バージョン管理システム入門

（初心者向け）

Subversionの基礎勉強
～Subversionによるバージョン管理を使う～

はじめに

tracpath

ようこそ、バージョン管理システム「Subversion」の基礎勉強です。バージョン管理システムの歴史は古く汎用機の時代からプログラムのソースコード管理に利用されてきました。現在はオープンソースで多くのバージョン管理システムがあり、SubversionやCVSのように集中管理型（クライアント・サーバ型）や、Git/Mercurial/Bazzarなどの分散型と呼ばれるバージョン管理システムがあります。

「Subversionの基礎勉強」では、Windows7のコマンドプロンプトで Subversionの基本的なコマンドを自分のマシンで動作させます。Subversionはこのチュートリアルで説明している以外にも多くの機能を備えたパワフルなバージョン管理システムです。使いこなすためには文章を読むだけではなく実際に試すことがとても重要です。

このドキュメントにならってコマンドを実行することで、基本的なSubversionのコマンドと使い方を学習することができます。

レッスン１．インストール

tracpath

最初にSubversionが利用できるように環境を作ります。Apache Subversionのサイトからインストール用のバイナリ（インストーラー）をダウンロードします。こちらのページ下部に移動してWin32Svn（32-bit client, server and bindings, MSI and ZIPs）をダウンロードしてください。
2013/05/21 現在、[Setup-Subversion-1.7.9.msi] がダウンロードされました。

ダウンロードしたインストールファイルをクリックしてウィザードに従って「次へ」をクリックすればインストールは完了します。

レッスン２．動作確認

インストールが完了したら動作の確認をしてみましょう。SubversionはCUIツールのためコマンドラインから利用します。コマンドプロンプトを起動してください。コマンドプロントからsvnを入力します。（Subversion のコマンドはすべて [svn] となります。）

```
C:¥>svn
使用方法を知りたいときは 'svn help' と打ってください。

C:¥>
```

```
C:¥>svn help
使用方法: svn <サブコマンド> [<オプション>] [<引数>]
...
C:¥>
```

動作しているようです。これでインストールと動作確認は完了です。次から早速Subversionをつかったバージョン管理を行います。

Note: 通常、インストーラーからインストールするだけでコマンドプロンプトからsvnコマンドが利用できるようになります。利用できない場合はご利用のPCの環境変数に「C:Program FilesSubversionbin」を追加しなければ行けないかもしれません。Subversionのインストールしたパスです。

レッスン３.リポジトリ用のディレクトリを作成

これでSubversionが利用できるようになりました。まだ開発に入ることはできません。Subversion でバージョン管理を行うためにリポジトリを作成します。

リポジトリ:Subversionで管理されるソースコードやファイルを格納する場所のことです。この格納場所を作成しなければ開発に進むことはできません。通常リポジトリの作成作業は最初の1回だけで良いです。

それでは、作成します。今回は以下の構成にします。 リポジトリ用: c:workrepos

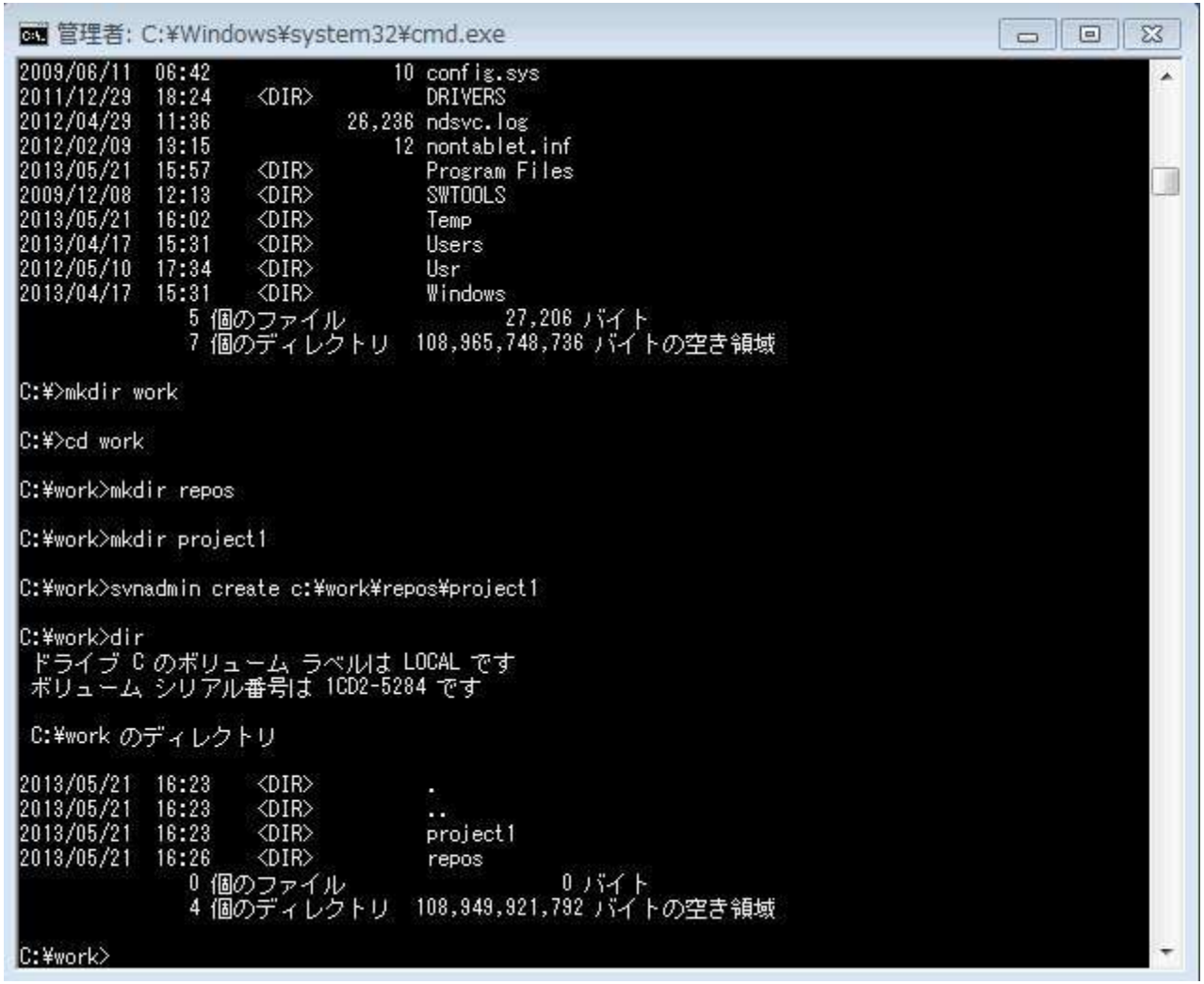
```
C:¥>mkdir work

C:¥>cd work

C:¥work>mkdir repos
```

reposフォルダはプロジェクト毎のリポジトリを格納するフォルダとして利用します。リポジトリを作成します。

```
C:¥work>svnadmin create c:¥work¥repos¥project1
```

警告 ： project1で利用するリポジトリを作成します。通常、リポジトリの作成は最初の1回だけです。

レッスン３．リポジトリ用のディレクトリを作成



レッスン４．最初のインポート

tracpath

Subversionで開発を進める場合の作法として、[trunk][tags][branches]フォルダをリポジトリに追加します。また、新規ファイル [readme.txt] を作成してインポートします。今回はインポートするフォルダとファイルを自分で用意しますが、既存のソースコードを利用しても良いです。また、tmp フォルダの配下にインポート対象のフォルダやファイルを作成しましたがリポジトリ登録後、この tmpフォルダは削除する予定です。

```
C:¥work>mkdir tmp
C:¥work>mkdir tmp¥trunk      #Subversionの作法で作成するフォルダ
C:¥work>mkdir tmp¥tags       #Subversionの作法で作成するフォルダ
C:¥work>mkdir tmp¥branches   #Subversionの作法で作成するフォルダ
C:¥work>echo "Sample Project readme file" > tmp¥trunk¥readme.txt
```

次に、インポート用コマンドを実行し、先ほど作成したリポジトリにインポートします。

```
C:¥work¥tmp>svn import file:///c:/work/repos/project1 -m "initial commit."    # -m はコメントを意味します。
追加しています          tags
追加しています          trunk
追加しています          trunk¥readme.txt
追加しています          branches

Committed revision 1.
```

レッスン４．最初のインポート

これで、リポジトリにインポートすることができました。Windowsで注意する点は、インポート時にリポジトリのパスを指定するとき「file:///c:/work/repos/project1」URL表記になっている点です。パスを間違えると以下のようなエラーになることがありますので注意してください。

```
C:¥work¥tmp>svn import c:¥work¥repos¥project1  -m "initial commit."
svn: E205000:  より詳しく知りたいときは 'svn help' を試してみてください
svn: E205000:  'C:/work/repos/project1'  は不正な URL です
```

インポートに利用した、tmpフォルダは削除しておきます。すでにリポジトリに登録されているため削除しても問題ありません。

```
C:¥work>cd tmp

C:¥work¥tmp>cd ..

C:¥work>del tmp
C:¥work¥tmp¥*、よろしいですか (Y/N)? yes
```

レッスン５．チェックアウト

チェックアウトは、リポジトリに登録されたソースコードを開発環境（自分の環境）にもってくることで開発作業を進めます。自分の環境にコピーを取得（今後、作業コピーといいます）するためのコマンドをチェックアウトと言います。

それでは、実際にプロジェクト用のフォルダに移動して作業コピーをチェックアウトします。

```
C:¥work>

C:¥work>svn checkout file:///c:/work/repos/project1
A    project1¥trunk                #A はファイルが追加されたことを意味します。
A    project1¥trunk¥readme.txt
A    project1¥branches
A    project1¥tags
リビジョン 1 をチェックアウトしました。
```

レッスン６．作業ディレクトリで作業開始

チェックアウトが完了しました。これからこのチェックアウトされた作業コピーフォルダで開発を進めていくことになります。 c:workproject1 を見てみます。

```
C:¥work¥project1 のディレクトリ
2013/05/21  17:52    <DIR>          .
2013/05/21  17:52    <DIR>          ..
2013/05/21  17:52    <DIR>          branches                  # リポジトリから取得
2013/05/21  17:52    <DIR>          tags                     # リポジトリから取得
2013/05/21  17:52    <DIR>          trunk                     # リポジトリから取得

C:¥work¥project1>dir trunk

C:¥work¥project1¥trunk のディレクトリ
2013/05/21  17:52    <DIR>          .
2013/05/21  17:52    <DIR>          ..
2013/05/21  17:52                31 readme.txt               # リポジトリから取得

C:¥work¥project1>type trunk¥readme.txt                       # インポートしたファイルの中身が確認できます
"Sample Project readme file"
```

レッスン７．新しいファイルを作成・追加

tracpath

リポジトリから作業コピーを取得しました。次にファイルを追加してみます。使い慣れたエディタを使って、以下のファイルを作成します。 [tutorial.html]を作成します。

```
 1 | <html>
 2 |   <body>
 3 |
 4 |   <h1>Subversion チュートリアル</h1>
 5 |   <ol>
 6 |     <li>リポジトリ作成</li>
 7 |     <li>作業コピー作成</li>
 8 |     <li>開発作業</li>
 9 |     <li>コミット</li>
10 |   </ol>
11 |
12 |   </body>
13 | </html>
```

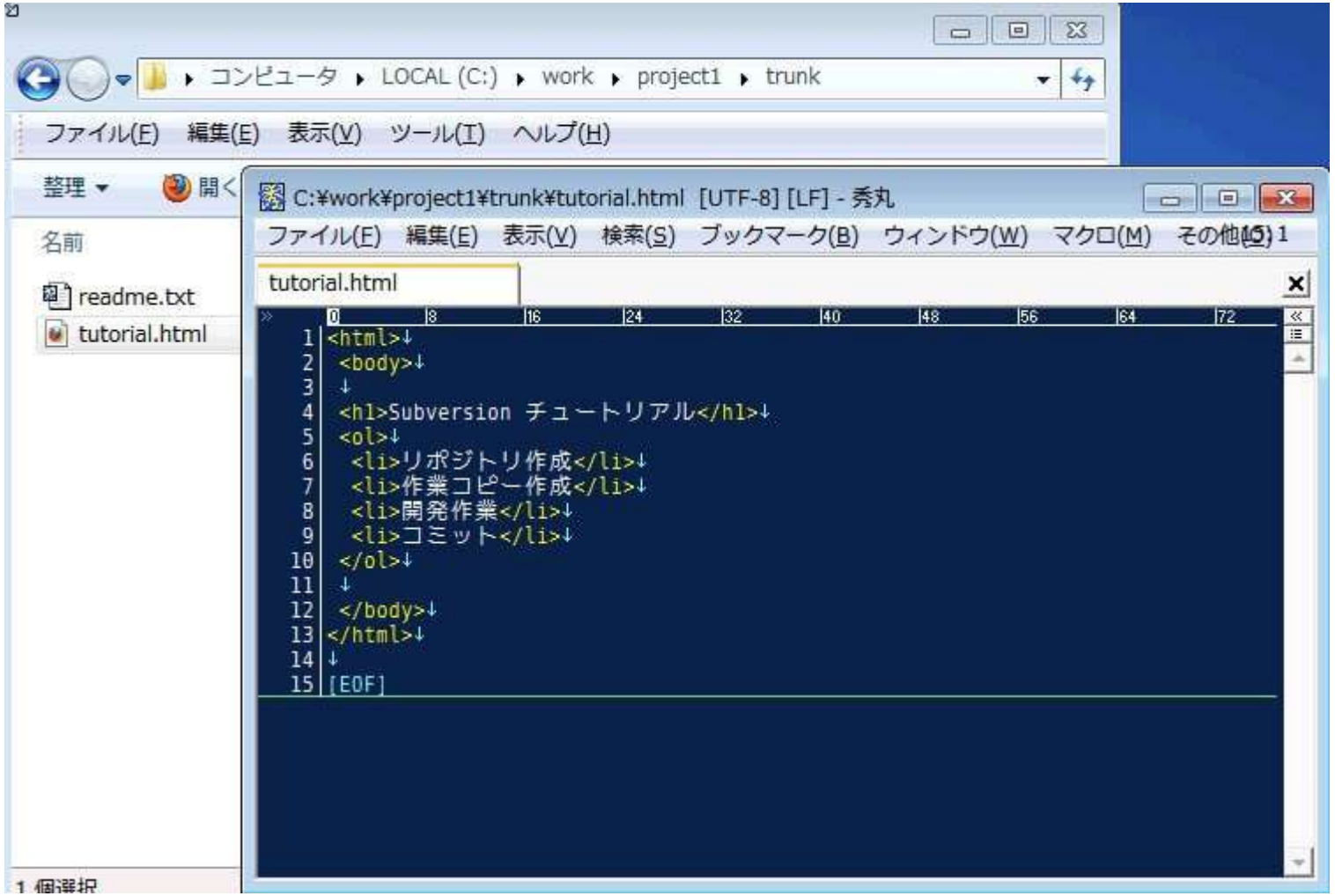
コマンドプロンプトからリポジトリにファイルを追加するコマンドを実行します。

```
C:¥work¥project1¥trunk>svn status     # status でリポジトリと作業コピーの状態を確認します。？ はリポジトリ管理外。
?       tutorial.html
```

これで、作業コピーで新しく追加ファイル「tutorial.html」をリポジトリにコミットして追加しました。他のメンバーがリポジトリから作業コピーをチェックアウトした場合、tutorial.htmlが追加された状態でチェックアウトされます。

警告 ： コミットとは、作業コピーの変更した内容をリポジトリに送ります。コミットして初めてリポジトリに反映されます。

レッスン７．新しいファイルを作成・追加



レッスン８. 新しいディレクトリを作成・追加

tracpath

ディレクトリの追加やディレクトリに含まれるファイル群の追加をやっていきます。やり方はファイルの追加と同じです。

フォルダA,BとBフォルダに2つのテキストファイルを作成します。

```
C:¥work¥project1¥trunk>mkdir A
C:¥work¥project1¥trunk>mkdir B
C:¥work¥project1¥trunk>echo hello world! > B¥hello.txt
C:¥work¥project1¥trunk>echo こんにちは > B¥こんにちは.txt
```

svn statusで状態を確認します。

```
C:¥work¥project1¥trunk>svn status
?       A
?       B
```

レッスン８. 新しいディレクトリを作成・追加

tracpath

フォルダAとフォルダBをリポジトリに追加するコマンドを実行します。フォルダBにはテキストファイルが含まれていることに注意してください。

```
C:¥work¥project1¥trunk>svn add A
A         A

C:¥work¥project1¥trunk>svn add B
A         B
A         B¥hello.txt
A         B¥こんにちは.txt
```

ファイルを追加したときと同じようにコミットします。「-m」はコミット時のコメントです。後で見て分かりやすいように変更点や修正内容を書くようにするべきです。

```
C:¥work¥project1¥trunk>svn commit -m "A,Bフォルダを一括登録した"
追加しています              A
追加しています              B
追加しています              B¥hello.txt
追加しています              B¥こんにちは.txt
ファイルのデータを送信しています ..
Committed revision 3.
```

レッスン９．ファイルを更新

tracpath

それでは、ファイルの中身を編集してバージョン管理システムの特長を見ていきましょう。テキストエディタで trunk/readme.txt を開きます。2行追加します。

```
<li>「-m]はコミット時の更新内容を記述します</li>
<li>バージョン管理システムの更新</li>
```

ファイルを保存して閉じた後、コマンドプロントに戻ります。ファイルの状態を確認してください。

```
C:¥work¥project1¥trunk>svn  status
M       tutorial.html
```

編集したファイルに「M」というマークが付いています。これは作業コピー内のファイルに変更があったことを示しています。 続いてコミットを行います。

```
C:¥work¥project1¥trunk>svn  commit -m "手順を追加した"
送信しています              tutorial.html
ファイルのデータを送信しています .
Committed revision 4.
```

もう一度同じファイルに以下の行を追加します。

```
<li>ファイルの更新</li>
```

レッスン９．ファイルを更新

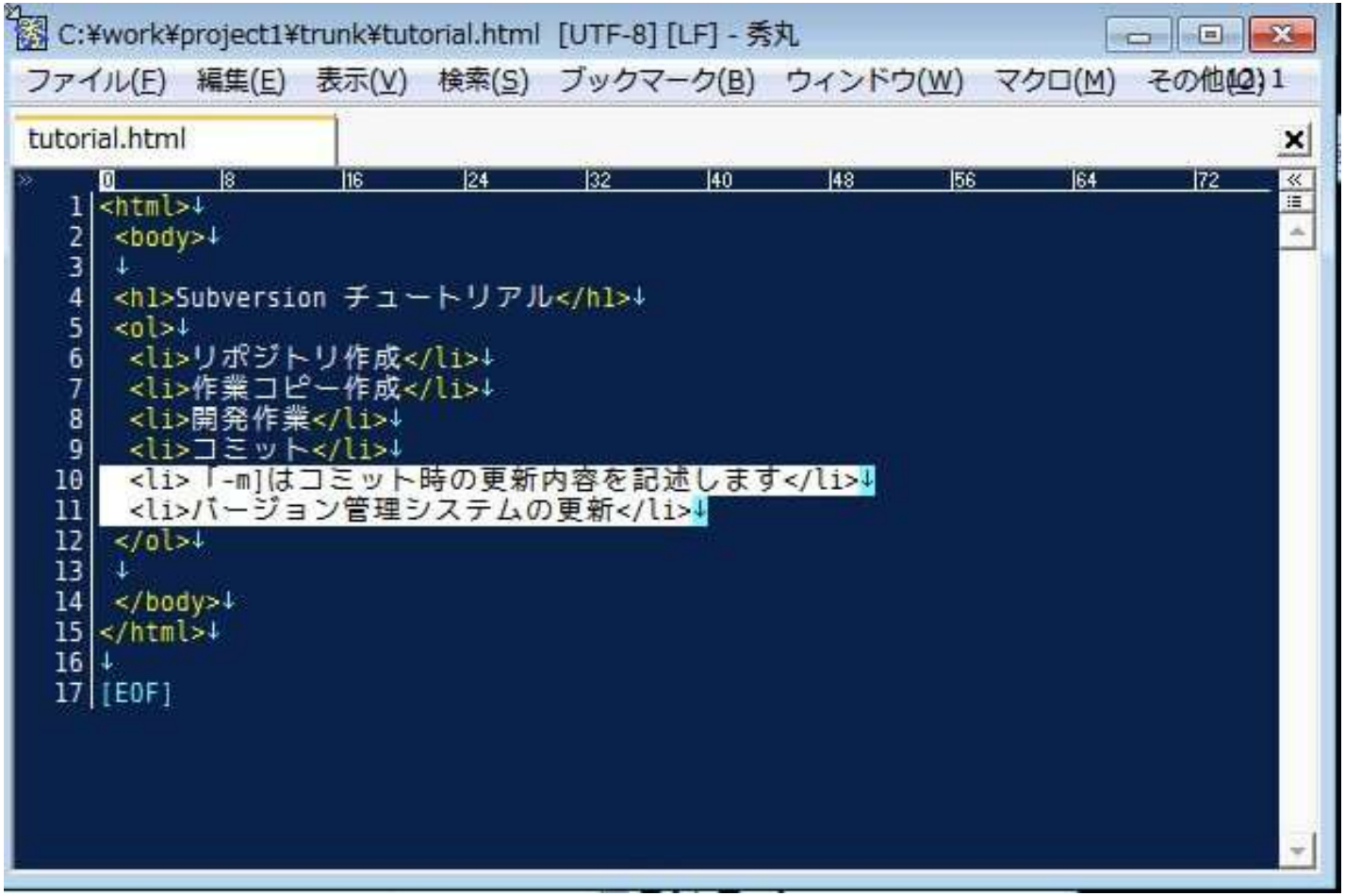
tracpath

今編集しているファイルはリポジトリのファイルより新しい修正が追加されています。コミットする前にその内容を確認しています。

```
C:¥work¥project1¥trunk>svn  diff tutorial.html
Index: tutorial.html
===================================================================
=======
--- tutorial.html       (リビジョン 6)
+++ tutorial.html       (作業コピー)
@@ -9,6 +9,7 @@
   <li>コミット</li>
   <li>「-m]はコミット時の更新内容を記述します</li>
   <li>バージョン管理システムの更新</li>
+  <li>ファイルの更新</li>                    # リポジトリと作業コピーの diff（比較）
  </ol>                                    # 作業コピーのみ「+」追加された行があることを示しています。

  </body>
```

レッスン９．ファイルを更新

修正内容を確認したらコミットします。コミットは何度もやっているので同じようにsvn commit-m “コメント” で実行します。



レッスン１０．ファイルを移動

tracpath

ファイルの移動はエクスプローラーを利用しておこなうとリポジトリと作業コピーの差異が生まれます。これはディレクトリが不完全になることを意味しています。リポジトリと作業コピーは同期されていることが必要です。例えば、/project1/trunk/tutorial.htmlをエクスプローラーで/project1 配下に移動した場合、svn statusはどうなるか見てみると、

```
C:¥work¥project1>svn status
!       trunk¥tutorial.html         # アイテムが失われた。ディレクトリが不完全な状態
?       tutorial.html               # リポジトリ管理外のファイルを見つけた
```

ファイルの移動は以下のように行います。

```
C:¥work¥project1>svn move trunk¥readme.txt .        # 移動元と移動先（ここでは、/trunk 配下）を指定
A       readme.txt                  # ファイルが追加
D       trunk¥readme.txt            # ファイルが削除
```

この操作もコミットしなければリポジトリに反映されないことに注意してください。

> **警告 ： ファイルの移動はsvn copyの後にsvn delete を実行することと同じです。**

レッスン１１．ファイル名を変更

tracpath

ファイル名の変更もファイルの移動と同じでエクスプローラーではなく、svn rename で変更する必要があります。

```
C:¥work¥project1¥trunk>svn rename tutorial.html tutorial2.html
A         tutorial2.html              # ファイルが追加
D         tutorial.html               # ファイルが削除
```

コミットは必要ですので忘れずに実行します。

レッスン１２．ファイルを削除

ファイルの削除を実行します。

```
C:¥work¥project1¥trunk>svn  delete tutorial2.html
D         tutorial2.html
```

コミットは必要です。

```
C:¥work¥project1¥trunk>svn  commit -m "tutorial2.html を削除"
削除しています                tutorial2.html

Committed revision 10.
```

ファイルの移動、ファイル名の変更、ファイルの削除はバージョン管理されているファイル群のみ使うことができます。

レッスン１３. 履歴を確認

tracpath

最後にこれまでの変更履歴を確認してみます。 履歴を確認する方法として、2つのコマンドを覚えておけば良いです。

svn log　ファイルやディレクトリの履歴情報を確認することができます。チュートリアルの trunk フォルダで確認します。

```
C:¥work¥project1>cd trunk

C:¥work¥project1¥trunk>svn log
------------------------------------------------------------------------
r4 | syoji | 2013-05-21 18:35:56 +0900 (火, 21 5 2013) | 1 line

手順を追加した
------------------------------------------------------------------------
r3 | syoji | 2013-05-21 18:18:27 +0900 (火, 21 5 2013) | 1 line

A,Bフォルダを一括登録した
------------------------------------------------------------------------
r2 | syoji | 2013-05-21 18:07:33 +0900 (火, 21 5 2013) | 1 line

tutorial 新しいファイル追加した
------------------------------------------------------------------------
r1 | syoji | 2013-05-21 17:05:04 +0900 (火, 21 5 2013) | 1 line

initial commit.
------------------------------------------------------------------------
```

svn diff　レッスン9. ファイルの更新で利用しています。ファイルの差分を表示することができます。

最後に

tracpath

これでSubversionの基本学習用チュートリアルは終了します。よく使うコマンドを中心に説明していますが、チームでの複数メンバーによる開発やSubversion/TortoiseSVNの高度な機能については説明していません。

このチュートリアルはいったんこれで終了します。さて、次は何をすればよいでしょうか？バージョン管理のアプリケーションを理解する一番の近道は実際に使ってみることです。あなたの開発で利用しはじめてください。あなたが所属する会社やチームでバージョン管理を使っていなかったり、別のツールを使っている場合があるかもしれません。そんなときでも、ひとりでSubversion/TortoiseSVNを使うことは可能です。

さらに、ネットには数多くの良質なコンテンツがたくさんあります。Subversion/ TortoiseSVNの専門書籍も多く出版されています。ぜひ参照して快適な開発ライフを。

参考資料

tracpath

- Subversion によるバージョン管理（1.4）

- Apache Subversion

- リモートリポジトリを使うなら、tracpath（トラックパス）が便利です！
  下記記事をぜひご参照下さい。
  ■tracpath（トラックパス）を使って、安全に複数名でバージョン管理を行う

tracpath（トラックパス）のご紹介

社内サーバにリモートリポジトリを作るのも一つですが、「開発にまつわる面倒事」をこの際全部、tracpath（トラックパス）に任せてみませんか？
バージョン管理サービス・プロジェクト管理サービスの「tracpath（トラックパス）」では、
ユーザー５名、リポジトリ数３つまで、永久無料で利用可能です。
さっそく実務でも使って見ましょう。
自らも開発を行う会社が作ったからこそ、開発チームの「作る情熱」を支える、やるべきことに集中出来るサービスになっています。
エンタープライズ利用が前提のASPサービスなので、セキュリティも強固です。