Subversion 入門

- Subversion でレポートを持って帰ろう -

Outline

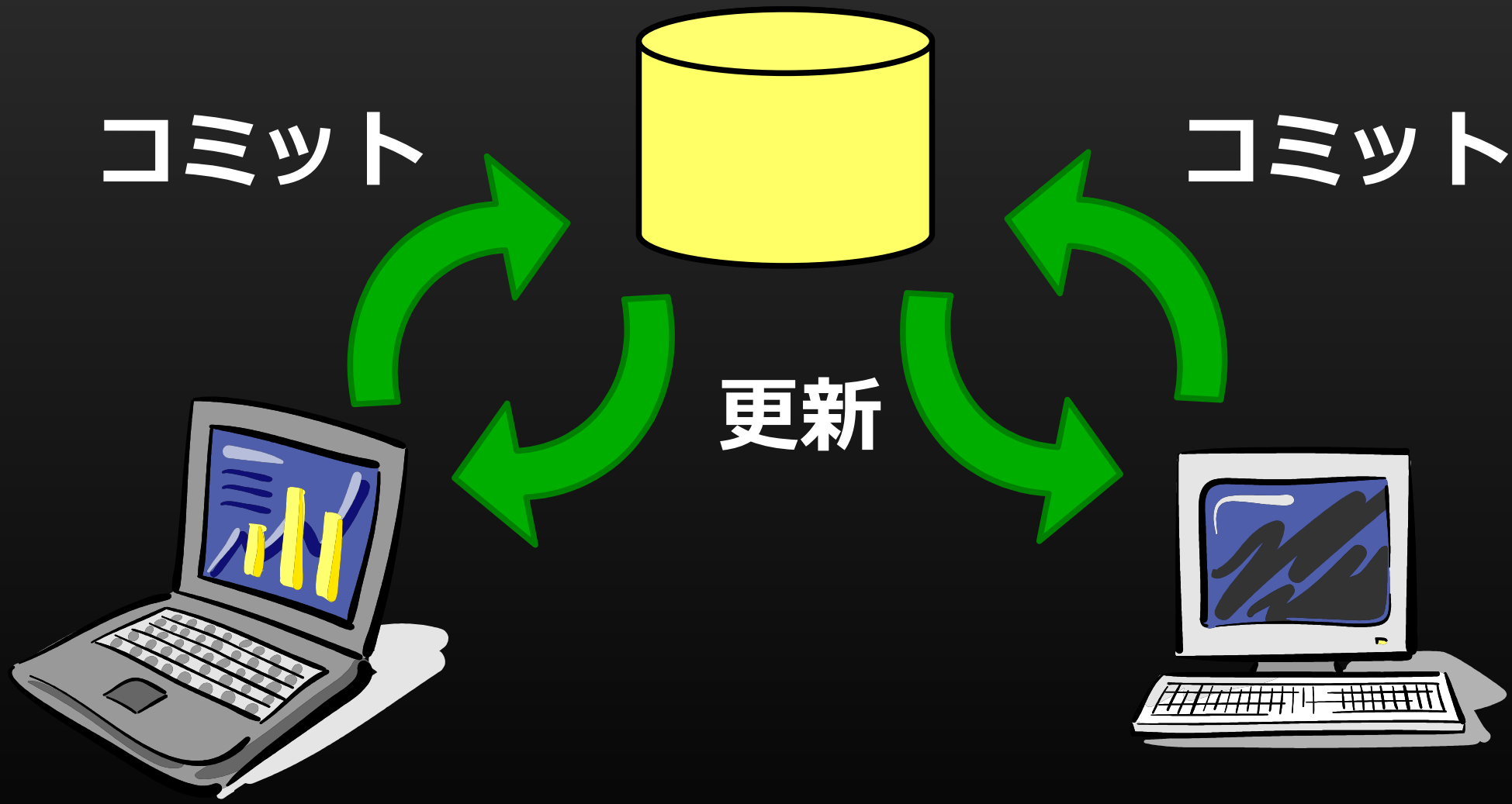
- はじめに
- 基本的な概念
- **実践**
- こんなこともできます

はじめに

Subversion って？

- 複数マシン間で
  - サーバを経由して
  - **指定されたファイルを同期するツール**
- **更新履歴を記録**

今日の目標！



USB メモリ比の メリット

- **手間が少ない！**
  - 抜き差し不要
  - クリック数回で同期（Add，Commit，Update）
- **更新履歴が残る**
  - 巻き戻し可能 etc
- バックアップしてます

USB メモリ比の デメリット

- **サーバが落ちたら使えない**
  - ネット切れ，停電，etc
- 冗長性：大事なファイルは複数の経路で運ぶ！
  - メール，Subversion，USB メモリ
- Add し忘れ，Commit し忘れ

基本的な概念

必要なもの

- サーバ
  - ファイルサーバにて **稼働中** です
- クライアント
  - Windows なら **TortoiseSVN**
  - Linux なら **svn** コマンド
  - **ブラウザ**からもアクセス可能（読み取りのみ）

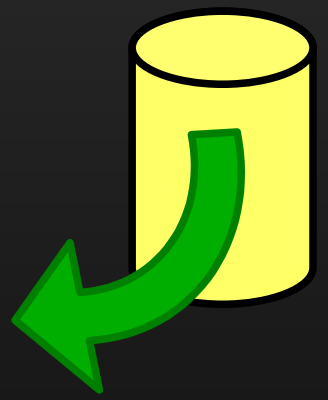
専門用語集

- 眠くなるけど，避けて通れない道

  - **リポジトリ　　… データベース（格納所）**
  - **リビジョン　　… バージョン番号**
  - チェックアウト　… リポジトリから取得
  - コミット　　… リポジトリに保存
  - 更新　　… 最新のリビジョンにする

サーバから取得

- **チェックアウト（checkout）**
  - サーバから，リポジトリを取得
  - 最初の一回だけ必要



- 個人別のリポジトリを用意してます
  - 好きなように使ってください.

ファイルを管理対象にする

- **追加（add）**
  - フォルダにあるファイルが自動的に管理対象になるわけではない
  - 明示的に，Add して**管理対象にする**

ファイルをサーバに保存

- **コミット（commit）**
  - ファイルの変更が自動で**保存**されるわけではない
  - 一定の区切り毎に，明示的に Commit する.
  - **要は，アップロード**
- USB メモリへのコピーに相当

ファイルを同期する

- **更新（update）**
  - 別のところでサーバに保存したファイルを**同期**する
  - これも，明示的に作業しないとだめ.
  - **要は，ダウンロード**
- USB メモリからの取り出しに相当

まとめ：作業の流れ

1. **アップデート**（サーバから更新する）
2. 編集する
3. **コミット**（サーバに更新を保存）

実 践

実践：レポートの持ち帰り

1. チェックアウト（Checkout）　最初に一回

2. 整理のために，ディレクトリを作る

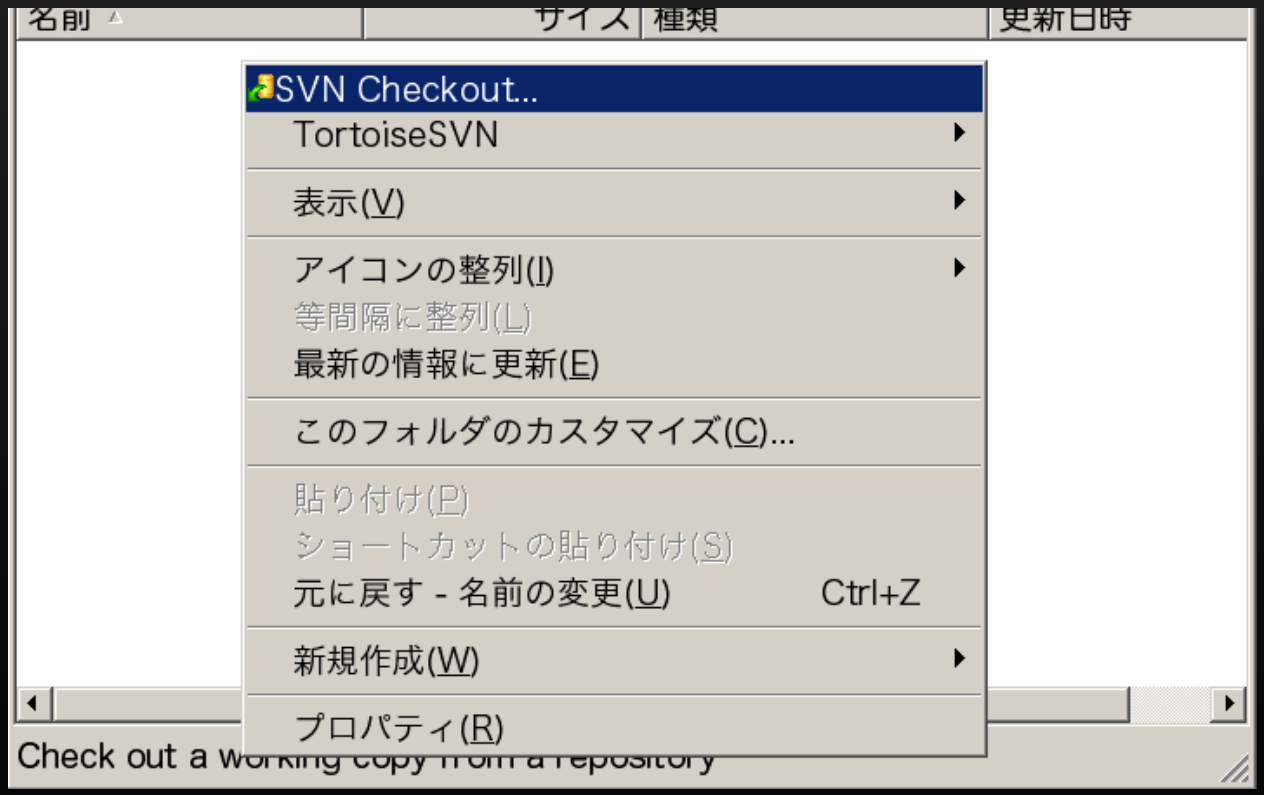
3. 追加（Add）　研究室

4. コミット（Commit）：サーバに保存

5. 更新（Update）：サーバから読み出し　家

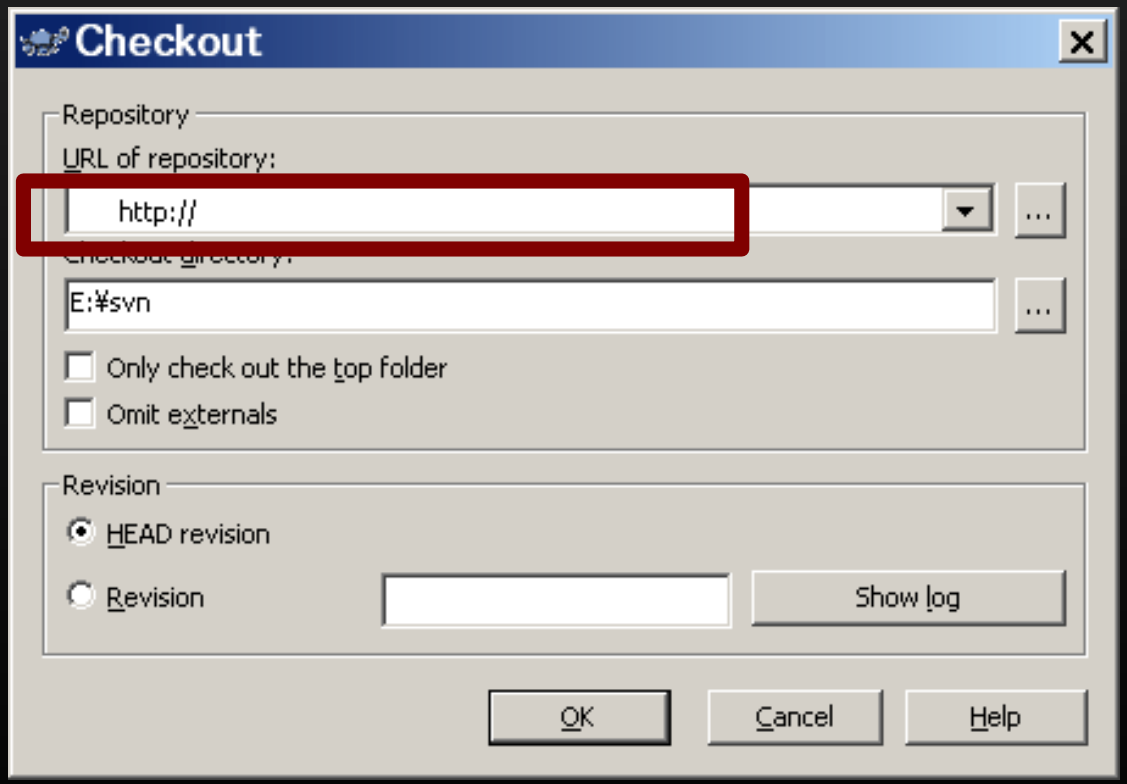
checkout する - 1

- パソコン内の好きな場所にディレクトリを作る
- そのディレクトリを開いて，右クリック



checkout する - 2

- checkout するリポジトリのアドレスを入力
  - http://hoge/foo/bar



チェックアウトした先へ移動

- このように，ディレクトリにマークが重なっているのが管理対象になっている証です

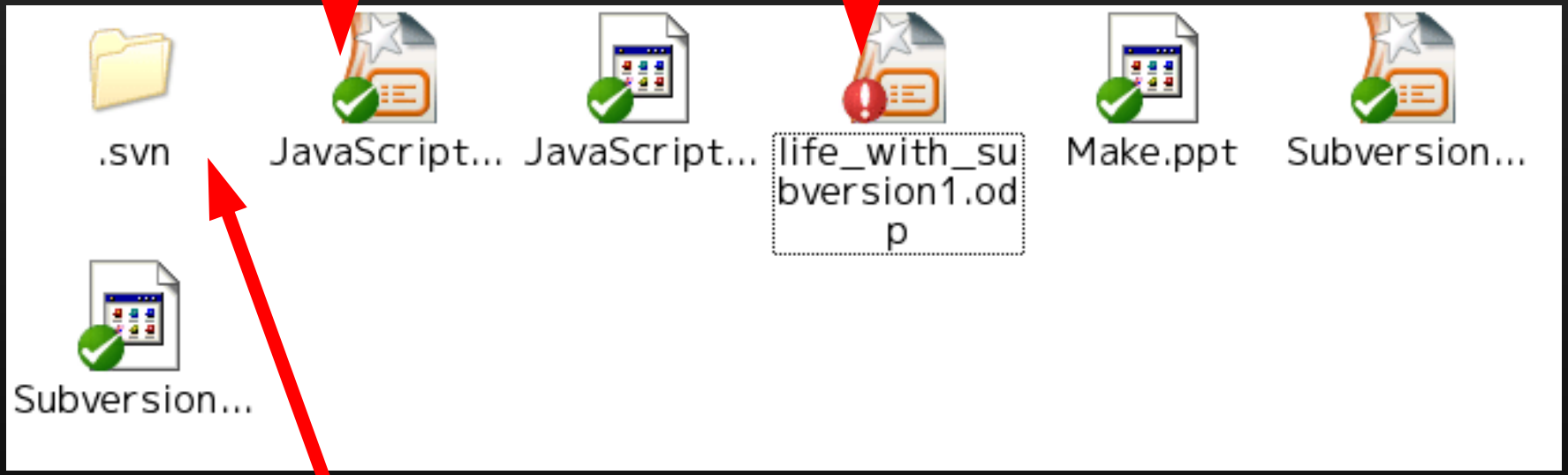
変更なし

変更したけど、
サーバには保存してない

エクスプローラ拡張

変更したけど、
サーバには保存してない

変更なし



管理情報を保存する隠しディレクトリ

ブラウザからのアクセス

- リポジトリは，ブラウザからもみれます
  - http://hoge/foo/bar
- 後の作業で使うので，開いてください

ディレクトリを作って Add

- チェックアウトしたディレクトリで **report** という名前のディレクトリを作る
- **作ったディレクトリを Add（追加）**
- ブラウザからリポジトリを確認
  - http://hoge/foo/bar
  - サーバにはまだディレクトリがありませんね？

コミットしてみよう

- **右クリックして，Commit（コミット）**
- **ログは好きなように書く**
  - **空白可**だけど、書いておいたほうが後から便利
- ブラウザからリポジトリにアクセスして確認
  - 新しいディレクトリが増えましたか？

ファイルもやってみよう

- Report ディレクトリに ファイルを作って
- 追加（Add）して
- コミット（Commit）する（サーバに保存）

ふつうは，複数まとめてやる

- 今回は説明のために一つずつ作業しましたがディレクトリを対象としたとき
- Update / Add / Commit などのコマンドはディレクトリ以下すべてが対象になります
- 個別に選ぶこともできます

自宅で読み込む

- 一回目だけ：リポジトリをチェックアウト
- 二回目以降：リポジトリを**更新**（Update）

継続して使おう

- 早速，レポートの持ち運びに使ってください
- 研究関係のファイルにも使ってみてください
- **ファイルの同期だけでも便利ですんで**

注意点：日本語ファイル名は？

- 日本語ファイル名も一応通りそうなのですが
- 英数字のみを推奨しておきます

注意点：ファイルサイズの制限

- **特にもうけてませんが**
- 共用リポジトリにはあんまり大きいファイルを入れないでください
  - 学校の回線が細く，でかいファイルをおいちゃうとアップデートに時間がかかるので
- **MP3 とかそういうのは勘弁**

まとめ

- Subversion は バージョン管理ソフト
- まずは チェックアウト
- 後は以下のループ
  - 追加
  - コミット
  - アップデート

こんなことも

できます

ファイルの移動

- **ファイル／フォルダの移動（Move）**
  - ディレクトリを管理対象にしてから
  - 右ボタンを押しながらドラッグして
  - 出てきたメニューで操作
- **ファイル名の変更 (Rename)**
  - 右クリックメニューから選ぶ

巻き戻し (Revert)

- たとえば
  - ソースコードを編集したら動かなくなった…
  - 移動しようと思ったけど，やっぱりやめる
- **巻き戻し (Revert)**
  - サーバに保存されている状態に戻せる
  - 差分をみて編集しても良いのですが，時によります

特定 Revision の取得

- Revision = リポジトリ の バージョン
- 過去にコミットしたファイルは全て取り出せる
  - 「昨日の状態に戻す」とかが可能
  - やりたいときは調べてください
- Update Revision to...

衝突 (conflict)

- 複数のコミットが重なると、衝突することが
  - hoge.txt … 自動で整合しようとした結果
  - hoge.txt.r001 … Revision 1 の hoge.txt
  - hoge.txt.r004 … Revision 4 の hoge.txt
  - hoge.txt.mine … 手元にあった hoge.txt
- これらを元に、望ましい hoge.txt を作って、残りを削除して、コミットする

ログを見たり，差分をみたり

- ログ (Log) を見たり
- 差分 (Diff) を見たり

お つ か れ

さ ま で し た