Git - Remote Repository

Kurt Schmidt

Dept. of Computer Science, Drexel University

August 25, 2016

(Much thanks to Mike Conway)

Intro

Remote Repository

- A repository in a different directory, or even on a distant machine
- Allows us to:
  - Play around a bit more safely
  - Keep directories on different computers synchronised
  - Collaborate; to have several people working on the same project (in a different set of notes)
- You might have only read access to remote (so, can only pull changes)
- You might have read and write (especially if it's yours), so, you can also push your changes back up

Define a Remote Repository

Bare Repository

- A bare repository is just a repository
  - Has no working directory
  - Contains no content files
- E.g., it might be the "Momma" repository from which everybody works
- Remember, Git is distributed. Might be many remotes floating around. It's up to the users to define workflow. Git can help

Creating Bare Repository

- Let's start in our `lab-git` directory from previously
- Commit all changes
- Create the directory:

```
$ mkdir -p ~/Git/practise-bare
```

- Initialise the directory:

```
$ pushd ~/Git/practise-bare
$ git init --bare
$ popd # return to your working git directory
```

Add Remote Repository

- Add the new, bare repository as an upstream remote to our working lab directory
- From our working repository, `git-lab`:

```
$ git remote add origin ~/Git/practise-bare
```

- `origin` is just a name you choose to refer to the distant repository
- Not so accurate in this case
- Maybe `practise` would be better:

```
$ git remote add practise ~/Git/practise-bare
```

Viewing Branches and Remotes


```
$ git branch --list
```

```
* master
```

```
$ git remote -v
```

```
practise /home/usr/Git/tutorial-bare/ (fetch)
practise /home/usr/Git/tutorial-bare/ (push)
```

Transport Protocols

- Local – file on local machine

```
$ git clone /home/kschmidt/public_html/CS265/Labs/Git/Sample
```

Or

```
$ git clone file:///~kschmidt/public_html/CS265/Labs/Git/Sample
```

- SSH

```
$ git clone ssh://abc123@tux.cs.drexel.edu/~kschmidt/Git/Sample
```

Or, scp-like syntax:

```
$ git clone abc123@tux.cs.drexel.edu:~kschmidt/Git/Sample
```

Transport Protocols

- Git
  - Seperate listener (port 9418)
  - No authentication
  - Repository needs `git-daemon-export-ok` file
- HTTP[S][1]
  - Needs a post-update hook

```
$ git clone https://www.cs.drexel.edu/~kschmidt/Git/Sample
```

[1] I've not gotten this working on tux, yet

Pushing to Remote

Pushing a Branch to a Remote Repository

- Make sure your local repository is committed
- Push your changes to the remote:

```
$ git push practise master
```

- `practise` is the local nickname of the remote repository to push to
- `master` is the local branch to be pushed
  - (`master` is the only branch you have right now)

Default Remote[2]

- The -u will have git remember the parameters, so you needn't type them in each time
- We need to set a global default value in git first:

```
$ git config --global push.default simple
```

```
$ git push -u practise master
```

[2] I've not found this useful, yet

Local Changes, Update Remote

- Commit when you get to a good spot (think you should)
- When you're ready to make your changes available to others, push your changes up

```
$ echo "Some chocolate to fix a bug" >> hello
$ git add hello
$ git commit -m "Fixed that missing chocolate bug in hello"
```

```
[master c421cc9] Fixed that missing chocolate bug in hello
 1 file changed, 1 insertion(+)
```

```
$ git push practise master
```

```
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 287 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To /home/user/Git/tutorial-bare/
   4c0270d..c421cc9 master -> master
```

Pulling Changes From Remote

Before pushing changes to remote:

- If others maybe changed the remote, pull down recent changes

```
$ git pull [git URL]
```

- (You can specify the remote repository, if not set)
- Resolve any conflicts
- If needed, commit changes (from resolution)
- Now push your changes up

Cloning an Existing Remote Repository

Working Off a Remote Repository

- You can clone a repository
  - On the same machine, as a place to work
  - Onto a different machine, to work, or to synchronise directories
  - You can make a *bare* clone (create a clone that is a bare repository)
- You create a new branch (in a different directory)

  `git clone` *`options repoUrl`* `[`*`directory`*`]`
- Continuing with the preceding example:

```
$ git clone ~/Git/practise-bare Here
Cloning into 'Here'...
done.
```

Changes to a Cloned Repo

- You can make changes in `Here` directory, as usual
- Commit them
- Before pushing changes up to parent repo, do a `pull` operation, in case others have pushed changes up since your last pull
- Resolve any merge conflicts
- Now, push the changes back up
  - Note, the cloned repository is listed as a remote
  - By default, `origin`

Cloned Repo – Example


```
$ git clone ~/Git/practise-bare Here
$ cd Here
$ # make some changes
$ git add -u
$ git pull
Already up-to-date.
$ git commit -m"Fixed the curtains"
...
$ git push
...
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working directory clean
$ git push
...
   0d13e1f..06ee293 master -> master
```

Pushing Changes – Conflicts

- Before pushing changes up, check for changes in the remote repository:

```
$ git pull
...
From /home/kschmidt/git/Play
 06ee293..767aa18 master   -> origin/master
Auto-merging c
...
Auto-merging a
CONFLICT (content): Merge conflict in a
Automatic merge failed; fix conflicts and then commit the result.
```

- If the deltas can be merged automatically, good
- If not, then manually fix the conflicts
  - git shows you the conflicts
  - `HEAD` is the local file

Fixing Conflicts

Listing 1: file1

```
Initial Content

Edit, made locally, earlier

<<<<<<< HEAD
Another local edit
=======
An edit pushed to origin, since last pull
>>>>>>> 767aa187c15f5aa06910809565e5f3c54a0b4dd8
```

Make your changes:

Listing 2: file1, fixed

```
Initial Content

Edit, made locally, earlier

An edit pushed to origin, since last pull
Another local edit
```

Fixing Conflicts – cont.

Now commit again, and push changes up

```
$ git add -u
$ git commit -m"Merged with origin"
[master 6ef80e8] Merged with origin
$ git push
Counting objects: 18, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (10/10), done.
Writing objects: 100% (10/10), 827 bytes | 0 bytes/s, done.
Total 10 (delta 6), reused 0 (delta 0)
To /home/you/Git/practise-bare
 767aa18..6ef80e8 master -> master
```