Using Git to Manage a Web Site

By ABHIJIT MENON-SEN

The HTML source for my web site lives in a Git repository on my local workstation. This article describes how I set things up so that I can make changes live by running just git push web.

The one-line summary: push into a remote repository that has a detached work tree, and a post-receive hook that runs git checkout -f.

The Local Repository
It doesn’t really matter how the local repository is set up, but for the sake of argument, let’s suppose you’re starting one from scratch.

$ mkdir website && cd website
$ git init
Initialized empty Git repository in /home/ams/web-
$ echo 'Hello, world!' > index.html
$ git add index.html
$ git commit -q -m "The beginnings of my web site."

Anyway, however you got there, you have a repository whose contents you want to turn into a web site.

The Remote Repository
I assume that the web site will live on a server to which you have ssh access, and that things are set up so that you can ssh to it without having to type a password (i.e., that your public key is in ~/.ssh/authorized_keys and you are running ssh-agent locally).

On the server, we create a new repository to mirror the local one.

$ mkdir website.git && cd website.git
$ git init --bare
Initialized empty Git repository in /home/ams/web-

Then we define and enable a post-receive hook that checks out the latest tree into the web server’s DocumentRoot (this directory must exist; Git will not create it for you):

$ mkdir /var/www/www.example.org
$ cat > hooks/post-receive
#!/bin/sh
GIT_WORK_TREE=/var/www/www.example.org
git checkout -f
$ chmod +x hooks/post-receive
Back on the workstation, we define a name for the remote mirror, and then mirror to it, creating a new master branch there.

$ git remote add web ssh://server.example.org/home/ams/website.git
$ git push web +master:refs/heads/master

On the server, /var/www/www.example.org should now contain a copy of your files, independent of any .git metadata.

The Update Process
Nothing could be simpler. In the local repository, just run:

$ git push web

This will transfer any new commits to the remote repository, where the post-receive hook will immediately update the DocumentRoot for you.

(This is more convenient than defining your workstation as a remote on the server, and running git pull by hand or from a cron job, and it doesn’t require your workstation to be accessible by ssh.)

Notes
First, the work tree (/var/www/www.example.org above) must be writable by the user who runs the hook (or the user needs sudo access to run git checkout -f, or something similar).

Also, the work tree does not need to correspond exactly to your DocumentRoot. Your repository may represent only a subdirectory of it, or even contain it as a subdirectory.

In the work tree, you will need to set the environment variable GIT_DIR to the path to website.git before you can run any git commands (e.g. git status).

Setting receive.denycurrentbranch to “ignore” on the server eliminates a warning issued by recent versions of git when you push an update to a checked-out branch on the server.

You can push to more than one remote repository by adding more URLs under the [remote "web"] section in your .git/config.

There are also other hooks. See githooks(5) [hn.my/githooks] for details. For example, you could use pre-receive to accept or deny a push based on the results of an HTML validator. Or you could do more work in the post-receive hook (such as send email to co-maintainers; see contrib/hooks/post-receive-email).

I wrote this after reading Daniel Miessler’s piece, “Using Git to Maintain Your Website [hn.my/gitmaintain].” His setup is straightforward: push to a bare repository on the server and pull the changes into a second clone that is used as the DocumentRoot. My implementation has the same effect, but there are fewer moving parts, and .git is far from the DocumentRoot.

Abhijit Menon-Sen is a freelance Unix programmer in New Delhi, India. He switched from Perforce to Git some years ago, and enjoys helping people to understand Git better.

Reprinted with permission of the original author. First appeared in hn.my/gitman.