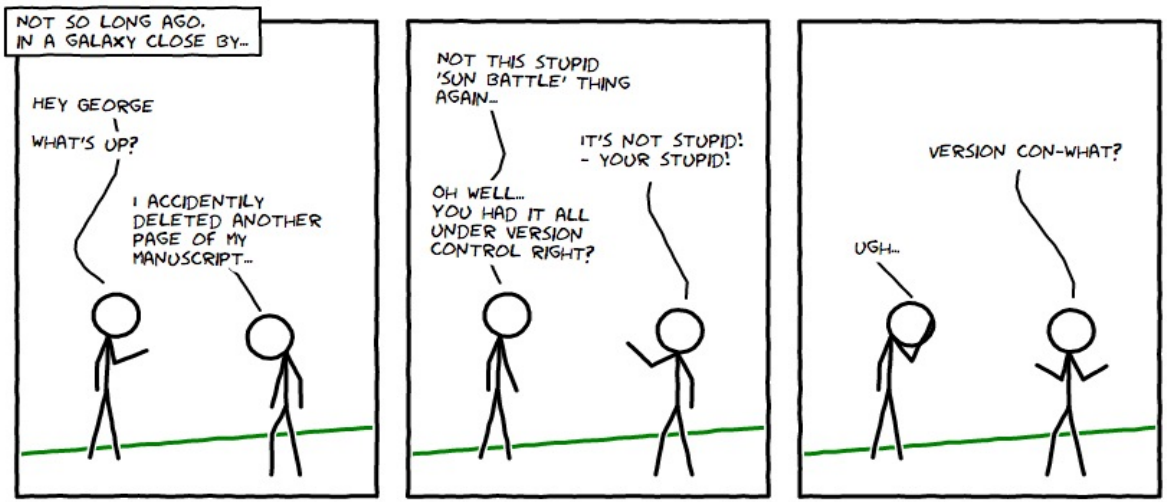
Version Control: Gitting Started

Cai Li

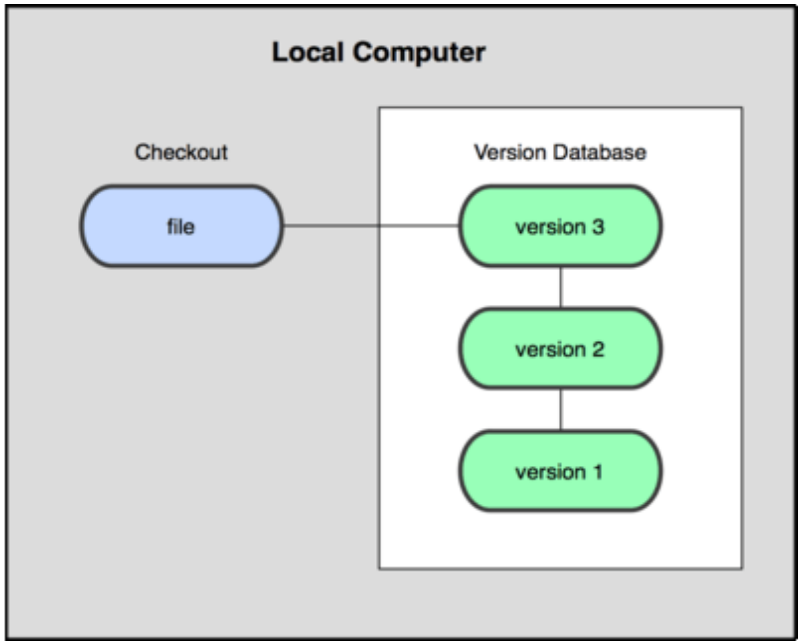
October 2014

What is Version Control?



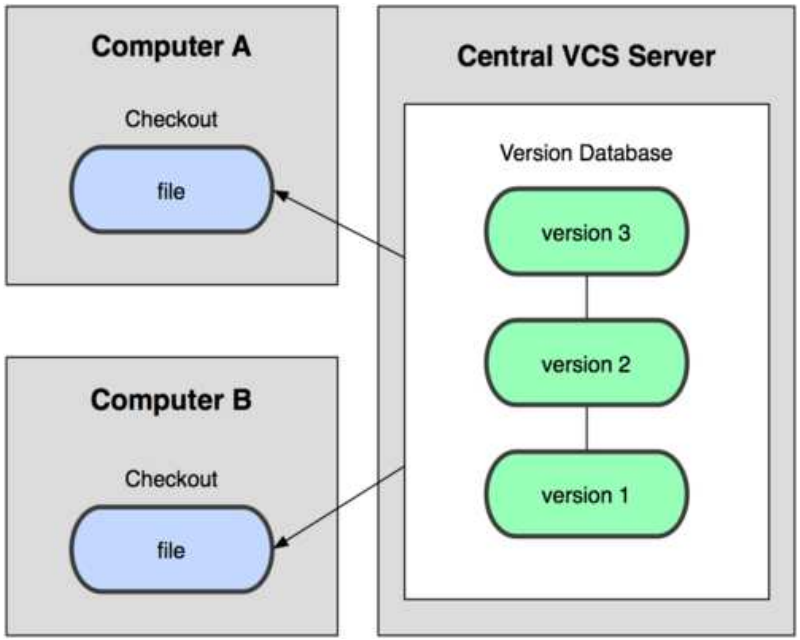
- Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.

Local Version Control System



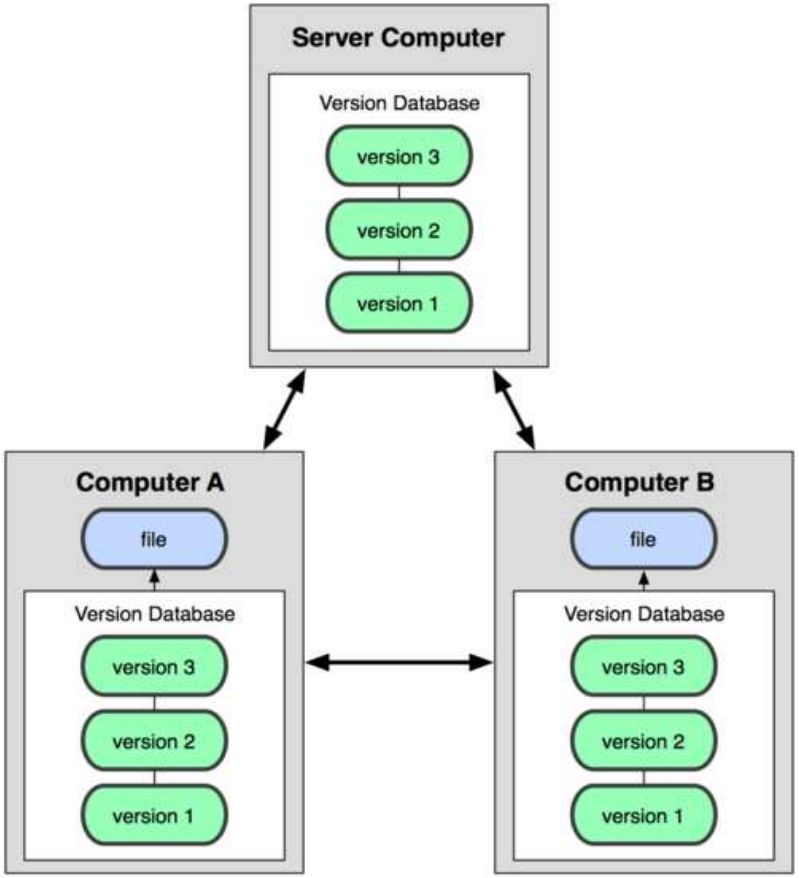
- rcs (still in use).
- This system keeps patch sets (the differences between files).

Centralized Version Control System



- CVS, Subversion, Perforce.
- It has been the standard for version control for many years.

Distributed Version Control System



- Git, Mercurial, Bazaar or Darcs.
- "Distributed-is-the-new-centralized".

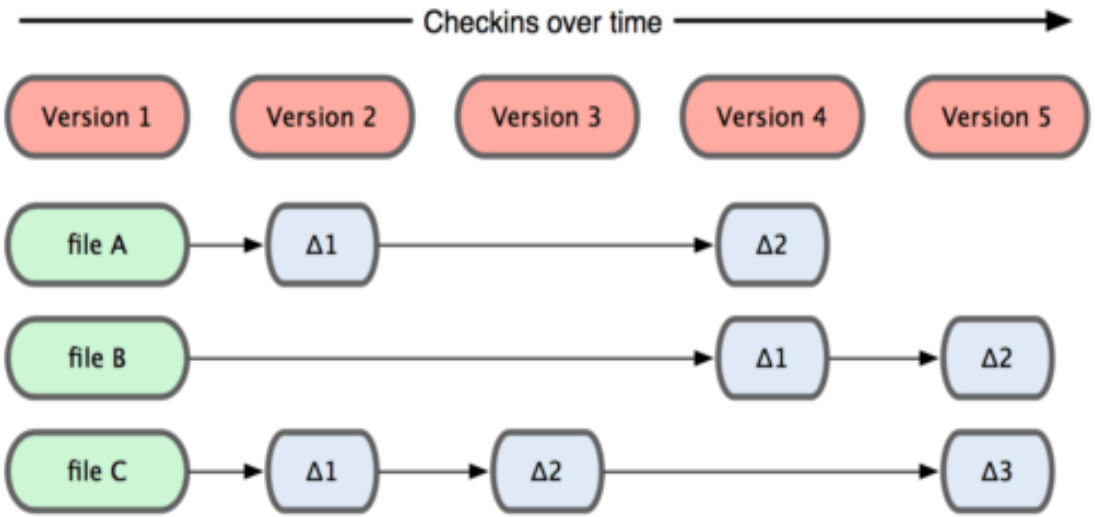
Short history of Git

- Linux kernel project.
- Learn from BitKeeper.
- Initially designed and developed by Linus Torvalds in 2005.

Advantages of Git

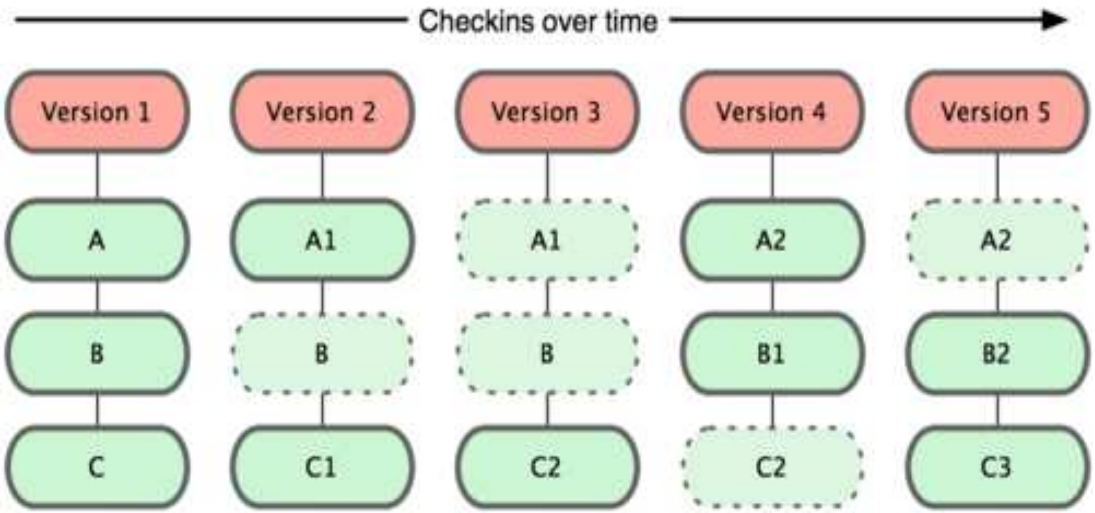
- Speed and simple design.
- Strong support for non-linear development (thousands of parallel branches).
- Fully distributed.
- Able to handle large projects like the Linux kernel efficiently.
- Free and open source.

What Makes Difference?



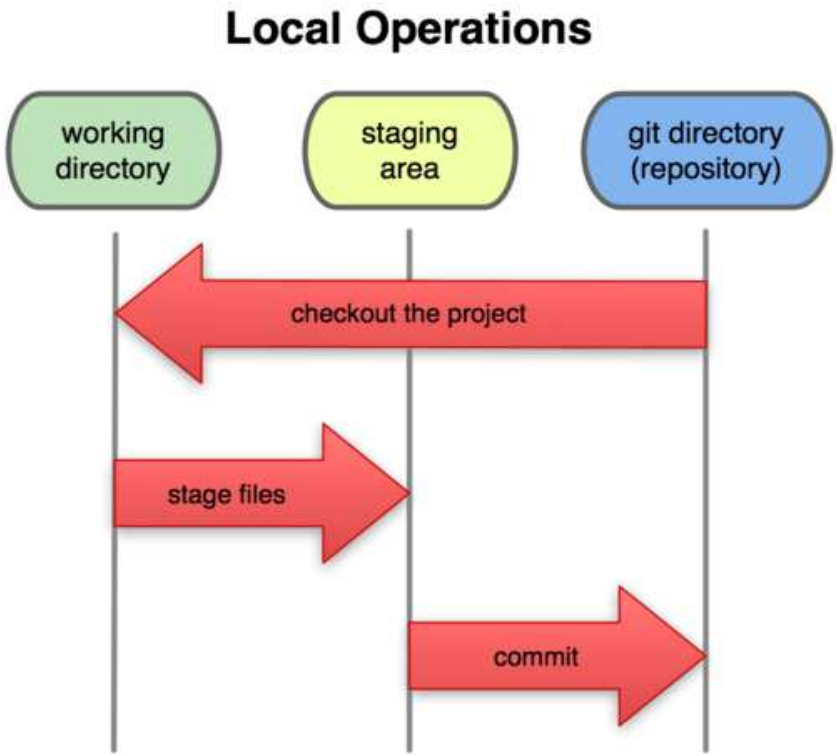
- Other systems tend to store data as changes to a base version of each file.

The way of Git



- Git stores data as snapshots of the project over time.

Workflows of Git



- Modify files in working directory.
- Add snapshots of them to staging area.
- Do a commit: store snapshot permanently to Git directory.

Setting Up

Identify yourself:

- $ git config --global user.name "cli9"
- $ git config --global user.email "cli9@ncsu.edu"

Initializing

Set up a project:

$ git init

```
Administrator@WINXVI71OST1SXY /f/Working Section/2014 Fall/Group Meeting/Example

$ git config --global user.email
cli9@ncsu.edu

Administrator@WINXVI71OST1SXY /f/Working Section/2014 Fall/Group Meeting/Example

$ git init
Initialized empty Git repository in f:/Working Section/2014 Fall/Group Meeting/E
xample/.git/
```

Adding

Keep things tracked and check status often

- $ git status
- $ git add ...

```
Administrator@WINXVI71OST1SXY /f/Working Section/2014 Fall/Group Meeting/Example
 (master)
$ git add README.txt

Administrator@WINXVI71OST1SXY /f/Working Section/2014 Fall/Group Meeting/Example
 (master)
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   README.txt
```

Committing

Once files are staged:

$ git commit -m "..."

```
Administrator@WINXVI710ST1SXY /f/Working Section/2014 Fall/Group Meeting/Example
 (master)
$ git commit -m "README"
[master 61b0bbd] README
 1 file changed, 1 insertion(+)
 create mode 100644 README.txt

Administrator@WINXVI710ST1SXY /f/Working Section/2014 Fall/Group Meeting/Example
 (master)
$ git status
On branch master
nothing to commit, working directory clean
```

Differences

By default, HEAD points to the most recent commit.

- $ git diff HEAD
- $ git diff --staged

```
Administrator@WINXVI71OST1SXY /f/Working Section/2014 Fall/Group Meeting/Example
 (master)
$ git add README.txt

Administrator@WINXVI71OST1SXY /f/Working Section/2014 Fall/Group Meeting/Example
 (master)
$ git diff HEAD
diff --git a/README.txt b/README.txt
index 69de6b8..1d2ff62 100644
--- a/README.txt
+++ b/README.txt
@@ -1 +1 @@
-This is an example.
\ No newline at end of file
+This is not an example.
\ No newline at end of file
```

Remote

Add a remote sever, like GitHub:

- $ git remote add origin ...
- $ git push -u origin master

```
Administrator@WINXVI71OST1SXY /f/Working Section/2014 Fall/Group Meeting/Example
 (master)
$ git push -u origin master
Counting objects: 9, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (9/9), 681 bytes | 0 bytes/s, done.
Total 9 (delta 1), reused 0 (delta 0)
To git@github.ncsu.edu:cli9/Example.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
```

Working with SSH Key

Add the public key to GitHub account:

- $ ls -al ~/.ssh
- $ ssh-agent bash
- $ ssh-add ~/.ssh/id_rsa

```
[cli9@zhou-lnx codebase]$ ssh-agent bash
[cli9@zhou-lnx codebase]$ ssh-add ~/.ssh/id_rsa
Enter passphrase for /home/cli9/.ssh/id_rsa:
Identity added: /home/cli9/.ssh/id_rsa (/home/cli9/.ssh/id_rsa)
[cli9@zhou-lnx codebase]$
```

History

Check history: $ git log

```
$ git log
commit 785e2a48cbb2d99754fdaf06930f9bbedba977af
Author: cli9 <cli9@ncsu.edu>
Date:   Fri Oct 3 18:00:41 2014 -0400

    Modify code

commit 3c0f8ceff1a16da93f05cf00ade562347027bea9
Author: cli9 <cli9@ncsu.edu>
Date:   Fri Oct 3 17:58:53 2014 -0400

    Add code

commit afc970efb22324deb2cc624e177c69db3de12b52
Author: cli9 <cli9@ncsu.edu>
Date:   Fri Oct 3 14:26:09 2014 -0400

    Modified README

commit 61b0bbdfc15179a38d1813cd508fcc3db2205bc4
Author: cli9 <cli9@ncsu.edu>
Date:   Fri Oct 3 14:18:46 2014 -0400

    README
```

Undo

- $ git checkout -- ...: go back to the point since the last commit.
- $ git reset ...: unstage files.
- $ git rm ...: different from $ rm ....

Goal

- Move "codebase" from the github.ncsu.edu to github.com.
- Don't move other folders in the repository.
- Preserve the Git commit history for the directory we are moving.

From Here (Step 1)

Get files ready for the move:

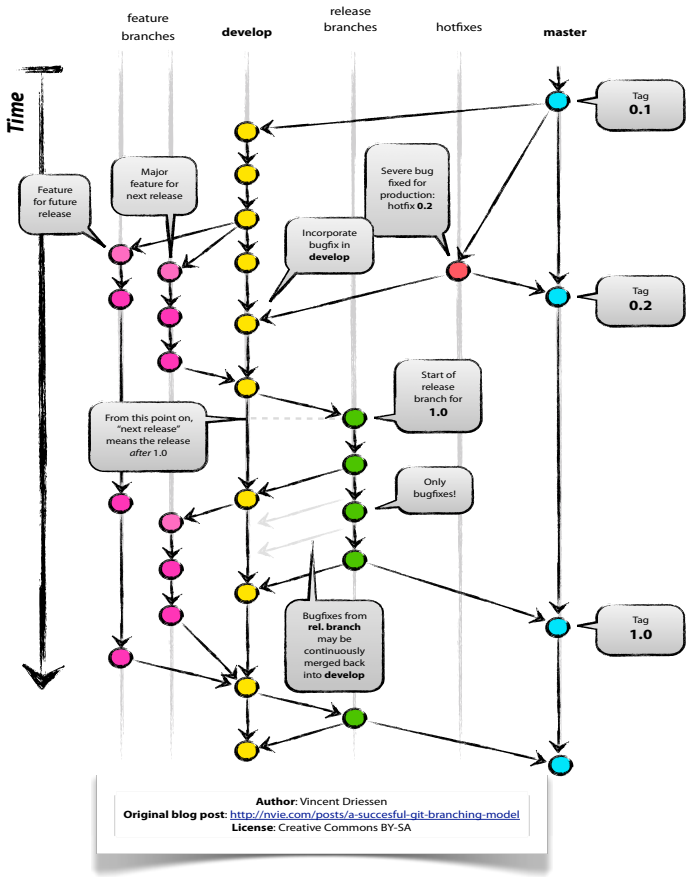
- # Don't mess up the original repo
- $ git clone git@github.ncsu.edu:cli9/Example.git
- $ cd Example
- # Don't affect remote server
- $ git remote rm origin
- # Only keep the codebase folder
- $ git filter-branch --subdirectory-filter codebase -- --all
- $ mkdir codebase
- $ git add .
- $ git commit

To There (Step 2)

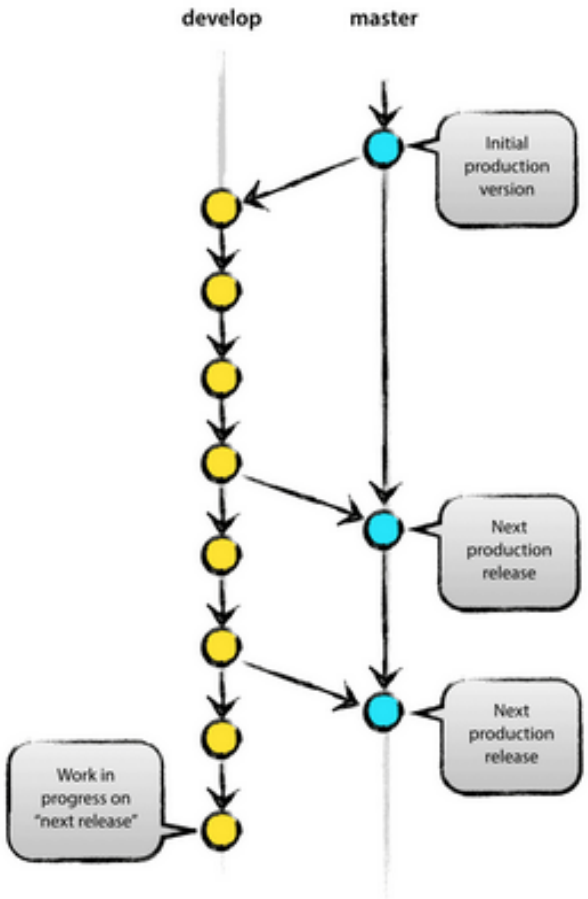
Merge files into new repository:

- # Don't mess up the original repo
- $ git clone git@github.com:cli9/Example-Pub.git
- $ cd Example-Pub
- # Add a remote repo
- $ git remote add repo-A-branch ./Example
- # Merge!
- $ git pull repo-A-branch master
- $ git remote rm repo-A-branch
- # Push to github.com
- $ git push

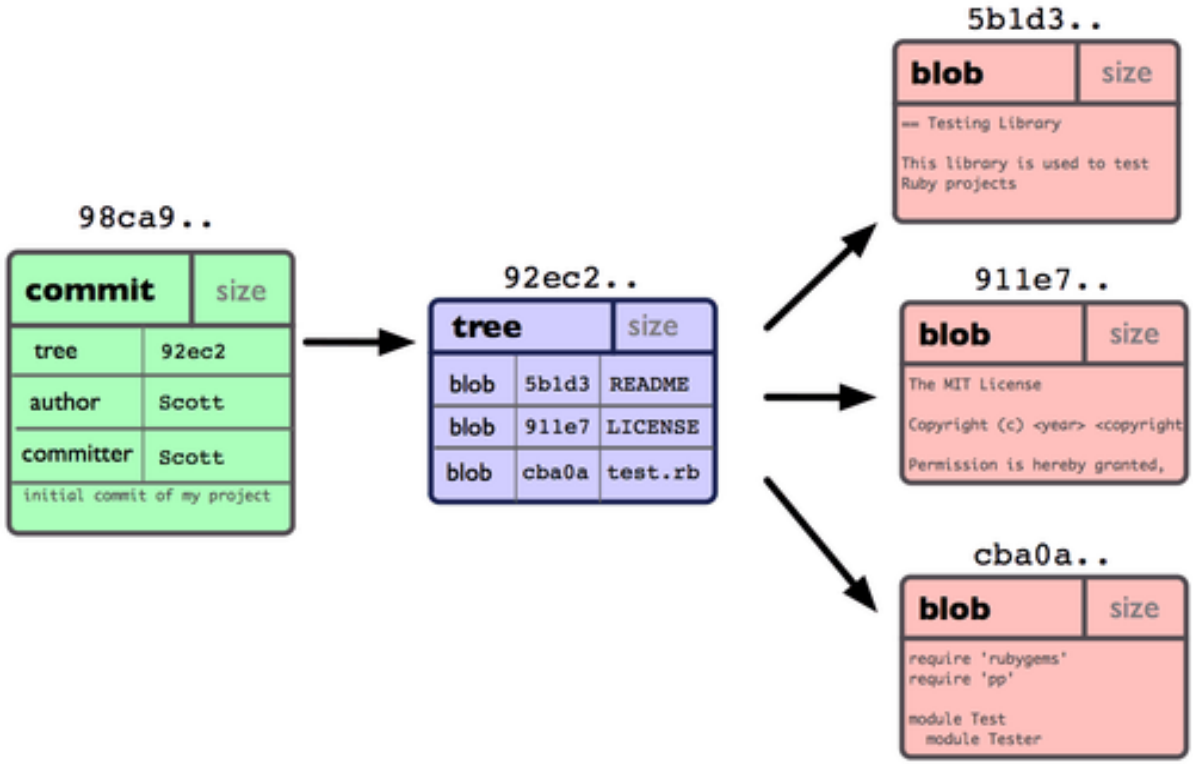
Branching Model



Simplified Version

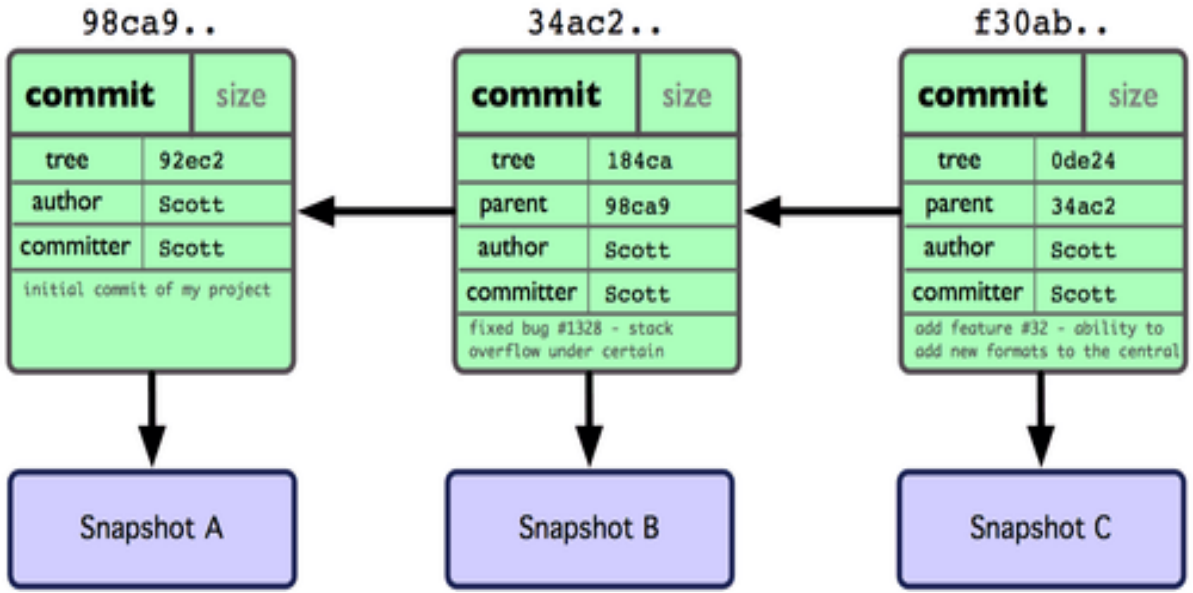


Getting to Know Commit



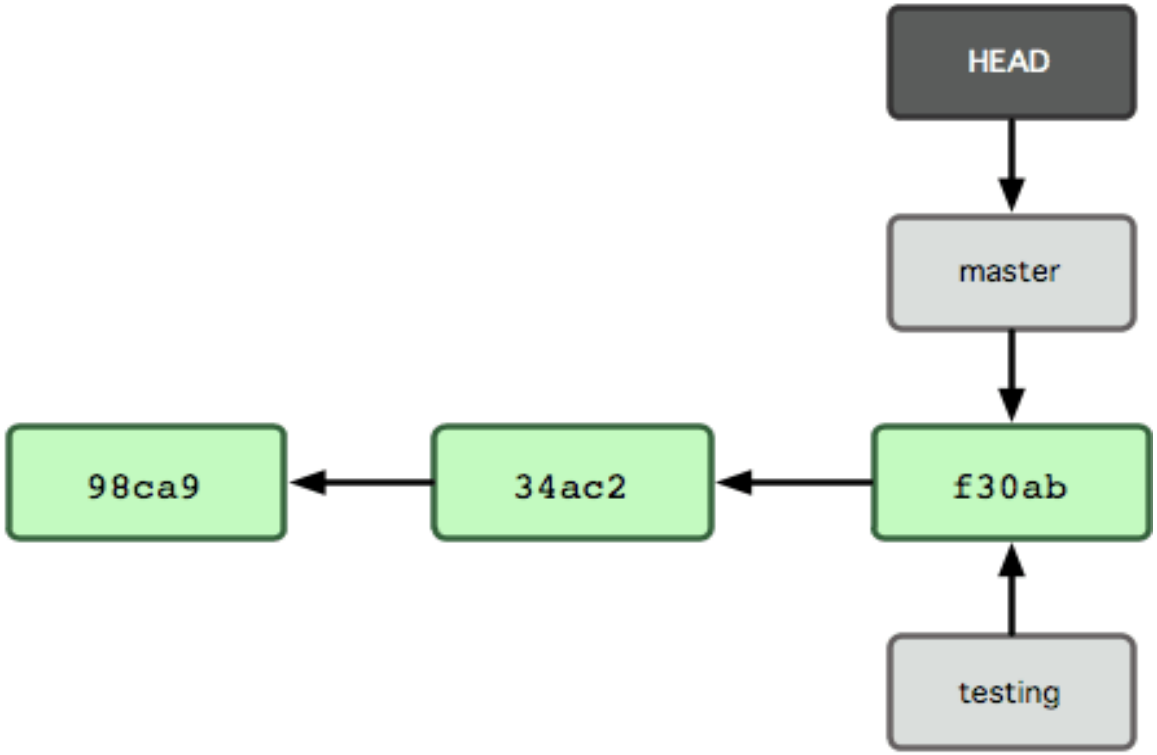
- Commit object has the metadata and a pointer to the root of project tree object so it can re-create that snapshot when needed.

Multiple Commits



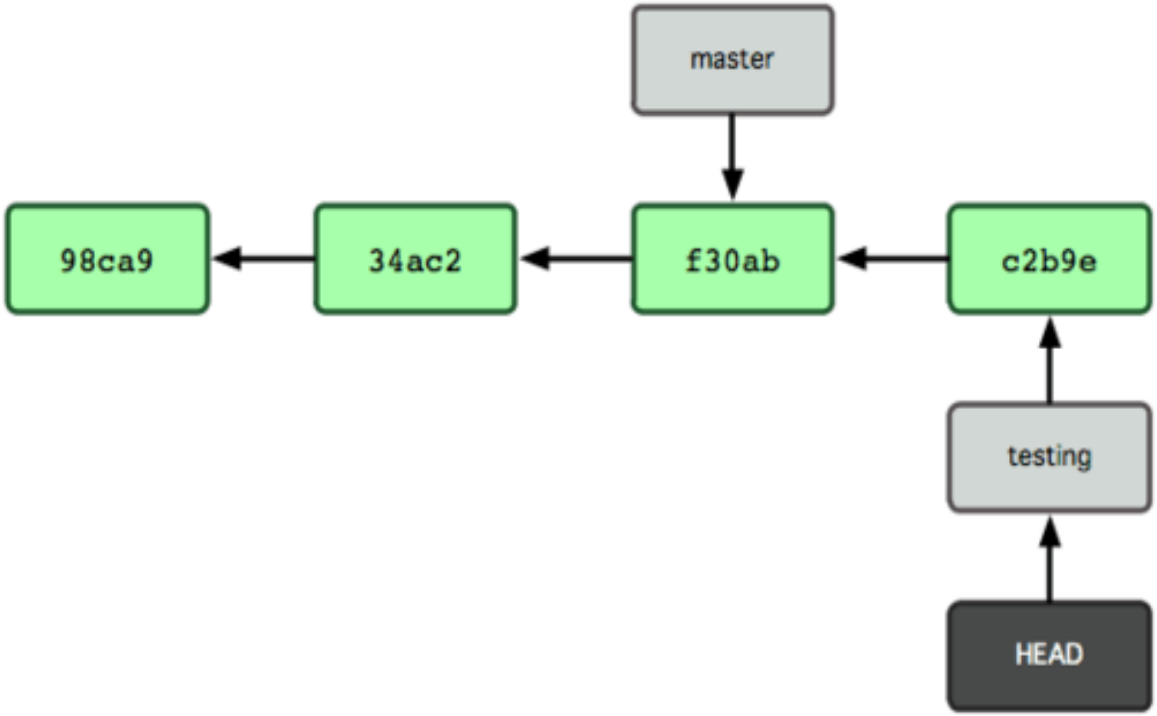
- The next commit stores a pointer to the commit that came immediately before it.

What is Branch



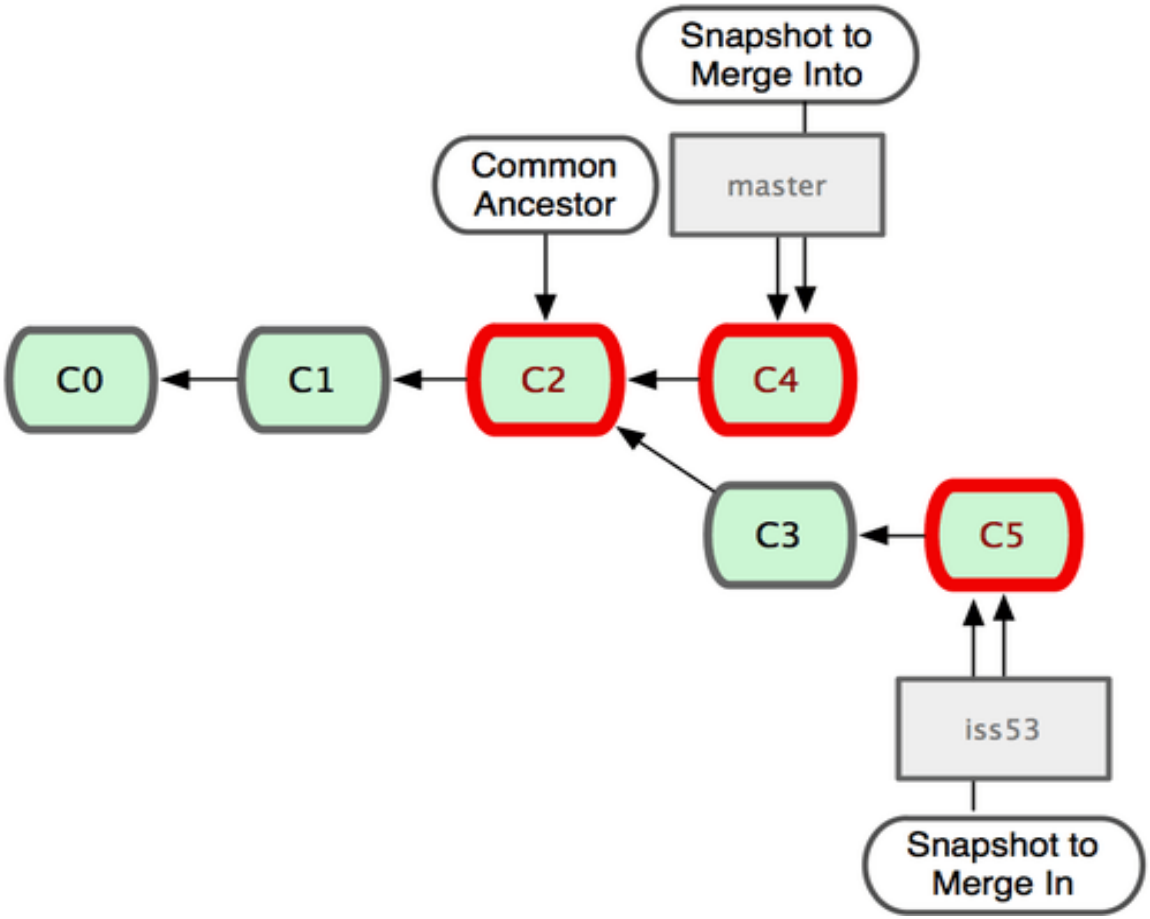
- HEAD is a pointer to the local branch youre currently on.
- Branch is a lightweight movable pointer (cheap).

One Ahead of Another

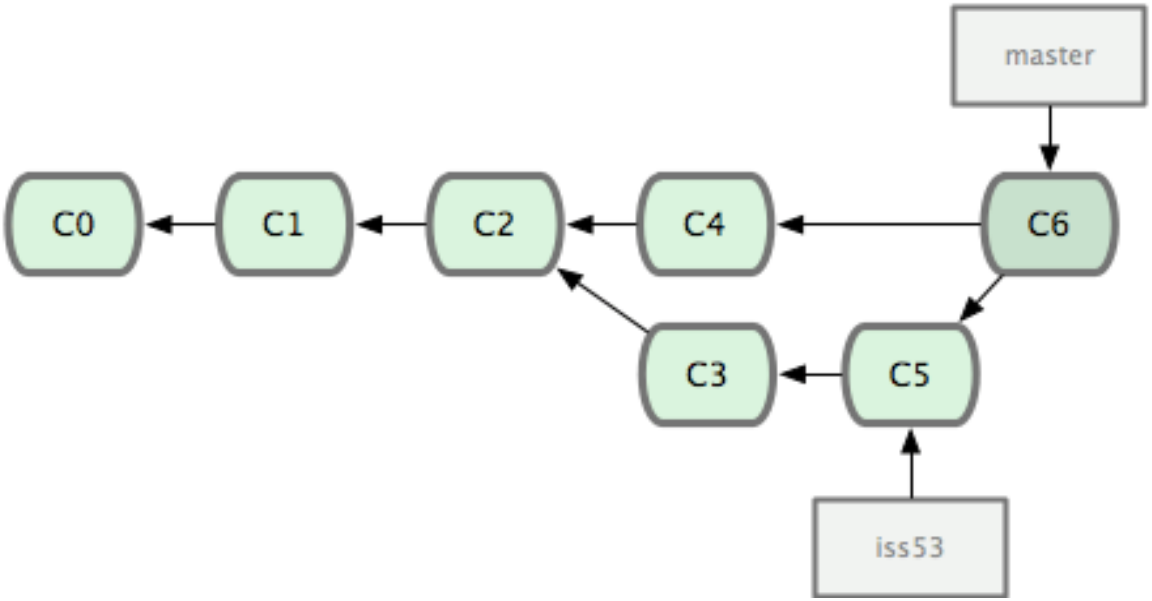


- Fast-forward merge: move the pointer forward.

Divergent Commits

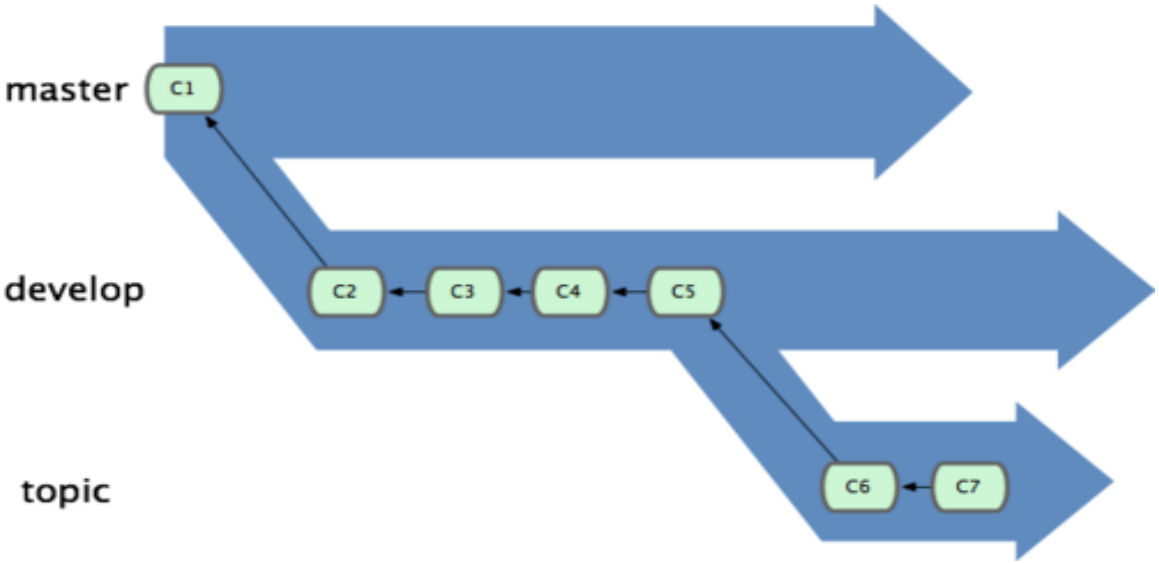


How Does Merge Work

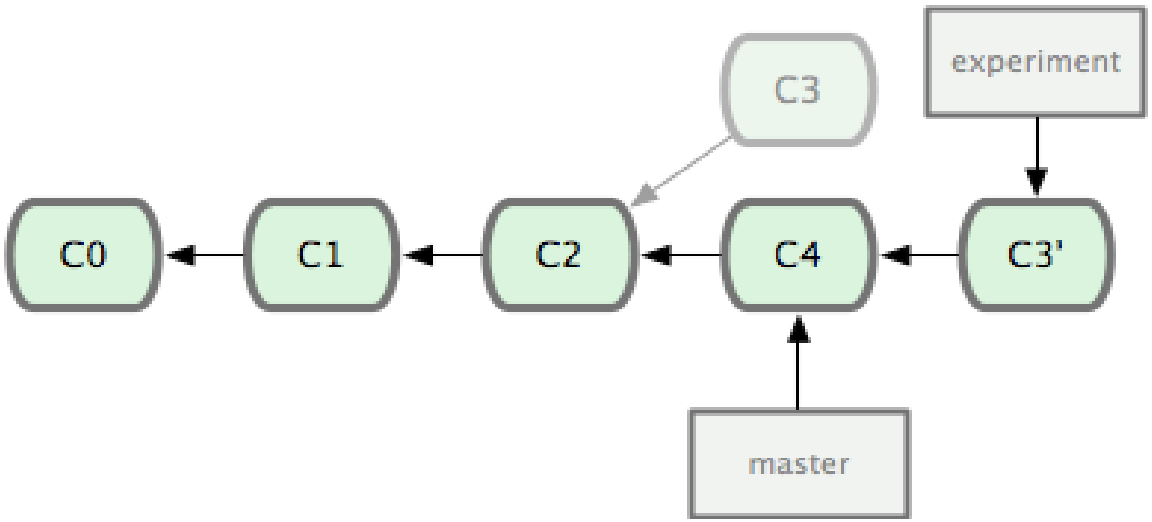


- Three-way merge made by the 'recursive' strategy.
- New commit has more than one parent.
- Automatically.

Branching Workflows



Another Way to Go



- Rebase: take the patch of the change introduced in C3 and reapply it on top of C4.
- Followed by fast-forward merge.
- Different history, same results.

Branching Out

- $ git branch ...: start a new branch.
- $ git checkout ...: switch to it.
- $ git branch: check the current branch.

```
[cli9@zhou-lnx Example-Pub]$ git checkout master
Switched to branch 'master'
[cli9@zhou-lnx Example-Pub]$ git branch
  develop
  gh-pages
* master
[cli9@zhou-lnx Example-Pub]$
```

Master Branch

There is a bug in version 2.0:

```
[cli9@zhou-lnx codebase]$ echo "This is a bug" >> bug.txt
[cli9@zhou-lnx codebase]$ ls
bug.txt  code.txt
[cli9@zhou-lnx codebase]$ git tag
v1.0
v2.0
[cli9@zhou-lnx codebase]$
```

Develop Branch

Sync with master branch:

```
[cli9@zhou-lnx codebase]$ git checkout develop
Switched to branch 'develop'
[cli9@zhou-lnx codebase]$ git pull origin master
From github.com:cli9/Example-Pub
 * branch            master     -> FETCH_HEAD
Updating cadd8be..7f14db5
Fast-forward
 codebase/bug.txt |    1 +
 1 files changed, 1 insertions(+), 0 deletions(-)
 create mode 100644 codebase/bug.txt
[cli9@zhou-lnx codebase]$ ls
bug.txt  code.txt
```

Debugging

Debug in develop branch, now it's ahead of master branch:

```
* develop
  gh-pages
  master
[cli9@zhou-lnx codebase]$ git rm bug.txt
rm 'codebase/bug.txt'
[cli9@zhou-lnx codebase]$ ls
code.txt
[cli9@zhou-lnx codebase]$ git status
# On branch develop
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       deleted:    bug.txt
#
[cli9@zhou-lnx codebase]$ git commit -m "Debug"
[develop 0dc1057] Debug
 1 files changed, 0 insertions(+), 1 deletions(-)
 delete mode 100644 codebase/bug.txt
```

Merging

Bug fixed for a new release version:

- $ git merge branch ...
- $ git pull origin ...

```
[cli9@zhou-lnx codebase]$ git checkout master
Switched to branch 'master'
[cli9@zhou-lnx codebase]$ ls
bug.txt  code.txt
[cli9@zhou-lnx codebase]$ git pull origin develop
From github.com:cli9/Example-Pub
 * branch            develop    -> FETCH_HEAD
Updating 7f14db5..0dc1057
Fast-forward
 codebase/bug.txt |    1 -
 1 files changed, 0 insertions(+), 1 deletions(-)
 delete mode 100644 codebase/bug.txt
[cli9@zhou-lnx codebase]$ ls
code.txt
[cli9@zhou-lnx codebase]$ git push origin master
Total 0 (delta 0), reused 0 (delta 0)
To git@github.com:cli9/Example-Pub.git
   7f14db5..0dc1057  master -> master
```

Tagging

Tag a new release:

```
[cli9@zhou-lnx codebase]$ git tag v3.0
[cli9@zhou-lnx codebase]$ git tag
v1.0
v2.0
v3.0
[cli9@zhou-lnx codebase]$ git show v3.0
commit 0dc1057a2416e606c391f48b7d46c59982715423
Author: cli9 <cli9@ncsu.edu>
Date:   Sat Oct 4 13:37:40 2014 -0400

    Debug

diff --git a/codebase/bug.txt b/codebase/bug.txt
deleted file mode 100644
index 6685723..0000000
--- a/codebase/bug.txt
+++ /dev/null
@@ -1 +0,0 @@
-This is a bug
[cli9@zhou-lnx codebase]$ git push origin v3.0
Total 0 (delta 0), reused 0 (delta 0)
To git@github.com:cli9/Example-Pub.git
 * [new tag]         v3.0 -> v3.0
[cli9@zhou-lnx codebase]$
```

Gh-pages Branch

- $ git checkout gh-pages
- $ git rm -rf .
- $ git add index.html
- $ git commit -a -m "First pages commit"
- $ git push origin gh-pages

Git Cheating Sheet

- Identity:
  - $ git config --global user.name ...: identify user name.
  - $ git config --global user.email ...: identify user email.
- Set up a repo:
  - $ git init: start a local repo.
  - $ git clone ...: clone a repo from local or remote repo.
- Add and Commit:
  - $ git add ...: stage files.
  - $ git add -A: stage all.
  - $ git stash: stash the changes away.
  - $ git commit -m "...": commit changes.
- Undo:
  - $ git reset ...: unstage files.
  - $ git checkout -- ...: restore changes.
  - $ git rm ...: delete files.

Git Cheating Sheet (Cont'd)

- Check and Inspection:
  - $ git status: status of files.
  - $ git diff HEAD: display differences.
  - $ git diff --staged: display differences of staged files.
- Remote:
  - $ git remote add origin ...: connect to remote repo.
  - $ git remote rm origin ...: disconnect to remote repo.
  - $ git push -u origin ...: push changes to remote repo.
  - $ git pull: update local repo with remote changes.
  - $ git fetch: download objects from remote repo.

Git Cheating Sheet (Cont'd)

- Branch and Merge:
  - $ git branch ...: create a new branch.
  - $ git push origin ...: add a remote branch.
  - $ git checkout ...: switch to a branch.
  - $ git checkout -b ...: create a new branch and switch to it.
  - $ git branch -d ...: delete a branch.
  - $ git push origin --delete ...: delete a remote branch.
  - $ git merge ...: merge changes from another branch.
  - $ git rebase ... : merge changes from another branch.
  - $ git pull origin ...: merge changes from remote branch.
- History and Tag
  - $ git tag ...: create a tag for release version.
  - $ git show ...: display information of a version.
  - $ git push origin ...: add a tag to remote repo.
  - $ git log: show history of commits.

Books and Resources

- Chacon, Scott (2009). *Pro Git*. New York: Apress.
  Free from here: (http://git-scm.com/book)
- Transfer a subdirectory:
  (http://gbayer.com/development/moving-files-from-one-git-repository-to-another-preserving-history/)
- A successful Git branching model:
  (http://nvie.com/posts/a-successful-git-branching-model/)
- Try Git: (https://try.github.io/)
- Git Real: (https://www.codeschool.com/courses/git-real)
- Git Real 2: (https://www.codeschool.com/courses/git-real-2)
- GitGuys: (http://www.gitguys.com/)