



Source control: a tool for every developer

Deblauwe Gino & Dirk Cleenwerck
Use It Group

🤔 Questions after the last slide please

Some say...

- I have a daily backup, so I'm protected
- I work alone
- It's too expensive
- Source control is too complex, I don't understand
- It's too much work



But did you ...

- Ever come in a situation where you have to find a bug that was probably introduced 3 versions ago?
 - ★ What was changed back then?
 - ★ By who?
 - ★ And why?



And did you ...

- Ever come in a situation where some developer claims that he did not cause that bug?
- Ever come in a situation where someone asks you what in your software changed in the last year to better their lives. Also known as "what do I pay you for?"
- Ever come in a situation that you really need to release a critical bugfix, but you're in the middle of creating new functionality?
- ...



And did you know ...



- It's cheap:
 - ★ If installed on your server, systems like **git** and **subversion** are free
 - ★ Free for teams up to 3 developers / 7\$ monthly for more on github <https://github.com/pricing#feature-comparison>
 - ★ Free on gitlab without support and basic functionality <https://about.gitlab.com/pricing/>
 - ★ Free on bitbucket for up to 5 developers / 2\$ monthly for each developer for more <https://bitbucket.org/product/pricing>
 - ★ Or if you want more choice: <https://www.slant.co/topics/153/~best-hosted-version-control-services>

But wait, there's more ...



- You can also use source control for analysis and documentation so you can see what was in customer documentation back then at that release. Or retrieve specs from that point in time.
- Setting up source control can be done within minutes

So to end the sales pitch: Why use source control?



- It makes working together in a team a lot easier
- You can keep track of code changes and compare versions if you need to
- Every commit you are advised to add a comment so in the future you have a log what was done
- You can undo changes that proved to be a bad idea
- You can create different versions (branches) of your software (at least: 'development' and 'production')
- There is at least 1 up to date backup of your entire project

Which version control do we choose?



There are 2 major systems for now:

- Client-Server systems like
 - + subversion
 - + cvs
 - + visual sourcesafe
- Distributed systems like
 - + git
 - + mercurial

Which version control do we choose?

- Client-Server systems:
 - + You can lock parts of the source, no one else works on it until you're done.
- Distributed systems:
 - + Faster to commit since you commit locally
 - + Available offline
 - + No single point of failure

Which version control do we choose?

- Both subversion and git are actively developed

General		
Project Activity	Activity Not Available	Very High Activity
Open Hub Data Quality	Updated 4 months ago	Updated about 9 hours ago
Homepage	subversion.apache.org	git-scm.com
Project License	apache_2	gpl
Estimated Cost	\$10,385,744	\$6,771,881
30 Day Statistics		
Contributors (Past 30 Days)	8 developers	35 developers
Commits (Past 30 Days)	75 commits	260 commits
Files Modified	50 files	395 files
Lines Added	3,675 lines	20,573 lines
Lines Removed	1,604 lines	5,127 lines

Which version control do we choose?

- Both are frequently used by developers in combination with Xoj
- 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 to make life easier
 - https://en.wikipedia.org/wiki/Comparison_of_Subversion_clients
 - <https://git-scm.com/downloads/guis>

Which version control do we choose?

So in short, between subversion en git there are no bad choices.

Git can also be used offline

Subversion supports file locking out of the box

All other features are available on both systems or easily configured in another way.

Just 1 warning about sharing code

Regardless of using sourcecontrol or not.

Xojo does never reload your code after opening a project, so always close your project before fetching the new version.

Otherwise you can overwrite changes another developer posted.

And sourcecontrol will blame you in the history.

Github

Why GitHub? Enterprise Explore Marketplace Pricing Search GitHub Sign in Sign up

Built for developers

GitHub is a development platform inspired by the way you work. From **open source to business**, you can host and review code, manage projects, and build software alongside 31 million developers.

Username
Email
Password

Make sure it's at least 15 characters OR at least 6 characters including a number and a lowercase letter. [Learn more.](#)

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account-related emails.

Github

- 1) Go to <http://github.com>
- 2) Create an account
- 3) Start a project

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Read the guide

Start a project

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

1) Owner: /

2) Description (optional):

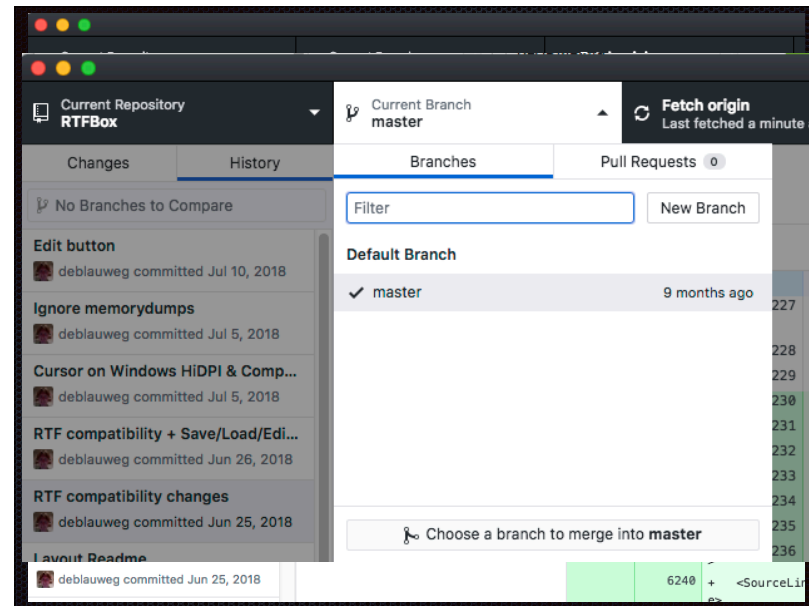
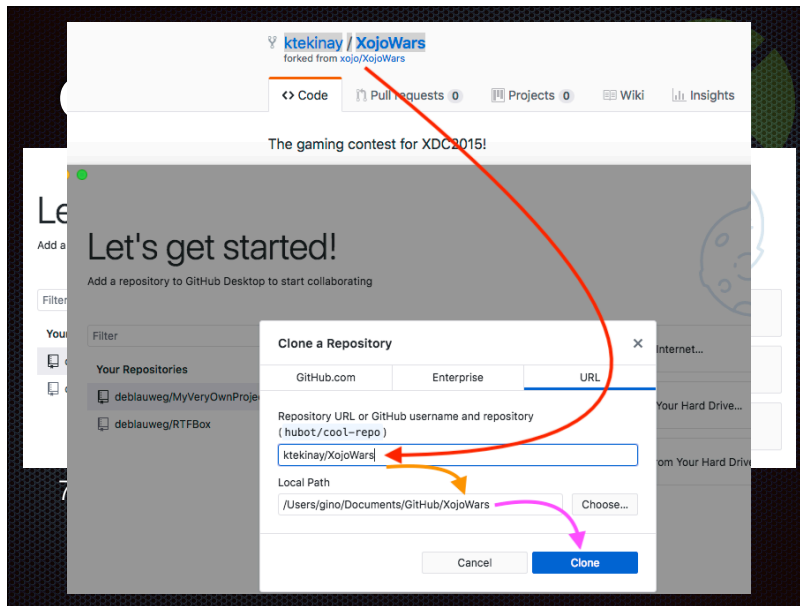
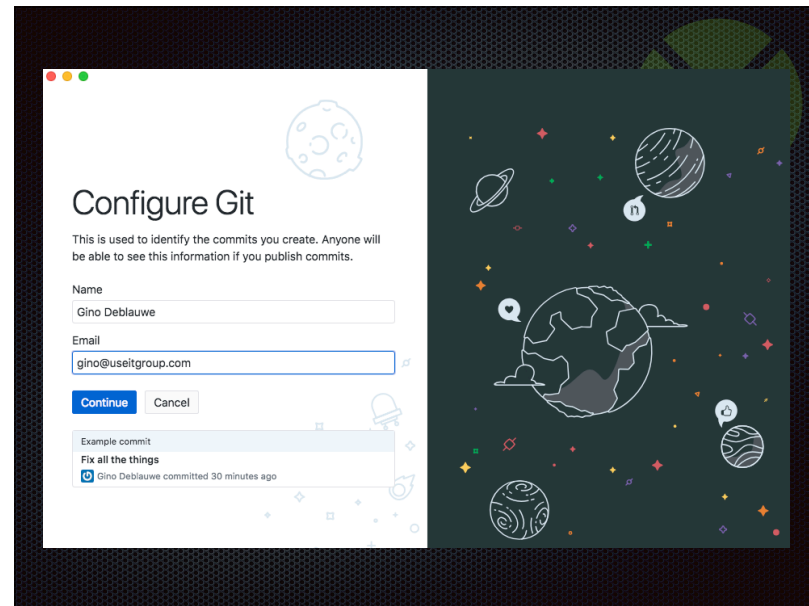
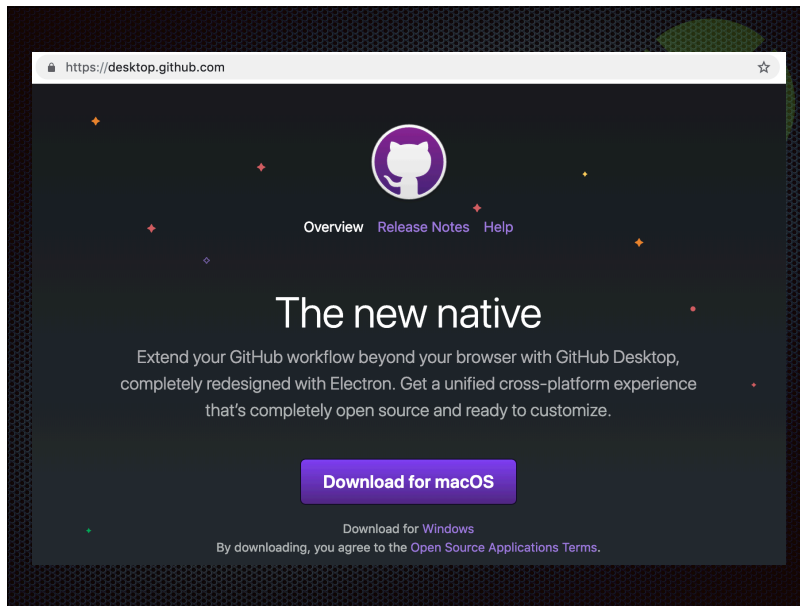
3) **Public**
Anyone can see this repository. You choose who can commit.

4) **Private**
You choose who can see and commit to this repository.


Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: Add a license: ⓘ

Create repository



Installing Subversion on Ubuntu 18.04



- `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`

Adding users



- Make a group for your svn users
`sudo groupadd svn`
- Change group ownership of the repositories directory to the new group
`sudo chgrp svn /usr/local/svn/repos`

Adding users



- 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 file create here will be owned by the group

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

```
dirk@ubuntu-s-1vcpu-1gb-ams3-01: /usr/local/svn$ ls -l
total 4
drw-rwSr-- 2 dirk svn 4096 Apr 11 17:29 repos
dirk@ubuntu-s-1vcpu-1gb-ams3-01: /usr/local/svn$
```

Adding users

- 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 you are a members off)

```
dirk@ubuntu-s-1vcpu-1gb-ams3-01:/usr/local/svn$ groups
dirk sudo svn
dirk@ubuntu-s-1vcpu-1gb-ams3-01:/usr/local/svn$
```

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

- `sudo 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`

```
root@ubuntu-s-1vcpu-1gb-ams3-01:/usr/local/svn/repos# ls -l
total 4
drw-rwSr-- 6 root svn 4096 Apr 11 18:19 myproject
root@ubuntu-s-1vcpu-1gb-ams3-01:/usr/local/svn/repos#
```

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

- We do this by editing svnserv.conf. Each repository has its own settings file.

```
sudo nano /usr/local/svn/repos/myproject/conf/svnserv.conf
```

- Put the following rules in the svnserv.conf file

```
anon-access = none
auth-access = write
password-db = passwd
```

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

```
GNU nano 2.5.3 File: ...repos/myproject/conf/svnserv.conf

### authenticated users, respectively.
### Valid values are "write", "read", and "none".
### Setting the value to "none" prohibits both reading and writing;
### "read" allows read-only access, and "write" allows complete
### read/write access to the repository.
### The sample settings below are the defaults and specify that anonym$
### users have read-only access to the repository, while authenticated
### users have read and write access to the repository.
anon-access = none
auth-access = write
### The password-db option controls the location of the password
### database file. Unless you specify a path starting with a /,
### the file's location is relative to the directory containing
### this configuration file.
### If SASL is enabled (see below), this file will NOT be used.
### Uncomment the line below to use the default password file.
password-db = passwd
### The authz-db option controls the location of the authorization
### rules for path-based access control. Unless you specify a path

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify
^X Exit ^R Read File ^N Replace ^U Uncut Text ^I To Spell
```

Set user passwords

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

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

- add users using the following syntax.

```
username = password
```

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

```
GNU nano 2.5.3 File: /usr/local/svn/repos/myproject/conf/passwd

### This file is an example password file for svnserve.
### Its format is similar to that of svnserve.conf. As shown in the
### example below it contains one section labelled [users].
### The name and password for each user follow, one account per line.

[users]
dirk = dirkspassword
gino = ginospassword

[ Read 8 lines ]
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text   ^J Justify
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell
```

Protect password file

- Since the passwords are stored unencrypted, it's important that you protect the passwords file by setting the proper permissions. The file should not be readable by anyone except the owner (which is root), so change its mode to 600:

```
chmod 600 /usr/local/svn/repos/myproject/conf/passwd
```

```
root@ubuntu-s-1vcpu-1gb-ams3-01: /usr/local/svn/repos/myproject/conf# ls -l
total 16
-rw-r--r-- 1 root svn 1080 Apr 11 18:19 authz
-rw-r--r-- 1 root svn  885 Apr 11 18:19 hooks-env.tmpl
-rw----- 1 root svn  305 Apr 11 19:00 passwd
-rw-r--r-- 1 root svn 3996 Apr 11 18:27 svnserve.conf
root@ubuntu-s-1vcpu-1gb-ams3-01: /usr/local/svn/repos/myproject/conf#
```

Make sure the svn server runs on startup

- Download the svnserve script
[wget http://odyniec.net/articles/ubuntu-subversion-server/svnserve](http://odyniec.net/articles/ubuntu-subversion-server/svnserve)
- Place the script in /etc/init.d

```
sudo cp ./svnserve /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

Make sure the svn server runs on startup

- run update-rc.d to install the script

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

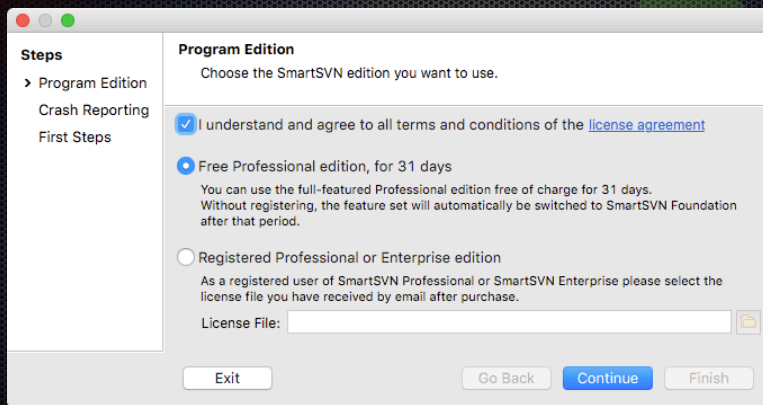
- That's it. svnserve will be started automatically when your system boots up.
- To start it manually, run

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

Make sure the svn server runs on startup

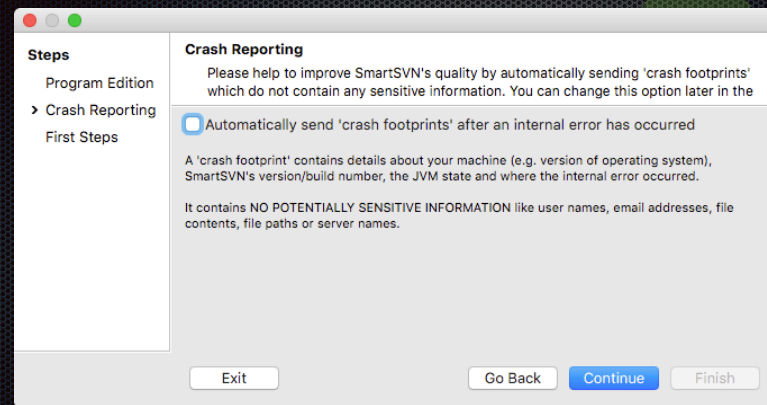
- Installing Subversion on an Ubuntu server takes about 20 commands including editing the configuration files
- Estimated time to install is under 10 minutes

Install and start SmartSVN <https://www.smartsvn.com/>



The screenshot shows the SmartSVN installation window. On the left, a 'Steps' sidebar lists 'Program Edition', 'Crash Reporting', and 'First Steps'. The main area is titled 'Program Edition' with the instruction 'Choose the SmartSVN edition you want to use.' There are three radio button options: 1) 'I understand and agree to all terms and conditions of the [license agreement](#)' (checked), 2) 'Free Professional edition, for 31 days' (selected), and 3) 'Registered Professional or Enterprise edition'. Below the second option, there is explanatory text: 'You can use the full-featured Professional edition free of charge for 31 days. Without registering, the feature set will automatically be switched to SmartSVN Foundation after that period.' Below the third option, there is text: 'As a registered user of SmartSVN Professional or SmartSVN Enterprise please select the license file you have received by email after purchase.' and a 'License File:' input field with a file selection icon. At the bottom, there are four buttons: 'Exit', 'Go Back', 'Continue', and 'Finish'.

Install and start SmartSVN <https://www.smartsvn.com/>

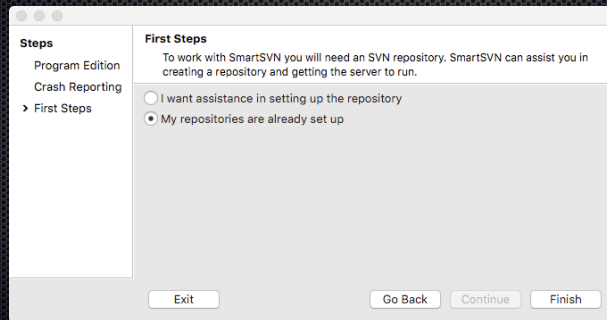


The screenshot shows the SmartSVN installation window at the 'Crash Reporting' step. The 'Steps' sidebar on the left is updated to show 'Crash Reporting' as the current step. The main area is titled 'Crash Reporting' with the instruction 'Please help to improve SmartSVN's quality by automatically sending 'crash footprints' which do not contain any sensitive information. You can change this option later in the'. There is one checkbox option: 'Automatically send 'crash footprints' after an internal error has occurred', which is currently unchecked. Below this, there is explanatory text: 'A 'crash footprint' contains details about your machine (e.g. version of operating system), SmartSVN's version/build number, the JVM state and where the internal error occurred. It contains NO POTENTIALLY SENSITIVE INFORMATION like user names, email addresses, file contents, file paths or server names.' At the bottom, there are four buttons: 'Exit', 'Go Back', 'Continue', and 'Finish'.

Install and start SmartSVN

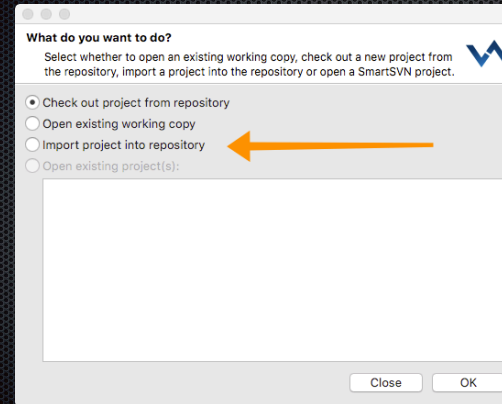
<https://www.smartsvn.com/>

Select "My repositories are already set up"



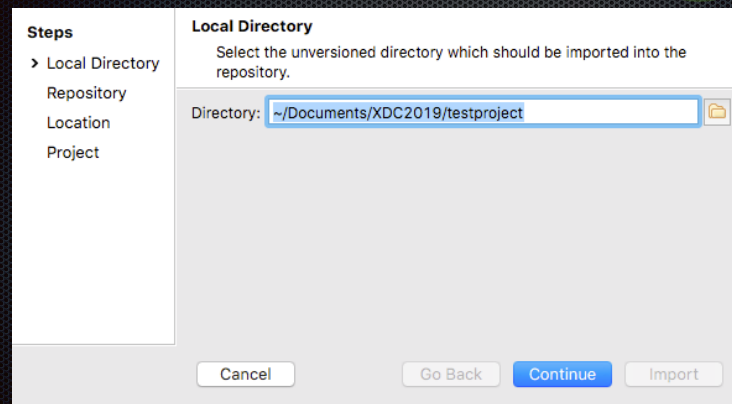
Import your project

Select "Import project into repository"



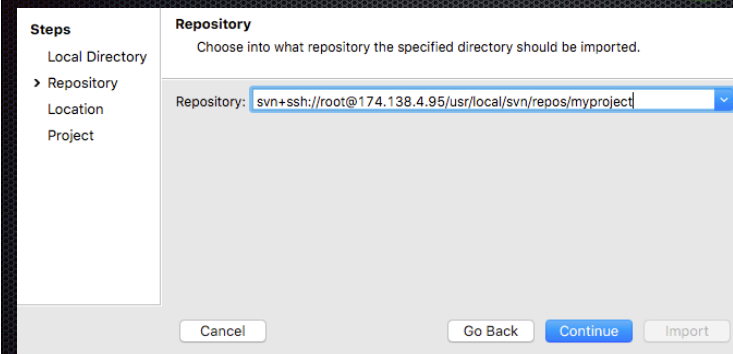
Import your project

Select your local folder that you want to import



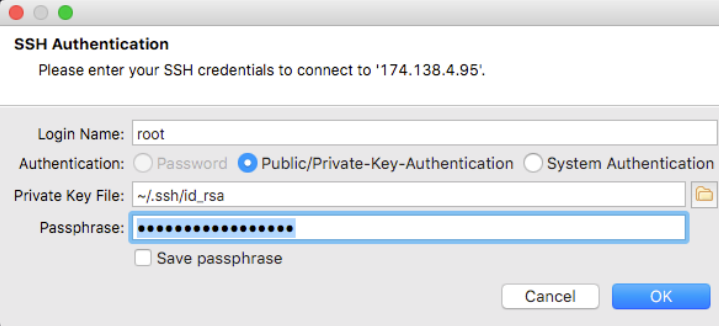
Import your project

Select your repository (we're using svn+ssh)



Import your project

Authenticate with your ssh credentials




SSH Authentication

Please enter your SSH credentials to connect to '174.138.4.95'.

Login Name:

Authentication: Password Public/Private-Key-Authentication System Authentication

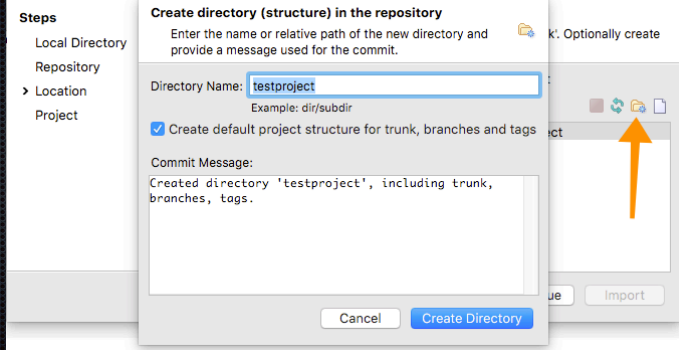
Private Key File: 

Passphrase:

Save passphrase

Import your project



Make a directory for your project and create the default project structure



Steps

- Local Directory
- Repository
- > Location
- Project

Create directory (structure) in the repository

Enter the name or relative path of the new directory and provide a message used for the commit.   k. Optionally create

Directory Name:

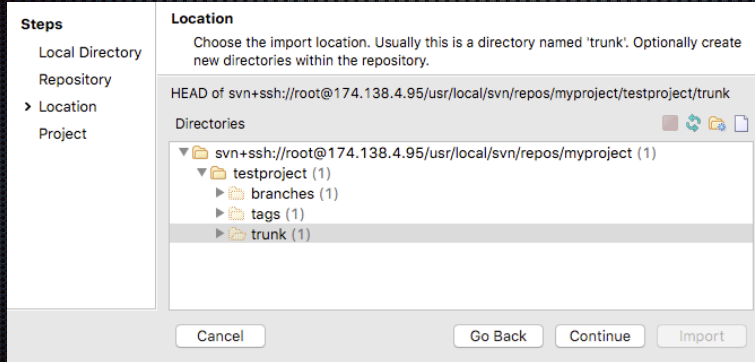
Example: dir/subdir

Create default project structure for trunk, branches and tags

Commit Message:
Created directory 'testproject', including trunk, branches, tags.

Import your project

Select the trunk of your new directory






Steps

- Local Directory
- Repository
- > Location
- Project

Location

Choose the import location. Usually this is a directory named 'trunk'. Optionally create new directories within the repository.

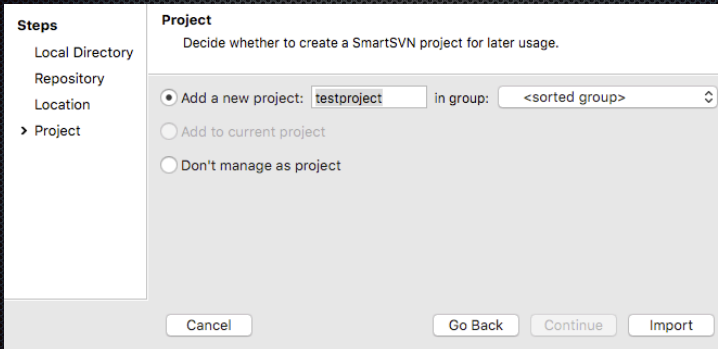
HEAD of svn+ssh://root@174.138.4.95/usr/local/svn/repos/myproject/testproject/trunk

Directories   

- svn+ssh://root@174.138.4.95/usr/local/svn/repos/myproject (1)
 - testproject (1)
 - branches (1)
 - tags (1)
 - trunk (1)

Import your project

Add the project in group "sorted group" and click Import



Steps

- Local Directory
- Repository
- Location
- > Project

Project

Decide whether to create a SmartSVN project for later usage.

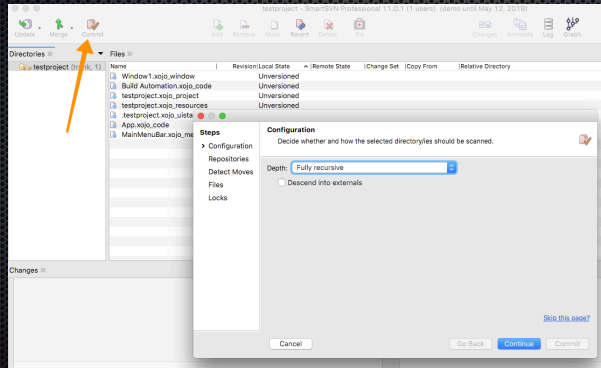
Add a new project: in group:

Add to current project

Don't manage as project

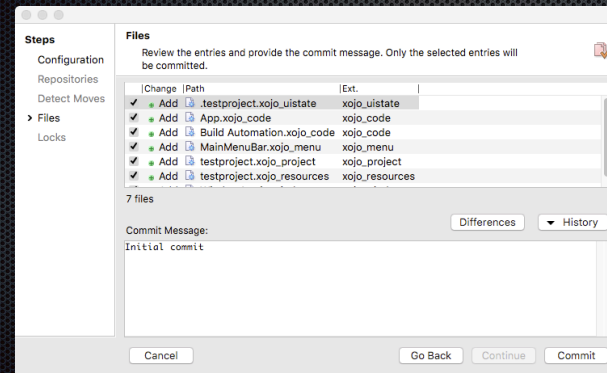
Commit your project

Commit your files to the repository.
Select "fully recursive"



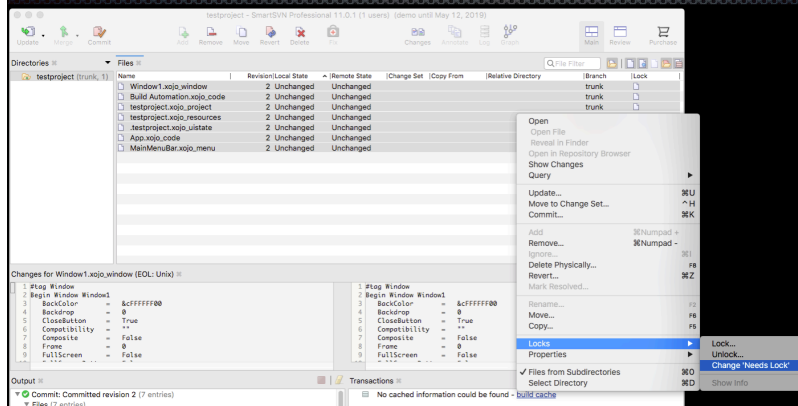
Commit your project

Enter a commit message and Commit



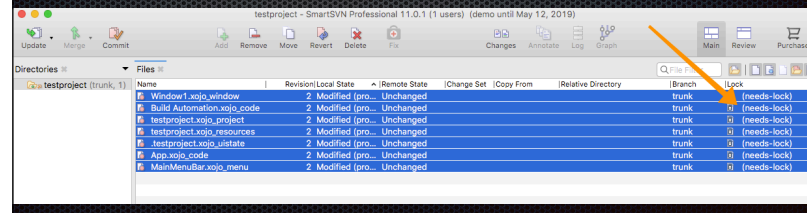
Set 'Needs Lock'

Select all your files and select
Locks -> Change 'Needs Lock'



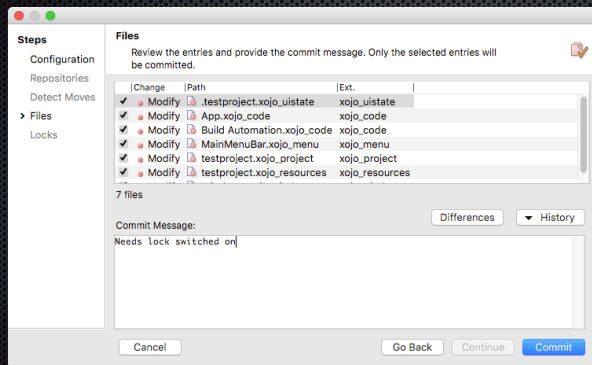
Set 'Needs Lock'

The files in the repo now need to be locked to be able to work on them. As long as you have not locked a file, it's read-only. If you want to work on a file, then lock it. Other people in your team will not be able to work on that file till you either unlock it, or commit your changes.



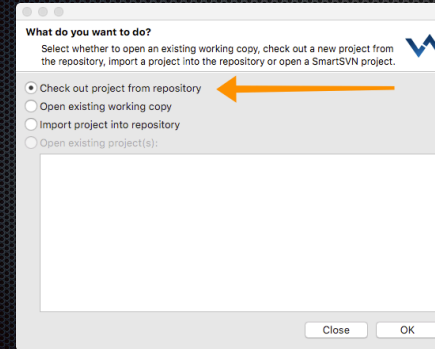
Set 'Needs Lock'

Commit the fact that the files need a lock to the repository.



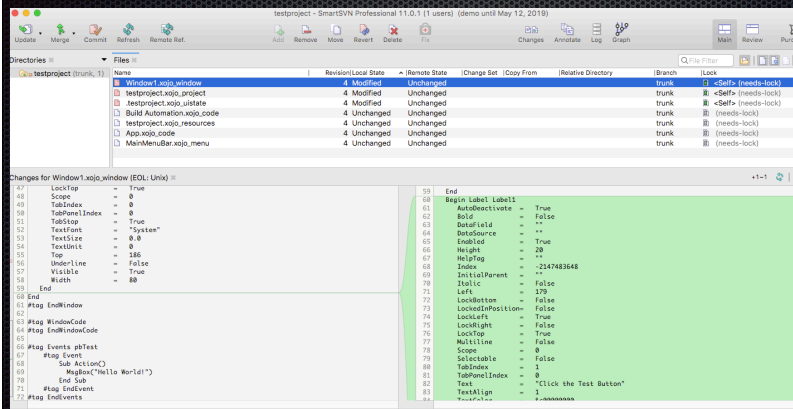
Add other team members

Other team members can get the project from the repository. Just make sure they select the trunk of the project.



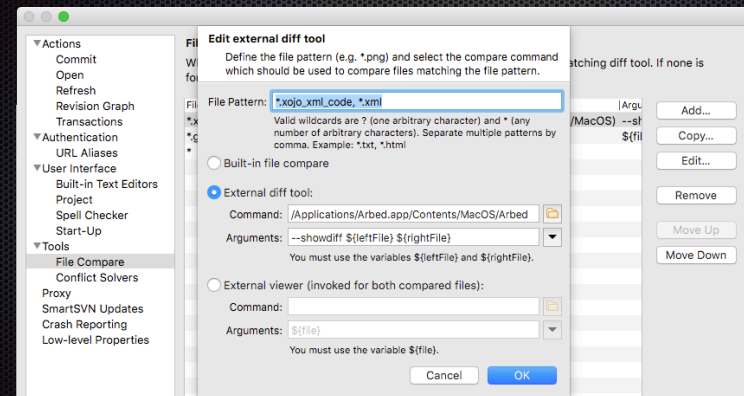
Reviewing changes

SmartSVN shows you the changes, so that you can review them before committing or reverting



Reviewing changes

If you work with the xml format for your Xojo program, you can use Arbed to show you the code differences (<http://www.tempel.org/Arbed/Arbed>)



Reviewing changes

This is how SmartSVN shows you the difference in code for an xml project

The screenshot shows the SmartSVN interface with two versions of an XML file side-by-side. The left pane shows the original code, and the right pane shows the modified code. A central pane highlights the differences in green. The XML code is as follows:

```
2203 <SourceLine>Dim ar As clsActiveReport</SourceLine>
2204 <SourceLine>Dim iTaalNr As Integer</SourceLine>
2205 <SourceLine>/</SourceLine>
2206 <SourceLine>ActiveReports ...</SourceLine>
2207 <SourceLine>If clsActiveReportSession.RpxFileExists("envelop.rpx") Then</SourceLine>
2208 <SourceLine> iTaalNr = Optics.GetOptValue_Long(Optics.OPTALGAFDRUKTAAL, "envelop")</SourceLine>
2209 <SourceLine>If iTaalNr = 0 Then</SourceLine>
2210 <SourceLine> iTaalNr = TAAL_GEBRUIKER</SourceLine>
2211 <SourceLine>End If</SourceLine>
2212 <SourceLine>/</SourceLine>
2213 <SourceLine>arSession = New clsActiveReportSession</SourceLine>
2214 <SourceLine>ar = New clsActiveReport</SourceLine>
2215 <SourceLine>/</SourceLine>
```

Reviewing changes

This is how Arbed shows you the difference in code for an xml project

The screenshot shows the Arbed interface with two versions of an XML file side-by-side. The left pane shows the original code, and the right pane shows the modified code. A central pane highlights the differences in pink. The XML code is as follows:

```
ActiveReports ...
If clsActiveReportSession.RpxFileExists("envelop.rpx") Then
ar = New clsActiveReportSession
ar = New clsActiveReport
ar.Title = Vertaal(LNGLAYOUT) + " + " + Vertaal(LNGENVELOPES)
arSession.Preview = True
arSession.Copies = 1
ar.RPXFile = "envelop.rpx"
ar.SourceID = iBedrijfId
```

Reviewing changes

You can view a log of the changes to a file

The screenshot shows the SmartSVN interface displaying a log of changes for a file. The log table is as follows:

Revision	Author	Commit Message	Date	File Count	Path
4244	Bert		Wednesday 07:14	1	RBMain\Windows\wndVoorbereidingBedrijfsgegevens.xml
4242	Bert	envelopes printen niet mogelijk op mac	Wednesday 02:20	1	RBMain\Windows\wndVoorbereidingBedrijfsgegevens.xml
4243	Bert	Ticket: #31853 - envelopes printen niet mogelijk op mac	Wednesday 02:20	3	RBMain\Windows\wndVoorbereidingBedrijfsgegevens.xml
4242	Bert	Ticket: #31853 - envelopes printen niet mogelijk op mac	Wednesday 02:20	3	RBMain\Windows\wndVoorbereidingBedrijfsgegevens.xml
3773	gjn	Aanpassingen voor externe rapport	12/19/2018 02:11	134	RBMain\Windows\wndVoorbereidingBedrijfsgegevens.xml
3664	brian	Ticket: #3923 - Eenheid afmetingen layout bedrijfsgegevens	11/26/2018 01:31	1	RBMain\Windows\wndVoorbereidingBedrijfsgegevens.xml

Reviewing changes

You can view a log (graph) of the revisions

The screenshot shows the Arbed interface displaying a log (graph) of the revisions. The log table is as follows:

Revision	Author	Date	Revision Info
4244	Bert	Wednesday 07:14	Ticket: #31853 - envelopes printen niet mogelijk op mac
4242	Bert	Wednesday 02:20	Ticket: #31853 - envelopes printen niet mogelijk op mac
4243	Bert	Wednesday 02:20	Ticket: #31853 - envelopes printen niet mogelijk op mac
4242	Bert	Wednesday 02:20	Ticket: #31853 - envelopes printen niet mogelijk op mac
3773	gjn	12/19/2018 02:11	Date Added
3664	brian	11/26/2018 01:31	Branch trunk Author: Bert

Q & A

Gino Deblauwe & Dirk Cleenwerck

gino@useitgroup.com

dirk@useitgroup.com

Give us feedback on this session in the XDC app!

