MigratingPrivateGitBranches

Migrating Private Git Branches

Summary: use `git format-patch` then `git am -3` to get your work atop the new Apache Hadoop git repository

The checksums of all the commits have changed, you can't rebase against the new trunk unless you want to suddenly get stuck trying to resolve conflict in ivy.xml files of branch 0.17.

If you have a set of commits rebased against a recent version of the (old) git.apache.org trunk, here's how to get it onto the new branch without losing all your commit history.

Preparing

Check in all your work.

Tag the final commit. This lets you get back if anything goes wrong:

(assuming `$JIRA` is set to your JIRA, or you are going to manually insert it)

```
> git tag tag_$JIRA_pre_migrate
```

Note the checksum of the last commit of the old tree *before your commits*. For me —and presumably others —this is 42a61

Tag it:

```
> git tag tag_final_old_trunk 42a61
```

Note the text of it

```
(HDFS-6899. Allow changing MiniDFSCluster volumes per DN and capacity per volume. (Arpit Agarwal))
```

Export your patches

Create a formatted patch sequence of all your commits

```
> git format-patch  --diff-algorithm=minimal -M --keep-subject --stdout  42a61 > changes.mbox
```

This is a history of all the changes. But: it cannot be applied to the new trunk directly as everything has changed. You need to merge it via the body of each patch, rather than against the commit IDs.

Import

Add the new remote https://git-wip-us.apache.org/repos/asf/hadoop.git as `asflive`

```
> git remote add asflive https://git-wip-us.apache.org/repos/asf/hadoop.git
> git fetch asflive
```

Check out `asflive/trunk` to new branch `live/trunk`

```
> git checkout -b live/trunk --track asflive/trunk
```

Locate the commit which matches the commit #41a61 in the old trunk

```
> git log --grep HDFS-6899

commit e871955765a5a40707e866179945c5dc4fefd389
Author: Arpit Agarwal <arp@apache.org>
Date:   Sat Aug 23 06:01:17 2014 +0000

    HDFS-6899. Allow changing MiniDFSCluster volumes per DN and capacity per volume. (Arpit Agarwal)

    git-svn-id: https://svn.apache.org/repos/asf/hadoop/common/trunk@1619970 13f79535-47bb-0310-9956-
ffa450edef68
```

Now checkout a branch off that revision

```
> git checkout -b svntrunk e871955765
```

Verify the log contains your last commits

```
> git log ^3
```

Tag this as `tag_svntrunk`

```
 > git tag tag_svntrunk
```

Check out a new branch for your merge

```
> git checkout -b merge/$JIRA
```

apply the merge

```
> git am -3 < changes.mbox
```

Verify that its log matches the last few of your original branch

```
> git log ^5
```

Tag this

```
> git tag tag_$JIRA_merged
```

Rebase and catch up

Now rebase this branch onto the full live trunk

```
git rebase asflive/trunk
```

Fix any problems you hit as you would on any other rebase. (for me it's invariably .pom files)

Maybe: rename the new branch to whatever you want to work with. You can also delete the original branch with a `git delete --force`; the tag you made ensures that it is still somewhere in your history.

Repeat for all other branches you've made. It is best to get this over with in one go.

branch-2 derived work: if your private branches are off branch-2, repeat as above, after checking out `asflive/branch-2` to `live/branch-2`

Finishing the migration

After you have have migrated all the branches you want to, you can clean up the branches.

Drop the old remote (assuming it was called `origin`)

```
> git remote remove origin
```

Rename `asflive`

```
> git remote rename asflive origin
```

Rename the trunk branch

After you have have migrated all the branches you want to, you can clean up the branches.

Drop the old remote (assuming it was called `origin`)

```
> git remote remove origin
```

Rename `asflive` to be `origin`

```
> git remote rename asflive origin
```

Rename {{{trunk

```
> git branch -m live/trunk origin
```