****

Contents

[Introduction to version control 2](#_Toc425520764)

[Why need version control? 3](#_Toc425520765)

[Different type of Version control tools 3](#_Toc425520766)

[Introducing SVN 4](#_Toc425520767)

[Introduction: 4](#_Toc425520768)

[Architecture: 5](#_Toc425520769)

[Installation: 8](#_Toc425520770)

[Creating Repository: 13](#_Toc425520771)

[Checking out repository and committing changes: 16](#_Toc425520772)

[The Working Cycle: 21](#_Toc425520773)

[Installing CLI for SVN (SlikSVN) : 22](#_Toc425520774)

[Creating branch from trunk to branch and adding changes: 26](#_Toc425520775)

[Merging changes from branch to trunk: 30](#_Toc425520776)

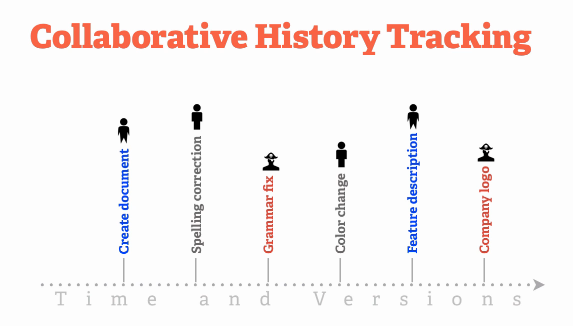
[Checking changes from SVN repository, SVN Log 34](#_Toc425520777)

# Introduction to version control

**Version control** is a system that records changes to a file or set of files over time so that you can recall specific versions later

Version control is also known as revision control or source control.

* + version control provides
  + Management of versions of information.
  + Keeps track of you creative output.
  + It tracks what is change made.
  + Tracks who made the changes
  + Tracks why the changes are made.
  + Documentation –
  + in a structured, standardized way
  + – with repositories.
  + centralized (SVN, CVS).
  + decentralized (GIT).



# Why need version control?

* + enables collaboration between developers.
  + understanding who made a change and when it happened.
  + centralized “main code” (trunk).
  + custom code alongside main code (branching).
  + eases release management (tags)
  + rollback to previous revisions.
  + integration with other tools.

# Different type of Version control tools

* + Github [<https://github.com/>]
  + SourceForge [<http://sourceforge.net/>]
  + Google Code [<http://code.google.com/>]
  + Git (open source) [<http://git-scm.com/>]
  + CVS (open source) [<http://www.nongnu.org/cvs/>]
  + Arch (open source) [<http://www.gnu.org/software/gnu-arch/>]
  + Subversion (open source) [<http://subversion.apache.org/>]
  + BitKeeper [<http://www.bitkeeper.com/>]
  + Perforce [<http://www.perforce.com/>]
  + AccuRev [<http://www.accurev.com/accurev-version-control.html>]
  + Visual Source Safe [<http://www.microsoft.com/vstudio/previous/ssafe/>]

# Introducing SVN

## Introduction:

**A software engineering discipline**

* + Comprised of tools and techniques
  + Used to manage change to software assets.

**A set of activities to control change by:**

* + Identifying the products & establishing relationships
  + Defining mechanisms for management, control, auditing and reporting on changes made.

**To manage: (Why do we need SCM?)**

* + Many people working on the same code.
  + Projects delivered in several releases (builds).
  + Rapid evolution of software and hardware.
  + Project surveillance: i.e. project’s status, bugs, features, etc.
  + Concurrent support and development.
  + Stress time requirements of projects

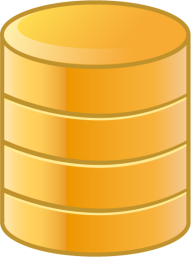
**Version management tool for all types of files and directories**

* + Manage documents over time
  + keep a history of all changes - multiple versions of every file
  + Coordinate work of multiple authors
  + Avoid conflicts ...and help resolve them
  + Permissions: authenticate and control access to files
  + Show differences between versions of a file
  + Document changes -- reason for change

Subversion is a free/open source version control system (VCS). That is, Subversion manages files and directories, and the changes made to them, over time. This allows you to recover older versions of your data or examine the history of how your data changed. In this regard, many people think of a version control system as a sort of “time machine.”

Subversion can operate across networks, which allows it to be used by people on different computers. At some level, the ability for various people to modify and manage the same set of data from their respective locations fosters collaboration. Progress can occur more quickly without a single conduit through which all modifications must occur. And because the work is versioned, you need not fear that quality is the trade-off for losing that conduit—if some incorrect change is made to the data, just undo that change.

## Architecture:



Berkley DB

FSFS

Subversion

Repository

Client Interface

Repository Interface

GUI client

apps

Command line

client apps

Client

Library

Working Copy Management

Library

Repository

access

DAV

SVN

Local

Apache

mod\_dav

mod\_dav\_svn

svnserve

Internet

(Any TCP/IP Network)‏



http://

https://

svn://

svn+ssh://

file://

**Directory structure**

* + $ ls repos

conf/ db/ format hooks/ locks/ README.txt

* + Here's a quick fly-by overview of what exactly you're seeing in this directory listing
  + conf --A directory containing configuration files
  + Db -- The data store for all of your versioned data
  + Format -- A file that contains a single integer that indicates the version number of the repository layout
  + Hooks --A directory full of hook script templates (and hook scripts themselves, once you've installed some)
  + Locks --A directory for Subversion's repository lock files, used for tracking accessors to the repository
  + README.txt --A file whose contents merely inform its readers that they are looking at a Subversion repository

Conf/

Db/

Format

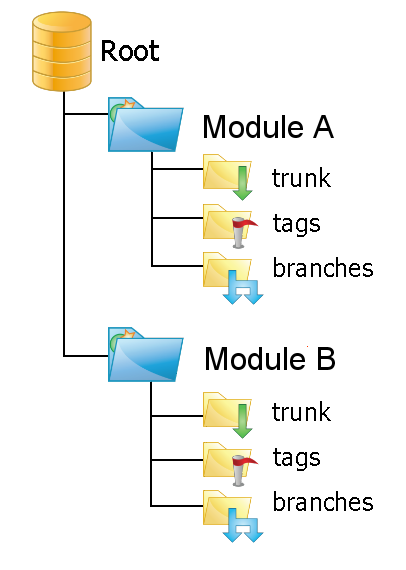
Hooks/

Locks/

README.txt

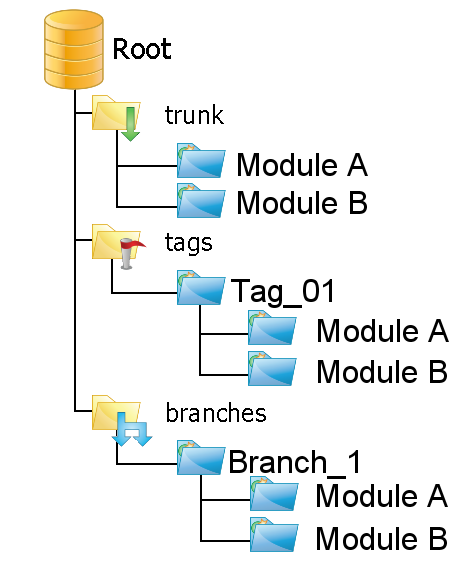
**Modules on root**

* + each module has to be checked out separately
  + commits cannot span multiple modules
  + tags cannot span multiple modules in single commit (workaround by 3rd party tools)‏
  + “official“ svn layout



**trunk/tags/branches on rootall modules will be tagged**

* + checkout of all modules is possible
  + commits can span multiple modules
  + tags can span multiple modules in single commit
  + maybe difficult to determine relevant modules of a tag

****

## Installation:

* + Download the Tortoise SVN client from

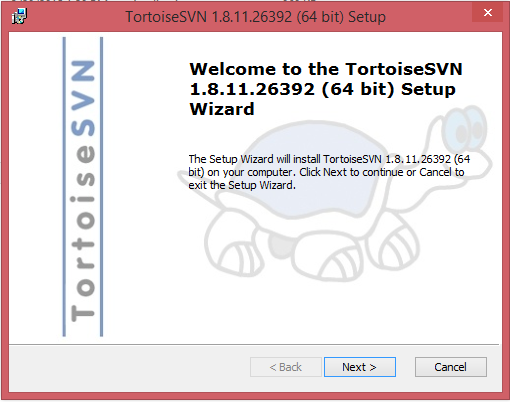
<http://tortoisesvn.net/>

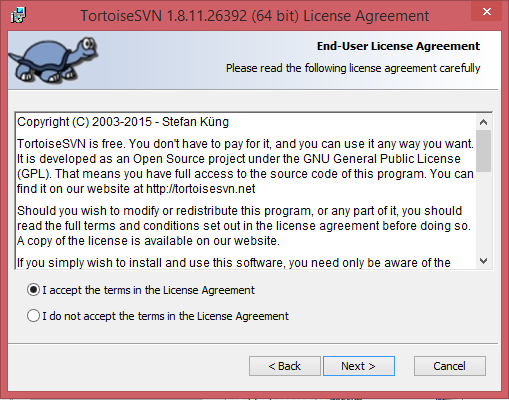
* + Depending on your operating system,you might choose the

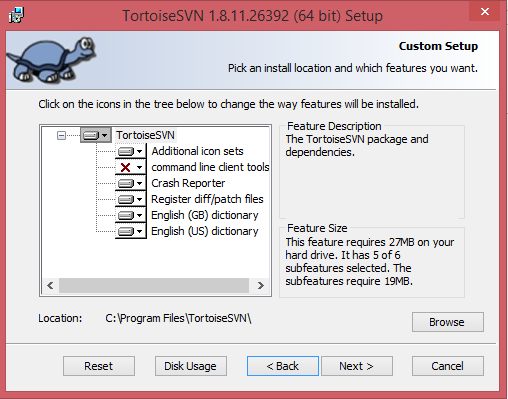
32-­‐bit or 64-­‐bit versions.

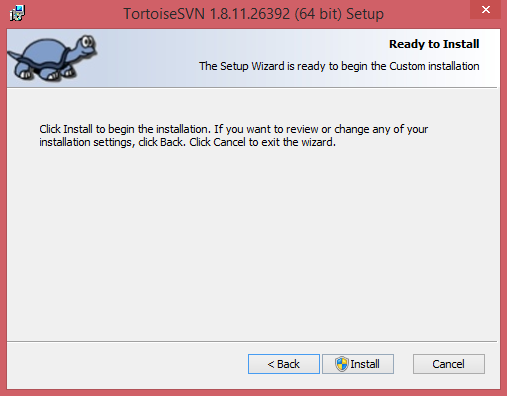
* + Install and restart.

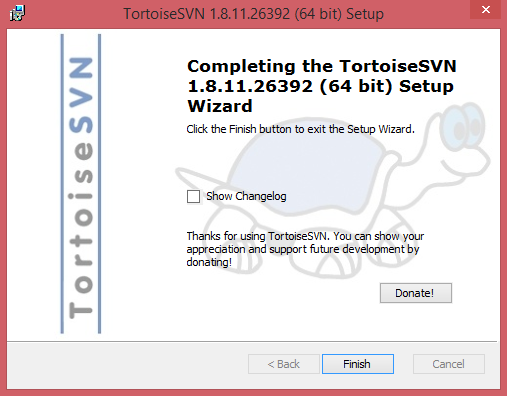
**Installation Steps**

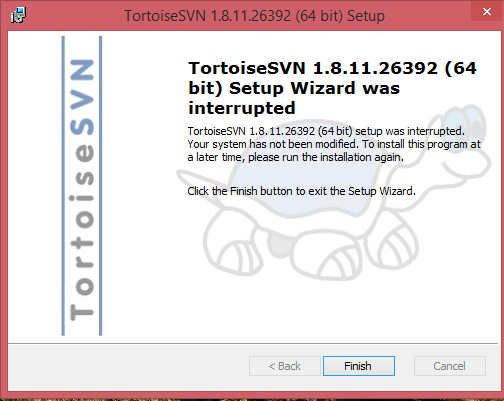




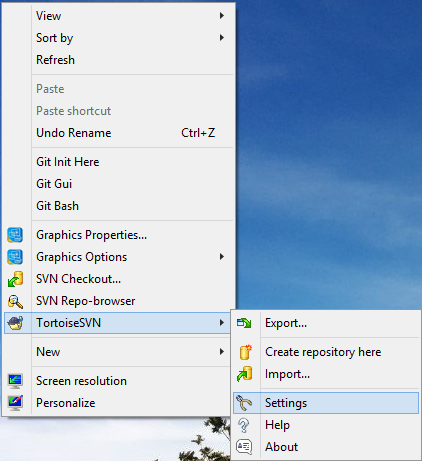


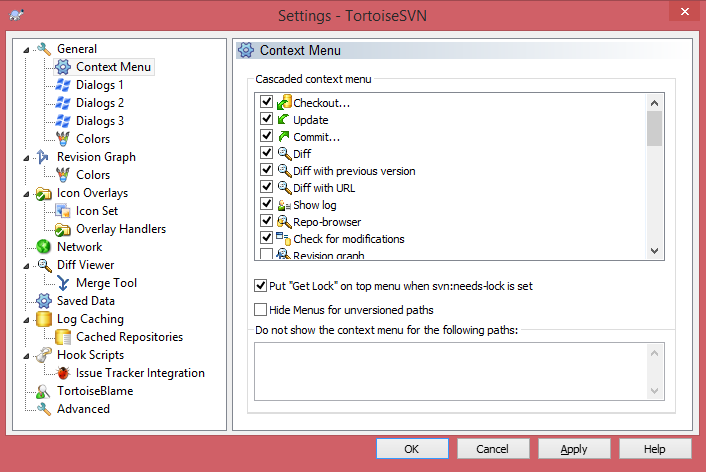
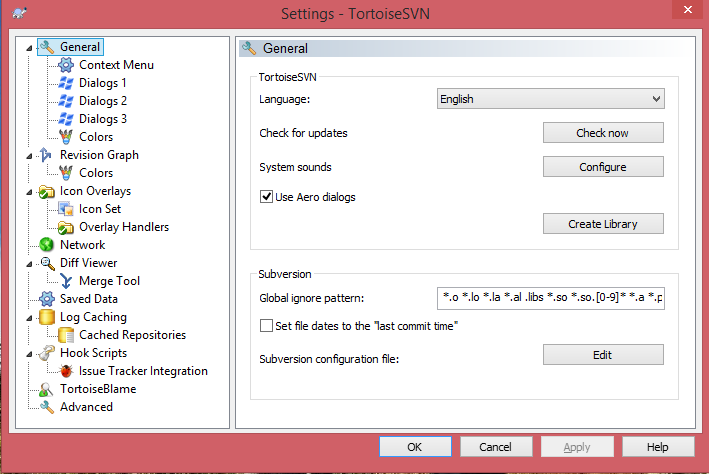






**Configure Additional Option**

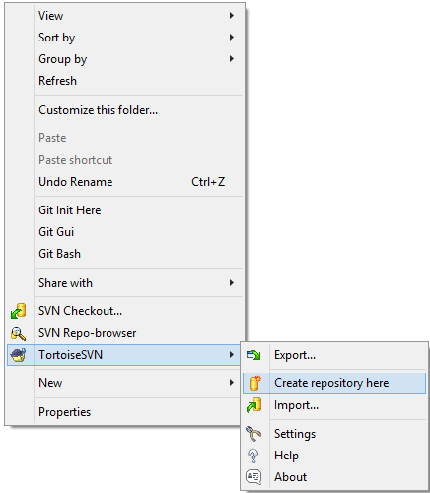


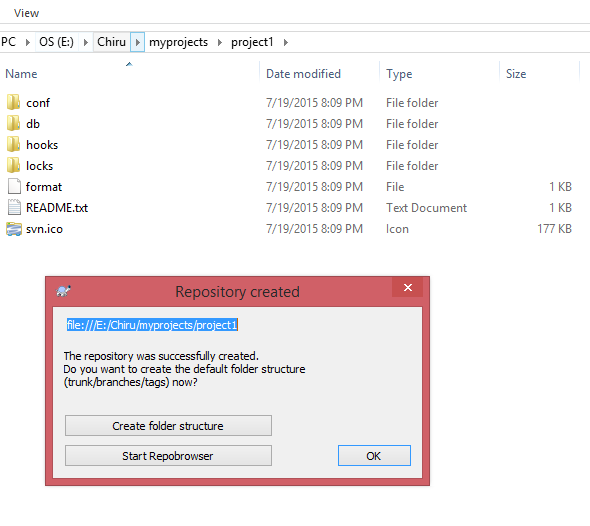
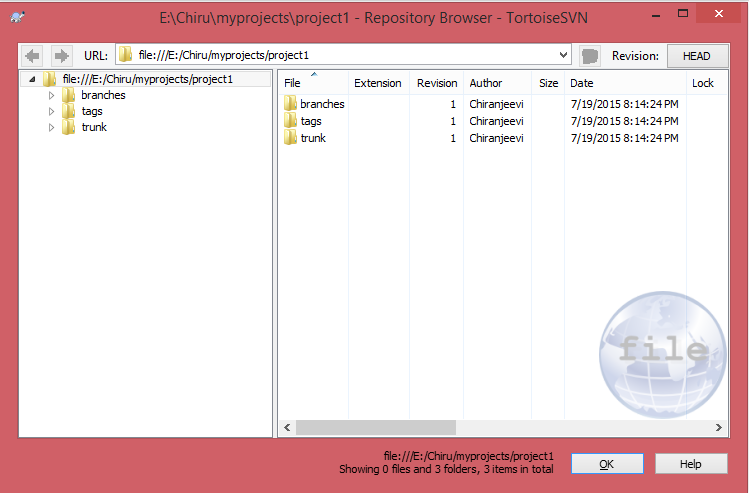
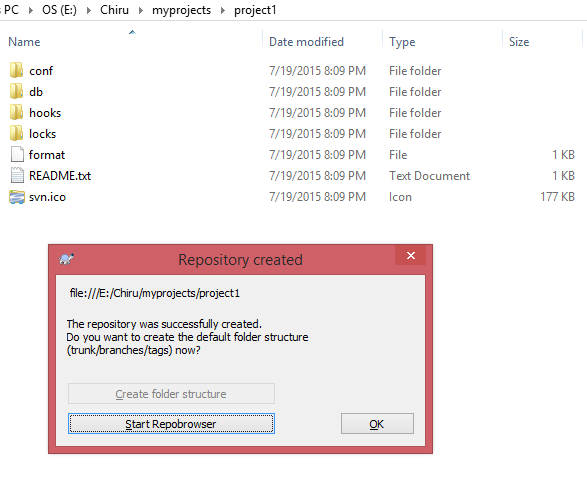
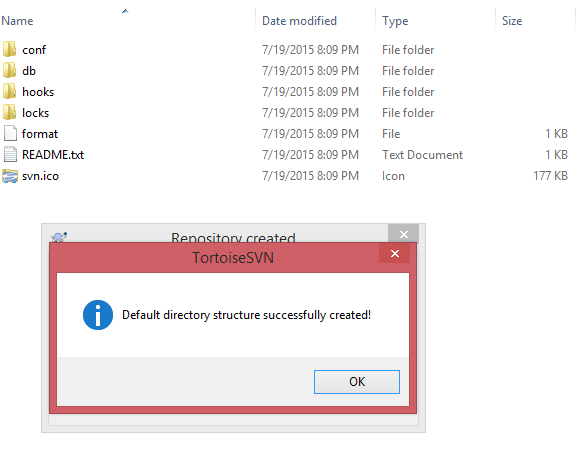


## Creating Repository:

**Creating local file repository**

* + Select the folder where repository needs to be created.
  + Example : E:\Chiru\myprojects\project1
  + Select create folder structure.
  + Folder structure will be created.
  + Start the repo browser, you can see the directory structure created.
  + We can use the same URL to check out the project created.

****

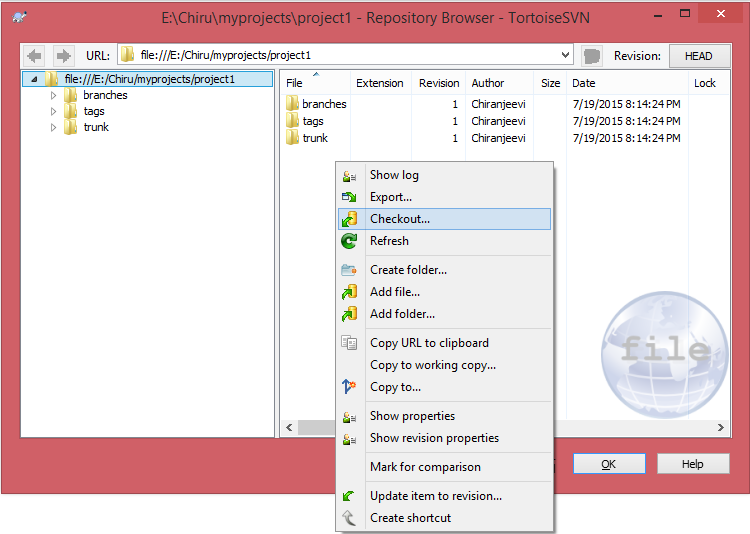
****

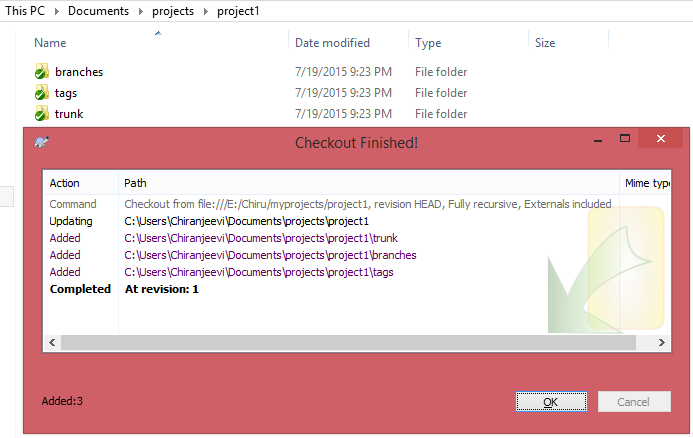
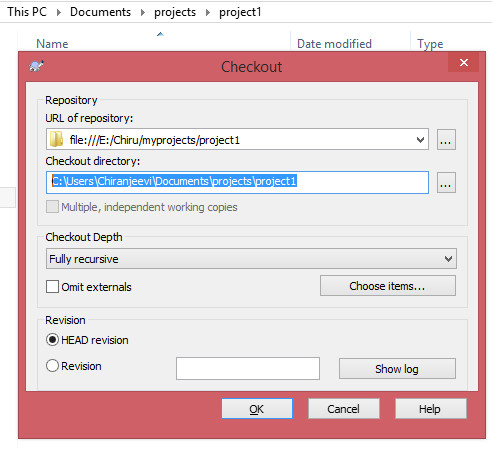
gjghjgjhgjgjhgj

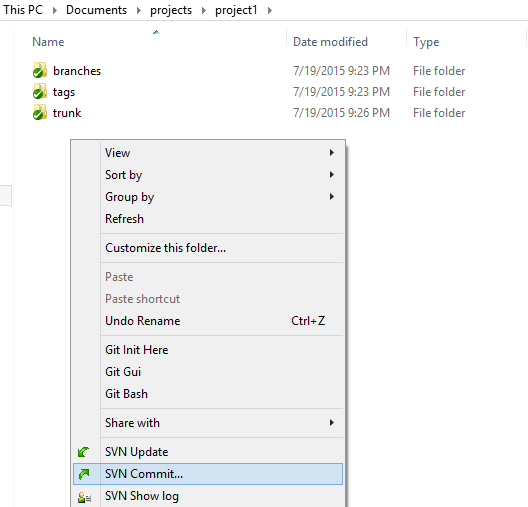
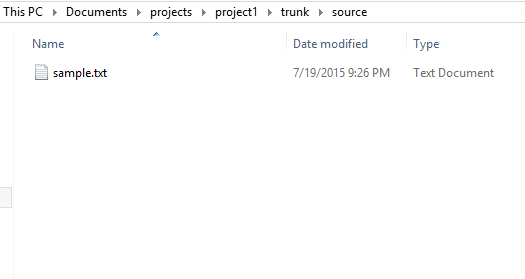
## Checking out repository and committing changes:

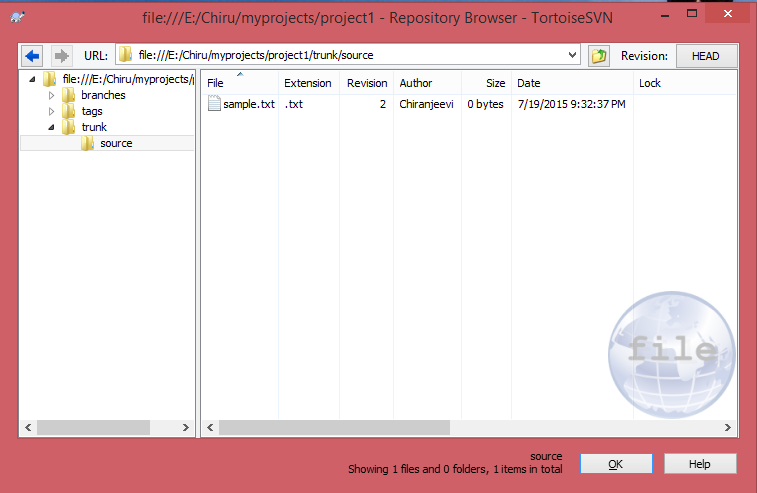
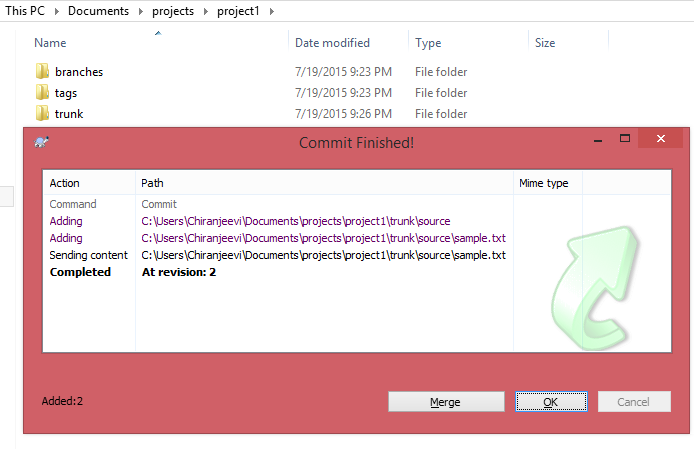
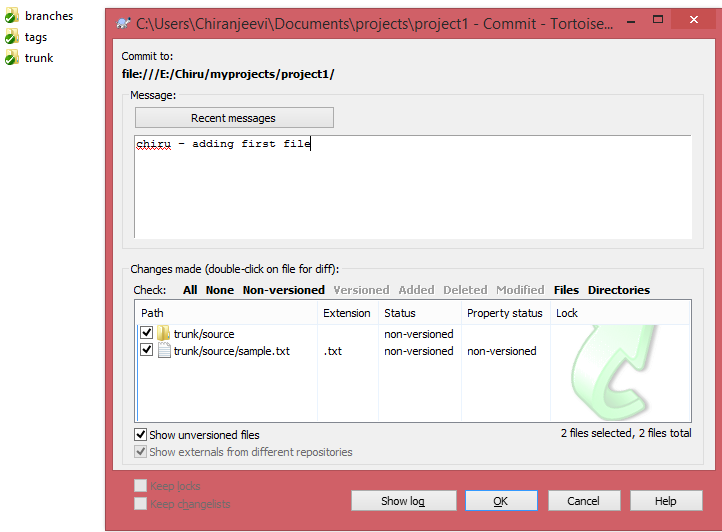
**Checking out directory and committing changes**

* + Right click on the repo browser and select checkout
  + Select the checkout directory where the project needs to be checked out. Example : C:\Users\Chiranjeevi\Documents\projects\project1
  + After checkout is completed. Do the changes in trunk.
  + Commit the changes i.e., newly added files.
  + You can see the newly added files in the repository.





****

****

## The Working Cycle:

**svn checkout**

**svn update**

get content

**svn add**

**svn move**

**svn delete**

Make changes



**svn status -u**

See what was changed

in the repository in the meantime



**svn update**

Update your local copy



**svn diff**

**svn resolved**

Merge your changes  
Resolve conflicts



**svn commit**

Submit your changes



105



100



106



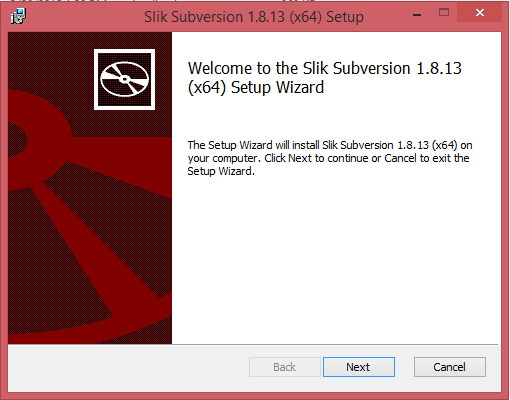
Subversion

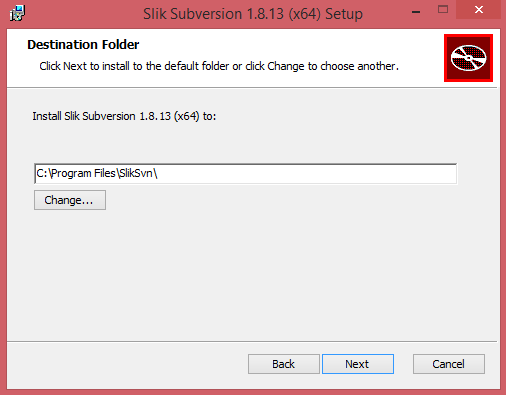
Repository

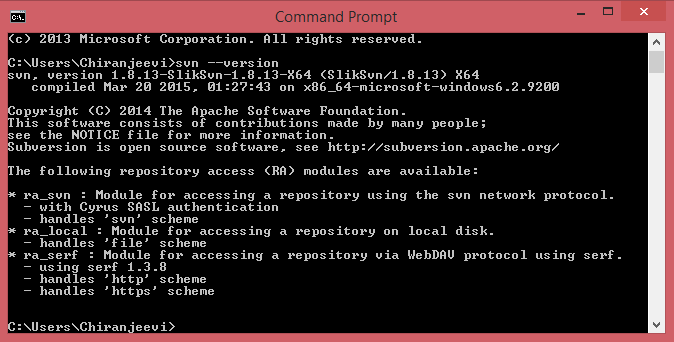
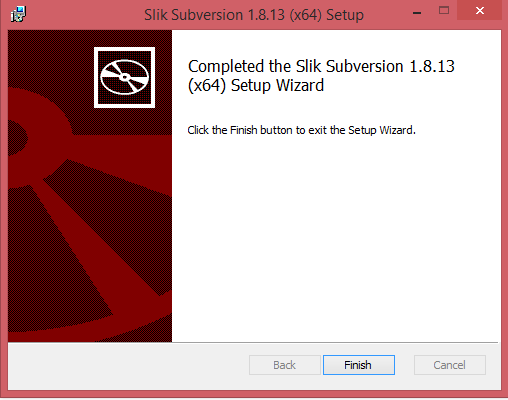
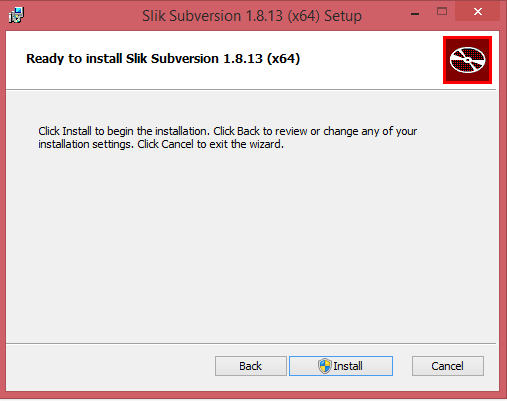
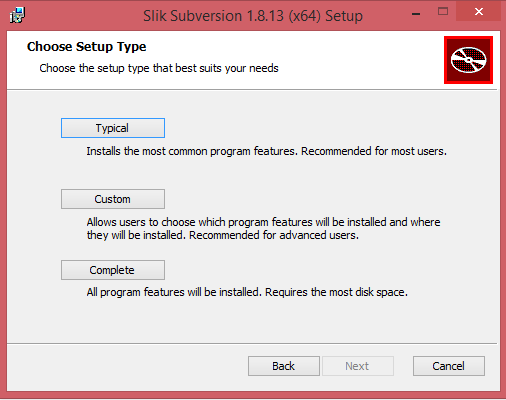
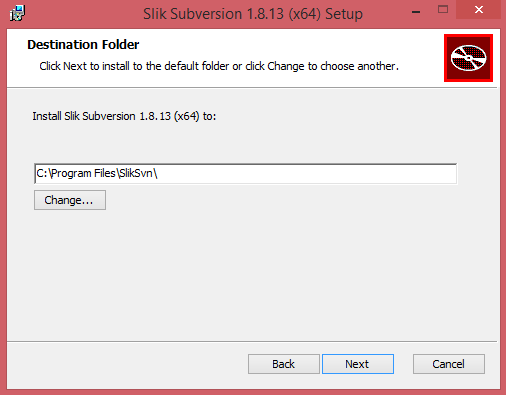
## Installing CLI for SVN (SlikSVN) :

**Installing command line interface for SVN - SlikSVN**

* + Download the CLI from the below location.
  + Once the installation is completed we can see the installed version in command prompt



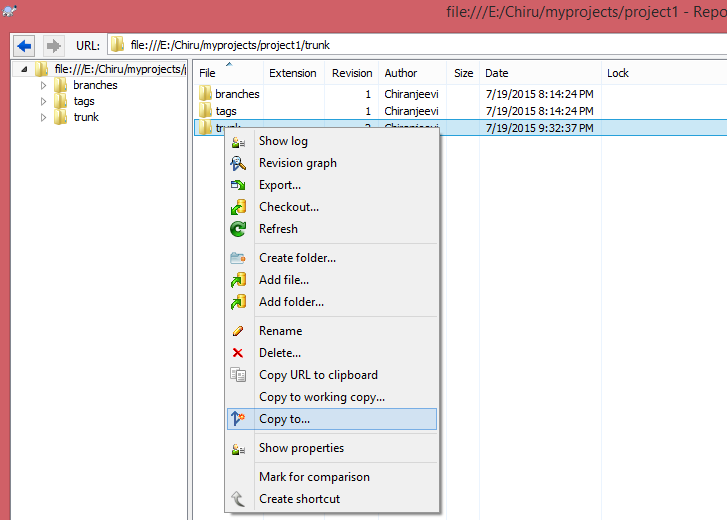


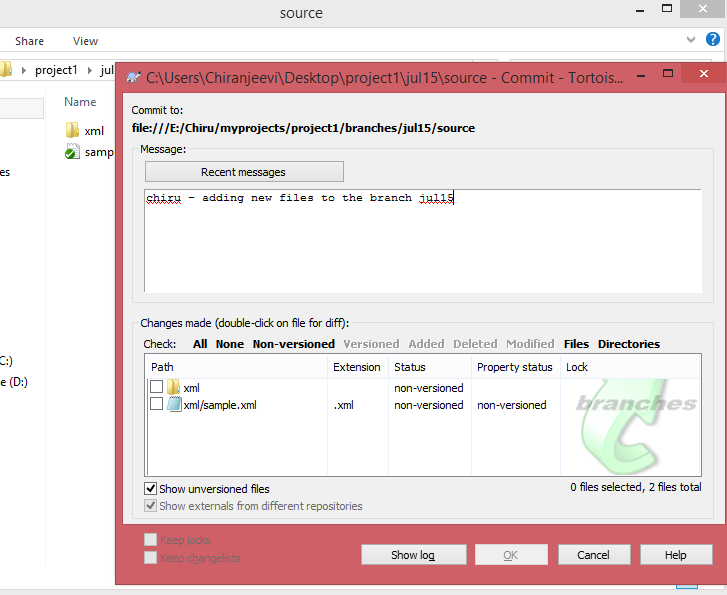
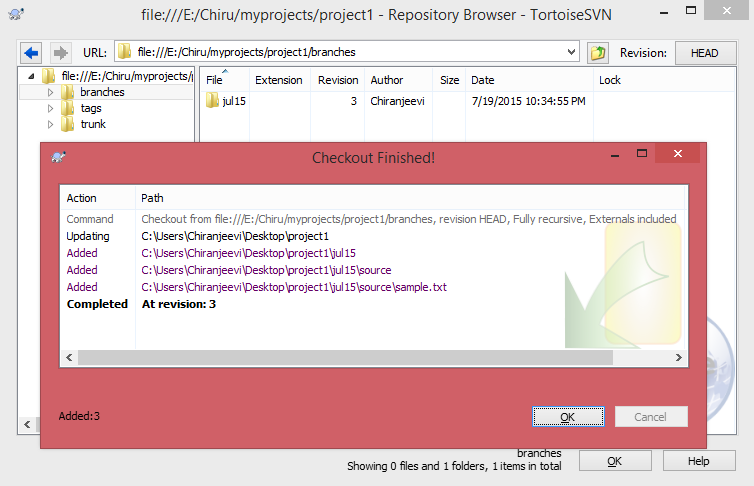
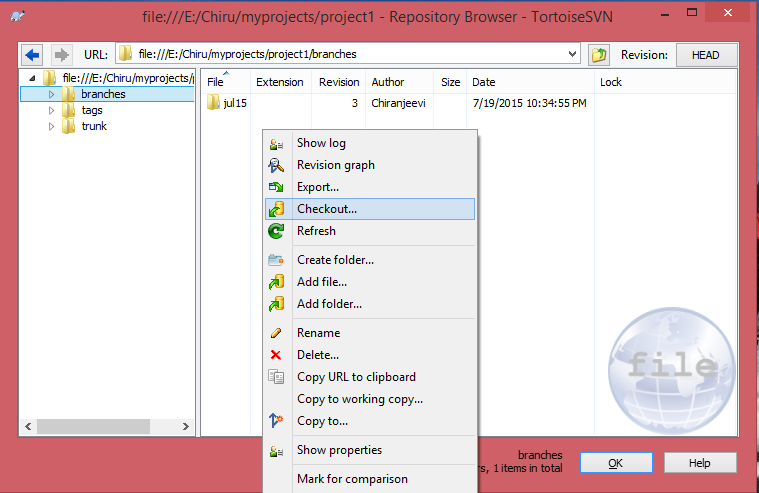
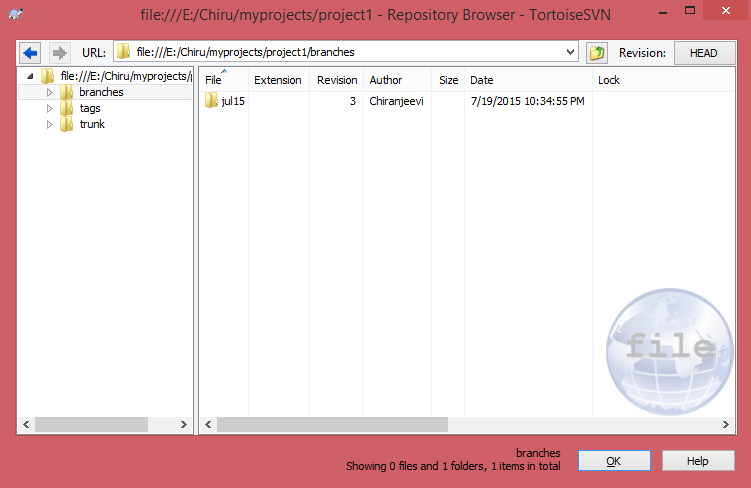
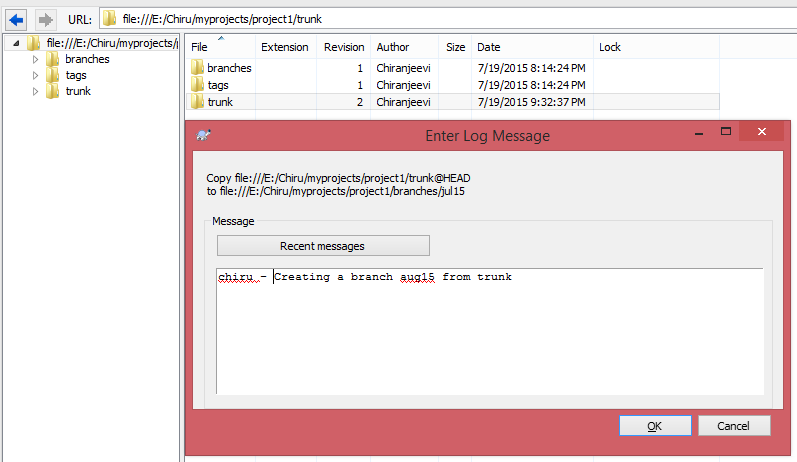
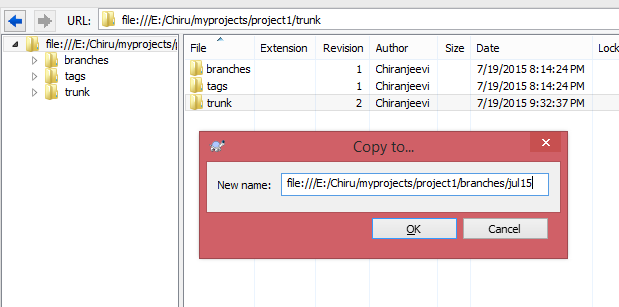


## Creating branch from trunk to branch and adding changes:

**Creating branch out of trunk**

* + Select trunk code, right click and copy to.
  + Give the destination to branches/jul15, jul15 is the new branch which is created.
  + Add the new files into the branch and commit the changes

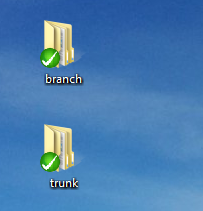


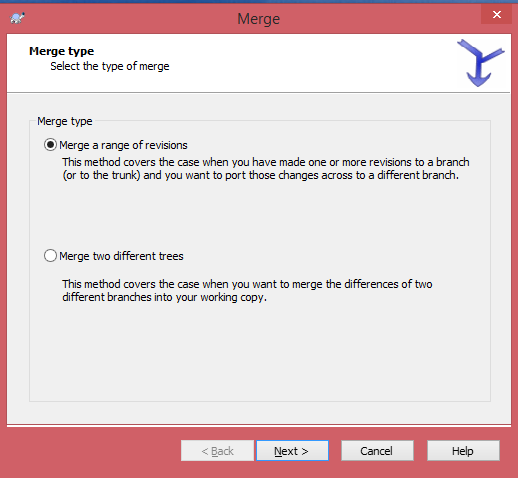


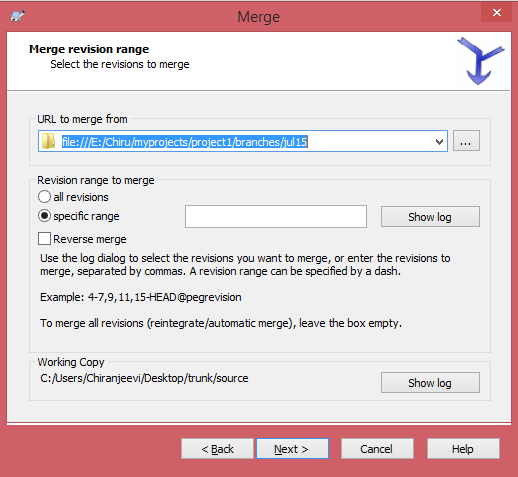
## Merging changes from branch to trunk:

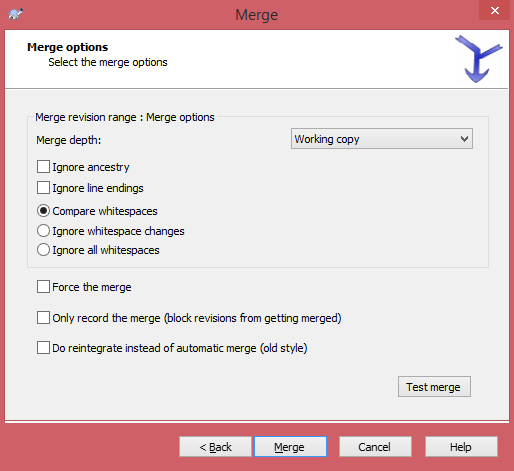
**Merging changes from branch to trunk**

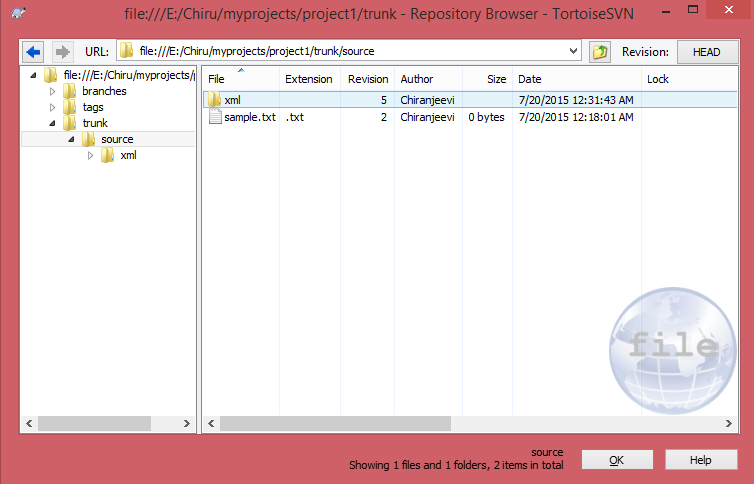
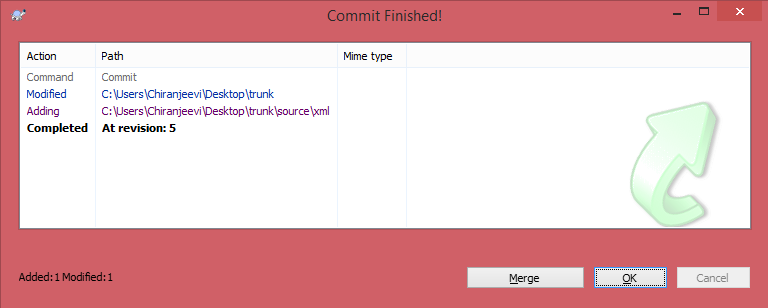
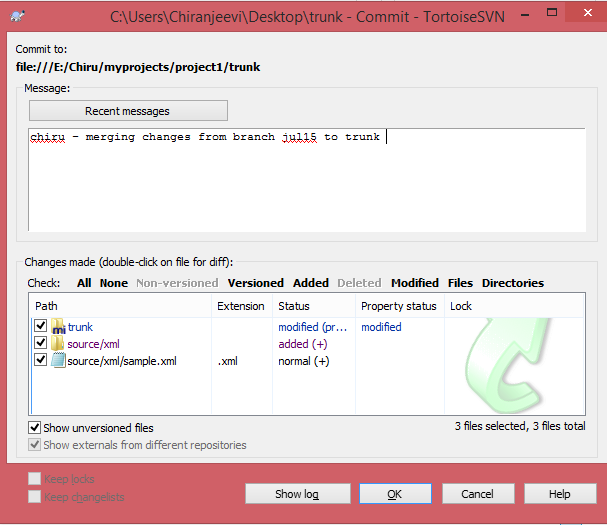
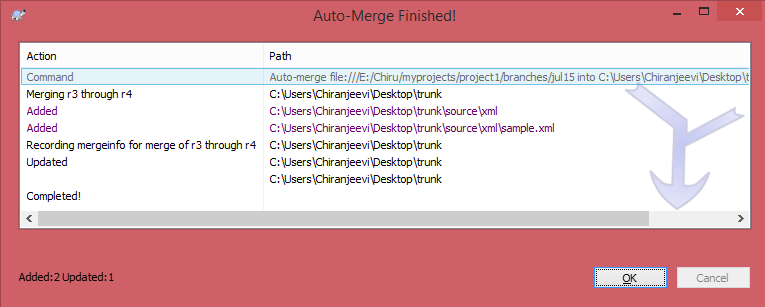
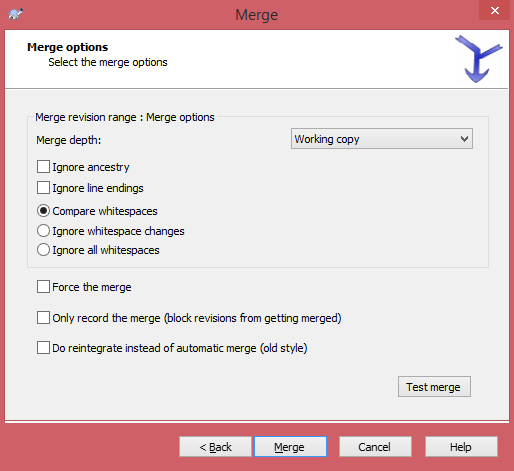
* + Checkout the trunk into the local copy
  + Right click and select merge option
  + Select the from directory of the branch to be merged
  + Click next and select working copy , merge
  + Contents will be merged to trunk working copy
  + Commit the changes in trunk working copy
  + Changes will be reflected in repository.











## Checking changes from SVN repository, SVN Log

**Merging changes from branch to trunk**

* + Right click on the SVN repository and select show log.
  + It can show you complete log on the author who committed the changes with time stamp.

