HBase Branching

Enis Soztutar (enis)
12/05/2014
Doc version: 1.0.2

Intro

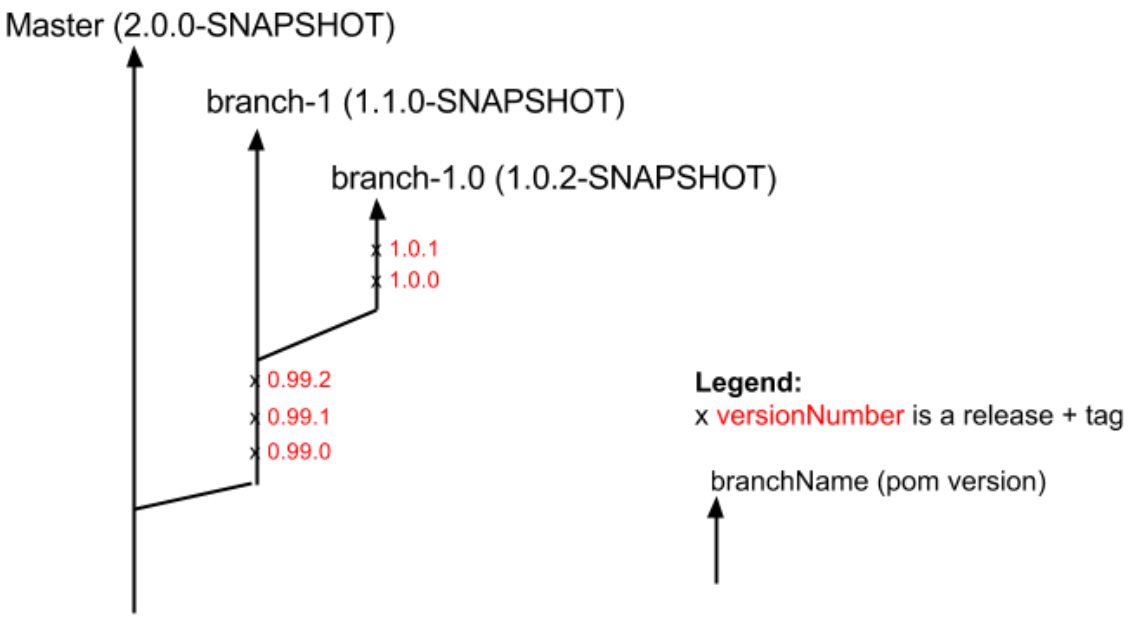
This doc contains a proposal for Apache HBase branching model for post-1.0. Hopefully it will help reduce the confusion around what patches can and should go where, and what are the release vehicles. This document assumes that we are following the semantic versioning that we agreed on previously.

For the short term, it is possible that we will have a couple of active branches and release trains. This document will only talk about branching for 1.0+.

Branching

For having releases in `major.minor.patch` format, we should have a branch, corresponding to every major version and a branch for every major.minor version that there is a .0 patch release of. Meaning, if we want to release x.y.0 then, there should be a branch called branch-x.y. All the patch releases (including x.y.0) will be done from a release branch named “branch-x.y”.

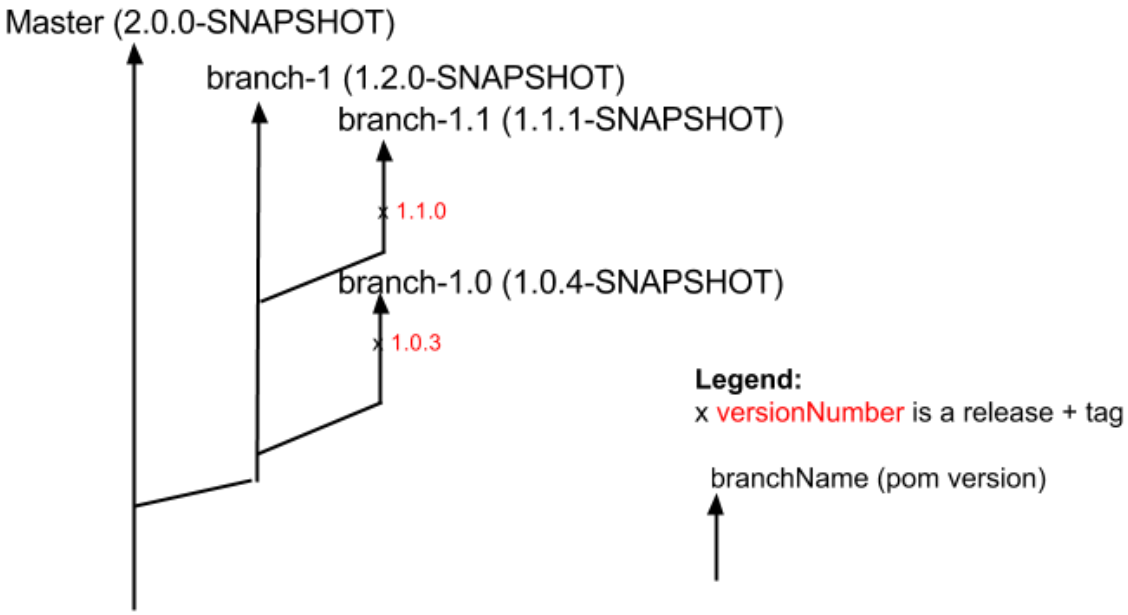
For example, for the 1.0.0 release, we will create a branch named branch-1.0 (as opposed to branch-1) and only do the 1.0.0RC0 from this branch. Short term, the branches will look like this:



Master branch will still be there, and will have a pom.xml version 2.0.0-SNAPSHOT, and we will have two branches `branch-1` and `branch-1.0`. The releases of 1.0.x will be made from branch-1.0. branch-1 will have 1.1.0-SNAPSHOT as the pom version, and will be a place for committing patches scheduled for 1.1.0 release.

We do not have to create another branch for x.y until the first RC for x.y.0 is in time, because branch-x will serve as the working place for the next x.y patches. When a release manager wants to release x.y.0, then they should create a branch named branch-x.y and then create the first release candidate for x.y.0RC0 from that branch. This ensures that the releases and branches are matched accordingly.

Post 1.0, this means that branch-1 will be used for the workspace for 1.1, and when in time of 1.1.0RC, the RM should create branch-1.1 and do the release from that branch. Some time later when 1.1.0RC is released, the branches might look like this:



Not creating the branch-1.1 until the 1.1.0RC timeframe arrives has the issue about maintaining branch-1. According to semantic versioning new features can be committed to branch-1, however BC should not be broken. Thus we should be careful for the commits going into branch-1 (and other earlier branches).

Committing

- As usual, all patches are committed to master first, then optionally cherry-picked to relevant branch.
- A patch cannot be committed to branch-x.y unless it is committed to branch-x first (unless a specific fix to the x.y branch)

Post-1.0 and Earlier branches

1.0 is the first release, we are starting to do semantic versioning. According to semVer, a patch release should not contain any new features (and arguably improvements). However, earlier 0.98 and 0.94 releases will be maintained for some time possibly accepting new features and improvements. This might result in some “feature gap”, where a feature/patch committed in a later 0.98.x release might not be found in any 1.0.z releases.