

A Very Very Short Introduction to Git

Tian Tian

January 6, 2017

Outline

What & Why Git?

Git Commands

Version-controlling: a comparison

Methods	Data Location	Pros	Cons
Manual Dating	L	SO EASY	What about merging?
Dropbox	L/R	Easy synchronizing	Only 1-way fallback
SVN	L/R	True VCS	Centralized
Git	L/R	Decentralized	Merging is verbose

Basic ideas of git

- Cache snapshots of file are stored, not the difference
- Local operations, mirroring remote source
 - A central server is actually **not** necessary
- Can revert to any state

States in git

- Not staged: changes are not recorded
- Staged (not committed): git knows your changes, but you haven't decided if you need to commit.
 - Normally it's easier to just stage and commit your changes
 - May be useful if you are doing a long fix
 - Unstage your changes does not revert the file.
- Committed: the changes are permanently recorded. You can either push your changes or continue working.
- **Anything you committed is safe!** (but the process can mess up)

Helpful utilities for git

- Command line (**of course**)
- The github client (good for visual branching)
- Magit-mode for emacs
- vim-fugitive for vim

Initialize a clean repository

#Initialize a clean git repo

```
git init
```

```
git status
```

Results:

```
Reinitialized existing Git repository in  
/Users/tiantian/polybox/Studies_ETH/Computation-Seminar/.git/  
On branch master
```

```
Initial commit
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be committed)
```

```
#Tian-git.org#  
.#Tian-git.org  
.DS_Store  
Tian-git.dvi  
Tian-git.org  
Tian-git.org~  
Tian-git.pdf  
Tian-git.tex  
Tian-git.tex~  
_minted-Tian-git/
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

Clone from an existing remote repository

```
# Clone from a remote repo to local dir
```

```
git clone https://github.com/lovaulonze/.matplotlib.git some
ls -al some
```

Results

```
total 56
drwxr-xr-x  6 tiantian  staff   204  1  6 10:59 .
drwxr-xr-x 14 tiantian  staff   476  1  6 10:59 ..
drwxr-xr-x 12 tiantian  staff   408  1  6 10:59 .git
-rw-r--r--  1 tiantian  staff    61  1  6 10:59 .gitignore
-rw-r--r--  1 tiantian  staff 24528  1  6 10:59 matplotlibrc
drwxr-xr-x  4 tiantian  staff   136  1  6 10:59 stylelib
```


Remember to use the .gitignore file

Git status w/o .gitignore rules

git status

Results

On branch master

Initial commit

Untracked files:

(use "git add <file>..." to include in what will be committed)

```
./#Tian-git.org
.DS_Store
Tian-git.dvi
Tian-git.org
Tian-git.org~
Tian-git.pdf
Tian-git.tex
Tian-git.tex~
_minted-Tian-git/
some/
```

nothing added to commit but untracked files present (use "git add" to track)

.gitignore (II)

Use .gitignore rules

```
cat .gitignore
```

```
echo ""
```

```
echo ""
```

```
git status
```

Results

```
***
```

```
*~
```

```
*.tex
```

```
_minted*
```

```
some/
```

```
*.dvi
```

```
On branch master
```

```
Initial commit
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be committed)
```

```
.DS_Store
```

```
.gitignore
```

```
Tian-git.org
```

```
Tian-git.pdf
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

Stage the untracked files:

```
# Add some file to the staged area  
# But we leave out some  
git add Tian-* .gitignore  
git status
```

Results

On branch master

Initial commit

Changes to be committed:

(use "git rm --cached <file>..." to unstage)

```
new file:   .gitignore  
new file:   Tian-git.org  
new file:   Tian-git.pdf
```

Untracked files:

(use "git add <file>..." to include in what will be committed)

```
.DS_Store
```

Finally the commit

Commit the stated files

```
git commit -m "Commit. Presentation to the commit"  
git status
```

Results

```
[master (root-commit) 45f73d7] Commit. Presentation to the commit  
3 files changed, 203 insertions(+)  
create mode 100644 .gitignore  
create mode 100644 Tian-git.org  
create mode 100644 Tian-git.pdf  
On branch master
```

Untracked files:

(use "git add <file>..." to include in what will be committed)

.DS_Store

no changes added to commit (use "git add" and/or "git commit -a")

Now use a remote for pushing

```
# Must add a remote for pushing!  
# origin is the default branch  
git remote add origin https://github.com/lovaulonze/git-slides.git  
# Push the changes to the master upstream  
git push -u origin master  
git status
```

Results

```
Branch master set up to track remote branch master from origin.  
On branch master  
Your branch is up-to-date with 'origin/master'.
```

```
Untracked files:  
  (use "git add <file>..." to include in what will be committed)
```

```
.DS_Store
```

```
no changes added to commit (use "git add" and/or "git commit -a")
```

The difference between the remote and local

```
# Now lets do some commit locally
git add Tian-git.org Tian-git.pdf
git commit -m "Now proceed to the remote part"
git status
```

Results

```
[master 262e532] Now proceed to the remote part
 1 file changed, 6 insertions(+), 14 deletions(-)
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 tree clean
```

Check the log of the git

Some formatting using git log

But I prefer to use a GUI or editor plugin

```
git log --pretty=format:"%h %ad | %s%d [%an]" --date=short
```

```
262e532 2017-01-06 | Now proceed to the remote part (HEAD -> master) [Tian Tian]
af9e6be 2017-01-06 | Add .DS_Store to ignore (origin/master) [Tian Tian]
0bc7411 2017-01-06 | Now proceed to the remote part [Tian Tian]
45f73d7 2017-01-06 | Commit. Presentation to the commit [Tian Tian]
```

Undoing

--amend option: add/something immediately after committing.

```
git commit --amend
```

Unstaging

#Add files to stage

```
git add Tian-*
```

```
git status -s
```

#I wanna remove the pdf from staging

```
git reset HEAD Tian-git.pdf
```

```
git status -s
```

```
M Tian-git.org
```

```
M Tian-git.pdf
```

```
Unstaged changes after reset:
```

```
M Tian-git.pdf
```

```
M Tian-git.org
```

```
M Tian-git.pdf
```


What if you don't like your current changes?

- checkout on file

```
# The checkout will revert the changes  
# in this file to the last commit  
git checkout -- $FILE
```

- **But** the change is never saved!

- checkout on version

```
# The checkout with a hash will  
# checkout on the specific version  
git checkout 45f73d7
```

- If local changes have been made after checkout to another version and you don't want to commit them: stash

```
# Stash local changes if you encounter  
# errors checking out to another version  
git stash save  
# Or just ignore the changes  
git stash drop
```

TAG: easier way to work with versions

Add a tag to the HEAD commit

```
git tag 0.1.0
```

You can use relative versions to checkout

^ is the parent commit

```
git checkout 0.1.0^
```

~[num] is the version offset

```
git checkout 0.1.0~2
```

Branching

When is a branching needed?

- Adding experimental features
- Restructuring code
- Your boss changes your L^AT_EXfiles

We are on the master branch!

```
git checkout alter-ego
```

Merge: alter-ego → master

From the alter-ego branch, checkout master

```
git checkout master
```

From the branch, merge with master

```
git merge alter-ego
```

- **CAUTION** editing the same region in both branches will cause unsuccessful merge!
- `git reset --merge` if you find the merge not necessary
- Manually reset the conflicts!
- Binary files cannot be merged.