changeset modified <repos> <rev> […]

Trac に 1 つ以上のチェンジセットに対するメタデータの変更がリポジトリに加えられたことを知らせる。

引数 <repos> にはリポジトリ名 (デフォルトのリポジトリには "(default)" を使用する) または、リポジトリへのパスを指定します。

Note: もし PYTHON_EGG_CACHE のロケーションをデフォルトから変更している場合、 trac-admin を起動する前にウェブサーバと同じ値の PYTHON_EGG_CACHE を環境変数として設定しないといけないかもしれません。詳細については、 Trac プラグイン を参照して下さい。

下記は、 Subversion の完璧な post-commit と post-revprop-change スクリプトの一例です。特定の環境用に編集、そして適切な実行権を付けて各リポジトリの hooks ディレクトリにおいてください。 UNIX における (post-commit) 例:

```
#!/bin/sh
export PYTHON_EGG_CACHE="/path/to/dir"
/usr/bin/trac-admin /path/to/env changeset added "$1" "$2"
```

Note: Ubuntu では /usr/bin/trac-admin を指定せずに以下のように指定ができます:

```
#!/bin/sh
export PYTHON_EGG_CACHE="/path/to/dir"
trac-admin /path/to/env/ changeset added "$1" "$2"
```

Windows における (post-commit.cmd) の例:

```
@C:\Python26\Scripts\trac-admin.exe C:\path\to\env changeset added "%1" "%2"
```

Subversion の post-revprop-change フックはとても似たものになります。 UNIX における (post-revprop-change) の例:

```
#!/bin/sh
export PYTHON_EGG_CACHE="/path/to/dir"
/usr/bin/trac-admin /path/to/env changeset modified "$1" "$2"
```

Windows における (post-revprop-change.cmd) の例:

```
@C:\Python26\Scripts\trac-admin.exe C:\path\to\env changeset modified "%1" "%2"
```

上記の UNIX の変数は、 Subversion のコミットの実行ユーザが Trac に対して、書き込み権限があると仮定したもので、リポジトリと Trac の両方がウェブサーバを使って、動作している一般的な設定に基づいています。もし、ウェブサーバ経由以外での方法でリポジトリにアクセス (例: svn+ssh://) しているならば、 trac-admin を別の権限で起動 (例: sudo コマンド) しなければならないかもしれません。

Note: Subversion のフックで trac-admin を呼び出すことは、クライアントサイドにおいては、コミットとログの編集操作に時間がかかってしまうことになります。非同期な方法で行なうために、trac-admin で始まる contrib/trac-svn-hook を使いたいと思うかもしれません。スクリプトもまた何回もの安全性のチェックと使い方のアドバイスを行なっているので、フックを設定したりテストするのはずっ Windows 用の trac-svn-hook.bat に匹敵するものはありませんが、スクリプトは Cygwin の bash で起動することができます。

より詳しい情報は、Subversion 本の フックについて を参照して下さい。他のリポジトリのタイプでは異なるフックの設定が必要です。

Git フックは、 Git リポジトリの同期と同じように使用することができます。.git/hooks/post-commit に以下を追加します:

```
REV=$(git rev-parse HEAD)
trac-admin /path/to/env changeset added <my-repository> $REV
```

Mercurial では、 Trac にアクセスされるリポジトリ毎の .hgrc ファイルに以下を追加します。(TracMercurial が Trac の plugins ディレクトリにインストールされている場合、 hooks.py をダウンロードし、どこかアクセス可能な場所に格納してください) :

```
[hooks]
; If mercurial-plugin is installed globally
commit = python:tracext.hg.hooks.add_changesets
changegroup = python:tracext.hg.hooks.add_changesets

; If mercurial-plugin is installed in a Trac plugins directory
commit = python:/path/to/hooks.py:add_changesets
changegroup = python:/path/to/hooks.py:add_changesets

[trac]
env = /path/to/env
trac-admin = /path/to/trac-admin
```

リクエスト毎の同期処理

もし post-commit フックが使用できないならば、その環境ではリクエスト毎の同期処理を設定することができます。この場合、 trac.ini のオプション `[trac] repository_sync_per_request` に同期対象のリポジトリをカンマ区切りでリストしなければなりません。

Note:
この場合、チェンジセットのリスナの拡張ポイントは呼び出されません。それゆえ、使用しているプラグインは正しく動かないかもしれません。

単一リポジトリからの移行手順（Subversion）

下記の手順は、 Subversion における単一リポジトリから複数のリポジトリの設定へ変更するための典型的な移行方法になります。

1. `[trac] repository_dir` オプションからリポジトリのデフォルトの設定を削除する
2. メインとなるリポジトリを名前付きのリポジトリとして登録する
3. メインのリポジトリを再同期する

   post-commit フックと post-revprop-change フックをメインのリポジトリに設定し、 `[trac] repository_sync_per_request` オプションに empty 値を設定します
   1. デフォルトリポジトリとしてメインリポジトリに alias 属性を追加します。 (`name` は除外。つまり、`.alias = main`)。これは移行前に作成されたすべてのリンクがメインリポジトリを解決できることを保証します
   2. ステップ 2,3,4 を他の "名前付きの" リポジトリに対して必要に応じて繰り返します

単一リポジトリからの移行手順（Mercurial）

下記の手順は、 Mercurial
における単一リポジトリから複数のリポジトリの設定へ変更するための典型的な移行方法になります。このドキュメントを執筆している時点では、 Mercurial 導入時の同期作業やフックは必要ではありません - 詳細は 本家チケット 9485 を参照して下さい。

1. TracMercurial プラグインを最新バージョンにアップグレードします
2. デフォルトリポジトリの設定を trac.ini の `[trac] repository_dir` オプションから削除します
3. メインとなるリポジトリを名前付きのリポジトリとして追加します
4. デフォルトリポジトリとしてメインリポジトリに alias 属性を追加します。 (`name` は除外。つまり、`.alias = main`)。これは移行前に作成されたすべてのリンクがメインリポジトリを解決できることを保証します
5. ステップ 3 を他の "名前付きの" リポジトリに対して必要に応じて繰り返します

トラブルシューティング

trac-post-commit-hook はもはや機能しません

今では、 `tracopt.ticket.commit_updater.*` のオプションのコンポーネントを使用しなければなりません。 Web
からの管理の一般設定配下のプラグインパネルか trac.ini の [components]
セクションを直接編集することによって有効にすることができます。上記で説明している 明示的な同期 を確実に行なってください。