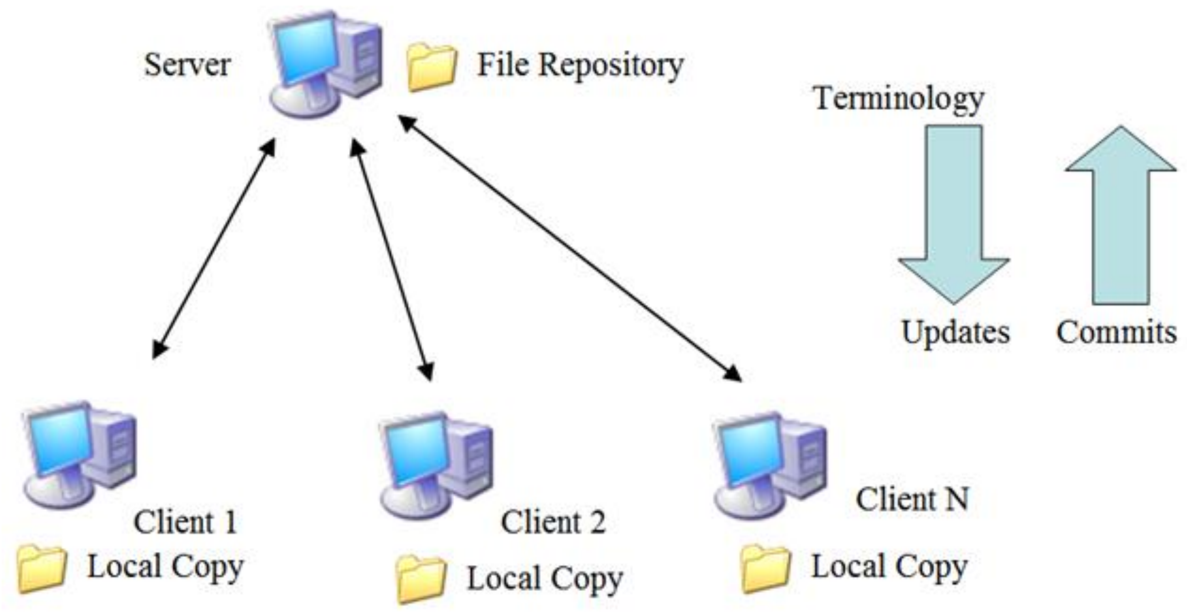
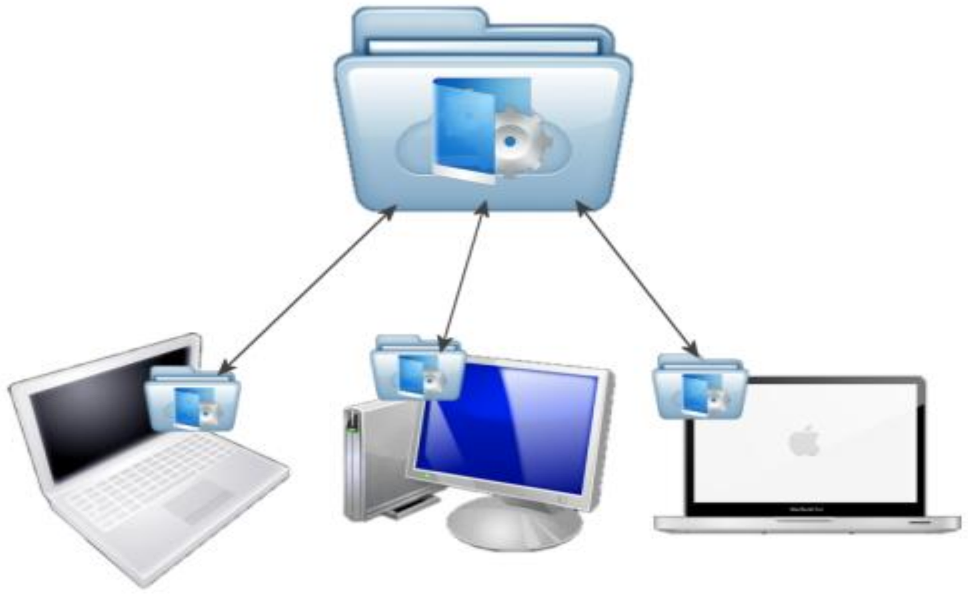
SVN

SVN is acronym as Subversion, a version control system. It lets several ***developers*** work on the same project at once without worrying about overwriting each other's changes, and keeps track of previous versions of files. It is used to maintain the revisions of files such as source code, web pages and documentation. It is commonly used in many open source projects.



The version control system puts all versions of a file in a place called **repository.**



The **repository** is much like an ordinary file server, except that it remembers every change ever made to your files and directories. This allows you to recover older versions of your code, or examine the history of how your code was changed.

Source control tools track all prior versions of all files, allowing developers to ***"time travel"*** backward and forward in their software to determine when and where bugs are introduced. If a mistake is made, files can be rolled back to previous version. Subversion was started in 2000 as an effort to write a free version control system.

Installation

```
[root@localhost ~]# yum install subversion mod_dav_svn
```

```
[root@localhost ~]# rpm -qa | egrep '(subversion|mod_dav_svn)'
```

mod_dav_svn-1.6.11-10.el5_8 : The mod_dav_svn package allows access to a Subversion repository using HTTP, via the Apache httpd server.

subversion-1.6.11-10.el5_8 : Subversion is a concurrent version control system which enables one or more users to collaborate in developing and maintaining a hierarchy of files and directories while keeping a history of all changes. Subversion only stores the differences between versions, instead of every complete file. Subversion is intended to be a compelling replacement for CVS.

Configuring SVN

```
root@localhost ~]# vi /etc/httpd/conf.d/subversion.conf
```

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

<Location /repo>
        DAV svn
        SVNPath /var/www/svn/repo
        AuthType Basic
        AuthName "Subversion repos"
        AuthUserFile /etc/svn-auth-conf
        Require valid-user
</Location>
```

:x! (save & exit

Creating SVN Users (Project Manager & Web-Developers)

```
[root@localhost ~]# htpasswd  -c /etc/svn-auth-conf PM
New password:
Re-type new password:
Adding password for user PM

[root@localhost ~]# htpasswd  /etc/svn-auth-conf developer1
New password:
Re-type new password:
Adding password for user developer1

[root@localhost ~]# htpasswd /etc/svn-auth-conf developer2
New password:
Re-type new password:
Adding password for user developer2


[root@localhost ~]# mkdir -p /var/www/svn
[root@localhost ~]# cd /var/www/svn/

[root@localhost repo]# svnadmin create repo
[root@localhost repo]# ls
repo

[root@localhost ~]# chmod -R 777 /var/www/svn/repo

[root@localhost svn]# chown -R apache.apache repo

[root@localhost svn]# service httpd restart
Stopping httpd:                                    [FAILED]
Starting httpd:                                    [  OK  ]
```

Creating Project and their Layouts

```
[PM@localhost ~]$ mkdir PerlProject PythonProject

[PM@localhost ~]$ mkdir PerlProject/modules
[PM@localhost ~]$ mkdir PerlProject/components
[PM@localhost ~]$ mkdir PerlProject/configurations

[PM@localhost ~]$ mkdir PythonProject/modules
[PM@localhost ~]$ mkdir PythonProject/components
[PM@localhost ~]$ mkdir PythonProject/configurations
```

```
[PM@localhost ~]$ svn import /home/PM/PerlProject
file:///var/www/svn/repo/PerlProject -m "Perl Project Layout"
Adding         /home/PM/PerlProject/components
Adding         /home/PM/PerlProject/modules
Adding         /home/PM/PerlProject/configurations

Committed revision 1.

[PM@localhost ~]$ svn import /home/PM/PythonProject
file:///var/www/svn/repo/PythonProject -m "Python Project Layout"
Adding         /home/PM/PythonProject/components
Adding         /home/PM/PythonProject/modules
Adding         /home/PM/PythonProject/configurations

Committed revision 2.
```

CheckOut

Checkout command is used to download sources from SVN repository to working copy. **If you want to access files from the SVN server, checkout is the first operation you should perform.**

SVN checkout creates the working copy, from where you can do edit, delete, or add contents. You can checkout a file, directory, trunk or whole project. To checkout you should know URL of the components you want to checkout.

Perl Project by Developer2

```
[developer2@localhost ~]$ svn co http://192.168.221.129/repo/PerlProject
Authentication realm: <http://192.168.221.129:80> Subversion repos
Password for 'developer2':

-----------------------------------------------------------------------
ATTENTION!  Your password for authentication realm:

   <http://192.168.221.129:80> Subversion repos

can only be stored to disk unencrypted!  You are advised to configure
your system so that Subversion can store passwords encrypted, if
possible.  See the documentation for details.

You can avoid future appearances of this warning by setting the value
of the 'store-plaintext-passwords' option to either 'yes' or 'no' in
'/home/developer2/.subversion/servers'.
-----------------------------------------------------------------------
```

```
Store password unencrypted (yes/no)? yes
A    PerlProject/components
A    PerlProject/modules
A    PerlProject/configurations
Checked out revision 2.

[developer2@localhost ~]$ ls
PerlProject
```

Python Project by Developer1

```
[developer1@localhost ~]$ svn co http://192.168.221.129/repo/PythonProject
Authentication realm: <http://192.168.221.129:80> Subversion repos
Password for 'developer1':

-----------------------------------------------------------------------
ATTENTION!  Your password for authentication realm:

   <http://192.168.221.129:80> Subversion repos

can only be stored to disk unencrypted!  You are advised to configure
your system so that Subversion can store passwords encrypted, if
possible.  See the documentation for details.

You can avoid future appearances of this warning by setting the value
of the 'store-plaintext-passwords' option to either 'yes' or 'no' in
'/home/developer1/.subversion/servers'.
-----------------------------------------------------------------------
Store password unencrypted (yes/no)? yes
A    PythonProject/components
A    PythonProject/modules
A    PythonProject/configurations
Checked out revision 2.

 [developer1@localhost ~]$ ls
PythonProject
```

Note : When you do a checkout, it creates hidden directory named .svn, which will have the repository details.

Python Code writing

```
[developer1@localhost ~]$ vi PythonProject/components/compCode.py
This is Component Code in Python
```

```
[developer1@localhost ~]$ vi PythonProject/modules/modCode.py
This is a Module Code in Python

[developer1@localhost ~]$ vi PythonProject/configurations/confCode.py
This is a First line in Configuration Code
```

Adding New files to the Python Project:

```
[developer1@localhost ~]$ svn add PythonProject/components/compCode.py
A         PythonProject/components/compCode.py

[developer1@localhost ~]$ svn add PythonProject/modules/modCode.py
A         PythonProject/modules/modCode.py

[developer1@localhost ~]$ svn add PythonProject/configurations/confCode.py
A         PythonProject/configurations/confCode.py
```

Commit

Whenever you do changes to the working copy, it will not reflect in SVN server. To make the changes permanent, you need to do SVN commit.

```
[developer1@localhost ~]$ svn commit -m "Committing first set of Python codes"
svn: '/home/developer1' is not a working copy

[developer1@localhost ~]$ cd PythonProject/
[developer1@localhost PythonProject]$ svn commit -m "Committing first set of
Python codes"
Adding         components/compCode.py
Adding         configurations/confCode.py
Adding         modules/modCode.py
Transmitting file data ...
Committed revision 3.
```

Developer2 wants to edit some more codes to Python Project

```
[developer2@localhost ~]$ svn co http://192.168.221.129/repo/PythonProject
A    PythonProject/components
A    PythonProject/components/compCode.py
A    PythonProject/modules
```

```
A    PythonProject/modules/modCode.py
A    PythonProject/configurations
A    PythonProject/configurations/confCode.py
Checked out revision 3.


[developer2@localhost ~]$ ls
PerlProject  PythonProject

[developer2@localhost ~]$ cd PythonProject/configurations/
[developer2@localhost configurations]$ vi confCode.py

This is a First line in Configuration Code

This is the Second line by Developer 2

[developer2@localhost PythonProject]$ svn commit -m "Committing further
changes in Python codes"
Sending        configurations/confCode.py
Transmitting file data .
Committed revision 4.
```

However Project Manager (PM) didn't like the code from Developer2 and wants to revert back to the code written by Developer1

Check the current version of the code

SVN remembers every change made to your files and directories. To know all the commands made in a file or directory, use SVN log command.

To check the versions at the repository level (all projects)

```
[PM@localhost ~]$ svn log http://192.168.221.129/repo/
------------------------------------------------------------------------
r4 | developer2 | 2013-03-04 10:31:52 -0800 (Mon, 04 Mar 2013) | 1 line

Committing further changes in Python codes
------------------------------------------------------------------------
r3 | developer1 | 2013-03-04 10:23:26 -0800 (Mon, 04 Mar 2013) | 1 line

Committing first set of Python codes
------------------------------------------------------------------------
r2 | PM | 2013-03-04 09:43:28 -0800 (Mon, 04 Mar 2013) | 1 line
```

```
Python Project Layout
------------------------------------------------------------------------
r1 | PM | 2013-03-04 09:42:56 -0800 (Mon, 04 Mar 2013) | 1 line

Perl Project Layout
```

To check the versions of only PythonProject

```
[PM@localhost ~]$ svn log http://192.168.221.129/repo/PythonProject
------------------------------------------------------------------------
r4 | developer2 | 2013-03-04 10:31:52 -0800 (Mon, 04 Mar 2013) | 1 line

Committing further changes in Python codes
------------------------------------------------------------------------
r3 | developer1 | 2013-03-04 10:23:26 -0800 (Mon, 04 Mar 2013) | 1 line

Committing first set of Python codes
------------------------------------------------------------------------
r2 | PM | 2013-03-04 09:43:28 -0800 (Mon, 04 Mar 2013) | 1 line

Python Project Layout
```

Note : Well , you can see (in **Green** color) the importance of using message (-m) at every commit or adding or deleting .

Revert Back

```
[PM@localhost LiveWorking]$ svn co -r 3
http://192.168.221.129/repo/PythonProject

A    PythonProject/components
A    PythonProject/components/compCode.py
A    PythonProject/modules
A    PythonProject/modules/modCode.py
A    PythonProject/configurations
A    PythonProject/configurations/confCode.py
Checked out revision 3.

[PM@localhost LiveWorking]$ cat PythonProject/configurations/confCode.py
This is a First line in Configuration Code
```

The code has been successfully reverted back to its original State.

SVN List

svn list is useful when you want to view the content of the SVN repository, without downloading a working copy.

```
[PM@localhost ~]$ svn list --verbose http://192.168.221.129/repo/PythonProject
      5 PM                  Mar 04 12:05 ./
      3 develope            Mar 04 10:23 components/
      5 PM                  Mar 04 12:05 configurations/
      3 develope            Mar 04 10:23 modules/
```

The following example lists all the files available in the given URL in the repository without downloading a working copy. When you execute svn list command with –verbose option it displays the following information.

- Revision number of the last commit
- Author of the last commit
- Date and time of the last commit

SVN Delete

SVN delete command deletes an item from the working copy (or repository). File will be deleted from the repository when you do a SVN commit.

```
[PM@localhost PythonProject]$ svn delete configurations/SecondCode.py
D         configurations/SecondCode.py

[PM@localhost PythonProject]$ svn commit -m "SecondCode.py Successfully
Deleted by PM"
Deleting       configurations/SecondCode.py

Committed revision 8.
```

Now you can do svn list and check whether the file was deleted from the repository.

SVN Diff

SVN diff displays the differences between your working copy and the copy in the SVN repository.

```
[PM@localhost modules]$ svn diff modCode.py
Index: modCode.py
===================================================================
--- modCode.py  (revision 6)
+++ modCode.py  (working copy)
@@ -1 +1,5 @@
 This is a Module Code in Python
+
+This is a Second Line in Module Code
+
+
```

SVN Status

Use svn status command to get the status of the file in the working copy. It displays whether the working copy is modified, or its been added/deleted, or file is not under revision control, etc.

```
[PM@localhost LiveWorking]$ svn status PythonProject/
M       PythonProject/modules/modCode.py
```

‘M’ represents that the item has been modified. “svn help status” command will explain various specifiers showed in SVN status command

SVN Update

svn update command brings changes from the repository into your working copy. If no revision is specified, it brings your working copy up-to-date with the HEAD revision. Otherwise, it synchronizes the working copy to the revision given in the argument.

Always before you start working in your working copy, update your working copy. So that all the changes available in repository will be available in your working copy. i.e latest changes.

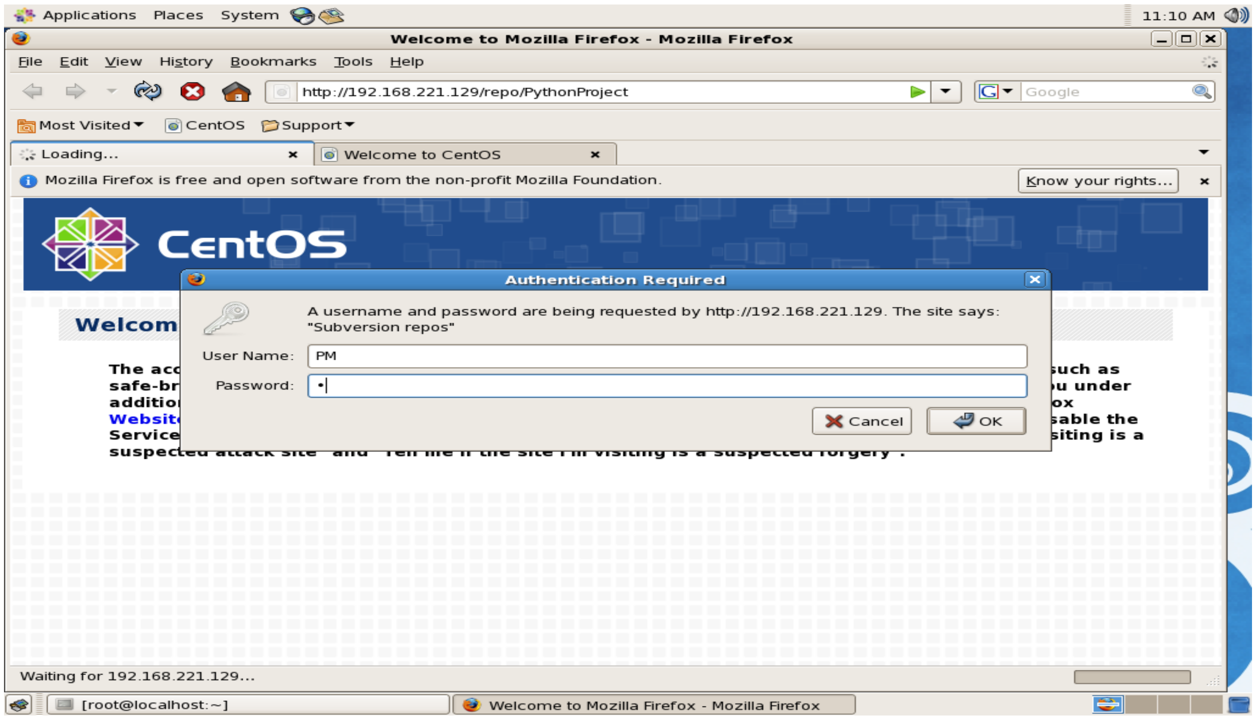
```
[developer2@localhost ~]$ svn add PythonProject/components/compCode2.py
A         PythonProject/components/compCode2.py

[developer2@localhost ~]$ svn add PythonProject/components/compCode3.py
A         PythonProject/components/compCode3.py


[developer2@localhost ~]$ svn commit -m "Some new files updated by
developer2"
Adding         components/compCode2.py
Adding         components/compCode3.py
Transmitting file data ...
Committed revision 9.


[PM@localhost PythonProject]$ svn update
A    components/compCode2.py
A    components/compCode3.py
Updated to revision 9.
```

SVN – on Browser


Applications Places System
11:10 AM
Welcome to Mozilla Firefox - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Most Visited CentOS Support
Loading... Welcome to CentOS
Know your rights...
Cancel OK
[root@localhost:~] Welcome to Mozilla Firefox - Mozilla Firefox

http://192.168.221.129/repo/PythonProject

Mozilla Firefox is free and open software from the non-profit Mozilla Foundation.

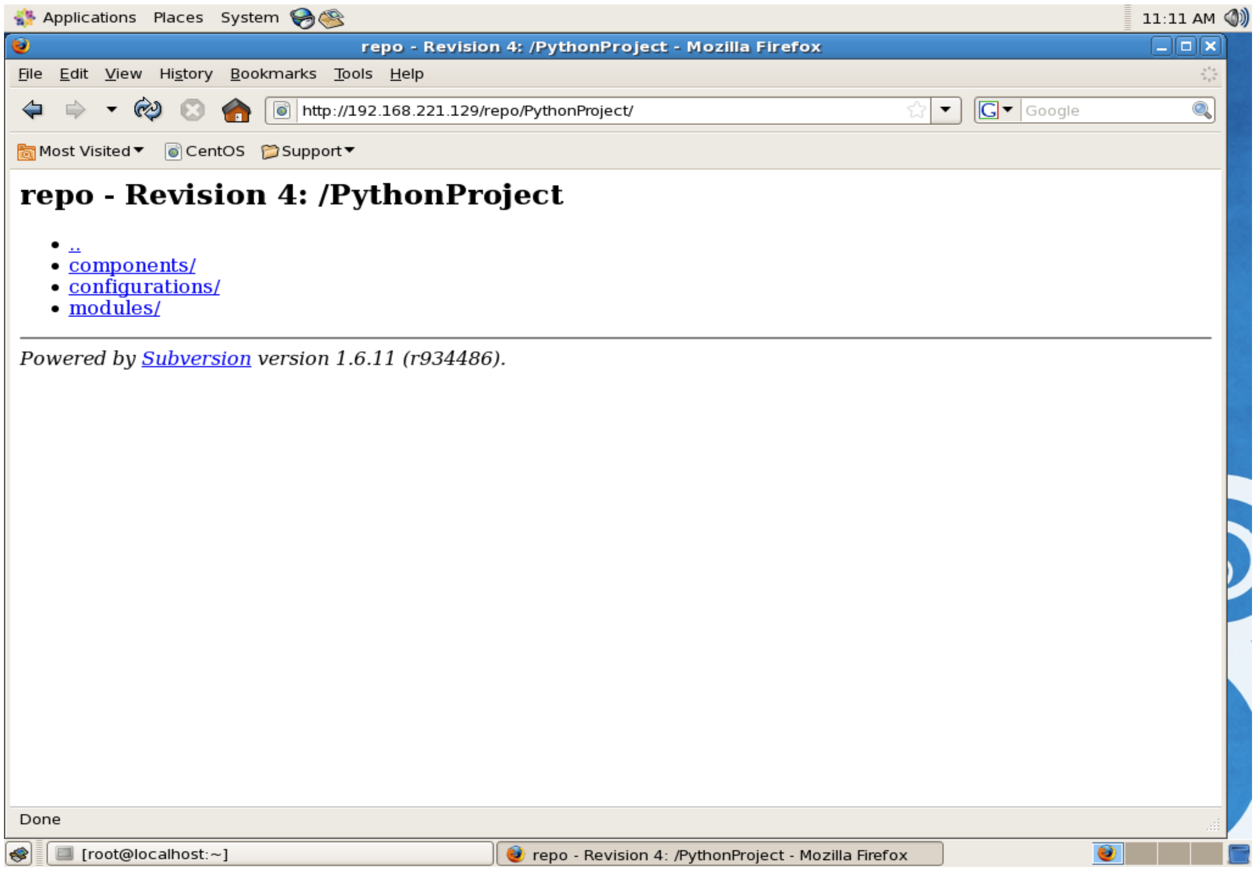
Authentication Required

A username and password are being requested by http://192.168.221.129. The site says: "Subversion repos"

User Name: PM

Password: •

Waiting for 192.168.221.129...


Applications Places System
11:11 AM
repo - Revision 4: /PythonProject - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Most Visited CentOS Support
[root@localhost:~] repo - Revision 4: /PythonProject - Mozilla Firefox

http://192.168.221.129/repo/PythonProject/

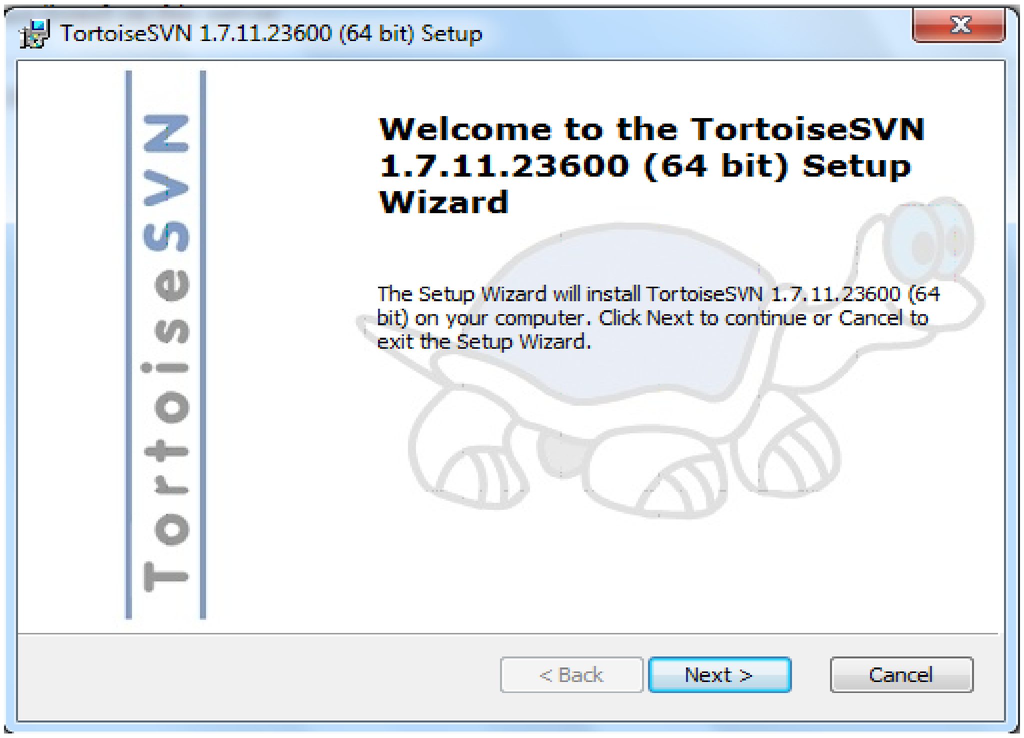
repo - Revision 4: /PythonProject

- ..
- components/
- configurations/
- modules/

Powered by Subversion version 1.6.11 (r934486).

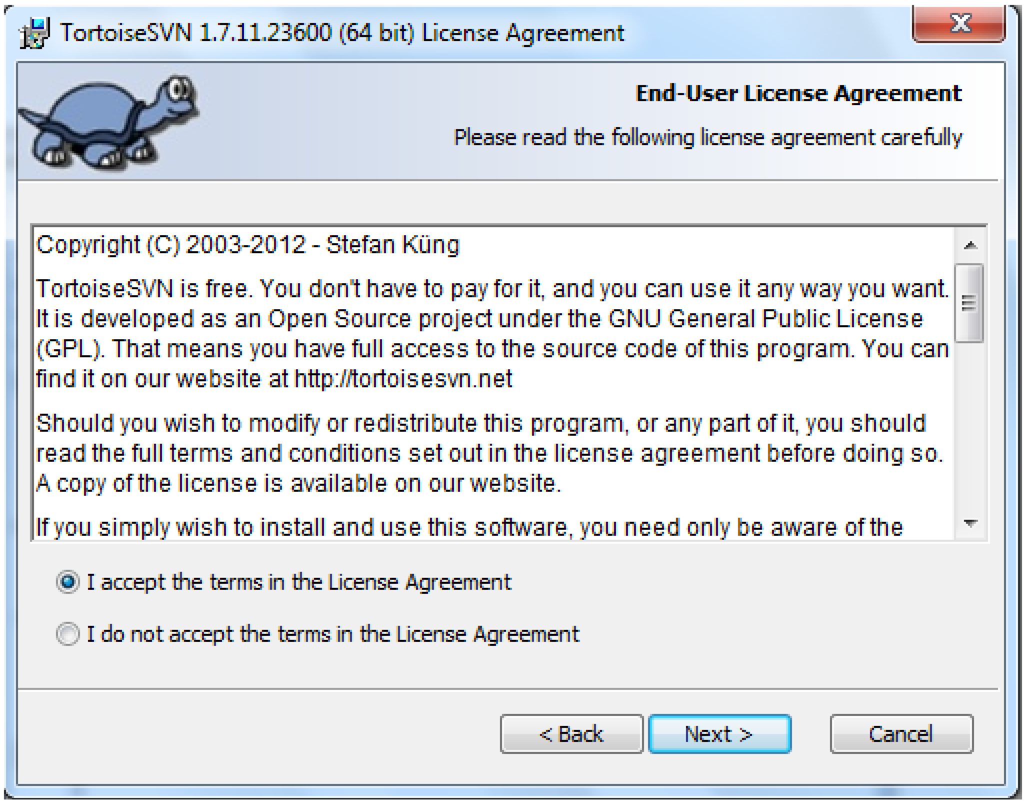
Done

TortoiseSVN



Welcome to the TortoiseSVN 1.7.11.23600 (64 bit) Setup Wizard

The Setup Wizard will install TortoiseSVN 1.7.11.23600 (64 bit) on your computer. Click Next to continue or Cancel to exit the Setup Wizard.



End-User License Agreement

Please read the following license agreement carefully

Copyright (C) 2003-2012 - Stefan Küng

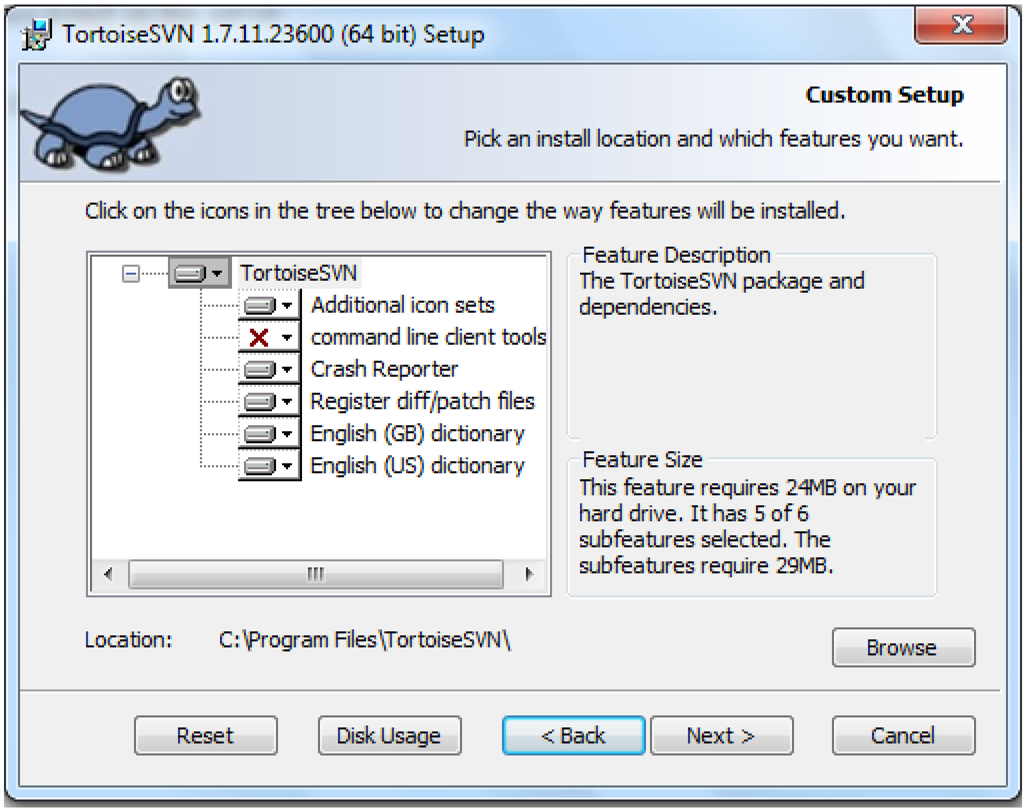
TortoiseSVN is free. You don't have to pay for it, and you can use it any way you want. It is developed as an Open Source project under the GNU General Public License (GPL). That means you have full access to the source code of this program. You can find it on our website at http://tortoisesvn.net

Should you wish to modify or redistribute this program, or any part of it, you should read the full terms and conditions set out in the license agreement before doing so. A copy of the license is available on our website.

If you simply wish to install and use this software, you need only be aware of the

[✓ I accept the terms in the License Agreement]

I do not accept the terms in the License Agreement


TortoiseSVN 1.7.11.23600 (64 bit) Setup

Custom Setup

Pick an install location and which features you want.

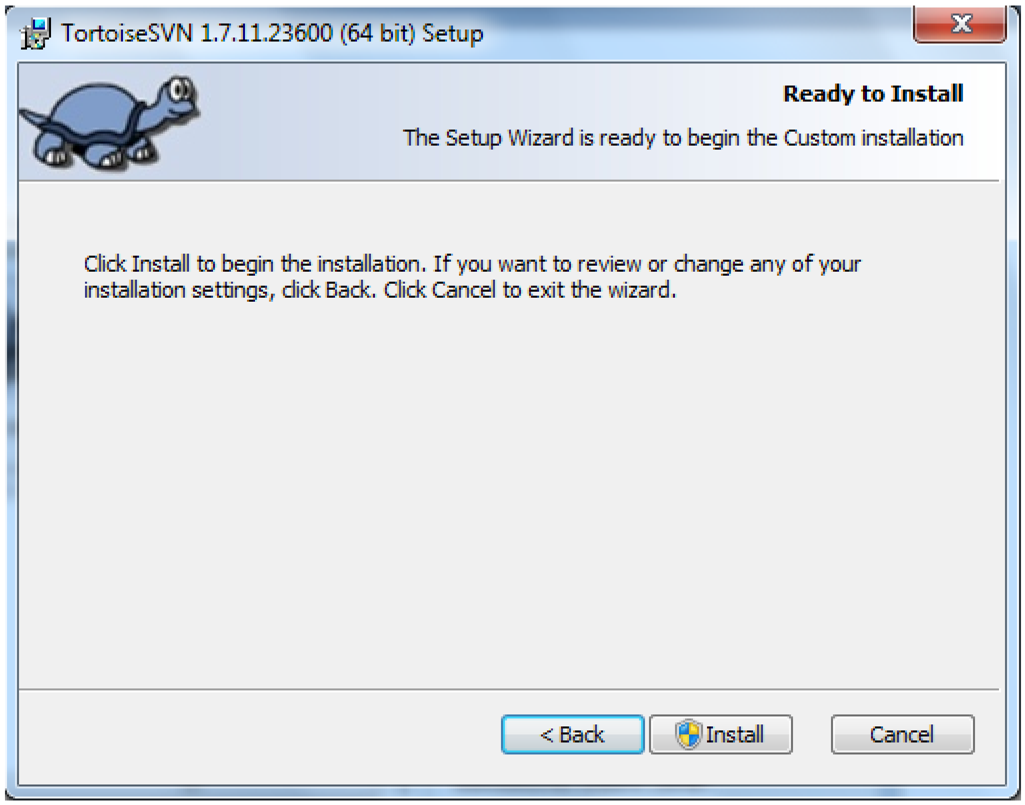
Click on the icons in the tree below to change the way features will be installed.

- TortoiseSVN
  - Additional icon sets
  - ✗ command line client tools
  - Crash Reporter
  - Register diff/patch files
  - English (GB) dictionary
  - English (US) dictionary

Feature Description
The TortoiseSVN package and dependencies.

Feature Size
This feature requires 24MB on your hard drive. It has 5 of 6 subfeatures selected. The subfeatures require 29MB.

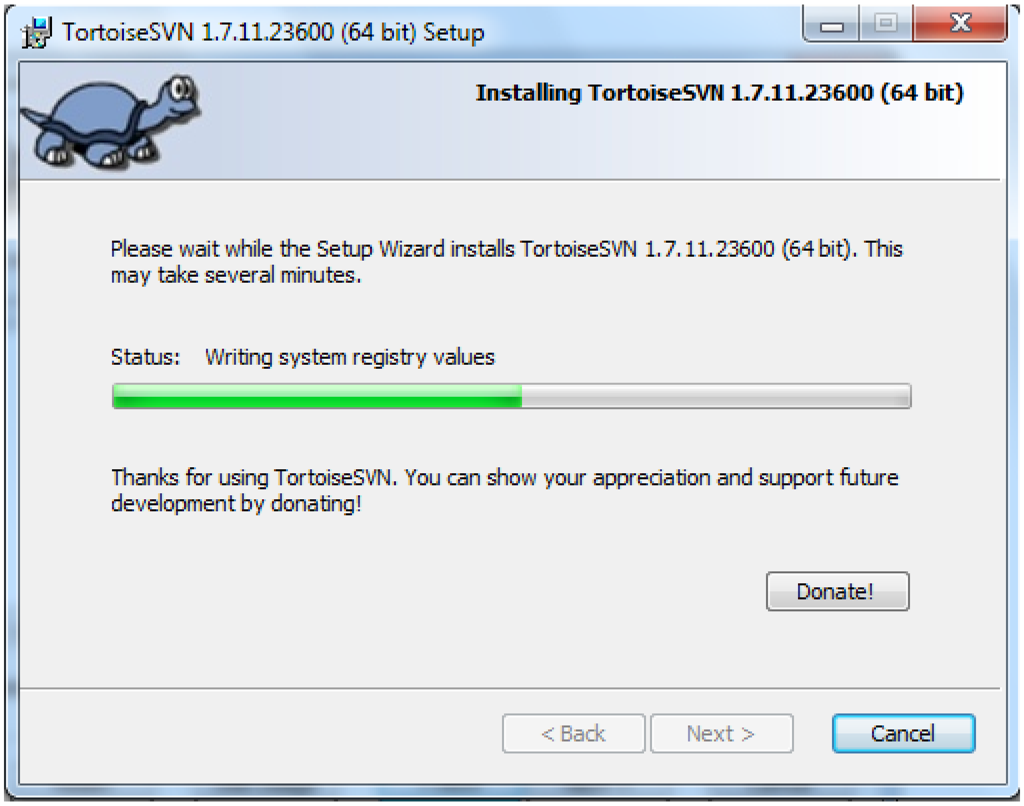
Location: C:\Program Files\TortoiseSVN\


TortoiseSVN 1.7.11.23600 (64 bit) Setup

Ready to Install

The Setup Wizard is ready to begin the Custom installation

Click Install to begin the installation. If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

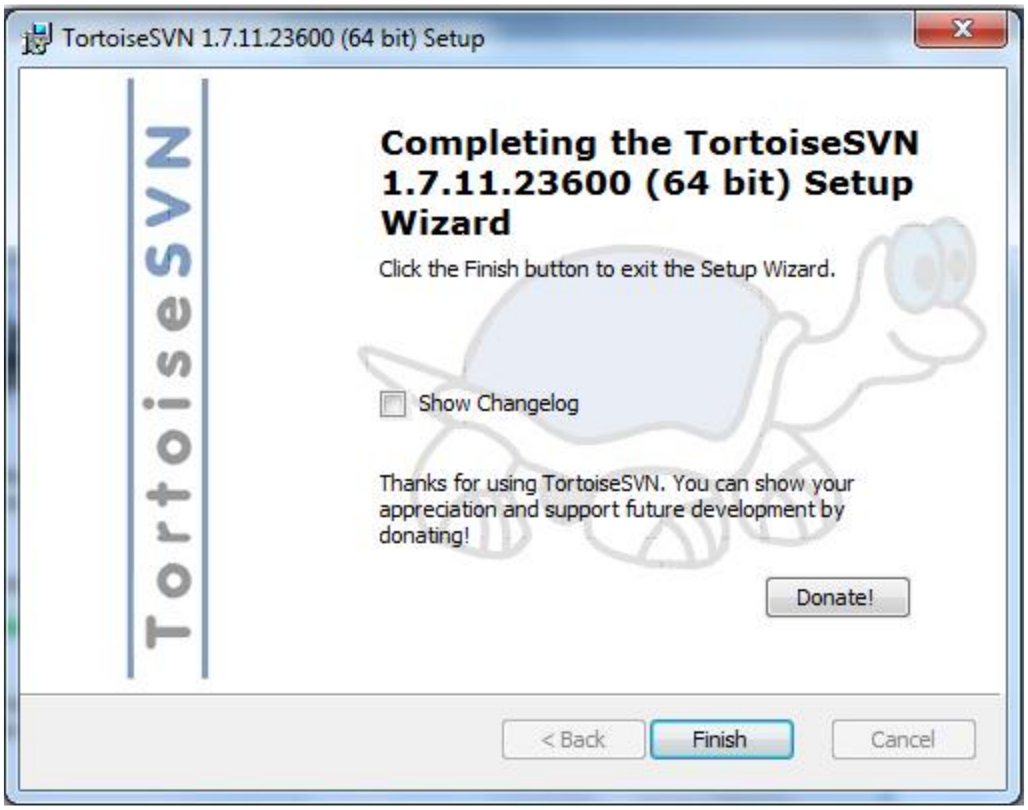

TortoiseSVN 1.7.11.23600 (64 bit) Setup

Installing TortoiseSVN 1.7.11.23600 (64 bit)

Please wait while the Setup Wizard installs TortoiseSVN 1.7.11.23600 (64 bit). This may take several minutes.

Status: Writing system registry values

Thanks for using TortoiseSVN. You can show your appreciation and support future development by donating!

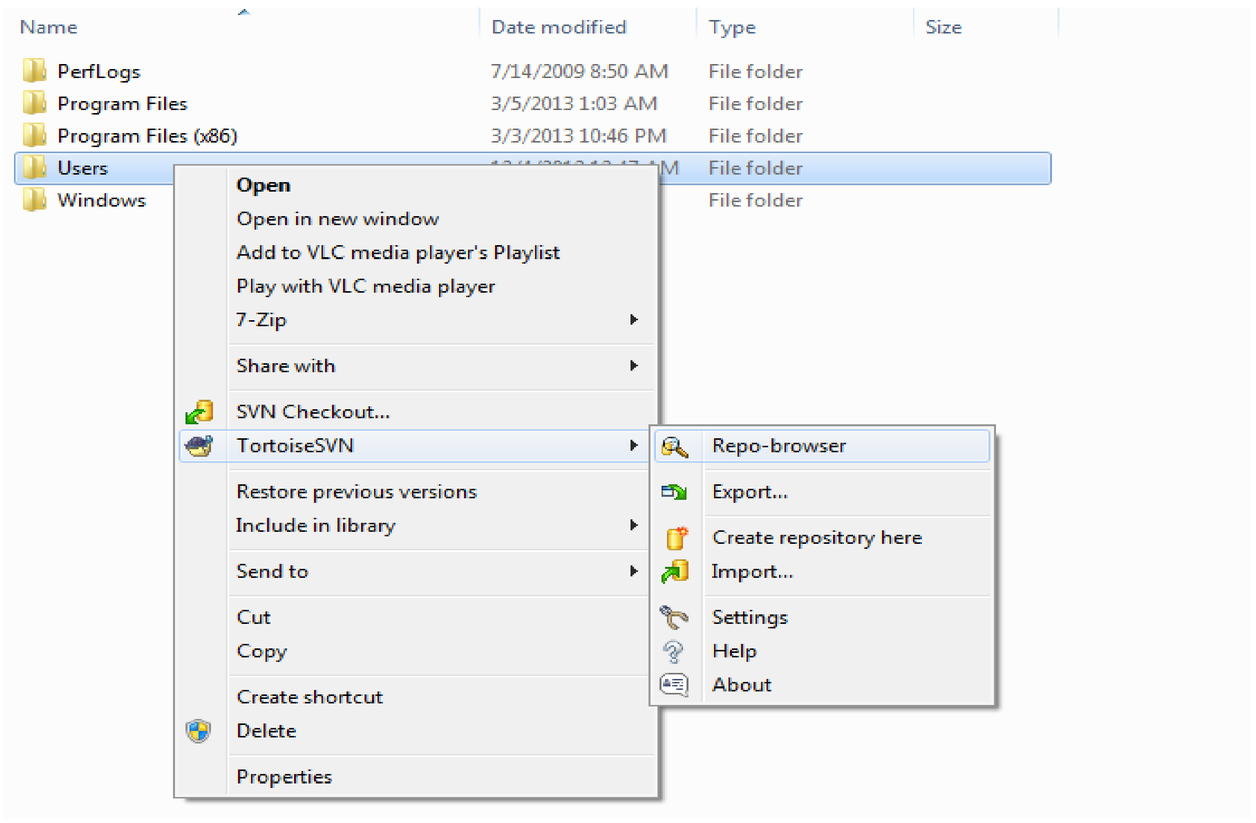

TortoiseSVN 1.7.11.23600 (64 bit) Setup

Completing the TortoiseSVN 1.7.11.23600 (64 bit) Setup Wizard

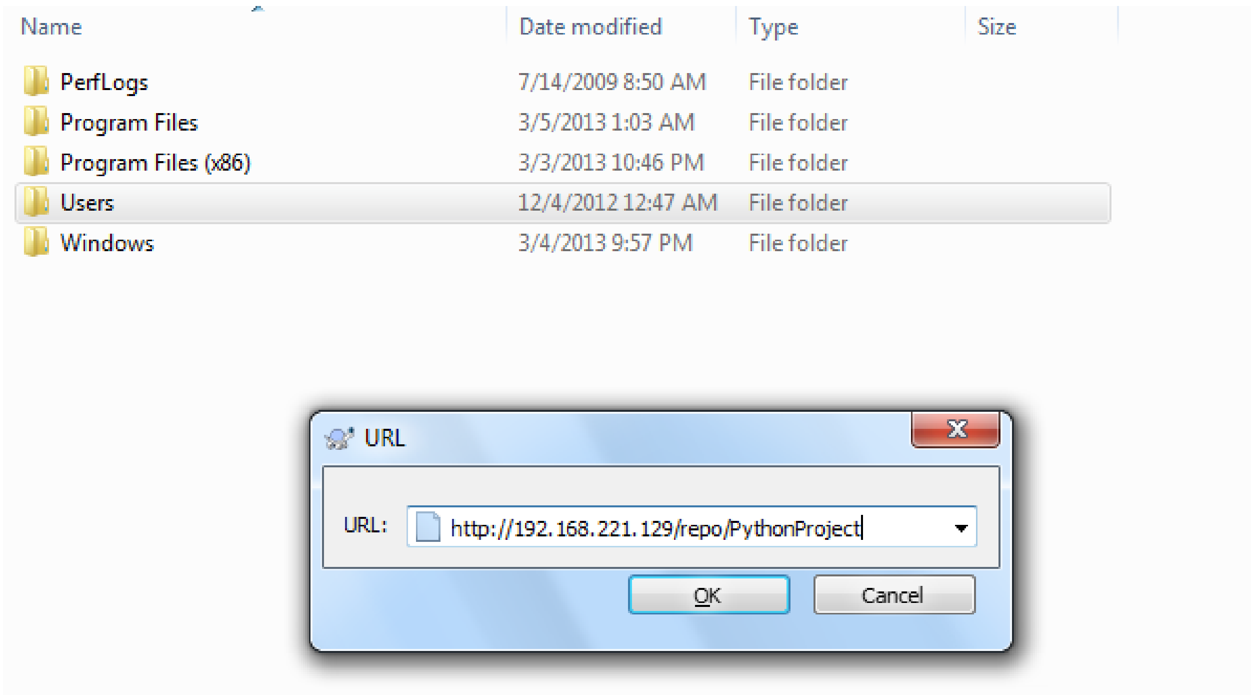
Click the Finish button to exit the Setup Wizard.

[ ] Show Changelog

Thanks for using TortoiseSVN. You can show your appreciation and support future development by donating!



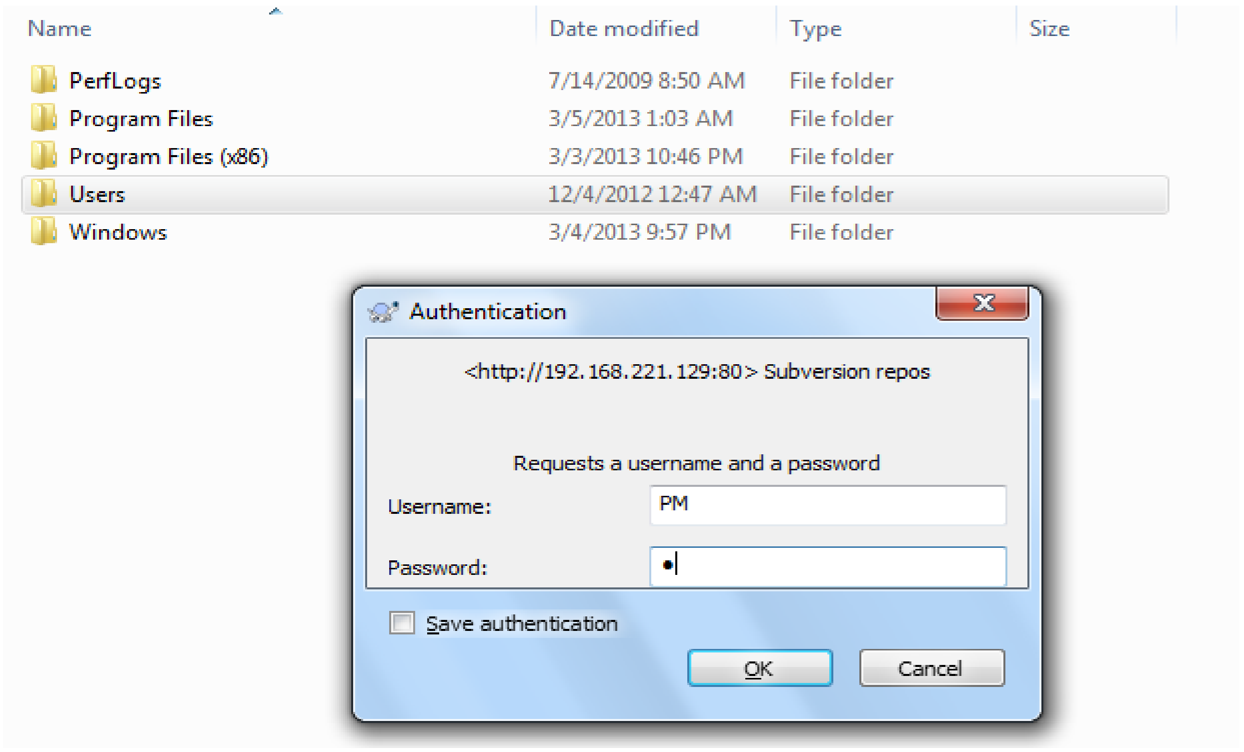
Name	Date modified	Type	Size
PerfLogs	7/14/2009 8:50 AM	File folder	
Program Files	3/5/2013 1:03 AM	File folder	
Program Files (x86)	3/3/2013 10:46 PM	File folder	
Users		File folder	
Windows		File folder	



Name	Date modified	Type	Size
PerfLogs	7/14/2009 8:50 AM	File folder	
Program Files	3/5/2013 1:03 AM	File folder	
Program Files (x86)	3/3/2013 10:46 PM	File folder	
Users	12/4/2012 12:47 AM	File folder	
Windows	3/4/2013 9:57 PM	File folder	

URL

URL: http://192.168.221.129/repo/PythonProject



Name	Date modified	Type	Size
PerfLogs	7/14/2009 8:50 AM	File folder	
Program Files	3/5/2013 1:03 AM	File folder	
Program Files (x86)	3/3/2013 10:46 PM	File folder	
Users	12/4/2012 12:47 AM	File folder	
Windows	3/4/2013 9:57 PM	File folder	

Authentication

<http://192.168.221.129:80> Subversion repos

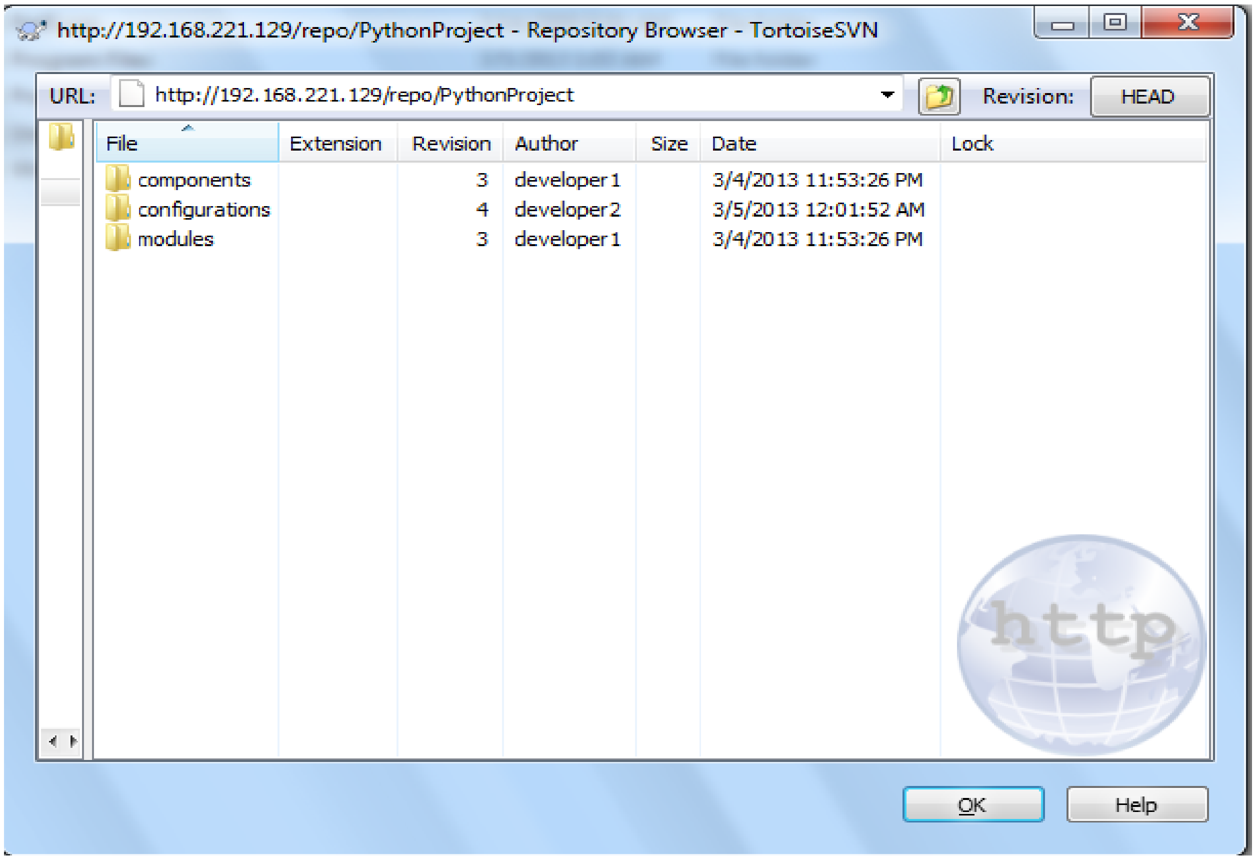
Requests a username and a password

Username: PM

Password: •

☐ Save authentication

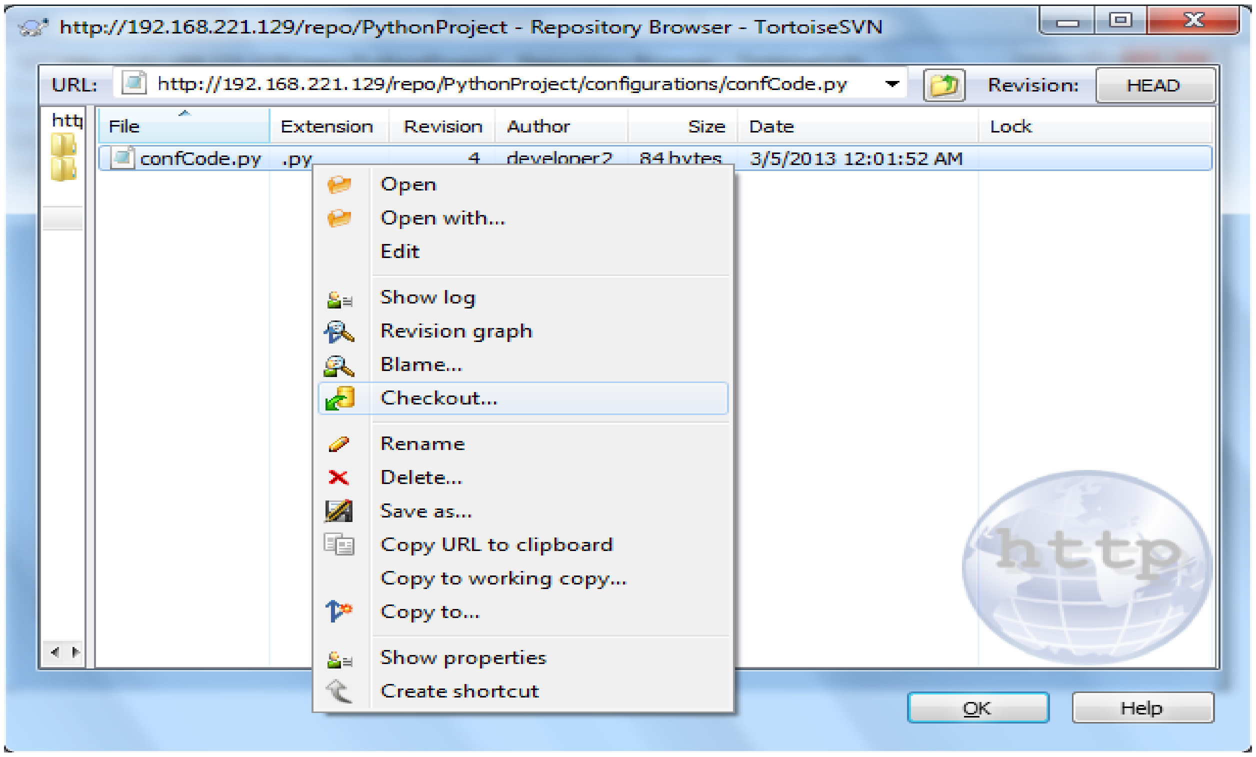
OK Cancel



http://192.168.221.129/repo/PythonProject - Repository Browser - TortoiseSVN

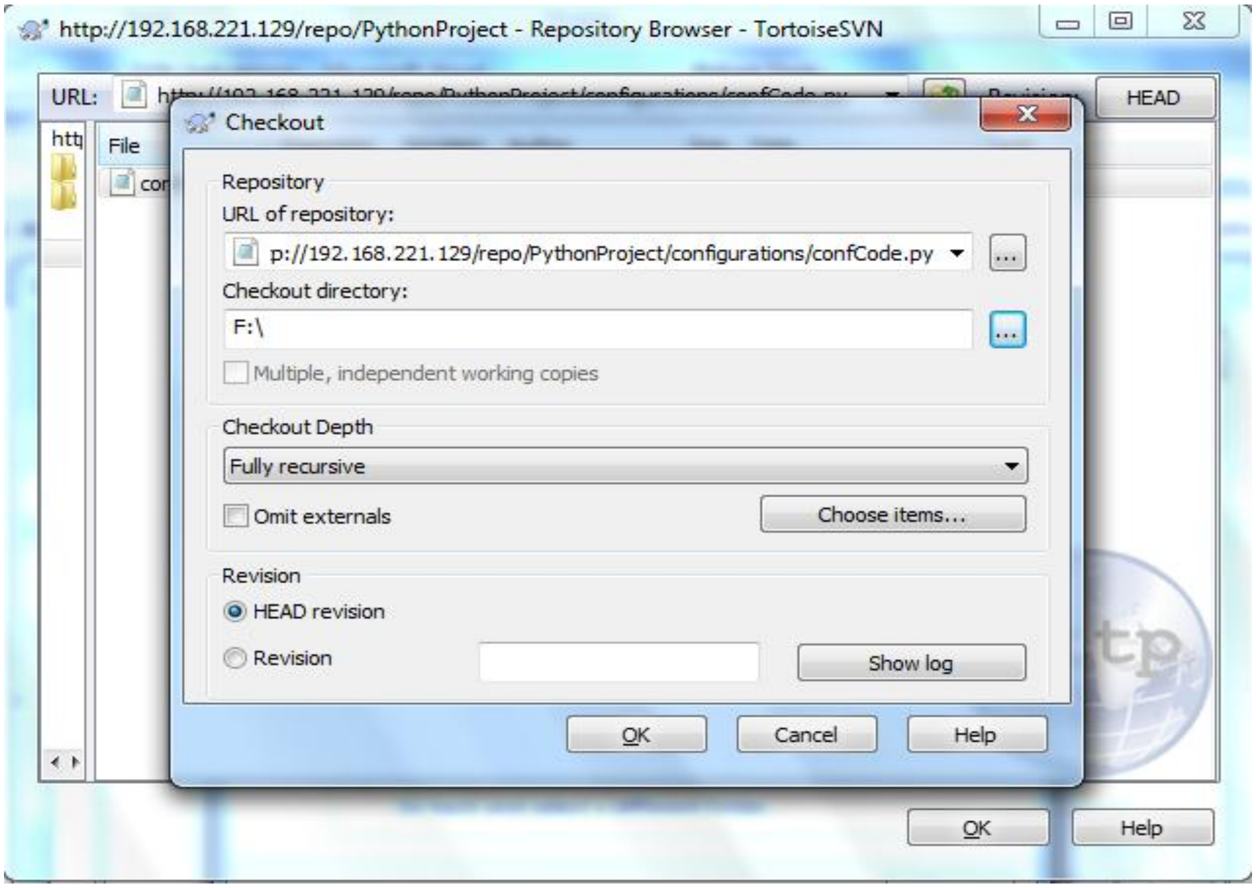
URL: http://192.168.221.129/repo/PythonProject Revision: HEAD

File	Extension	Revision	Author	Size	Date	Lock
components		3	developer1		3/4/2013 11:53:26 PM	
configurations		4	developer2		3/5/2013 12:01:52 AM	
modules		3	developer1		3/4/2013 11:53:26 PM	


http://192.168.221.129/repo/PythonProject - Repository Browser - TortoiseSVN

URL: http://192.168.221.129/repo/PythonProject/configurations/confCode.py Revision: HEAD

File	Extension	Revision	Author	Size	Date	Lock
confCode.py	.py	4	developer2	84 bytes	3/5/2013 12:01:52 AM	


Checkout

Repository

URL of repository: p://192.168.221.129/repo/PythonProject/configurations/confCode.py

Checkout directory: F:\

[ ] Multiple, independent working copies

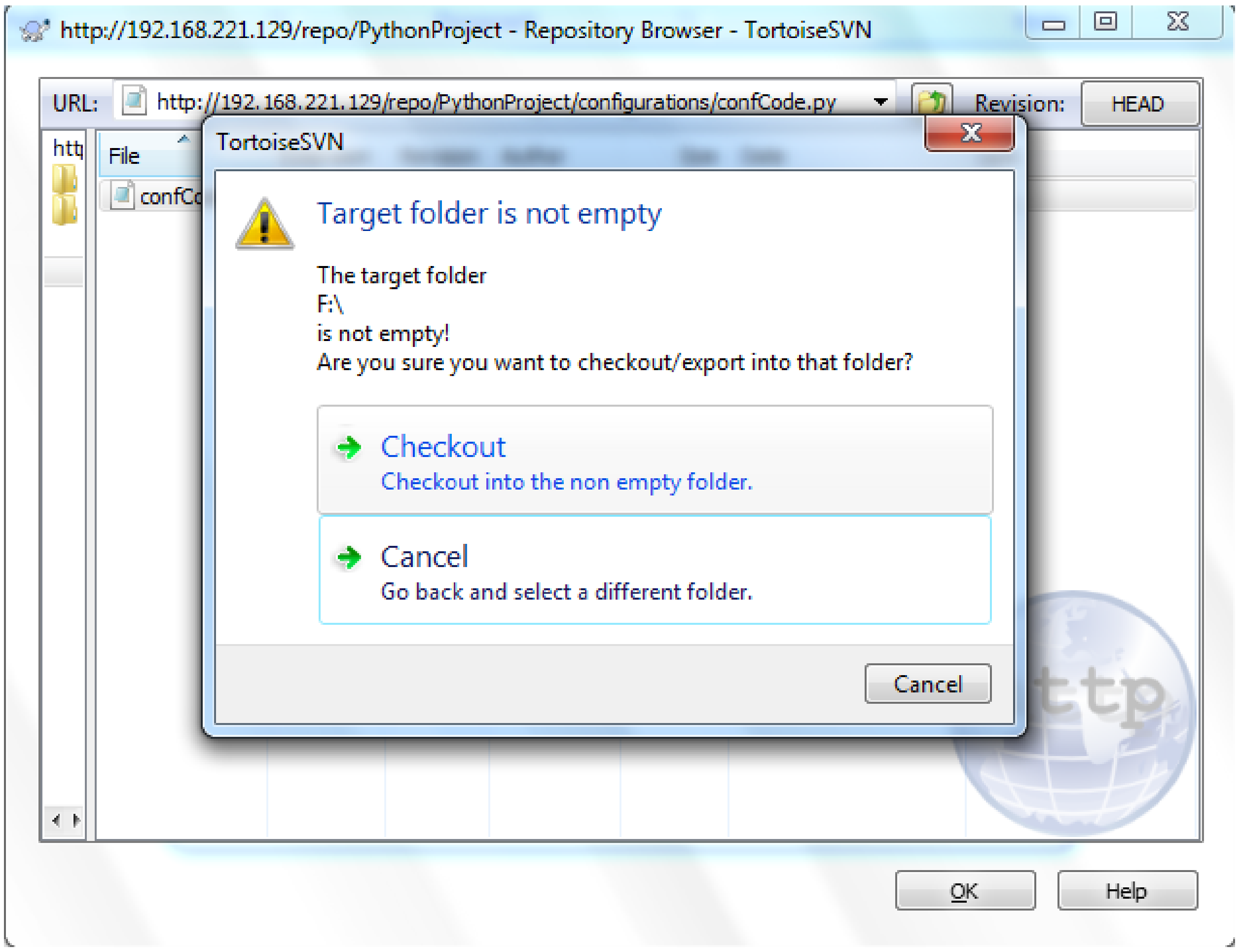
Checkout Depth: Fully recursive

[ ] Omit externals

Revision

[●] HEAD revision

[○] Revision


http://192.168.221.129/repo/PythonProject - Repository Browser - TortoiseSVN

URL: http://192.168.221.129/repo/PythonProject/configurations/confCode.py Revision: HEAD

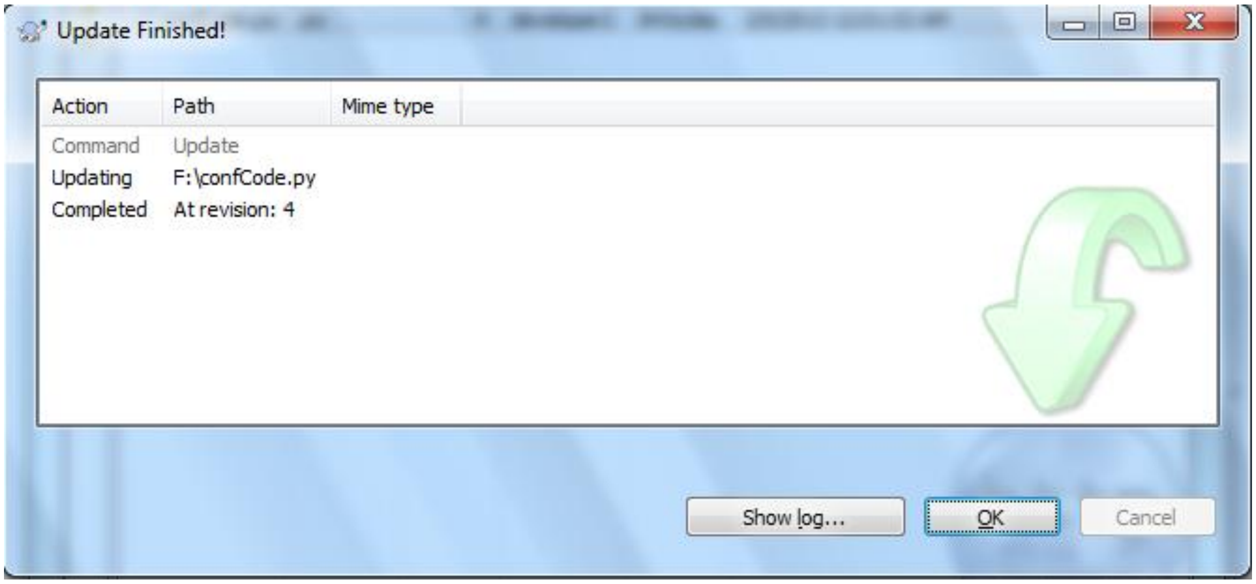
TortoiseSVN

Target folder is not empty

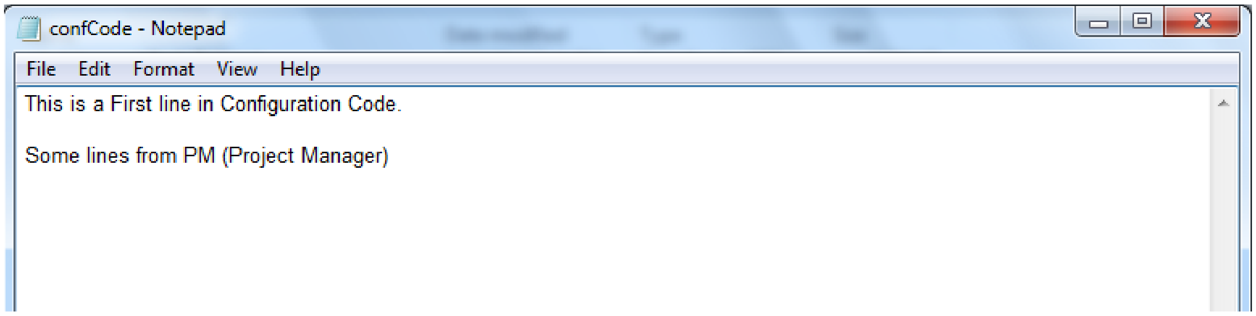
The target folder
F:\
is not empty!
Are you sure you want to checkout/export into that folder?

Checkout
Checkout into the non empty folder.

Cancel
Go back and select a different folder.

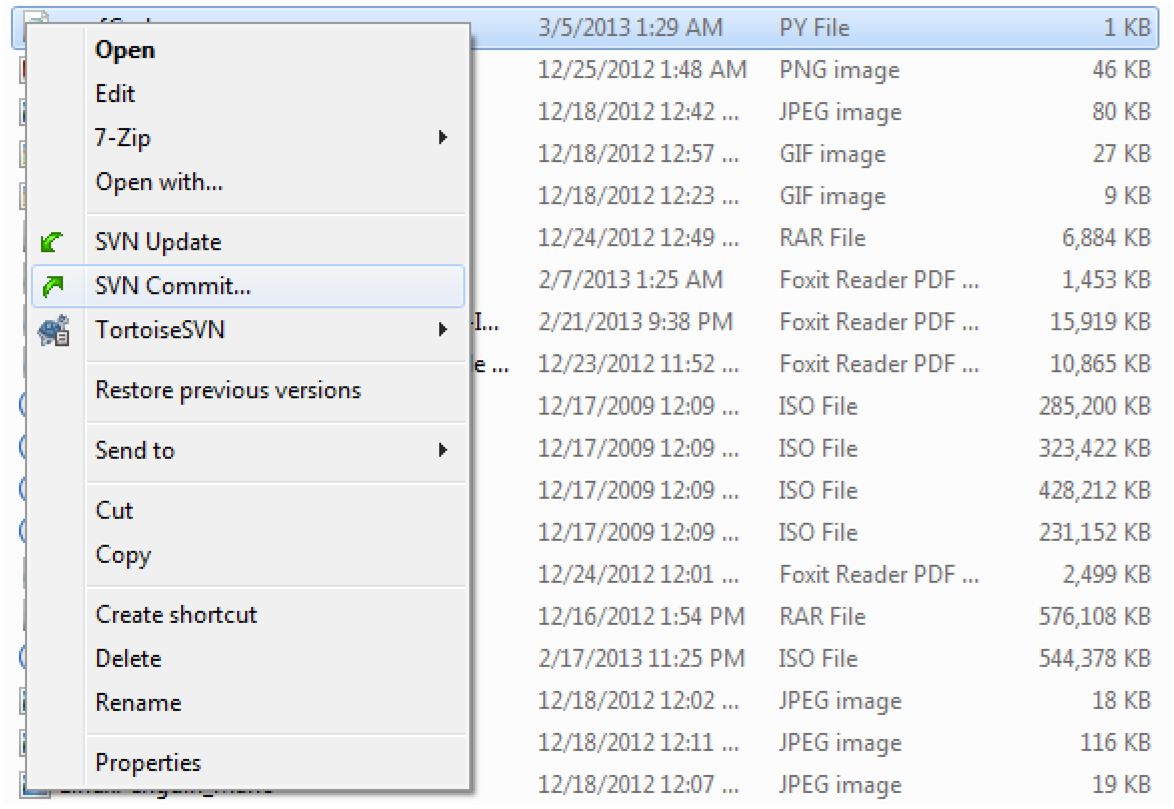

Update Finished!

```
Action      Path              Mime type
Command     Update
Updating    F:\confCode.py
Completed   At revision: 4
```

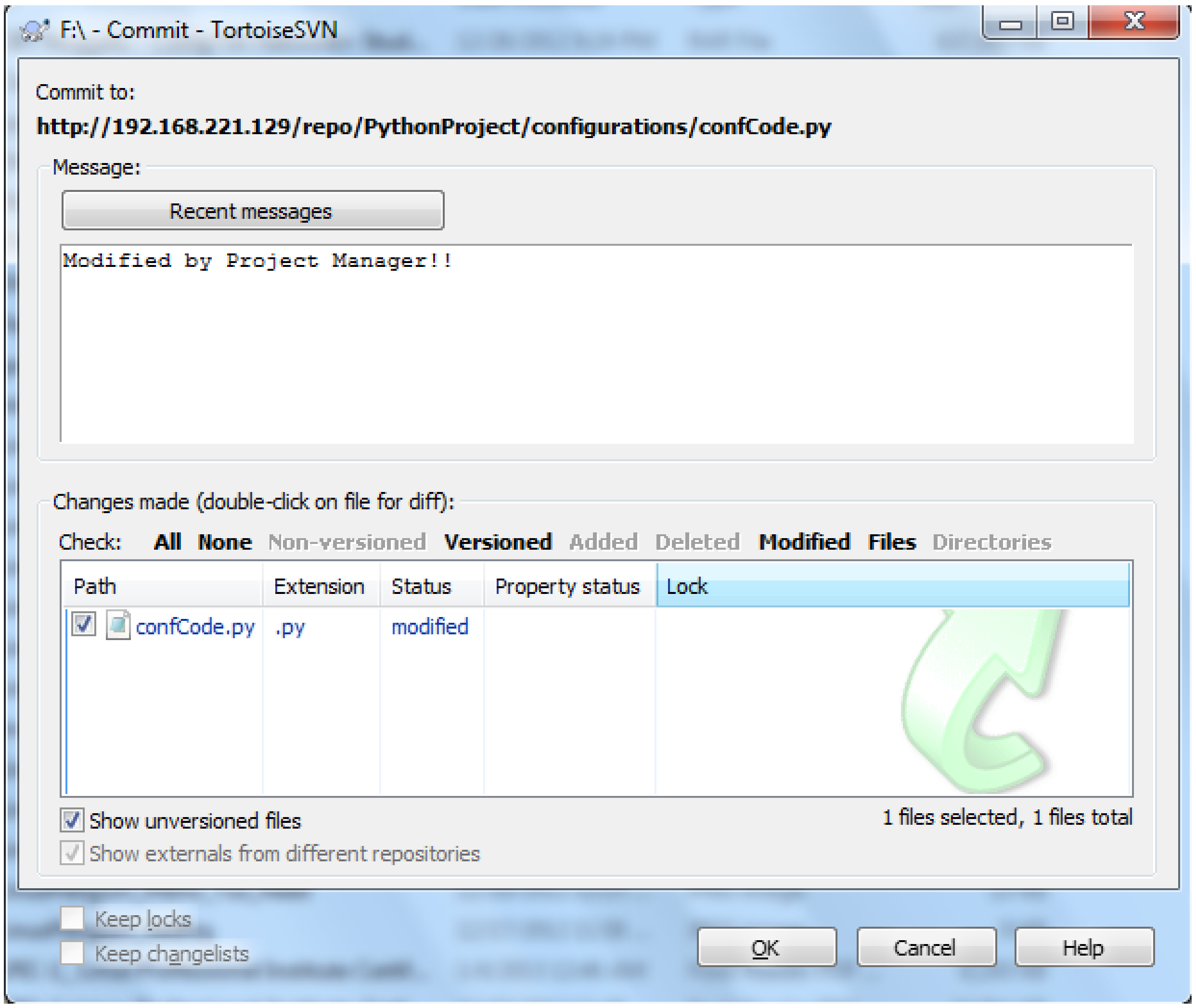


```
This is a First line in Configuration Code.

Some lines from PM (Project Manager)
```



3/5/2013 1:29 AM	PY File	1 KB
12/25/2012 1:48 AM	PNG image	46 KB
12/18/2012 12:42 ...	JPEG image	80 KB
12/18/2012 12:57 ...	GIF image	27 KB
12/18/2012 12:23 ...	GIF image	9 KB
12/24/2012 12:49 ...	RAR File	6,884 KB
2/7/2013 1:25 AM	Foxit Reader PDF ...	1,453 KB
2/21/2013 9:38 PM	Foxit Reader PDF ...	15,919 KB
12/23/2012 11:52 ...	Foxit Reader PDF ...	10,865 KB
12/17/2009 12:09 ...	ISO File	285,200 KB
12/17/2009 12:09 ...	ISO File	323,422 KB
12/17/2009 12:09 ...	ISO File	428,212 KB
12/17/2009 12:09 ...	ISO File	231,152 KB
12/24/2012 12:01 ...	Foxit Reader PDF ...	2,499 KB
12/16/2012 1:54 PM	RAR File	576,108 KB
2/17/2013 11:25 PM	ISO File	544,378 KB
12/18/2012 12:02 ...	JPEG image	18 KB
12/18/2012 12:11 ...	JPEG image	116 KB
12/18/2012 12:07 ...	JPEG image	19 KB


F:\ - Commit - TortoiseSVN

Commit to:
http://192.168.221.129/repo/PythonProject/configurations/confCode.py

Message:

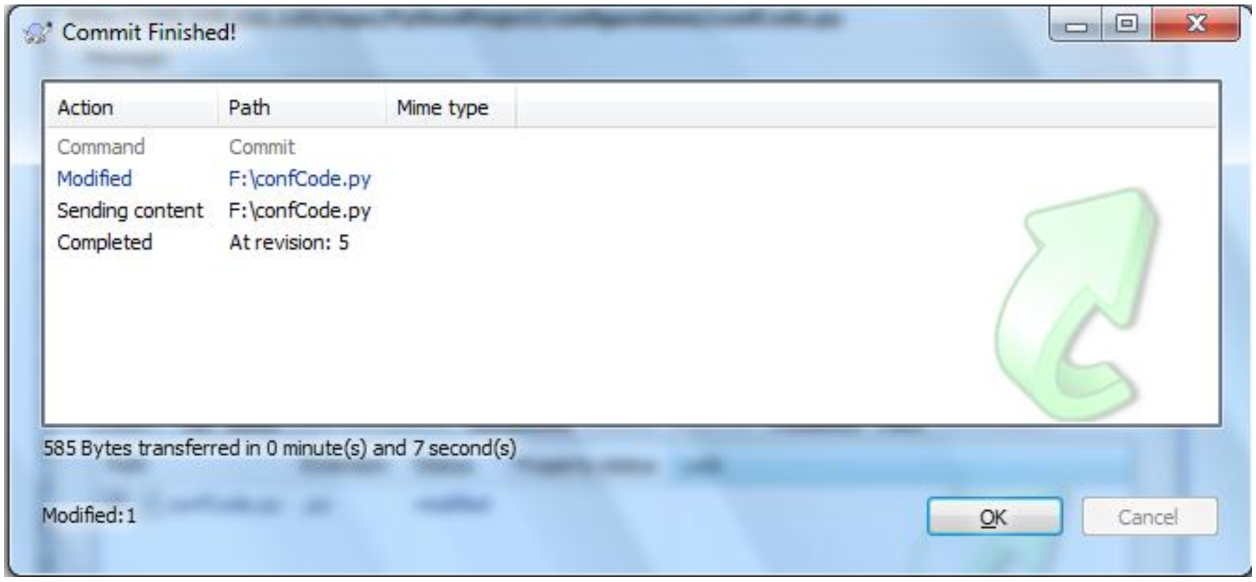
```
Modified by Project Manager!!
```

Changes made (double-click on file for diff):

Check: All None Non-versioned Versioned Added Deleted Modified Files Directories

Path	Extension	Status	Property status	Lock
☑ confCode.py	.py	modified		

☑ Show unversioned files — 1 files selected, 1 files total
☑ Show externals from different repositories
☐ Keep locks
☐ Keep changelists



Action	Path	Mime type
Command	Commit	
Modified	F:\confCode.py	
Sending content	F:\confCode.py	
Completed	At revision: 5	

585 Bytes transferred in 0 minute(s) and 7 second(s)

Modified: 1

ALTNIX Research & Development Team

- Mr. Sadhiq Bashir Ahmed
- Mr. Manoharan Nadar