Tracking your source code with Git

If you check out the DSpace source code from SVN, you get stuck in a rut where you start to work on modifications, but can't contribute them back to the community for various reasons:

- You're not a commiter with commit rights to SVN
- The feature is not quite "finished" and it would be somewhat premature to send it in
- Your modifications include local customizations that are unfit for community contribution.
- You're getting annoyed that have had uncommited changes for several weeks

This guide will possibly help with all of those problems.

Git, is a version control system that allows you to make commits locally that don't bother anyone else's system. So you can work on your feature locally for a week or so, committing every couple of hours as you make progress, and once its finished and tested, you can then push it to a remote/central server. Thats what distributed means.

Git Resources

A list of some possibly useful external Git resources:

- Intro to Git for those who know SVN: http://git.or.cz/course/svn.html
- Fedora Developers' list of useful Git Resources: Git Resources
- Fedora Developers' Best Practices (not all are Fedora specific): Git Guidelines and Best Practices
- Fedora Developers' Git Quick Start Guide: Git Quick Start Guide
- Git Reference: http://gitref.org/
- Pro Git Book (Online): http://progit.org/book/
- Chris Wilper's (Fedora Committer) tips/rules on using Git: https://wiki.duraspace.org/display/~cwilper/How+I+Use+Git

Still want to use SVN locally?

- GitHub does have some basic SVN client support: https://github.com/blog/966-improved-subversion-client-support
- Or, you could obviously download the Zipped up DSpace release packages and import them into your local SVN

Working on a big feature in SVN, but wanting to be able to locally commit every so often (with Git)

If you are working on a project that is too small to warrant the effort of making a branch in SVN, and using the DSpace sandbox directory, then you can just initialize a Git repository on top of your svn checkout, and it will allow you to make commits in git, that doesn't interfere with SVN.

```
svn co http://scm.dspace.org/svn/repo/dspace/trunk/
cd trunk
git init .
git add .
git commit -a -m "Initializing Git repo on top of svn repo"
```

From there, svn and Git will show that you've got a mostly clean system. SVN will notice that you've got a new .git file, but you can just ignore that. At this point you may want to start developing a new feature and it will take two weeks, and you'll need to rebuild DSpace dozens of times, checking that at each point that you've modified things that everything still works. In SVN, you'll have a dirty tree for the entire period of time, so the output of svn diff is mostly useless, especially if you've created a new file, and especially if you're coming in fresh on a Monday, and don't recall where you left off last week. A good practice in Git will be to commit your changes frequently. With Dspace, you have a decent amount of downtime waiting for maven to recompile, so during that period would be a perfect time to commit to your local Git branch.

```
## Commit all changes
git commit -a

## OR
## Commit just a particular file to git
git commit -- dspace-api/src/main/java/org/dspace/content/Community.java
```

So then, when you are working on something you can then just say git log, or git diff, and it will show you what you've recently committed, or what is still dirty since last commit. Once you've finished the project or feature that you are working on, you can then use svn to commit the "perfect" commit to trunk. Behind the scenes it took you countless rebuilds and code changes, but all that is officially committed to svn is the cleaned up "works good" commit.

Using the Git repository on Github

You can also clone a recent copy of DSpace directly from GitHub. You could fork that project, and work on your personal instances code there. From there interactions will be the same as Git interactions. The code repository in GitHub is not designed to sync changes down to SVN. So you can't make your changes there to commit to the svn repository. You could create a patch of your changes, and then apply that to a svn checkout, and commit there. We are hopeful that the GitHub location facilitates users to actively develop their work, and potentially in a public place so that many can benefit.

See: https://github.com/DSpace/DSpace/wiki

Git adoption by the DSpace developers

Final Note: The DSpace developers (committers), are currently firmly committed to using SVN for official contributions to the DSpace source code. The cost of migrating to Git is still substantial, especially as many modules are not being mirrored on Github, not all developers are fluent in Git, and our release process is only configured for subversion.

There has been a DSpace Developers Meeting on Git: DevMtg 2011-02-09

And the IRC chat log is: http://irclogs.duraspace.org/index.php?date=2011-02-09