



+



Arbed

 SmartSVN

X O J OTM

Xojo team development with Subversion, SmartSVN and Arbed (<http://www.xplatdev.com/xdcdenhaag.pdf>)

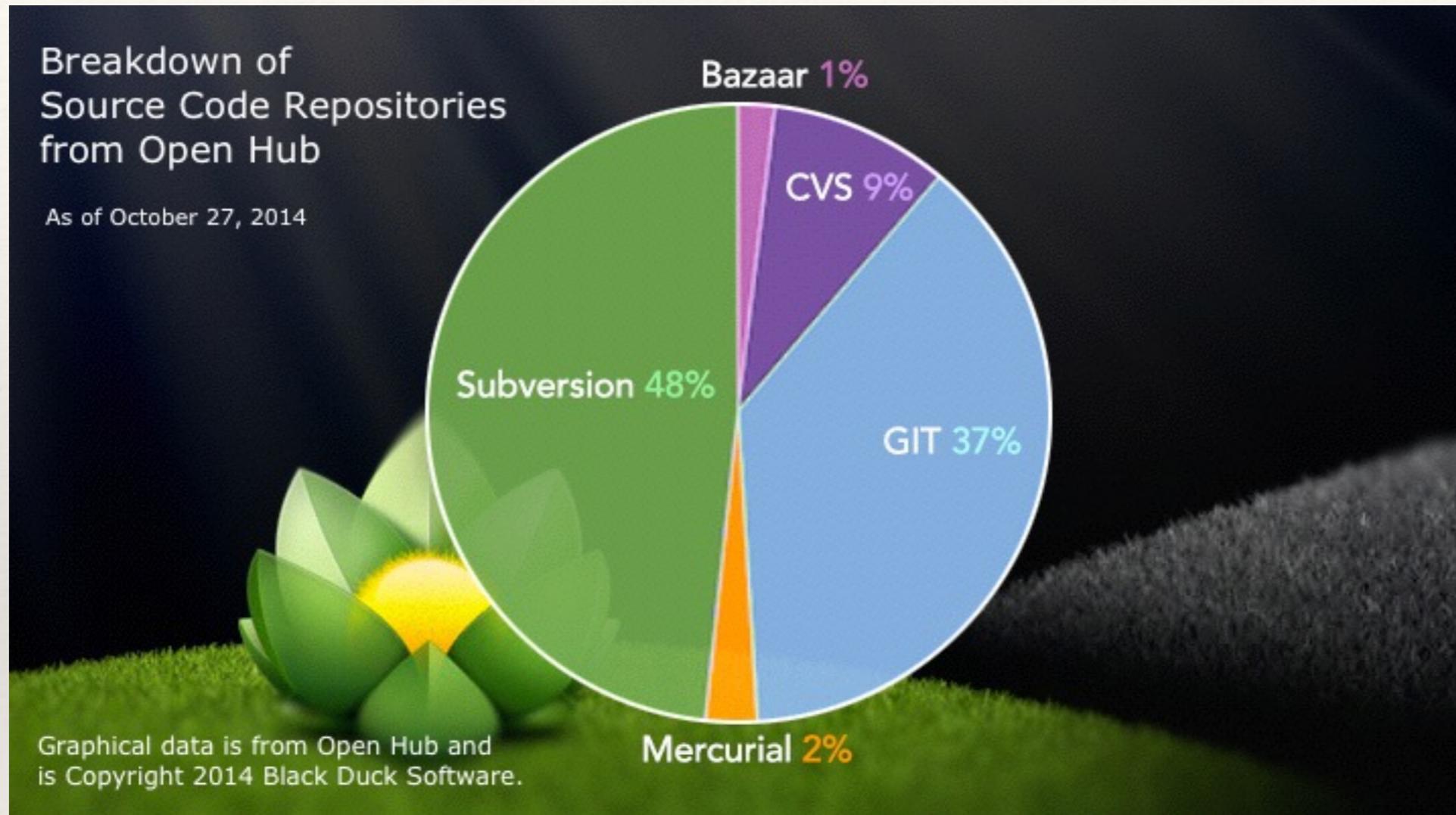
Xojo Developer Conference

Den Haag (Scheveningen)

Why version control?

- ❖ Working together in a team (central repository)
- ❖ Keeping track of changes to your code (diffs)
- ❖ Adding comments to the changes (commit)
- ❖ Ability to undo changes if you made a mistake (revert and rollback)
- ❖ Create several versions of your software (branches)
- ❖ Helps with cross-platform development (PC / Mac / Linux)
- ❖ Having a backup of your project on the server

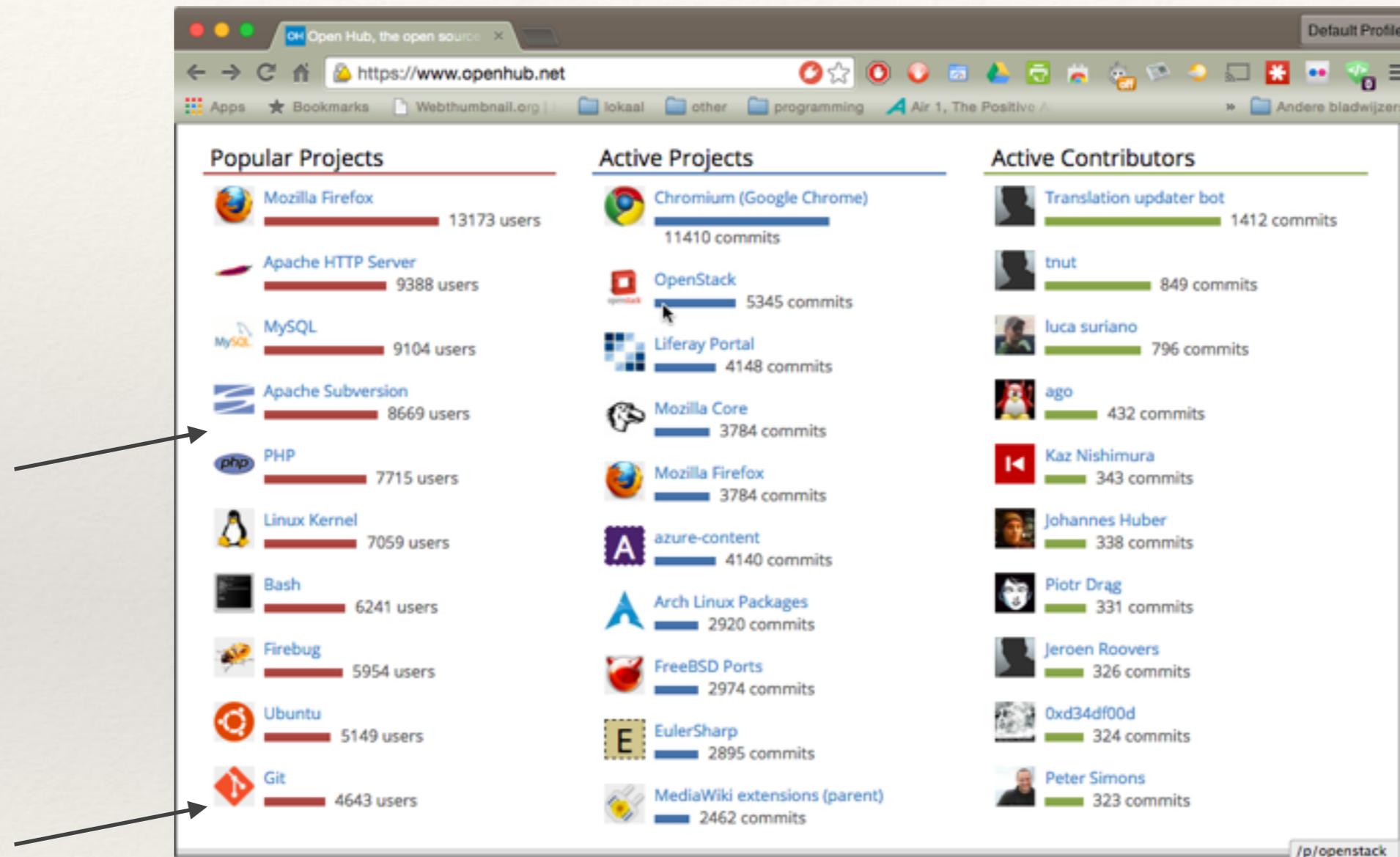
Which version control do we choose?



The ones most used are Subversion and GIT
(according to Duck Software Open Hub, formerly ohloh.net)

The Black Duck Open Hub (formerly Ohloh.net) is an online community and public directory of free and open source software (FOSS), offering analytics and search services for discovering, evaluating, tracking, and comparing open source code and projects. Open Hub Code Search is free code search engine indexing over 21,000,000,000 lines of open source code from projects on the Black Duck Open Hub.

Which version control do we choose?



Subversion and GIT are both popular and under active development.

Which version control do we choose?

GIT is available for:
Mac OS X
Windows
Linux
Solaris

git --distributed-even-if-your-workflow-isnt

Search entire site...

Downloads

Mac OS X Windows
Linux Solaris

Older releases are available and the Git source repository is on GitHub.

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos →](#)

Git via Git

If you already have Git installed, you can get the latest development version via Git itself:

```
git clone https://github.com/git/git
```

You can also always browse the current contents of the git repository using the [web interface](#).

<https://www.kernel.org/pub/software/scm/git/>

Which version control do we choose?

Subversion is available for:

Mac OS X

Windows

Linux

Solaris

AIX

HP-UX

FreeBSD

NetBSD

OpenBSD

Apache Subversion Binary Packages

Find packages for your operating system:

[AIX](#) | [Centos Linux](#) | [Debian Linux](#) | [Fedora Linux](#) | [FreeBSD](#) | [HP-UX](#) | [NetBSD](#) | [OpenBSD](#) | [Mac OS X](#) | [Red Hat Linux](#) | [Solaris](#) | [SUSE Linux](#) | [Ubuntu Linux](#) | [Windows](#)

The Apache Subversion project **does not officially endorse or maintain** any binary packages of the Subversion software. However, volunteers have created binary packages for different distributions and platforms, and as a convenience, we maintain a list of links to them here. If there are any problems with or questions about the different binary packages please send email to the [Subversion users mailing list](#).

Note that binary packages usually come out about a week after the corresponding [source release](#). Please don't post to the mailing lists asking when a binary package for a given platform will be ready. The packagers already know when new source releases come out, and work as fast as they can to make binaries available.

Binaries are typically built from the latest stable release.

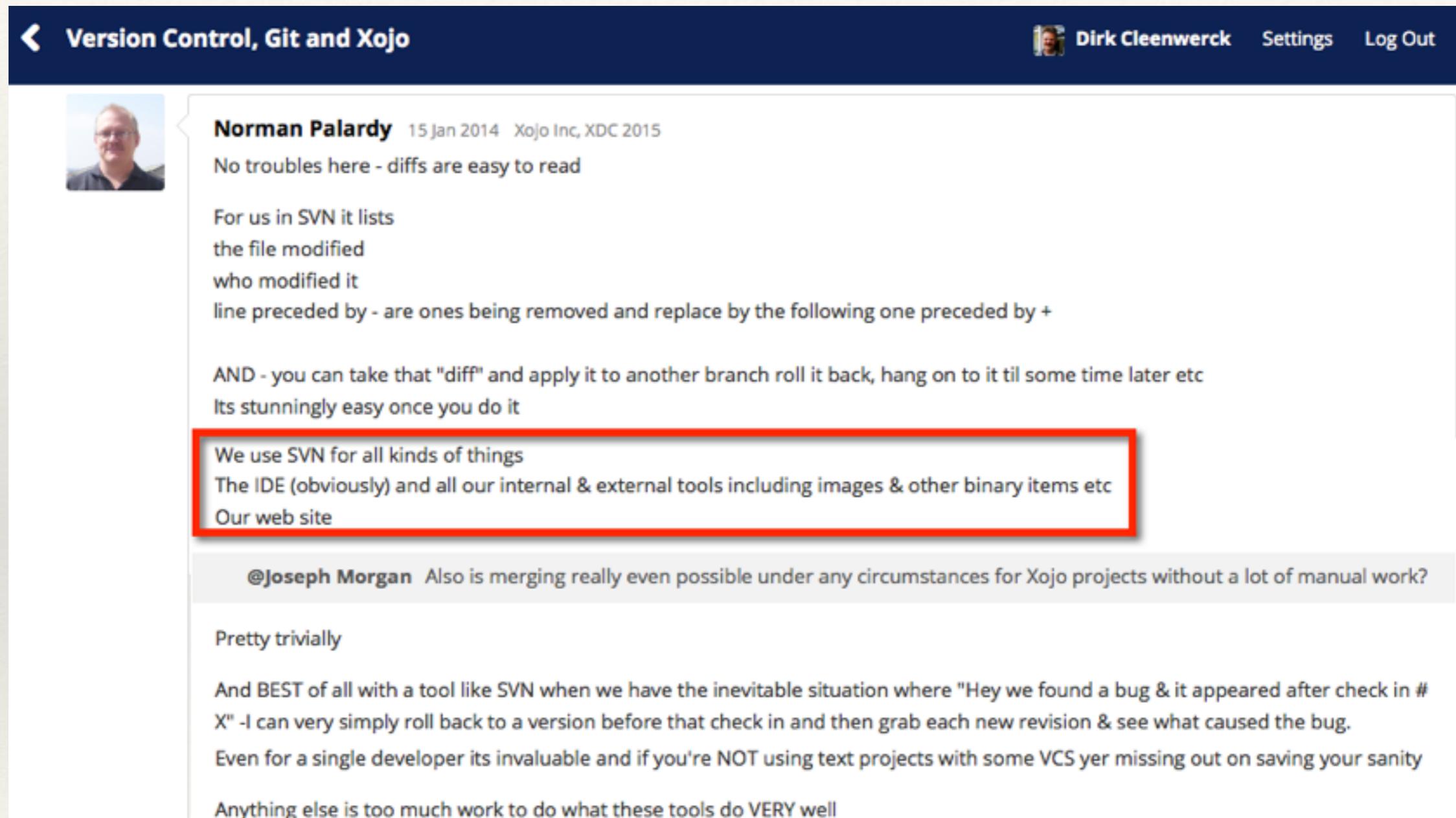
Note also that this list does not include distributions of larger collections of software of which Subversion is but one piece. Several vendors offer such things, but we concern ourselves primarily with Subversion itself. As such, the listing here is limited to those packages which may be re...

Apache Subversion Binary Packages

Which version control do we choose?

- ❖ Both Subversion and Git are used by developers in combination with Xojo
(see <https://forum.xojo.com/8379-version-control-git-and-xojo>)
- ❖ Both can be used from the command line
(<http://svnbook.red-bean.com/en/1.7/svn.ref.svn.html>)
(<http://git-scm.com/book/en/v2/Getting-Started-The-Command-Line>)
- ❖ Both have clients available to make life easier
(https://en.wikipedia.org/wiki/Comparison_of_Subversion_clients)
(<https://git-scm.com/downloads/guis>)

Why did we choose Subversion?



Version Control, Git and Xojo Dirk Cleenwerck Settings Log Out

 **Norman Palardy** 15 Jan 2014 Xojo Inc, XDC 2015

No troubles here - diffs are easy to read

For us in SVN it lists
the file modified
who modified it
line preceded by - are ones being removed and replace by the following one preceded by +

AND - you can take that "diff" and apply it to another branch roll it back, hang on to it til some time later etc
Its stunningly easy once you do it

**We use SVN for all kinds of things
The IDE (obviously) and all our internal & external tools including images & other binary items etc
Our web site**

@Joseph Morgan Also is merging really even possible under any circumstances for Xojo projects without a lot of manual work?

Pretty trivially

And BEST of all with a tool like SVN when we have the inevitable situation where "Hey we found a bug & it appeared after check in # X" -I can very simply roll back to a version before that check in and then grab each new revision & see what caused the bug.
Even for a single developer its invaluable and if you're NOT using text projects with some VCS yer missing out on saving your sanity

Anything else is too much work to do what these tools do VERY well

The Xojo dev team uses SVN.

Why did we choose Subversion?

- ❖ Easy to install
(<https://help.ubuntu.com/community/Subversion#Installation>)
- ❖ Easy to find documentation
(<http://svnbook.red-bean.com/>)
- ❖ Easy to use client available for Mac/Windows
(<http://www.smartsvn.com/>)
- ❖ Part of command line developer tools for Xcode on Mac
(<http://osxdaily.com/2014/02/12/install-command-line-tools-mac-os-x/>)
- ❖ It's possible to require locks on files for the developers
(http://tortoisesvn.net/docs/nightly/TortoiseSVN_en/tsvn-dug-locking.html)

Installing Subversion on Ubuntu 14.04 LTS

If you will access subversion over http, make sure your Apache is installed first (won't do that here)

```
sudo tasksel install lamp-server
```

(<http://howto.blbosti.com/2010/02/4-easiest-ways-to-install-lamp-server-on-ubuntu/>)

install subversion

```
sudo apt-get install subversion
```

or if you will access subversion over http

```
sudo apt-get install subversion libapache2-svn
```

make a directory for svn and one for the repositories

```
sudo mkdir /usr/local/svn
```

```
sudo mkdir /usr/local/svn/repos
```

Installing Subversion on Ubuntu 14.04 LTS

Make a group for you svn users

```
sudo groupadd svn
```

Change group ownership of the repositories directory to the new group

```
sudo chgrp svn /usr/local/svn/repos
```

Give members of the svn group write access to the repositories directory

```
sudo chmod g+w /usr/local/svn/repos
```

Set the group-ID of the repositories directory so that new files created here will be owned by the group

```
sudo chmod g+s /usr/local/svn/repos
```

Installing Subversion on Ubuntu 14.04 LTS

Add yourself to the svn group (add other users as necessary)

```
sudo usermod -a -G svn dirk
```

Log out and back in to check you belong to the group

```
groups
```

(you should see the svn group among the groups of which you are a member)

Create a repository for your project (change umask so users of the svn group will have write access)

```
svnadmin create /usr/local/svn/repos/myproject  
sudo chgrp svn /usr/local/svn/repos/myproject  
sudo chmod g+w /usr/local/svn/repos/myproject  
sudo chmod g+s /usr/local/svn/repos/myproject
```

Installing Subversion on Ubuntu 14.04 LTS

configure subversion to allow access through the custom protocol (svn://)

We do this by editing `svnserve.conf`. Each repository has its own settings file.

```
nano /usr/local/svn/repos/myproject/conf/svnserve.conf
```

Put the following rules in the `svnserve.conf` file:

```
anon-access = none
```

```
auth-access = write
```

```
password-db = passwd
```

After changing the `.conf` file you can add the user list to the `passwd` file that can be found in the same directory.

```
nano /usr/local/svn/repos/myproject/conf/passwd
```

Add users using the following syntax.

```
username = password
```

Installing Subversion on Ubuntu 14.04 LTS

Make sure the svn server runs on startup

Download the svnserve script from <http://odyniec.net/articles/ubuntu-svnserve-server/>

Place the script in `/etc/init.d`

Make the script executable

```
sudo chmod +x /etc/init.d/svnserve
```

If you chose anything other than `/usr/local/svn/repos` for the repository directory, make sure to change the path in the init script

run `update-rc.d` to install the script

```
sudo update-rc.d svnserve defaults
```

That's it. `svnserve` will be started automatically when your system boots up.

To start it manually, run

```
sudo /etc/init.d/svnserve start
```

Install SmartSVN

Free foundation edition available, Professional version (max. 99\$/license/yr, volume discounts)

Download and install SmartSVN

(available for Windows 7+, Mac OS 10.7.3+ and Linux)

(<http://www.smartsvn.com/download>)

After installing, prepare a folder with your Xojo project

- ❖ Make a new folder
- ❖ Save your project to this folder as a .xojo_project (VCP format)

Import project into repository

- ❖ Start SmartSVN
- ❖ Select 'Import project into repository'
- ❖ Select the folder with your project
- ❖ Select your repository
(for instance svn://192.168.128.94/myproject)
- ❖ Enter your svn username and password

Import project into repository

- ❖ Make a folder in the repository for your project
Check 'Create default project structure for trunk, branches and tags'
- ❖ Select the trunk
- ❖ Add your new project in group <sorted group>
- ❖ Import

Import project into repository

- ❖ Commit your project
- ❖ Select Depth 'Fully recursive'
- ❖ Type 'initial commit' as your Commit Message
- ❖ Select all files in the toplevel (show subdirectories and unchanged)
- ❖ Under 'Locks' select 'Change Needs Lock'
- ❖ Commit the changes with 'needs lock' as your commit message

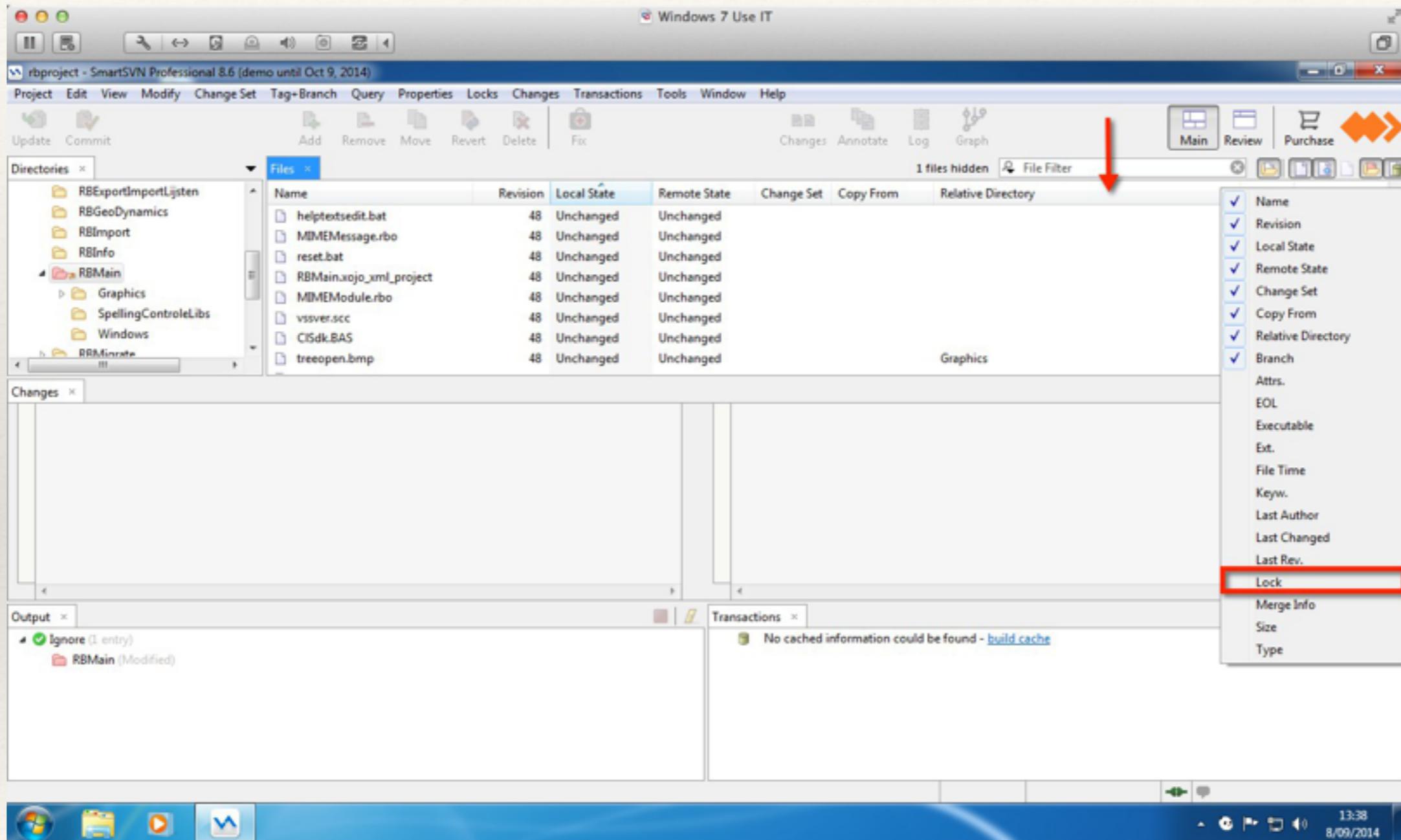
You are now ready to start using SVN

Start using SVN

If you are on a new PC, first check out the repository

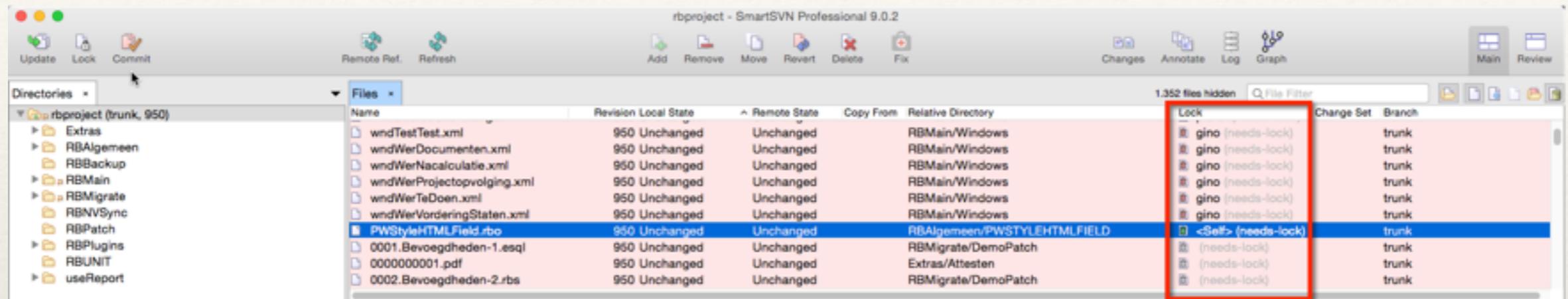
- ❖ Start SmartSVN
- ❖ Check out project from repository (repositories are already set up) (for instance svn://192.168.128.94/myproject)
- ❖ Enter your username and password
- ❖ Select the trunk of the project
- ❖ Select a local directory for your project
- ❖ Checkout Depth: fully recursive
- ❖ Check out a working copy

SmartSVN functionality

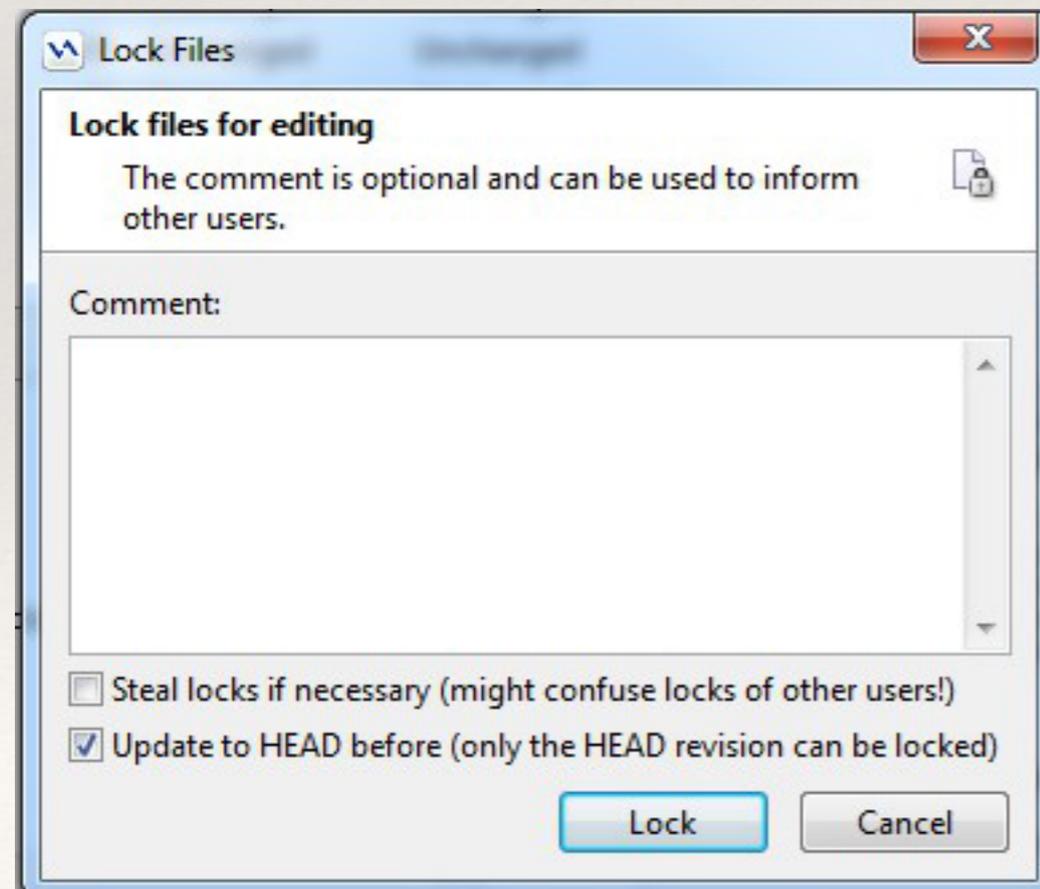


If you will use locks, make sure you can see the column in SmartSVN

SmartSVN functionality



In the Lock column you can see if a file is locked and who locked it



As long as you haven't locked a file, the file will be read-only. If you want to edit a file, you therefore need to lock it.

You do this by selecting the files and then choosing 'Locks, Lock' in the right-click menu.

SmartSVN functionality

Don't steal locks!

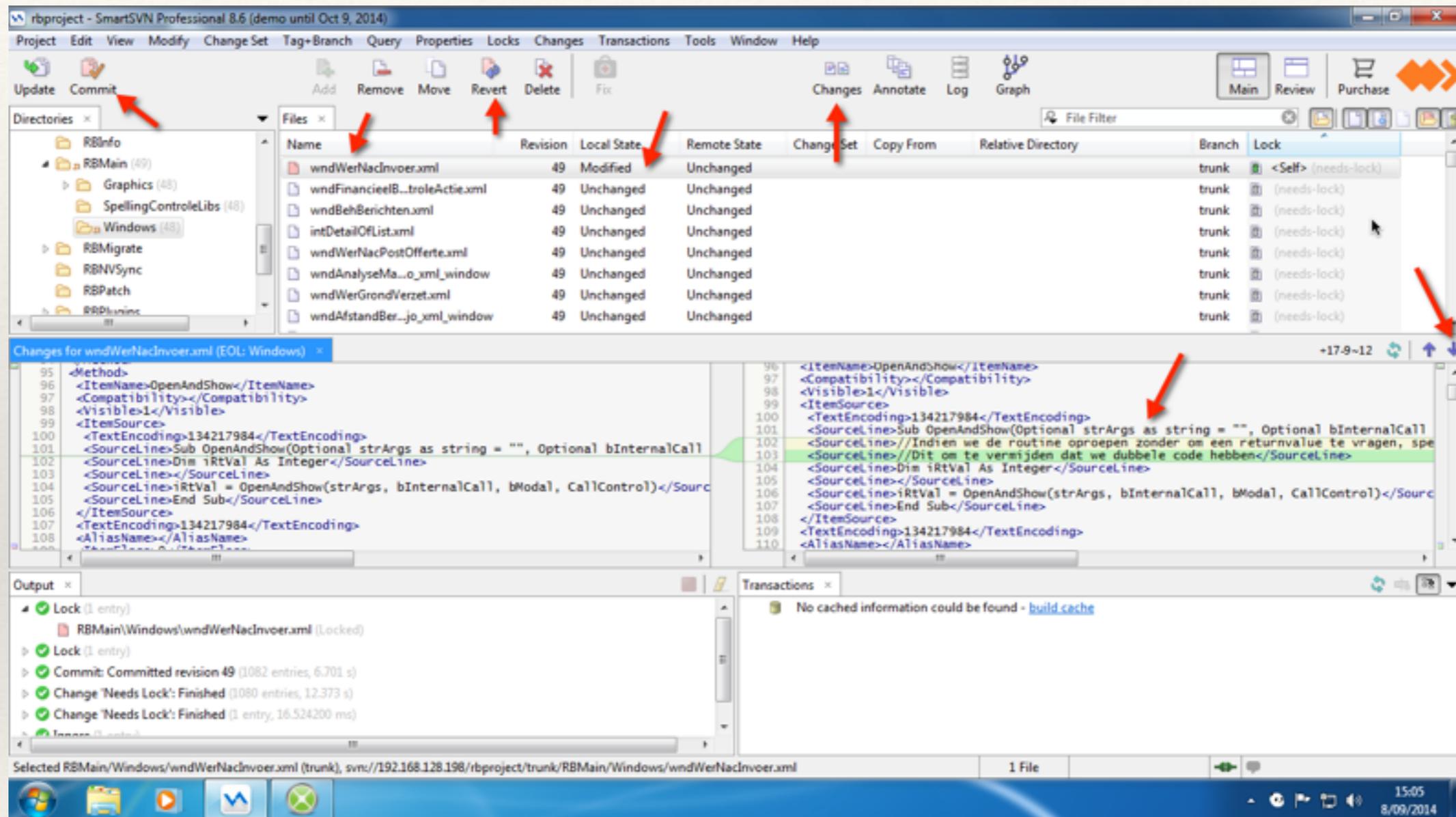


Avoid Breaking and Stealing Locks

If you break or steal someone else's lock without telling them, you could potentially cause loss of work. If you are working with unmergeable file types and you steal someone else's lock, once you release the lock they are free to check in their changes and overwrite yours. Subversion doesn't lose data, but you have lost the team-working protection that locking gave you.

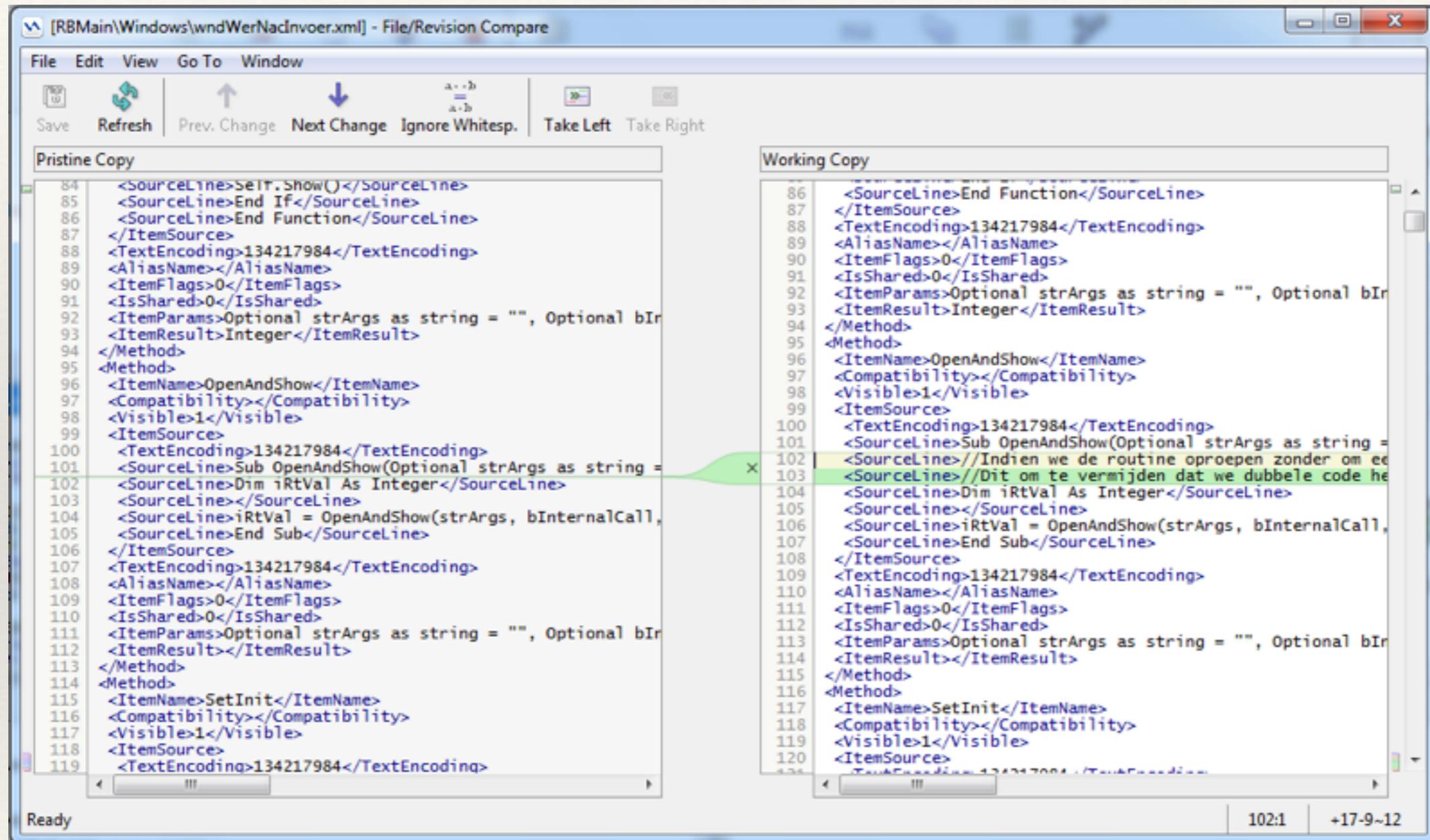
see http://tortoisesvn.net/docs/nightly/TortoiseSVN_en/tsvn-dug-locking.html

SmartSVN functionality



When you make changes, you will see that the local state of the file changes to Modified. You can see the changes made to the file in the 'Changes' section

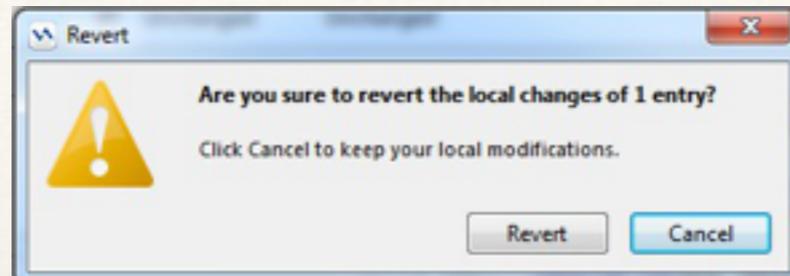
SmartSVN functionality



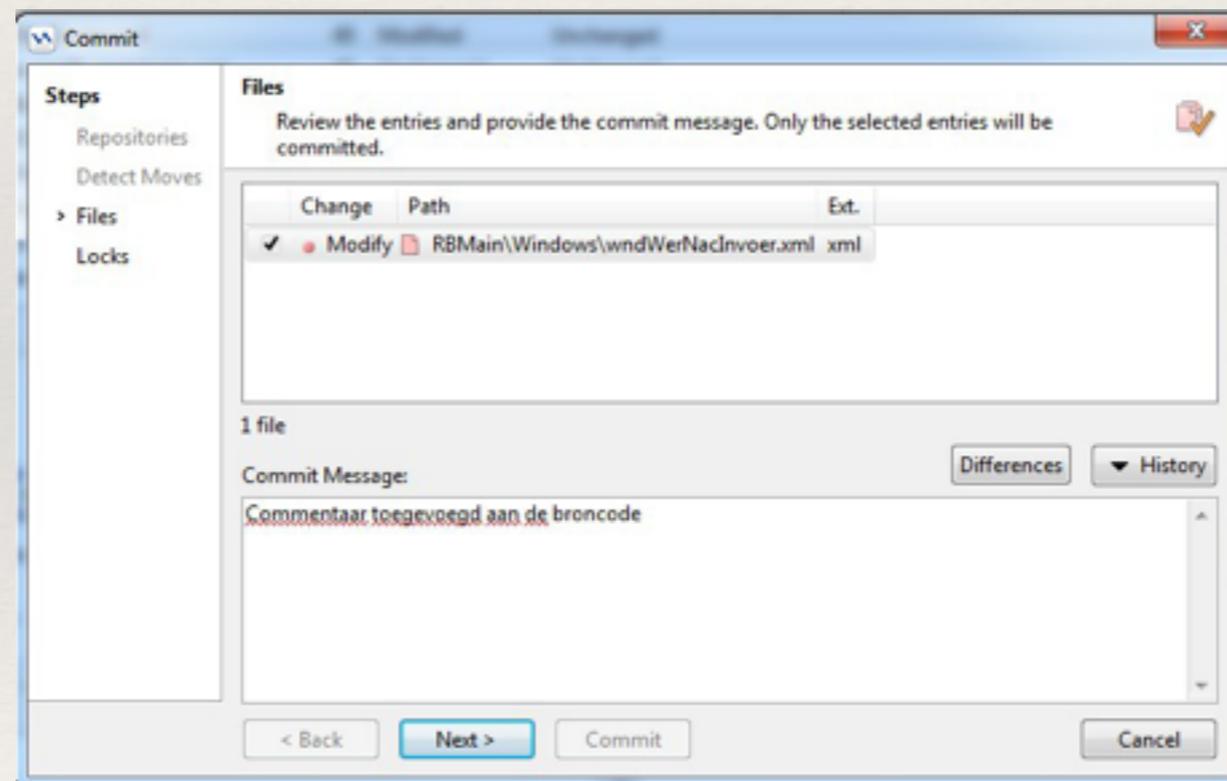
The screenshot shows the 'Revision Compare' window in SmartSVN. The window title is '[RBMain\Windows\wndWerNacInvoer.xml] - File/Revision Compare'. The menu bar includes 'File', 'Edit', 'View', 'Go To', and 'Window'. The toolbar contains icons for 'Save', 'Refresh', 'Prev. Change', 'Next Change', 'Ignore Whitesp.', 'Take Left', and 'Take Right'. The window is split into two panes: 'Pristine Copy' on the left and 'Working Copy' on the right. Both panes display XML source code. The 'Pristine Copy' pane shows lines 84 through 119. The 'Working Copy' pane shows lines 86 through 120. A green callout bubble with an 'x' icon points to line 102 in the 'Working Copy' pane, which contains the text: '<SourceLine>//Indien we de routine oproepen zonder om ee'. The status bar at the bottom indicates 'Ready' on the left, and '102:1' and '+17-9-12' on the right.

If you want to have a better view of the changes, you can open the revision screen by clicking the 'Changes' button in the main menu.

SmartSVN functionality



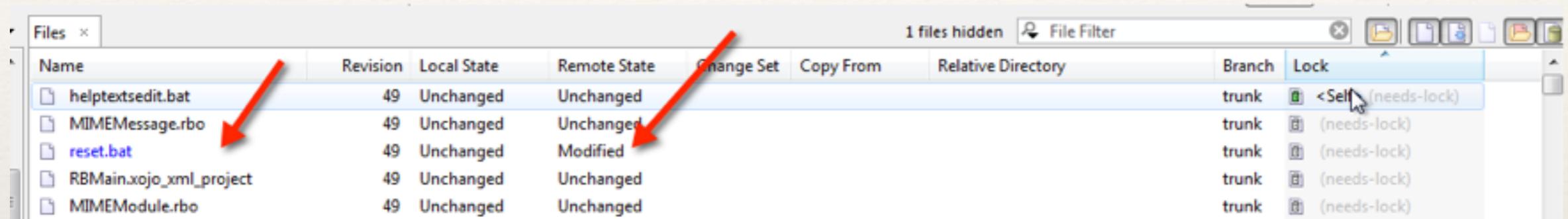
If you made a mistake, you can revert to the code that is in the repository. You need to confirm that you want to undo your changes. This will only roll back your local copy.



When you are happy with your code, you need to commit the code to the repository. Don't forget to add a commit message that explains what was changed and why. After committing you will be asked if you want to unlock your files.

SmartSVN functionality

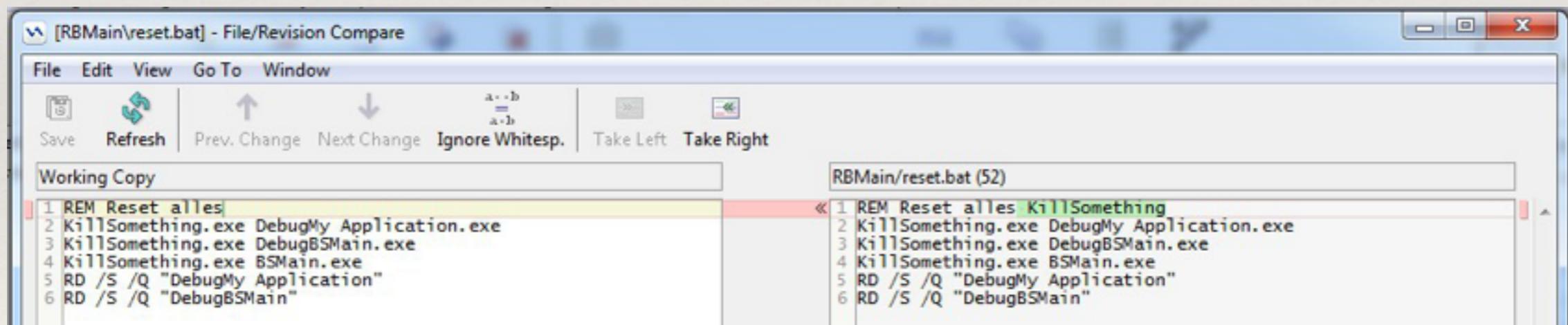
To see if something changed on the server, you choose 'Query, Refresh Remote State' in the menu.



The screenshot shows a file list in SmartSVN. The 'reset.bat' file is highlighted in blue, and its 'Remote State' is 'Modified'. Red arrows point to the 'reset.bat' file name and the 'Modified' text. The 'Lock' column shows a lock icon and '<Self (needs-lock)'. The 'Branch' column shows 'trunk' for all files.

Name	Revision	Local State	Remote State	Change Set	Copy From	Relative Directory	Branch	Lock
helptextsedit.bat	49	Unchanged	Unchanged				trunk	<Self (needs-lock)
MIMEMessage.rbo	49	Unchanged	Unchanged				trunk	(needs-lock)
reset.bat	49	Unchanged	Modified				trunk	(needs-lock)
RBMain.xojo_xml_project	49	Unchanged	Unchanged				trunk	(needs-lock)
MIMEModule.rbo	49	Unchanged	Unchanged				trunk	(needs-lock)

If there are any changes, you will see 'Modified' in the Remote State column. You can then click Update and choose 'Update to HEAD' to get the latest version of the project's code.



The screenshot shows the 'File/Revision Compare' window for 'reset.bat'. The 'Working Copy' is on the left and the 'RBMMain/reset.bat (52)' is on the right. The 'Working Copy' has 6 lines of code, and the 'RBMMain/reset.bat (52)' has 6 lines of code. The first line of the 'RBMMain/reset.bat (52)' is highlighted in green and has a double arrow pointing to the left, indicating a change from the working copy.

```
Working Copy:
1 REM Reset alles
2 KillSomething.exe DebugMy Application.exe
3 KillSomething.exe DebugBSMain.exe
4 KillSomething.exe BSMain.exe
5 RD /S /Q "DebugMy Application"
6 RD /S /Q "DebugBSMain"

RBMMain/reset.bat (52):
1 REM Reset alles KillSomething
2 KillSomething.exe DebugMy Application.exe
3 KillSomething.exe DebugBSMain.exe
4 KillSomething.exe BSMain.exe
5 RD /S /Q "DebugMy Application"
6 RD /S /Q "DebugBSMain"
```

If you just want to see the changes on the server instead of updating, you can use 'Query, Compare with HEAD'. Then you can see what changed to the code on the server side and compare it with your local code in the File/Revision screen.

SmartSVN functionality

Revision	Author	Commit Message	Time	File Count	Path
952	gino	Grayed Text wordt nu gereset in geresized	16:43	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
951	gino	Vertalingen Bouwsoft Favorieten	16:03	4	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
950	gino	Vertalingen Bouwsoft Favorieten	14:29	4	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
948	gino	Vertalingen favorieten	11:53	4	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
912	gino	EmptyIfNull in AlgSearch	08/09/2015 14:08	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
906	gino	Scrollen in Treelist	03/09/2015 18:56	52	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
904	gino	MakeThisControlVisible works fine for combos now + horizontal scrollbar <-> 0 while clicking an editfield do...	03/09/2015 18:00	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
902	gino	typo in ThreadEnd	03/09/2015 16:24	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
899	gino	typo in inladen mrfav	03/09/2015 13:13	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
890	gino	Minimumbreedte verandert naar 10	02/09/2015 19:19	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
889	gino	Code check minimumbreedte in AddTitleItem iets duidelijker gemaakt (werking zelfde als vroeger)	02/09/2015 18:57	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
888	gino	Regeloverloop RTF, refreshissue solved	02/09/2015 17:46	2	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
881	gino	+ in zoekvelden wordt nu als tekst behandeld indien autoand aanstaat (bv algsearch of combo)	01/09/2015 17:49	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
880	gino	Extra velden met komma in de SQL tussen quotes worden nu ook correct verwerkt	01/09/2015 17:23	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
878	gino	Code was weg	01/09/2015 16:23	1	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
877	gino	Pijltje naar beneden in een record terwijl nog records voorhanden zijn buiten dit scherm werkt terug. Expor...	01/09/2015 14:52	5	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)
867	bert	Kostprijsberekening: verliesfactor ook gebufferd	27/08/2015 10:01	3	RBAlgemeen/Controls/useSubMultiRow.xml (trunk)

If you made a mistake and you already checked the file into the repository, you can go to the history of the file by clicking 'Log'. You will then get a list of all the revisions of the selected file(s). If you want to go back to an earlier copy of your code, you select 'Rollback' in this screen

Potential issues

Main project file is read only

Solution: Save your project under another name. Then put this file on the SVN ignore list.

Outside files get overwritten (feedback://showreport?report_id=11725)

To completely understand the danger as the implementation is now:

- 1) Export a window to an xml file
- 2) Add it to source control

- 3) Person A and B get the latest changes from that file (they have the same file now) and open their IDE

- 4) Person A checks the file out
- 5) Person A makes some changes and saves
- 6) Person A checks the file in (changes are now in Sourcecontrol and in the version of Person A)

- 7) Person B checks the file out (now the file has the new code, but the IDE still has the old code)
- 8) Person B makes some changes and saves (the new code is overwritten with the old code and the changes of person B)
- 9) Person B checks the file in (the changes of Person A are completely ignored and gone from Sourcecontrol now)

This all goes without any warning at all and is dangerous when you work in team.

Solution: close the IDE, checkout the file that was changed on the server side, open the IDE again

Potential issues

No externals in VCP ([feedback://showreport?report_id=3624](https://showreport?report_id=3624))

When sharing code between projects, we normally make an external that we share between the different projects. You cannot save an external in VCP format though.

Solution:

Save externals in .xojo_xml format and share this between the projects.

Additional problem:

A diff between xml files is not as nice as a diff between VCP format.

Solution to this:

Use Arbed to show the diffs.

You can add Arbed as a tool for 'File Compare' under SmartSVN Preferences, Tools.

(<http://www.tempel.org/Arbed>) Free version available, basic license 29\$, complete 89\$

Potential issues

No externals in VCP (feedback://showreport?report_id=3624)

When sharing code between projects, we normally make an external that we share between the different projects. You cannot save an external in VCP format though.

Alternative solution: (not yet tested by us)

Use advanced externals in SVN. For additional information read:

the feedback ticket

the discussion on the forum

<https://forum.xojo.com/12202-why-use-make-external/0>

‘sharing code between projects’ on Bob Keeney’s website

<http://www.bkeeneybriefs.com/2012/06/sharing-code-between-projects/>

Potential issues

Externals or external files in different source folder

Solution: Use Arbed to assemble the external files.

You can use Arbed's convert function for this.

Files that need to be ignored (.xojo_uistate, build folder,...)

Solution: Use the right click menu in SmartSVN and select 'ignore'.

You can ignore a file explicitly, ignore by pattern and ignore folders.

(<http://blogs.wandisco.com/2013/04/22/ignoring-files-with-smartsvn-2/>)

Commit too many files in one commit

Committing too many files in one go makes it confusing to keep track of changes. Make sure you commit often and enter commit messages that make it clear what was changed and why.

Potential issues

Don't forget to lock

Your files are read-only as long as you don't lock a file. After locking, make sure you click another window or class in Xojo and then click back so that the IDE knows you can now write to this file. If you forget to do this, you won't be able to edit the code in Xojo, since the IDE will think the file is readonly still.

Watch out for breakpoints

Remembering breakpoints

Do you use breakpoints often, would you like to keep them when you save and re-open your project? Note that only the XML and RBP project file formats preserve breakpoints, the VCP format doesn't. (<http://www.realsoftwareblog.com/2012/05/ide-tips-and-tricks.html>)

The other developers probably won't want to have your breakpoint stop their debug run. Therefore if you use our way to save to Subversion, don't forget to clear your breakpoints before checking in.

Xojo, Subversion, SmartSVN, Arbed

Enjoy Team development.

Dirk Cleenwerck

Use IT Group NV