

The Wiert Corner – irregular stream of stuff

Jeroen W. Pluimers on .NET, C#, Delphi, databases, and personal interests

Some notes and links on DVCS (dynamic version control systems)

Posted by [jpluimers](#) on 2013/10/01

I love DVCS for one single reason: merges are so much easier than with traditional version control systems, and you can use it in a disconnected way for almost 100% of the time. Coming from a traditional world, interoperability is key. So here are some links that greatly helped my getting started with DVCS (a really long while ago; this post has been way overdue).

The meaning of D

Usually, the D in DVCS is regarded as distributed. But it can have multiple meanings (even at the same time), depending on how you use it.

- Distributed (the repository is available on all systems that have forked it, lessening the need centralized backups)
- Decentralized (there is no need for a central server, though a certain structure in how you fork, pull and push (or pull request) can make life a lot easier)
- Disconnected (a big plus: you can work off-line most of the time, and still perform your day to day work, but sometimes you want it to be truly distributed, and tools for that exist too)

The reason to go D

Because it leads to experimenting, which makes you a better programmer: [Version Control Makes You A Better Programmer | Cocoa Is My Girlfriend.](#)

Because you will never loose your repository: [Embarcadero Discussion Forums: XanaNews users ...](#).

Because... [Intro to Distributed Version Control \(Illustrated\) | BetterExplained.](#)

Because of the demise of traditional version control systems: SVN projects on SourceForge, CodePlex and other sites are massively migrated to DVCS.

Hosting

[Bitbucket.org](#)

can host both GIT and HG for you. (it is from Atlassian; both public and private projects; free for private projects up to 5 users). Note it is USA based, so the NSA can watch your back. Note that [HG forking from mercurial to mercurial can be a bit of a pain when you want to pull to the original](#) though the [documentation indicates this should be painless](#). Make sure you have a good README file in your rout ([you can use various extensions](#)), as it will get a prominent place at your bitbucket repository page.

[GitHub.com](#)

can host GIT for you. It can also hosts GISTs (small code snippets). Note it is USA based, to the NSA can watch your back.

[tfs.VisualStudio.com](#)

can host both TFVC and GIT for you (it is from Microsoft; payed if you have over 5 developers). Note it is USA based, so the NSA can watch your back.

www.CodePlex.com

can host TFVC, GIT and Mercurial for you (also from Microsoft, but free, though supports only public projects) . Note it is USA based, to the NSA can watch your back.

Bridging / conversion

SVN2TFS

can put migrate from SVN to TFVC in TFS. I have not used this myself, but did use:

SVNBridge

an SVN front-end to TFVC. CodePlex uses this a lot, and I used it client-side to pull a TFS 2005 repository to a user that could only do SVN. I could never get this to work on TFS 2005 server side, then never bothered as I switched to GIT/Mercurial soon after.

SVN to Mercurial: HGSVN and HGSubversion

There are three tools that can help you integrate SVN and Mercurial:

- [hgsvn 0.1.9 : Python Package Index](#). that page includes an example on how to clone from SVN, then push to HG.
- [durin42 / hgsubversion / wiki / Home — Bitbucket](#). example of usage is at [Interacting with Subversion](#). and [svn – Is there a way to use bitbucket to fork a subversion project? – Stack Overflow](#).
- [ConvertExtension – Mercurial](#). this is built in into HG, is not limited to SVN as source, but is also less sophisticated than HGSVN and HGSubversion.

GIT and Visual Studio / TFS

Getting Started with Git in Visual Studio and Team Foundation Service – Visual Studio ALM + Team Foundation Server Blog – Site Home – MSDN Blogs. GIT TF: Announcing Git Integration with TFS – Brian Harry’s blog – Site Home – MSDN Blogs. GIT TFS: similar to GIT-TF, but works cross platform. git-tfs; How do I use git-tfs and idiomatic git branching against a TFS repository? – Stack Overflow.

GIT / Mercurial

GIT and Mercurial are so so similar that there is both a Hg-Git Mercurial Plugin (free command-line tool) and Kiln Harmony (payed hosting service, but free for teams up to 2 developers).

D Workflow

Normally you work on a local for of a more global repository. Locally you branch/tag on work (then merge back into a more mainstream branch) and finally push your work (or create a pull request) to integrated it in other repositories. But if you are going to do major work that you are not sure you are willing to push back, then you are going to fork locally as well: Forking a Repository – Bitbucket – Atlassian Documentation.

Note that pull requests between BitBucket/GitHub/GoogleCode are not possible (hopefully yet). Actually it takes a bit of effort to push to multiple servers so I’ll ever that to another blog post.

There are various pages with nice graphs and diagrams on what kind of workflows you can use, for instance:

- Git Workflows and Tutorials | Atlassian.
- A handy workflow image for newbie mercurial users.
- Some people go as far as locally committing on every build they make: Mercurial workflow for personal projects (with a .net bias), which you can do in Delphi with postbuild events.
- The Tuenti Release and Development Process: Once Upon a Time | Tuenti Corporate.
- A successful Git branching model » nvie.com.
- Integration Manager Workflows & Pull Requests with Git and Mercurial.

Various VCS/DVCS links

Maintain the integrity of your .hg, .git and .svn directories

One of the drawbacks of files-based repository databases is that if they get damaged, you are hosed as downloading a fresh repository can take a long time. HG, GIT and SVN store their database in local directories named .hg, .git and .svn. You should be extra careful with them. So first and for all: make backups! Then: make sure you disable virus scanners for your .hg, .git and .svn directories.

Installation / Commandline versions

All Tortoise* downloads, except TortoiseGit (go figure!) can install commandline versions of the tools. TortoiseGIT requires the commandline tools for msysgit to function at all. If you want to do command-line only, here are some sources:

- HG for Windows builds: download the Mercurial* from the Selenic downloads.
- GIT for Windows builds: download from msysgit full+installer+official+git (they are marked 'preview' but actually are the correct files, note that for confusion, TortoiseGit 1.8.4 corresponds with msysgit 1.8.3).
- SVN for Windows builds: alagazam.net.

Encoding

The best is to use ASCII filenames and comments. Be very careful when using non-ASCII filenames, especially across operating systems. The main reason is that the Windows Console does not use the UTF-8 codepage. Some links (mainly about HG, but all version control systems have issues here):

- SVN Character Set Conversion Errors.
- HG Windows EncodingStrategy.
- WindowsUTF8Plan – Mercurial.
- unicode filenames on windows mercurial 2.5 (or future) – Stack Overflow.
- GIT for Windows Unicode Support.

Be careful with file content. For instance, TortoiseHG does not like UTF-16 files with BOM: it thinks they are binary: Converted `GroupProj.xsd` from UTF-16 to UTF-8 as TortoiseHg does not like UTF-16 files (it thinks they are binary).

Atomicity

Since the local operations are on your file system, they are not atomic. Which means that when a command is interrupted, your local state can be corrupted. This can hold for any version control (SVN, GIT, HG, TFS, etc). This leads to answers and comments like with these questions:

- [svn – Status “S” in Subversion – Stack Overflow](#).
- [SVN: What does the status “switched relative to its parent” mean? – Stack Overflow](#).

Status info

One of the really powerful features of SVN is that you can get status info in XML format for easier parsing. Without it, you get questions like these:

- [version control – Can I see the currently checked out revision number in Tortoise SVN? – Stack Overflow](#).
- [batch – Getting the current Revision number on command line via TortoiseSVN – Stack Overflow](#).

For GIT and HG you need this a lot less, but when you need it, they don't have it.

Relative URLs

SVN 1.5 added a great feature (improved with 1.6) called [relative URLs](#) (they start with ^, .., / or //) which made a lot of commands easier. Since HG and GIT contain the complete repository history, [virtually any path can be relative](#).

Ignore lists

Ignore lists are path patterns to ignore in your version control system. There is a whole bunch of [pattern examples for various languages at Github](#). Setting up Ignore lists under GIT and Mercurial is easy: add a [.gitignore](#) or [.hgignore](#) file in the root of your repository. They are version controlled, so apply to all clients. [HG ignore allows to kinds of patterns: glob and regex](#). For SVN it is a lot harder, so here are some links that can serve as a good starting point:

- [svn:ignore is not recursive!](#)
- [TortoiseSVN: Ignoring Files And Directories](#).
- [svn – How do I list subversion repository's ignore settings – Stack Overflow](#).
- [svnignore – How to create ignore list of several items in SVN? – Stack Overflow](#). (trick to import a file with ignore patterns using the -F or -file switch; you can add that file to your version control

system to make the intent more clear).

- [How do I remove a file from SVN's ignore list, using command-line? – Stack Overflow](#). (note that for `'propedit'` to work, you have to have an editor configured)
- There is no way for `"svn:ignore"+comments`. ([GIT allows them](#) and [HG allows them too](#)).
- It is easy to delete the svn:ignore list: [How do I remove a file from SVN's ignore list, using command-line? – Stack Overflow](#).

For all version control systems: Don't use a client-side global ignore list; they only apply at that particular client. Even if you have ignore lists, things can seem to be strange. A few examples:

- Be careful to add files on the commandline using wildcards (like `*`) as they will expand, therefore ignore the ignore list. See for instance [version control – How to ignore a directory with SVN? – Stack Overflow](#).
- If you edit the ignore list after files matching the patterns are already in your repository, then those files will not be ignored. See for instance [svn – TortoiseSVN ignore list is not working – Stack Overflow](#).
- The `svn propset` will overwrite the existing property. You need to work round this with a trick like in [Append directories to ignored list in SVN – Stack Overflow](#).
- again: [svn:ignore is not recursive!](#) (git and hg ignore lists are)

Renaming / Moving files

This is a bit tricky as the standard rename/move in Windows explorer does not notify the version control system about the operation. So you have to perform the rename or move in your version control tool or on the commandline.

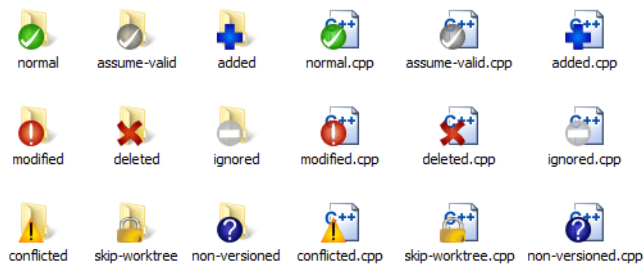
- [TortoiseHG rename/move](#). [HG rename](#) (move is just specifying a longer path including directories) Note that the default HG log will not follow a move/rename, but [you can fix that](#).
- GIT
- SVN

Cheat sheets

- [\\$ cheat svn](#).
- [Subversion Cheat Sheet by DaveChild – Cheatography.com](#).
- [Mercurial \(Hg\) Cheat Sheet by codeshane – Cheatography.com](#).
- [QuickReferenceCardsAndCheatSheets – Mercurial](#).
- [Dave's Visual Guide to TortoiseHg and Mercurial's Named Branches – Ratfactor](#).
- [Git Cheat Sheet by SamCollett – Cheatography.com](#).

- [Github – Codeschool Git Real Cheat Sheet by wdfelippe – Cheatography.com.](#)
- [Git Flow Cheat Sheet by vmalkani – Cheatography.com.](#)
- [git branch, fork, fetch, merge, rebase and clone, what are the differences? – Stack Overflow.](#)
- [Git-svn for Beginners Cheat Sheet by ezk – Cheatography.com.](#)
- [Git hg rosetta stone · sympy/sympy Wiki.](#)
- [Git equivalents of most common Mercurial commands? – Stack Overflow.](#)
- [What is the deal with the Git Index? What is the Git Index? – GitGuys – GitGuys.](#)

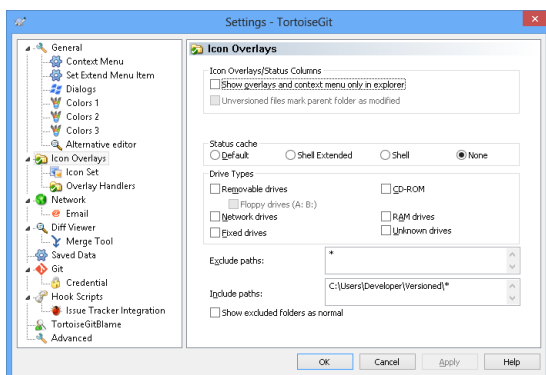
Explorer overlay icons



Version control Icon overlays in the Windows Explorer

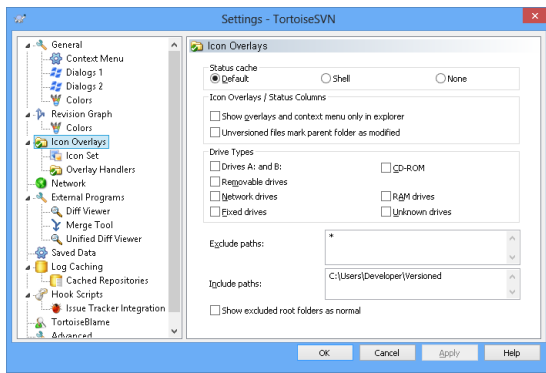
Version control Icon overlays in the Windows Explorer can have a big system impact. TortoiseHG does handle this in a very smart way, that's why TortoiseHG does not have Icon Overlay include, exclude paths. TortoiseSVN and TortoiseGIT however do, as they share large parts of the code (hence the dialogs look very similar), and this is how you configure them:

- Both:
 - Disable default showing on all kinds of media.
 - Exclude all paths.
 - Include only the paths you perform version control in.
- TortoiseGIT: Make sure that you have a * at the end of the include paths:



20130820-TortoiseGit-overlay-icons-settings (click to enlarge)

- TortoiseSVN: No need to include a * at the end of the include paths:



20130820-TortoiseSVN-overlay-icons-settings (click to enlarge)

For all three, there can be various reasons for overlay icons not to show up, varying from wrong configuration to crossing the Windows limit of 15 overlay icons (which is the reason that all three use the same overlay icon set).

External Comparison Tools example: Beyond Compare integration

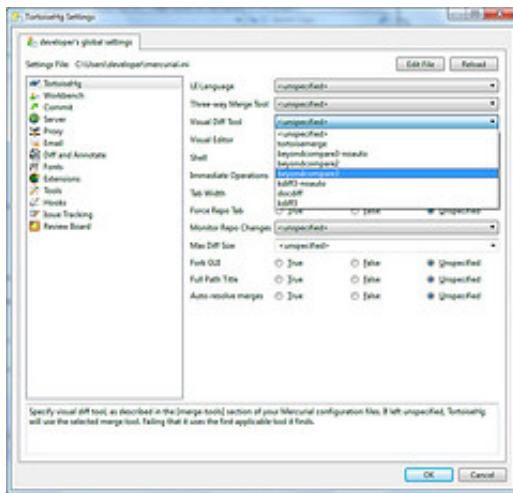
I love Beyond Compare very much as a diff/merge tool as it supports to many file and archive formats.

Their support list lists a large number of other tools they integrate with, including many version control systems.

Integration screenshots (click on each to get a larger version):

TortoiseHG

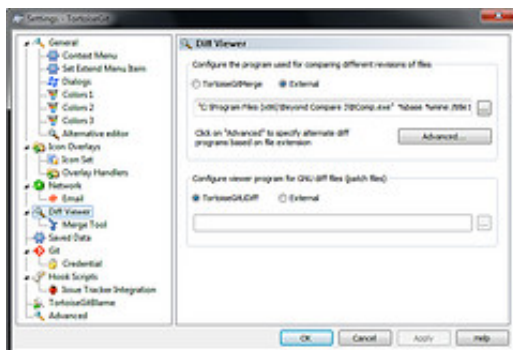
- A dropbox selection



Select “beyondcompare3” from the dropdowns in both “Three-way Merge Tool” and “Visual Diff Tool”

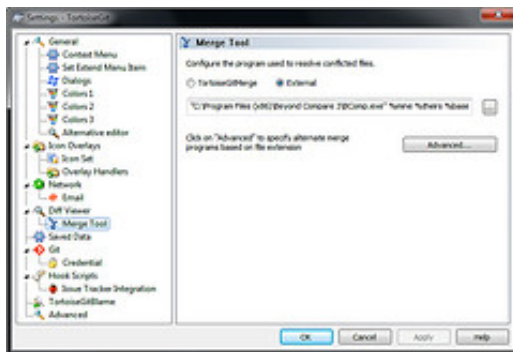
TortoiseGit

- Compare: Fill in the path plus parameters:
“C:\Program Files (x86)\Beyond Compare 3\BComp.exe” %base %mine /title1=%bname /title2=%yname /leftreadonly



Switch the radiobutton to “External” then fill in the path and parameters.

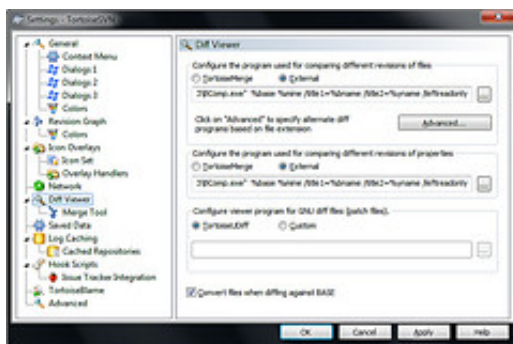
- Diff: Fill in the path plus parameters:
“C:\Program Files (x86)\Beyond Compare 3\BComp.exe” %mine %theirs %base %merged /title1=%yname /title2=%tname /title3=%bname /title4=%mname



Switch the radiobutton to “External” then fill in the path and parameters.

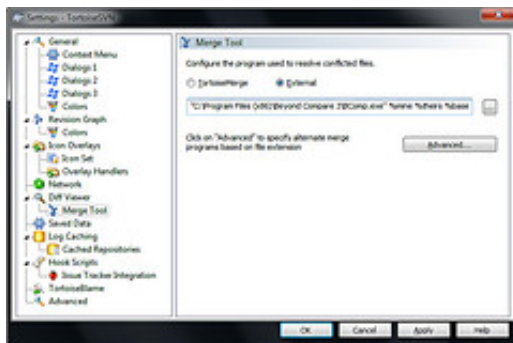
TortoiseSVN

- Compare: Fill in the path plus parameters:
`“C:\Program Files (x86)\Beyond Compare 3\BComp.exe” %base %mine /title1=%bname /title2=%ynname /leftreadonly`



Switch the radiobutton to “External” then fill in the path and parameters.

- Diff: Fill in the path plus parameters:
`“C:\Program Files (x86)\Beyond Compare 3\BComp.exe” %mine %theirs %base %merged /title1=%ynname /title2=%tname /title3=%bname /title4=%mname`



Switch the radiobutton to “External” then fill in the path and parameters.

Line ending conversion

Be careful that you do not accidentally change CRLF line endings into LF ones: Delphi still does not like those. See [You Clumsy GIT – DelphiFeeds.com](#).

Multiple upstreams

Both Mercurial/HG and GIT support multiple upstreams.

I need to put some more research in this though.

- [push – How to set Mercurial upstream – Stack Overflow](#).
- [github – Git adding multiple upstream servers and tracking their respective branches – Stack Overflow](#).
- [Pushing Changes from Development to Production? / Forums / Community / EllisLab](#).

Making the move from SVN to HG (or GIT)

- [Making the Switch to DVCS: The FishEye Teams move from Subversion to DVCS | Atlassian Blogs](#).
- [Convert from Subversion to Mercurial – Bitbucket – Atlassian Documentation](#).
- [Use subversion to access a Bitbucket repo – Bitbucket – Atlassian Documentation](#) (unsupported, works only with Mercurial).

Mercurial/HG

1. [HgInit: Subversion Re-education](#).
2. [HgInit: Ground Up Mercurial](#).
3. [HgInit: Setting up for a Team](#).
4. [HgInit: Fixing Goofs](#).
5. [HgInit: Merging](#).
6. [HgInit: Repository Architecture](#).
7. [Mercurial: The Definitive Guide](#). (hosted at [Red Bean](#), see below)
8. [Mercurial repository identification – Stack Overflow](#).
9. [Do not fork an empty repository](#): as soon as you commit changes to both, they loose their connection because an empty repository has a nullid first changeset to which the fork is bound, but the nullid gets overwritten at the first commit.

GIT

1. [Learning GIT | gitmap.](#)
2. [Git Workflows, Branching & Merging Q&A – blogs.collab.netblogs.collab.net.](#)
3. [Introduction to Git Concepts | Intertech Blog.](#)

Red Bean: Books and more about VCS and OSS

Next to [Mercurial: The Definitive Guide](#), there is quite bit of other stuff around version control and open source software at [Red Bean](#), for instance these three books are great:

- [Version Control with Subversion.](#)
- [A CVS Book.](#)
- [Producing Open Source Software.](#)

DVCS differences

- [version control – What is the Difference Between Mercurial and Git? – Stack Overflow.](#)
- [Git vs Mercurial: Why Git? | Atlassian Blogs.](#)
- [Mercurial vs Git: Why Mercurial? | Atlassian Blogs.](#) (with a great table comparing the SVN/GIT/HG command parameters)
- [Mercurial/Git Command equivalence table.](#)
- [10 things I hate about Git | Steve Bennett blogs.](#)

Syncing a fork on BitBucket HG or GitHub GIT

You use forks when you do not have write access to a public repository. Pull requests than allows you to offer your changes back to the original repository. But what if the original repository changes? You want to sync your fork, right? On BitBucket with HG, this is easy: they offer a sync button. On GitHub with GIT it is a bit more work: you have to sync through your local clone through what is usually called the `upstream` (the original repository), then push the changes back to your fork. GitHub explains this using the command-line tools at [Syncing a fork · GitHub Help](#), and [manojlds](#) for TortoiseGit at [github – Can I update a forked project, on git, to the original/master copy? – Stack Overflow.](#)

TFS and Mercurial, SVN, and GIT

Real-World use of Mercurial with a Team Foundation Server? – Stack Overflow. This explains how to use HG, and how to push/pull from the TFVC. The comments and answers also talk about SVN, and other version control systems.

TFS migration itself

- TFS Integration Platform – Home.
- TFS Integration Tools – How can I validate that a migration or sync is working? Q&A-34 – Willy's Reflections – Site Home – MSDN Blogs.

–jeroen

This entry was posted on 2013/10/01 at 12:00 and is filed under CodePlex, Development, DVCS - Distributed Version Control, git, Mercurial/Hg, Software Development, Source Code Management, Subversion/SVN, TFS (Team Foundation System), Visual Studio 11, Visual Studio 2005, Visual Studio 2008, Visual Studio 2010, Visual Studio and tools. You can follow any responses to this entry through the RSS 2.0 feed. You can leave a response, or trackback from your own site.

« Sciences jokes...

Blog at WordPress.com. | Customized Andreas09 Theme.