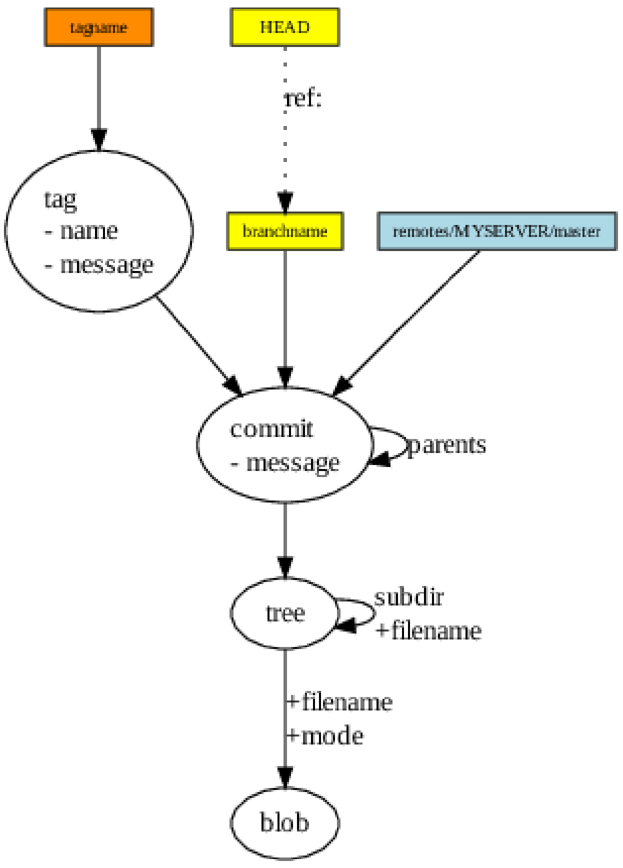
tagging

committed 03 Feb 2009

Tagging in Git is a great way to denote specific release versions of your code, or perhaps if you need a way to refer to exactly one commit in your history for some reason. This post is going to over the right (and wrong) ways to use `git tag`.

Probably the best way to describe a tag is a post-it note that refers to one commit. It contains a name, so something like `v1.0.0` or `production`, and a message too if you want. Git for Computer Scientists visualizes a tag like so:



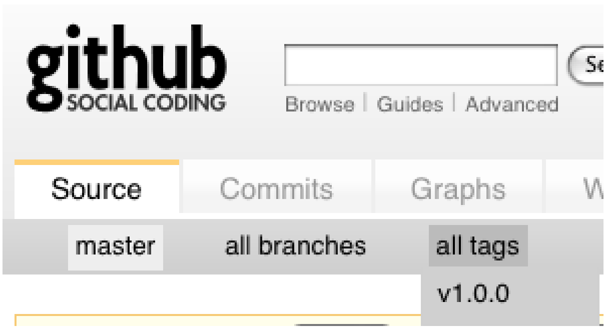
So, how does one create a tag? Just `git tag v1.0.0` right? WRONG. You **usually** don't want to do that. In fact, some suggest that this command does the wrong thing by default. Without arguments, `git tag` creates a "lightweight" tag that is basically a branch that never moves. Lightweight tags are still useful though, perhaps for marking a known good (or bad) version, or a bunch of commits you may need to use in the future. Nevertheless, you probably don't want to push these kinds of tags.

Normally, you want to at least pass the `-a` option to create an unsigned tag, or sign the tag with your GPG key via the `-s` or `-u <key-id>` options. Once this has been done you can then use `git describe` to figure out how many commits you've created since the last tag or the given tag. Git-fu has a great tutorial on how to use this command (and it deserves a tip of its own for the future too!).

So let's make a new tag and see how we can refer to it even after commits have been made, then we'll push it to a remote. It's a lot easier than you think.

```
$ git tag -a v1.0.0 -m "Creating the first official version."

$ git show v1.0.0
  tag v1.0.0
  Tagger: Nick Quaranto <nick@quaran.to>
  Date:   Tue Feb 3 20:37:45 2009 -0500

  Creating the first official version.
  commit a8d1b55c5d4bdec843d9942cabf1b678bc1d4eed
  Merge: 00b9675... 1b487b8...
  Author: Nick Quaranto <nick@quaran.to>
  Date:   Sun Nov 30 00:41:08 2008 -0500

      Merge branch 'master' of git@github.com:qrush/config

$ git describe --tags
  v1.0.0

$ touch test
$ git add test
$ git commit -am "Adding test files"
  Created commit a7aafb8: Adding test files
   0 files changed, 0 insertions(+), 0 deletions(-)
    create mode 100644 test

$ git describe --tags
  v1.0.0-1-ga7aafb8

$ git push --tags
  Counting objects: 40, done.
  Compressing objects: 100% (37/37), done.
  Writing objects: 100% (40/40), 29.32 KiB, done.
  Total 40 (delta 15), reused 9 (delta 2)
  To git@github.com:qrush/config.git
   * [new tag]         v1.0.0 -> v1.0.0
```

The second describe command shows us that we're one commit ahead of where the last tag was done. This is an easy way to figure out how much work has been done since the last commit. Also, the `--tags` option was used to push here, but just the one tag could have been pushed with `git push origin : v1.0.0`. On GitHub (or on your remote) you should see the tag pushed up:



Another side note, if you have pushed tags that you want to move or rename, Git makes it ridiculously annoying to do so. Check out the manpage for an explanation as to why it's complicated and should be done with caution.

If you have a unique way of integrating tags into your Git workflow, let us know in the comments or even submit a tip so others can learn!