

Mini-Howto for Subversion User

Last update: December 15, 2005

Gordon Grubert (grubert_NOSPAM-AT-physik.uni-greifswald.de)

The goal of this short tutorial is to show ordinary users the simple usage of the version control system SUBVERSION and to convince them to apply it.

Contents

1 Basic Concept	1
2 Requirements	1
3 Basic Commands	1
3.1 Import	1
3.2 Check out	2
3.3 Working Copy	2
3.4 Commit changes	3
3.5 Update work copy	4

1 Basic Concept

The basic concept of version control systems is that everything of the project is located at a central position and the *complete* history of the project will be saved, too. In many cases a documentation of projects is missed but a version control system forces the user to document changes. Version control based projects consist of one central repository and the local work copies of the current authors. The great advantage is that more than one person can work with the project and no changes will be lost or conflicts can occur.

2 Requirements

On the user's computer SUBVERSION has to be installed. In this tutorial a web based repository access is used. Therefore, it is assumed, that a subversion server is available in the network and accessible over the *https* protocol. In this examples the short name of the server is *svn* and the location of the repositories will be */repos/*.

3 Basic Commands

To view the content of a repository, open a web browser and go to the page

https://svn/repos/example

You see, that the repository's name has to be specified explicitly. In this tutorial a project *apache_hash* will be created and modified.

3.1 Import

To create a project under version control you have to perform an initial import of the complete project with its structure. This can be done easily by

```
svn import /tmp/apache_hash https://svn/repos/example/apache_hash \  
-m "Initial import."
```

The result of this import is presented in Fig. 1. Comments (development documentation) are given by the option "-m", respectively. If you do not add a comment to every action you do (in this case the import command) SUBVERSION prods you to the missing documentation.

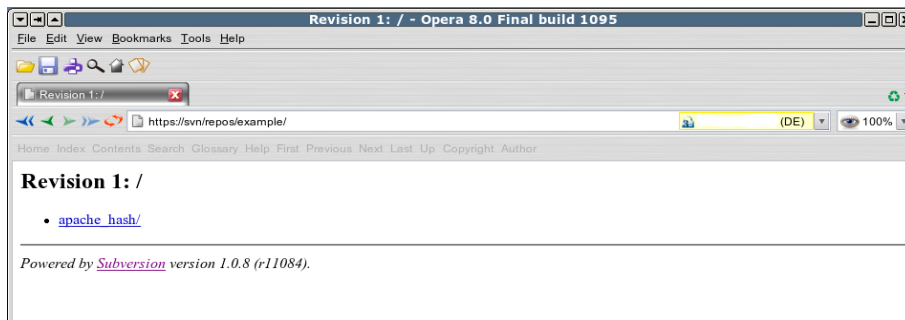


Figure 1: Repository's content after import procedure.

3.2 Check out

The first step to work with SUBVERSION is to check out your private local work copy of the project. That means, you copy the current version of the project (or parts of it) into your own directory. There you can modify it without destroying anything done before. The local copy can be obtained by

```
svn checkout https://svn/repos/example/apache_hash
```

On this way you will get the current version of the complete project. You can also specify a subdirectory or a special revision for check out (see Fig. 2).

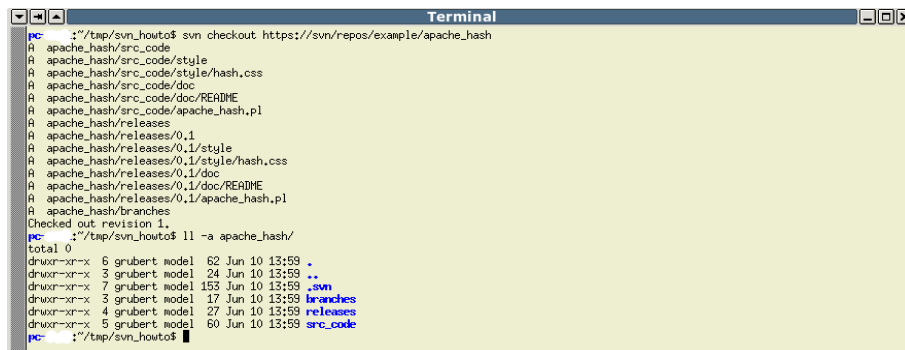


Figure 2: How to check out a working copy.

3.3 Working Copy

After the *checkout* command you will find a directory *apache.hash* on your hard disk which contains the complete project. In every directory a subdirectory *.svn* is located which contains all relevant version control information.

If you change into the project directory you can use all of the subversion commands, e.g, you can get information about the project or look at the history: *svn info*, *svn log* (see Fig. 3).

To work with your project (perform changes) you can do this as before. But after your work SUBVERSION is important again. You can ask the status of the project or get information about your changes: *svn status*, *svn diff* (Fig. 4). Here, you have also the possibility to use the *diff* function for different revisions of the project.

Notice that file system operations like *copy*, *move*, *rm*, *mkdir* or *rmdir* have to be performed by means of SUBVERSION. That means, that a new directory has to be created by

```
svn mkdir newdir
```

```

Terminal
PC- ~/tmp/svn_houto/apache_hash$ svn info
Path:
URL: https://svn/repos/example/apache_hash
Repository UUID: 25693bb8-17f9-0310-b930-de6d547d39c6
Revision: 1
Node Kind: directory
Schedule: normal
Last Changed Author: grubert
Last Changed Rev: 1
Last Changed Date: 2005-06-09 10:22:47 +0200 (Thu, 09 Jun 2005)

PC- ~/tmp/svn_houto/apache_hash$ svn log -v
-----
r1 | grubert | 2005-06-09 10:22:47 +0200 (Thu, 09 Jun 2005) | 1 line
Changed paths:
   R /apache_hash
   R /apache_hash/branches
   R /apache_hash/releases
   R /apache_hash/releases/0.1
   R /apache_hash/releases/0.1/apache_hash.pl
   R /apache_hash/releases/0.1/doc
   R /apache_hash/releases/0.1/doc/README
   R /apache_hash/releases/0.1/style
   R /apache_hash/src_code
   R /apache_hash/src_code/apache_hash.pl
   R /apache_hash/src_code/doc
   R /apache_hash/src_code/doc/README
   R /apache_hash/src_code/style
   R /apache_hash/src_code/style/hash.css

Initial Import.
PC- ~/tmp/svn_houto/apache_hash$

```

Figure 3: Get information about your working copy and the corresponding repository.

```

Terminal
PC- ~/tmp/svn_houto/apache_hash/src_code$ svn status
M   apache_hash.pl
PC- ~/tmp/svn_houto/apache_hash/src_code$ svn diff
Index: apache_hash.pl
=====
--- apache_hash.pl      (revision 1)
+++ apache_hash.pl      (working copy)
@@ -5,7 +5,7 @@
 #
 #   E-Mail:      grubert@physik.uni-greifswald.de
 #   Copyright:   (C) 2005 by Gordon Grubert
 #   Version:     0.1
 #   Last modified: Apr 2005
+#   Last modified: May 2005
 #
 #   Licence:     GPL (http://www.gnu.org/licenses/gpl.html)
 #               This program is free software; you can redistribute it and/or modify
PC- ~/tmp/svn_houto/apache_hash/src_code$

```

Figure 4: Query the status of your working copy or compare your local copy with the repository.

Now, SUBVERSION immediately knows everything about the new directory. If you use the operating system command `mkdir`, you can create a new directory, too. But this one is **not** under version control. You have to add it *manually* by

```
svn add newdir
```

3.4 Commit changes

The final step of your work with the local copy of the project is to commit all of your changes to the central repository. To do this, use

```
svn commit -m "Correction of the date."
```

```

mc - /home/SVN/access
PC- ~/tmp/svn_houto/apache_hash/src_code$ svn commit -m "Correction of the date."
Sending
  src_code/apache_hash.pl
Transmitting file data .
Committed revision 2.
PC- ~/tmp/svn_houto/apache_hash/src_code$ svn status
PC- ~/tmp/svn_houto/apache_hash/src_code$ svn log
-----
r1 | grubert | 2005-06-09 10:22:47 +0200 (Thu, 09 Jun 2005) | 1 line
Initial Import.
-----
PC- ~/tmp/svn_houto/apache_hash/src_code$ svn log -r 1:2
-----
r1 | grubert | 2005-06-09 10:22:47 +0200 (Thu, 09 Jun 2005) | 1 line
Initial Import.
-----
r2 | grubert | 2005-06-10 14:25:43 +0200 (Fri, 10 Jun 2005) | 1 line
Correction of the date.
-----
PC- ~/tmp/svn_houto/apache_hash/src_code$

```

Figure 5: Commit your local working copy to the server's repository.

When you have done this, your changes of the project are available in the repository so that everybody can use them. Every commit procedure gets an own new revision number (see Figs. 5

& 6).

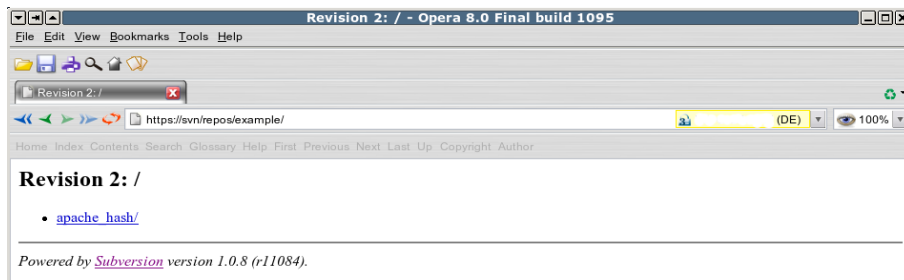


Figure 6: Web view of the repository after commit procedure.

3.5 Update work copy

Last but not least there is the update option. The starting point is a situation where you have a local working copy. If there are more authors for a project you cannot be sure that your copy is up-to-date. Now, you have two possibilities:

1. you can remove your local copy (*rm -rf your_copy*) and check out again or
2. you can update your local copy by

```
svn update
```

If you do it like in the second point, your local copy becomes up-to-date with the current version of the SUBVERSION server's repository.

References

- [1] <http://subversion.tigris.org/faq.html>
- [2] http://gentoo-wiki.com/HOWTO_Subversion
- [3] B. Collins-Sussman, B.W. Fitzpatrick, and C.M. Pilato, "Version Control with Subversion – For Subversion 1.0", book compiled from Revision 10945 (2002)
- [4] http://www-ai.math.uni-wuppertal.de/~huesken/div/svn_conf.pdf