The PracTEX Journal, 2007, No. 3
Article revision 2007/08/17

LaTeX Document Management with Subversion

Uwe Ziegenhagen

Email latex@ziegenhagen.info
Website http://www.uweziegenhagen.de
Address Humboldt-Universitt zu Berlin, Germany
Center for Applied Statistics and Economics

Abstract From the single-author composition of a Bachelor thesis to the creation of a book by a team there are many occasions, where version management of a document may be helpful. With the aim of overcoming the shortcomings of CVS (Concurrent Version System) the Subversion version control system was implemented.

In this article I will describe the Subversion setup on Windows and Linux systems, the elementary steps of document management and various LaTeX packages working hand in hand with Subversion.

1 CVS versus Subversion

Contrary to CVS the versioning scheme of Subversion does not refer to single files anymore but to a whole tree of files. Each revision number n refers to the state of the repository after the n-th commit. When we speak about a file in revision 4 we mean the file in the state of revision 4.

The revision numbers of a single file may even have gaps if it had not been changed on every commit to the repository. Table 1 illustrates an example: Up to revision 4 all files have been changed before each commit, so the revision of the repository and the revision numbers of the files are equal. Before the commit to revision 5 only chapter1.tex is modified however the whole repository receives the revision number 5, before the commit to version 6 all files were modified again and therefore have the revision number 6.

On each checkout from a Subversion repository the highest revision number of each file will be checked out which is smaller or equal to the desired revision

Revision 4	Revision 5	Revision 6
thesis.tex:4	thesis.tex:4	thesis.tex:6
preamble.tex:4	preamble.tex4	preamble.tex:6
chapter1.tex:4	chapter1.tex:5	chapter1.tex:6

Table 1: Gaps in Subversion revisions

number. Subversion stores a second copy of each file in a special directory (`.svn`) on each checkout, update and commit.

Although this doubles the required space on the hard-disk it has certain advantages, especially when dealing with remote repositories: Viewing local changes can be made without access to the network, when committing a file, Subversion has only to send the changed parts whereas, CVS calculates the changes on the server and has to send the whole file on each commit. Commits are atomic, which means a change to a file is either completely stored or not stored at all. Thus network issues or concurrent commits cannot lead to inconsistent status.

2 Installation

There are different options for the installation of Subversion. One can either use svnserve [3] or install Subversion as an Apache 2 module, which uses WebDAV[1].

In this article I will use the latter option by installing Subversion as an Apache2 module, since the integration into Apache 2 provides a few interesting features such as the possibility of surfing through repositories in a web browser and the use of Apaches authentification mechanisms.

1. *Web-based Distributed Authoring and Versioning*, a set of extensions to the HTTP protocol which allows users to collaboratively edit and manage files on remote web servers.

2.1 Windows XP

2.1.1 Apache Setup

Binary versions of Apache 2 are available from [1], however I usually prefer to use a WAMP[2]-solution provided by apachefriends.org. We extract the xampp.zip[3] to e.g. `C:/xampp` and start the Apache server using `xampp-control.exe`. When we open `http://localhost` in a web browser the XAMPP start page should be displayed as shown in Figure 1.

Figure 1: Screenshot of xampp starting page

As we assume that only the local computer should be allowed to access the webserver it is strongly recommended securing against access from outside. For details please see the respective chapter in the Apache documentation [7].

2. Windows-Apache-MySQL-PHP
3. current version at printtime: 1.6.2

2.1.2 Subversion

We download Subversion[4] from [2] and extract all files from the zip archive to e.g. `C:/Program Files/Subversion`. After adding the path to the `C:/Program Files/Subversion/bin` directory to the PATH environment variable from Windows, we can call `svn help` from the commandline to check if our installation is working. In the next step we copy `mod_authz_svn.so` and `mod_dav_svn.so` from the `subversion/bin` directory to the Apache modules directory and overwrite older versions of this file if necessary.

In the final step we enable WebDAV and the Subversion module by adding

- `LoadModule dav_svn_module modules/mod_dav_svn.so` and
- `LoadModule authz_svn_module modules/mod_authz_svn.so`

to the `httpd.conf` in the Apache `/conf` directory. Before we restart Apache to load the modules we need to make further adjustments to this file. We create a root directory for all our repositories (e.g. `c:/allMyRepositories`) and add the code from Listing 1 to `httpd.conf`:

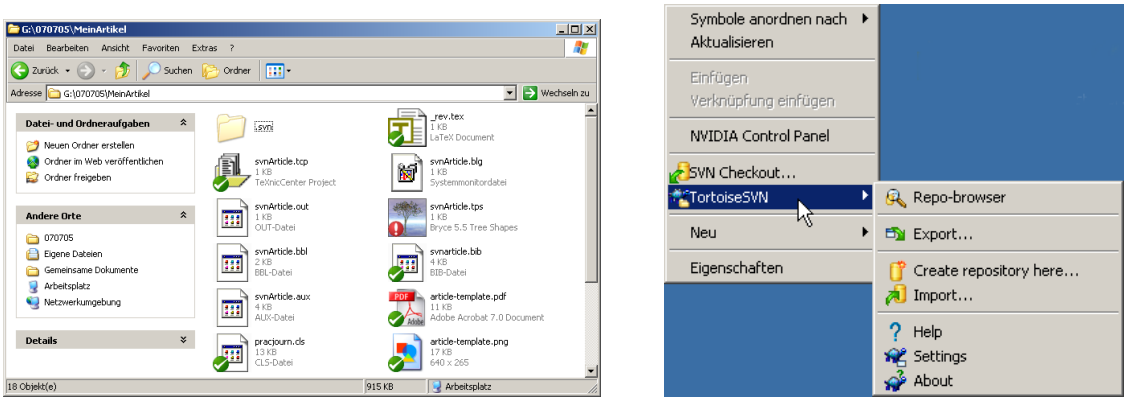
```
1 <Location /svn>
2 DAV svn
3
4 SVNParentPath c:/allMyRepositories
5 </Location>
```

Listing 1: Setup code for the Windows repository root

Using the command line we go to `c:/allMyRepositories` and create our first repository by executing `svnadmin create firstSample`.

If we now open `http://localhost/svn/firstSample/` in a browser we should see an empty directory listing with the headline `Revision 0: /`. The basic installation of Subversion is now done, however we can achieve a much more convenient way of handling repositories by installing TortoiseSVN.

4. current version at printtime: 1.4.4



(a) (b)

Figure 2: Screenshot of a Working Directory with Subversion installed and context menu of Subversion 1.4.4.

2.1.3 TortoiseSVN

TortoiseSVN[5] [4] is a free Subversion client, implemented as a Windows shell extension. It features a multilingual interface with Windows Explorer integration, its icon overlays show immediately which files/folders have been changed and need to be committed to the repository. The installation is straightforward, after rebooting the computer we find various entries in the context menu to manage our repositories. Besides there are more clients available, for example RapidSVN (Windows, Unix/Linux) and SVNcommander (Linux).

2.2 Linux (Ubuntu 7.04)

The installation on a Linux system is much easier than the installation on Windows. Using `apt-get install` or the Synaptic package manager we install the following packages:

- apache2.2-common and apache2-utils
- libapache2-svn
- subversion

5. current version at printtime: 1.4.4.9706

Additional packages are selected automatically by the package management tool. After the installation of these packages the remaining step is to make the necessary adjustments to `/etc/apache2/sites-available/default` and to set the access rights for this directory via `chmod -R 770 /home/uwe/repositoryRoot`:

```
<Location /svn>
DAV svn

SVNParentPath /home/uwe/repositoryRoot
</Location>
```

Listing 2: Setup code for the Linux repository root

3 First steps

To fill the repository we created during the installation we create a empty directory (all files in this directory will be imported to the repository) which we populate with a small LaTeX document (article-template.tex):

```
\documentclass{article}
\begin{document}

Hello World!

\end{document}
```

Listing 3: A simple LaTeX file

Using the the command line (`svn import http://localhost/svn/firstSample/ -m "import"`) or the TortoiseSVN context menu or we can now import the file using our URL for the repository `http://localhost/svn/firstSample/` and use "import" as comment. Apache will list now `Revision 1: /` when we browse the repository (see Figure 3. To work with the file again we need to make a check-out into a `working directory`. The files in the working directory are the files we edit, all future revisions will be committed from this directory.

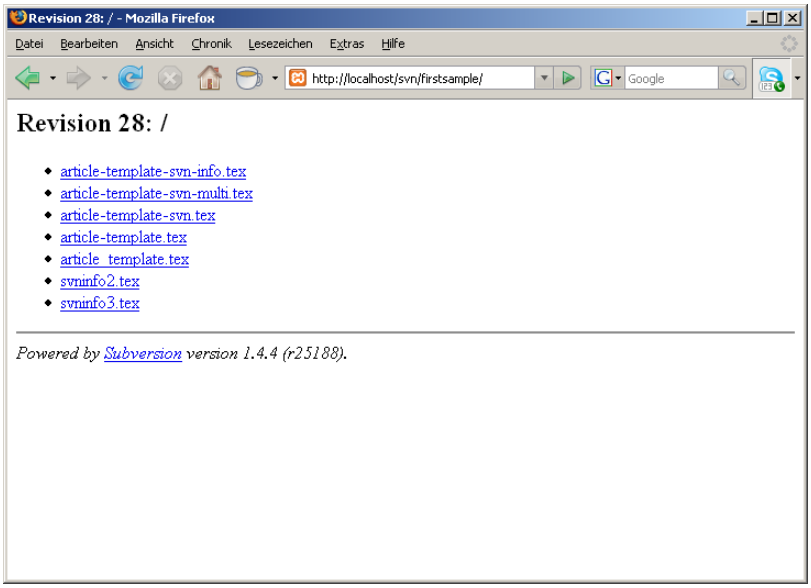


Figure 3: Repository browsing with Apache and Subversion Module

4 Integration with LaTeX

To integrate the Subversion metadata in our LaTeX files we need to tell Subversion to include them. The following list contains the available keywords and their description:

Date (LastChangedAt) date and time of last check-in
Revision: (LastChangedRevision) the number of the revision
Author: (LastChangedBy) name of the submitting author
HeadURL: the URL of this file
Id: a summary of the above keywords

After using `svn propset svn:keywords "Date HeadURL Revision Id" article_template.tex` from the commandline Subversion will expand those keywords (enclosed in $) in our file when we include them in the LaTeX code. Subversion will expand the keywords as following:

July 15, 2007
2007-07-15 17:33:30 +0200 (So, 15 Jul 2007)
17:33:30
article-template.tex 12 2007-07-15 15:33:30Z
http://localhost/svn/firstSample/article-template.tex

Figure 4: Output of article-template.tex with svn package

```
% $Revision$
% $HeadURL$
% $Date$
% $Author$
% $Id$

\documentclass{article}
\begin{document}
 Hello World!
\end{document}
```

Listing 4: A sample file with expanded Subversion keywords

All LaTeX packages introduced below are based on the evaluation of these keywords.

4.1 svn

The svn package allows access the metainformation by evaluating the Subversion information using a \SVN $Keyword: <metadata>$ syntax. If the keywords are correctly expanded, then the svn package defines:

- \SVNDate for the date of the checkin, \SVNTime as the check-in time and \SVNRawDate as raw date and time if Keyword was $Date$.
- \SVNKeyword otherwise (Examples: \SVNId, \SVNHeadURL)

```
\documentclass{article}
\usepackage{svn}

\SVN $Id$
\SVN $Date$
\SVN $Id$
\SVN $HeadURL$

\begin{document}

\SVNDate \\
\SVNRawDate \\
\SVNTime \\
\SVNId \\
\SVNHeadURL
\end{document}
```

Listing 5: A sample file using the `svn` package

4.2 svninfo

The `svninfo` package needs information from the Id keyword only which need to follow the `\svnInfo` command: \svnInfo *Id* : *article* − *template* − *svn* − *info.tex*182007 − 07 − 1516 : 11 : 21*Z*

To use the meta information the package defines the following commands:

- `\svnInfoFile` the name of the file
- `\svnInfoRevision` the revision number
- `\svnInfoDate` the date of the last check-in
- `\svnInfoTime` the time of the last check-in
- `\svnInfoYear` the year of \svnInfoDate
- `\svnInfoMonth` the month of \svnInfoDate
- `\svnInfoDay` the day of \svnInfoDate
- `\svnInfoOwner` the owner of the file (if specified at check-in)
- `\svnToday` date of last check-in in the `\today` format
- `\svnInfoMinRevision` the minimum revision of the document
- `\svnInfoMaxRevision` the maximum revision of the document

\svnInfoMinRevision and \svnInfoMaxRevision are useful for multi-file documents. Furthermore the packages allows a few optional parameters such as fancyhdr,eso-foot, scrpage2 to typeset Subversion information in the margin or the footer of the document. For details please see the manual.

4.3 svn-multi

The svnmulti package provides two commands, \svnid and \svnidlong, to capture the input from Subversion. To use the variables, the package provides the following commands:

- \svnrev the revision
- \svndate the date of the last check-in
- \svnauthor the author
- \svnfilerev the revision of the current file if it contains a \svnid or \svnidlong or the values of the last file if it does not contain one of these commands
- \svnmainurl and \svnmainfilename typeset the URL respectively name of the main file, as it was defined by the internal command \svnmainfile at the end of the preamble

Furthermore svn-multi uses \svn{*keyword*} and \svnkw{*keyword*} to print Subversion keywords directly. To access date information the package provides some more commands, explanations can be seen directly from each respective name: \svnfileyear, \svnfilemonth, \svnfileday, \svnfilehour, \svnfileminute, \svnfilesecond, \svnfiletimezone, \svnyear, \svnmonth, \svnday, \svnhour, \svnminute, \svnsecond and \svntimezone.

5 Conclusion

This article described the basic usage of Subversion from LaTeX and the most common features of the three LaTeX packages svn, svninfo and svn-multi. More information about integration with LaTeX or Subversion itself can be found in the documentation of the packages or in books on Subversion ([6, 8]. Feedback on this article is welcome, if you find any mistakes or have comments please send me an email. Updates of the article can later be found online at http://www.uweziegenhagen.de/latex/

References

[1] Apache 2 web server. URL http://httpd.apache.org.

[2] Subversion. URL http://subversion.tigris.org/.

[3] Svnserve based server. URL http://tortoisesvn.net/docs/nightly/TortoiseSVN_en/tsvn-serversetup-svnserve.html.

[4] TortoiseSVN. URL http://tortoisesvn.tigris.org.

[5] apachefriends.org. Xampp. URL http://www.apachefriends.org/en/.

[6] Ben Collins-Sussman, Brian W. Fitzpatrick, and C. M. Pilato. *Version Control with Subversion. Next Generation Open Source Version Control*. O'Reilly, 2004.

[7] Apache Foundation. Apache HTTP server version 2.2 documentation. URL http://httpd.apache.org/docs/2.0/en/.

[8] Mike Mason. *Pragmatic Version Control Using Subversion*. Pragmatic Programmers LLC., 2006.