chem-bla-ics

CDK development with branches using Git

Egon Willighagen

Published September 7, 2008

Citation
Willighagen, E. (n.d.). In *chem-bla-ics*. chem-bla-ics. https://doi.org/10.59350/vt3yd-8nb03

Keywords
Cdk, Svn

Copyright
Copyright © Egon Willighagen 2008. Distributed under the terms of the Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Christoph pointed me to a video on Git by Linus. CDK is now using branches extensively in development, and just set up a branch for the upcoming 1.2.0 release later this year (end of October, see cdk-1.2.x). Christoph has just reviewed the branch containing the API move to Iterable. This patch now allows to do this (which would really deserve a blog item by itself):

```
for (IAtom atom : molecule.atoms()) {
  System.out.println("Symbol: " + atom.getSymbol());
}
```

Now, while branching in SVN is easy (`svn copy`), merging is a pain, something Miguel and I found out in the last half year, where he and I experimented with using branches in development (see also Comparing Branches). We discovered that porting bug fixes from trunk to a branch, or just keeping the branch synchronized with trunk, simply does not work. And merging itself, after a while, became a tedious process. So, when watching Linus' movie on Git where he mentions being able to merge several branches a day, I knew I had to switch. A full switch for the CDK depends on an always accessible repository (I have been thinking about GitHub; anyone with an opinion on that?).

However, you can start using Git without a central Git repository, including branch support. This blog by Bart has the juicy details, which I'll apply here to CDK, for easy copy/pasting. This replaces the earlier writing on Offline CDK development using git-svn .

First step is to get yourself a Git mirror of SVN (which will take a long time; do it overnight(s)):

```
$ git svn clone https://cdk.svn.sourceforge.net/svnroot/cdk/cdk/ -T trunk -b branches -t
$ git gc
```

The second command compresses commits to reduce the size of your local Git copy, resulting in a cdk folder of about 300MB. Enter the directory, and check that it has the default `master` branch:

```
$ git branch
* master
```

In SVN one must always do a svn update before one starts coding. Similarly, in git you do (and I found this important to keep your local repository consistent):

```
$ git svn rebase
```

Committing has not changed, and a simple change would go via:

```
$ nano build.xml
$ git commit -m "Changed something, but too lazy to write up what I actually changed" bui
$ git svn dcommit
```

Branches

Now, before we move to setting up branches, one must realize that there are SVN branches and (local) Git branches. Keep that in mind, and consider that we have Git to realize how to keep them synchronized. The check the Git branches one uses `git branch` as shown above; to view the SVN branches, however, we type (which should produce a quite long list for the CDK; only a few listed below):

```
$ git branch -r
 trunk
 tags/cdk-2003-Oct-17
 cdk-1.2.x
 mesprague-iterators
```

Here, the first is CDK trunk, the second a tag tags/cdk-2003-Oct-17, and the last two are the branches cdk-1.2.x and *mesprague-iterators* (no longer existing). I am not sure why the *branches/* is missing here; some git-svn magic I presume.

Now, to create local Git branches that are synchronized with the SVN *cdk-1.2.x* and *cdk-1.0.x* branches, we type:

```
$ git checkout -b my-local-1.2 cdk-1.2.x
$ git checkout -b my-local-1.0 cdk-1.0.x
$ git branch
* master
  my-local-1.0
  my-local-1.2
```

You can now easily change branches with `git checkout <BRANCH>`, and check which SVN path you are working against with `git log -1`:

```
$ git checkout my-local-1.2
$ git log -1
commit 93bd0b22bbad31897eed6686e5b208c5e23505f7
Author: egonw
Date:   Sun Sep 7 08:13:38 2008 +0000

    Fixed inline citation (closes #1987947)


    git-svn-id: https://cdk.svn.sourceforge.net/svnroot/cdk/cdk/branches/cdk-1.2.x@12215
```

Inspection of the output shows the `git-svn-id` line which indicates that that patch was indeed commited against *cdk/branches/cdk-1.2.x*.

chem-bla-ics

With this set up, I can easily changes between trunk and branches, and backport patches from trunk to the *cdk-1.2.x* branch (using `git cherry-pick`) and merge all commits to the branch into trunk using:

```
$ git checkout master
git merge cdk-1.2.x
```

Git does an excellent job here. It recognizes when the branch was last merged with trunk, and will not attempt to apply patches twice. Even better, it also recognized patches that were backported from trunk to the branch, and will not attempt to merge that either.

The result: I can easily merge branches now, generally speeding up CDK development! For example, it reduces the time between someone submits a patch, and when I apply it to *trunk* (or *cdk-1.2.x* in case of a bug fix). I just set up a local branch, apply the patch, and tune until I am happy; I do not keep trunk unstable, as I am doing this in a separate branch. Similarly, if people develop there patch in an SVN branch, I can just as easily switch branches (as described above) and check things, before I merge).

Setting up new SVN branches

As far as I know, `git-svn` cannot create or delete new SVN branches. But this is easy enough with SVN command:

```
$ svn copy https://cdk.svn.sourceforge.net/svnroot/cdk/cdk/trunk https://cdk.svn.sourcefo
$ git svn fetch
$ git checkout -b my-local-newbranch egonw-mynewbranch
$ # hack in my-local-newbranch
$ git commit -a
$ git svn dcommit
$ git checkout master
$ git merge my-local-newbranch
$ svn remove https://cdk.svn.sourceforge.net/svnroot/cdk/cdk/branches/egonw-mynewbranch
```

Enough for now.