

Wrangling Git

Scott Chacon

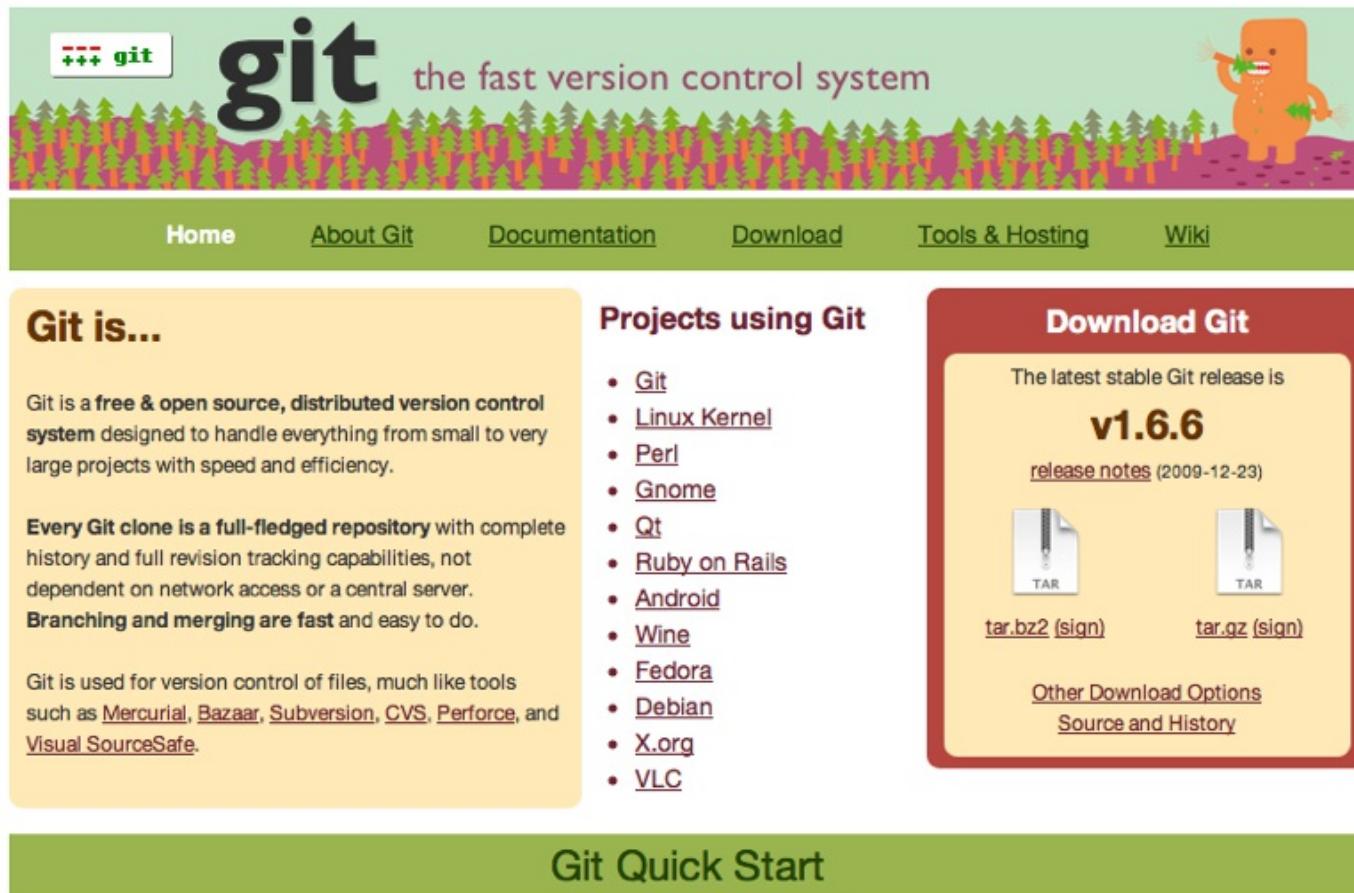
Me

Scott Anthony Chacon



github
SOCIAL CODING

<http://git-scm.com>



The screenshot shows the official website for Git. At the top, there's a green header bar with the Git logo and the tagline "the fast version control system". Below the header is a navigation bar with links for Home, About Git, Documentation, Download, Tools & Hosting, and Wiki. The main content area has three main sections: "Git is...", "Projects using Git", and "Download Git". The "Git is..." section contains text about Git being a free & open source, distributed version control system. The "Projects using Git" section lists various projects that use Git, including Git, Linux Kernel, Perl, Gnome, Qt, Ruby on Rails, Android, Wine, Fedora, Debian, X.org, and VLC. The "Download Git" section features a large red button for "Download Git v1.6.6" with release notes from 2009-12-23, and links for TAR and TAR.gz file downloads.

git the fast version control system

Home About Git Documentation Download Tools & Hosting Wiki

Git is...

Git is a free & open source, distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Every Git clone is a full-fledged repository with complete history and full revision tracking capabilities, not dependent on network access or a central server. Branching and merging are fast and easy to do.

Git is used for version control of files, much like tools such as [Mercurial](#), [Bazaar](#), [Subversion](#), [CVS](#), [Perforce](#), and [Visual SourceSafe](#).

Projects using Git

- [Git](#)
- [Linux Kernel](#)
- [Perl](#)
- [Gnome](#)
- [Qt](#)
- [Ruby on Rails](#)
- [Android](#)
- [Wine](#)
- [Fedora](#)
- [Debian](#)
- [X.org](#)
- [VLC](#)

Download Git

The latest stable Git release is
v1.6.6
[release notes](#) (2009-12-23)

 [tar.bz2 \(sign\)](#)  [tar.gz \(sign\)](#)

[Other Download Options](#)
[Source and History](#)

Git Quick Start

Git Reference

[Reference](#) [About](#) § [Site Source](#)

Getting and Creating Projects

- [init](#)
- [clone](#)

Basic Snapshotting

- [add](#)
- [status](#)
- [diff](#)
- [commit](#)
- [reset](#)
- [rm, mv](#)

Branching and Merging

- [branch](#)
- [checkout](#)
- [merge](#)
- [log](#)
- [tag](#)

Sharing and Updating Projects

- [fetch, pull](#)
- [push](#)
- [remote](#)

Inspection and Comparison

INTRODUCTION TO THE GIT REFERENCE

This is the Git reference site. This is meant to be a quick reference for learning and remembering the most important and commonly used Git commands. The commands are organized into sections of the type of operation you may be trying to do, and will preset the common options and commands needed to accomplish these common tasks.

Each section will link to the next section, so it can be used as a tutorial. Every page will also link to more in-depth Git documentation such as the official manual pages and relevant sections in the [Pro Git book](#), so you can learn more about any of the commands. First, we'll start with thinking about source code management like Git does.

HOW TO THINK LIKE GIT

This first thing that is important to understand about Git is that it thinks about version control very differently than Subversion or Perforce or whatever SCM you may be used to. It is often easier to learn Git by trying to forget your assumptions about how version control works and try to think about it in the Git way.

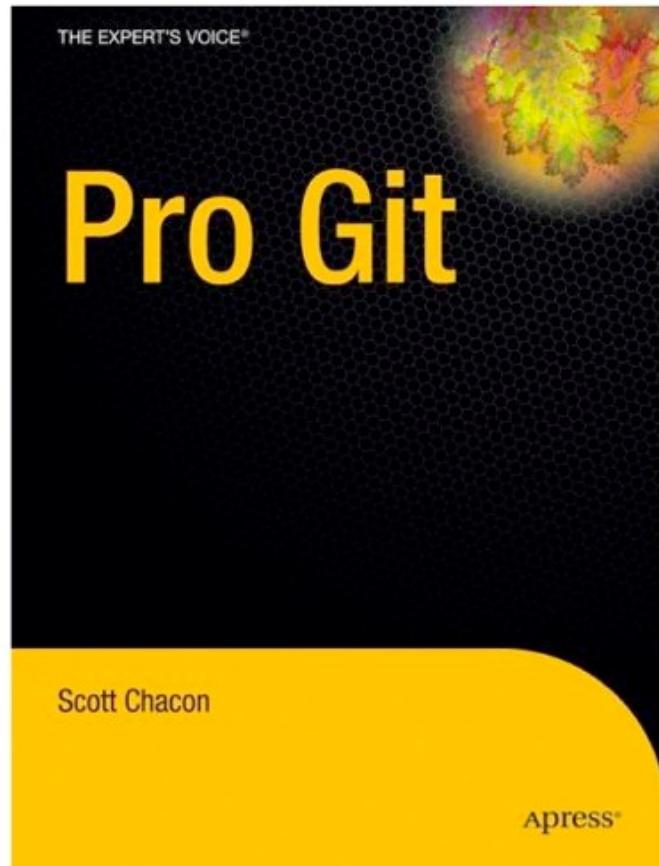
Let's start from scratch. Assume you are designing a new source code management system. How do you do basic version control before you used a tool for it? Chances are that you simply copied your project directory to save what it looked like at that point.

```
$ cp -R project project.bak
```

That way, you can easily revert files that get messed up later, or see what you have changed by comparing what the project looks like now to what it looked like when you copied it.

If you are really paranoid, you may do this often, maybe putting the date in the name of the backup:

<http://gitref.org>



<http://progit.org>

@chacon

schacon@gmail.com

</me>



Get the Tutorial Repo

```
git clone git://github.com/ghtraining/recipes.git
```

New Stuff

Colon Syntax

```
$ git show :/fix
```

```
# shows the last commit which has the  
word "fix" in its message
```

```
$ git show :/love
commit 1eee16988ed737b1805a5ed022bfa3f37dce8da5
Author: rick <technoweenie@gmail.com>
Date:   Wed Aug 11 20:43:57 2010 -0700

    love ie8

diff --git a/public/javascripts/github/editbar.js b/public/javascripts/github/editbar.js
index d461822..d6c2ad4 100644
--- a/public/javascripts/github/editbar.js
+++ b/public/javascripts/github/editbar.js
@@ -170,7 +170,7 @@ $(function(){
    var classes = $(this).attr('class').split(' ')
    var name = classes[0]
    var format = $('#guides .write select#wiki_format option')
-   if (classes.indexOf('gollum') == -1) {
+   if ($(this).hasClass('gollum')) {
        $('#editbar .sections .page.' + name + '.' + format).addClass('current')
    } else {
        $('#editbar .sections .page.' + name).addClass('current')
```

```
$ git show :/^Merge  
# shows the last merge commit
```

```
$ git show :/^Merge
commit 1549652ce43f07ad53baaa2ce4898a7df1a3d727
Merge: ec4b82b 3e92ceb
Author: Ryan Tomayko <rtomayko@gmail.com>
Date:   Thu Aug 12 04:05:27 2010 -0700

Merge branch 'locale-whole-hash-cache'
```

Group Fetching

```
$ git config remotes.mygroup 'remote1 remote2 ...'  
$ git fetch mygroup
```

Short Status

```
git status -s -b
```

```
$ git status -sb

## master...origin/master [ahead 6]
M ext/fsevent/fsevent_watch.c
A Makefile
?? SCEvents/
?? bin/fsevent_watch
```

Word Diffing

```
$ git diff
diff --git a/timer.js b/timer.js
index 4595967..8ba61dd 100644
--- a/timer.js
+++ b/timer.js
@@ -2,7 +2,7 @@ var timerSetUp = false;
var timerRunning = false;
var intervalRunning = false;
var seconds = 0;
-var totalMinutes = 45;
+var totalMinutes = 120;
```

```
$ git diff --word-diff
```

```
diff --git a/timer.js b/timer.js
index 4595967..8ba61dd 100644
--- a/timer.js
+++ b/timer.js
@@ -2,7 +2,7 @@ var timerSetUp = false;
var timerRunning = false;
var intervalRunning = false;
var seconds = 0;
var totalMinutes = [-45;-] {+120;+}
```

Git Notes

**for commenting on a commit
without changing the SHA**

```
$ git log
commit 30e367cef2203eba2b341dc9050993b06fd1e108
Author: Chris Wanstrath <chris@ozmm.org>
Date:   Sun Mar 30 20:50:08 2008 -0700
```

timeout code and tests

```
commit 5a0943123f6872e75a9b1dd0b6519dd42a186fda
Author: Chris Wanstrath <chris@ozmm.org>
Date:   Sun Mar 30 16:31:20 2008 -0700
```

add timeout protection to grit

```
$ git notes edit 30e367
```

```
#commit 30e367cef2203eba2b341dc9050993b06fd1e
#Author: Chris Wanstrath <chris@ozmm.org>
#Date:   Sun Mar 30 20:50:08 2008 -0700
#
#       timeout code and tests
~
~
~
"~/projects/grit2/.git/new-notes-30e367cef220
```

```
#commit 30e367cef2203eba2b341dc9050993b06fd1e
#Author: Chris Wanstrath <chris@ozmm.org>
#Date:   Sun Mar 30 20:50:08 2008 -0700
#
#       timeout code and tests
Bugzilla: #2143
~
~
~
"~/projects/grit2/.git/new-notes-30e367cef220
```

```
$ git log
commit 30e367cef2203eba2b341dc9050993b06fd1e108
Author: Chris Wanstrath <chris@ozmm.org>
Date:   Sun Mar 30 20:50:08 2008 -0700
```

 timeout code and tests

Notes:

Bugzilla: #2143

```
commit 5a0943123f6872e75a9b1dd0b6519dd42a186fda
Author: Chris Wanstrath <chris@ozmm.org>
Date:   Sun Mar 30 16:31:20 2008 -0700
```

 add timeout protection to grit

```
$ git notes show HEAD
Bugzilla: #2143
```

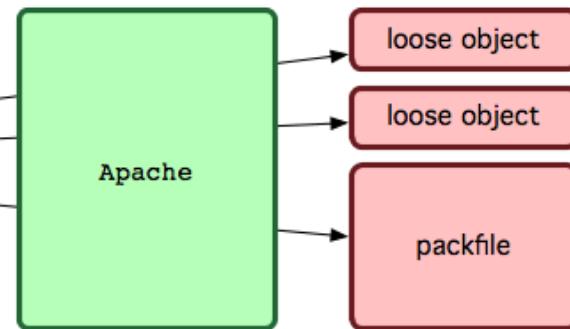
Smart HTTP Transport

"Dumb" HTTP

the user



your server

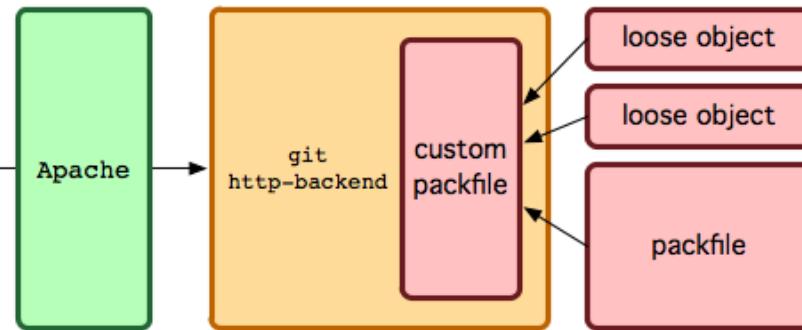


"Smart" HTTP

the user



your server



```
SetEnv GIT_PROJECT_ROOT /var/www/git
SetEnv GIT_HTTP_EXPORT_ALL
ScriptAlias /git/ /usr/libexec/git-core/git-http-backend/
```

Grack

<http://github.com/schacon/grack>

```
$ (edit config.ru to set git project path)
$ rackup --host 127.0.0.1 -p 8080 config.ru
$ git clone http://127.0.0.1:8080/schacon/grit.git
```

Old Stuff

Patch Staging

git add -p

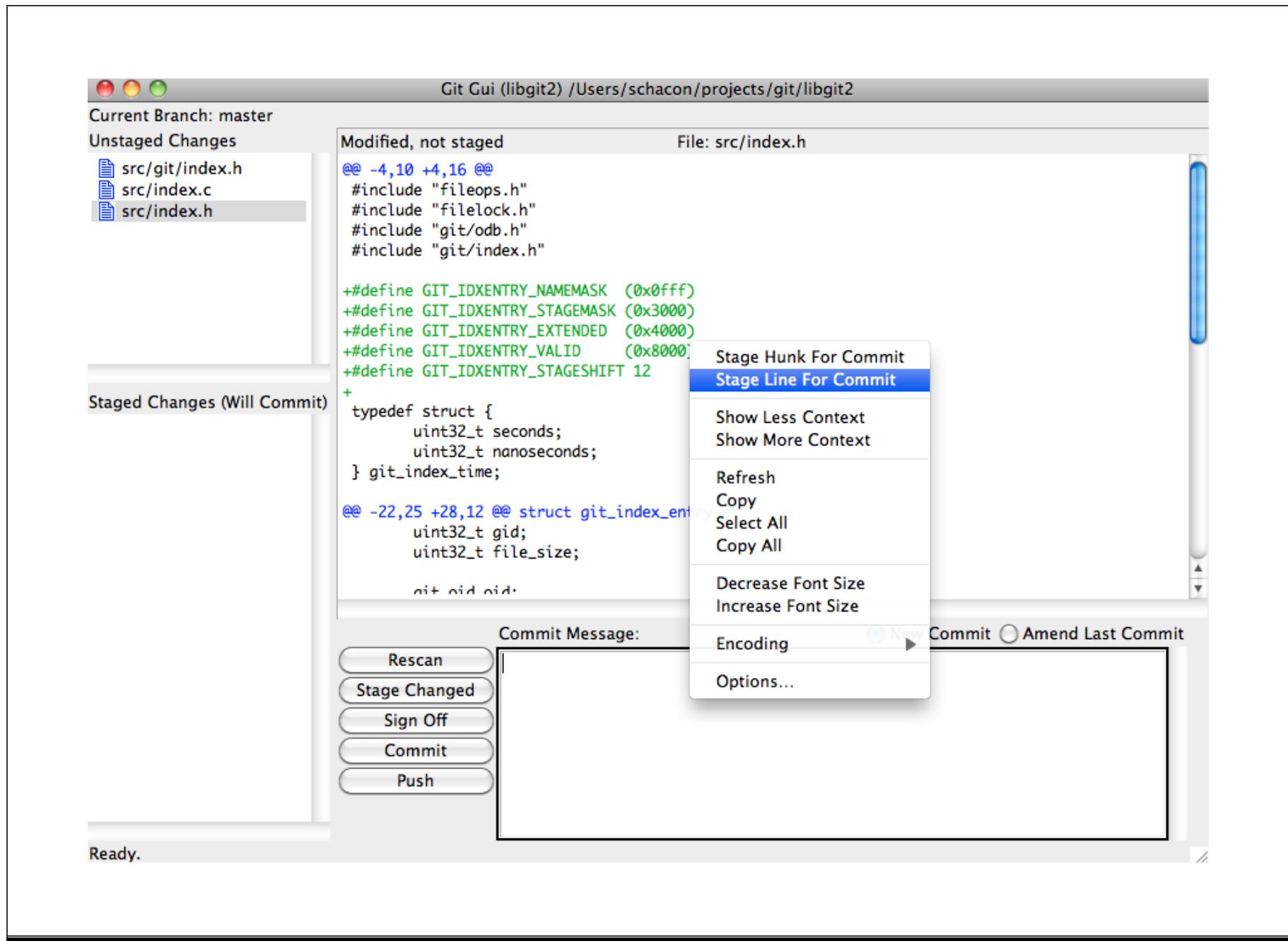
```
$ git add -p
diff --git a/src/index.h b/src/index.h
index 864af6d..7c106af 100644
--- a/src/index.h
+++ b/src/index.h
@@ -6,6 +6,12 @@
 #include "git/odb.h"
 #include "git/index.h"

+#define GIT_IDXENTRY_NAMEMASK (0xffff)
+#define GIT_IDXENTRY_STAGEMASK (0x3000)
+#define GIT_IDXENTRY_EXTENDED (0x4000)
+#define GIT_IDXENTRY_VALID (0x8000)
+#define GIT_IDXENTRY_STAGESHIFT 12
+
typedef struct {
    uint32_t seconds;
    uint32_t nanoseconds;
Stage this hunk [y,n,q,a,d/,j,J,g,e,?]?
```

```
# Manual hunk edit mode -- see bottom for a quick guide
@@ -6,6 +6,12 @@
 #include "git/odb.h"
 #include "git/index.h"

+#define GIT_IDXENTRY_NAMEMASK    (0xffff)
+#define GIT_IDXENTRY_STAGEMASK   (0x3000)
+#define GIT_IDXENTRY_EXTENDED    (0x4000)
+#define GIT_IDXENTRY_VALID       (0x8000)
+#define GIT_IDXENTRY_STAGESHIFT 12
+
typedef struct {
    uint32_t seconds;
    uint32_t nanoseconds;
# ---
# To remove '-' lines, make them ' ' lines (context).
# To remove '+' lines, delete them.
# Lines starting with # will be removed.
#
# If the patch applies cleanly, the edited hunk will immediately
# be marked for staging. If it does not apply cleanly, you will
# have an opportunity to edit again. If all lines of the hunk are
# then the edit is aborted and the hunk is left unchanged.
```

git gui



Git Gui (libgit2) /Users/schacon/projects/git/libgit2

Current Branch: master

Unstaged Changes

- src/git/index.h
- src/index.c
- src/index.h

Portions staged for commit File: src/index.h

```
@@ -4,11 +4,16 @@
 #include "fileops.h"
 #include "filelock.h"
 #include "git/odb.h"
 #include "git/index.h"

+#define GIT_IDXENTRY_NAMEMASK (0x0fff)
+#define GIT_IDXENTRY_STAGEMASK (0x3000)
+#define GIT_IDXENTRY_EXTENDED (0x4000)
#define GIT_IDXENTRY_VALID (0x8000)
#define GIT_IDXENTRY_STAGESHIFT 12

+
typedef struct {
    uint32_t seconds;
    uint32_t nanoseconds;
} git_index_time;

@@ -23,25 +28,12 @@
 struct git_index_entry {
    uint32_t gid;
    uint32_t file_size;

    git_oid oid;
}
```

Staged Changes (Will Commit)

- src/index.h

Commit Message:

New Commit Amend Last Commit

Rescan
Stage Changed
Sign Off
Commit
Push

Ready.

Tutorial

edit a file, changing 2 different lines, one at the top of the file and one at the bottom

stage and commit each change separately using
git add -p

do the same, but use git gui instead

use git log -p to make sure they were recorded seperately

Describing Commits

git describe

```
$ git describe HEAD  
v0.2.4-25-g8a3f93b
```

```
$ git describe HEAD@{1.month.ago}  
v0.2.4-6-gf51a8ba
```

```
$ git describe 9903167e0c638d0e134d7e23bd43e66d97a51401  
v0.1.4-46-g9903167
```

git name-rev

```
$ git name-rev 9903167e0c638d0e134d7e23bd43e66d97a51401  
9903167e0c638d0e134d7e23bd43e66d97a51401 tags/v0.2.0~4  
  
$ git name-rev --name-only 9903167e0c638d0e134d7e23bd43e66d97a  
tags/v0.2.0~4  
  
$ git name-rev --name-only --refs=refs/heads/* 9903167e0c638d0  
master~36  
  
$ git describe 9903167e0c638d0e134d7e23bd43e66d97a51401  
v0.1.4-46-g9903167
```

```
$ git config --global alias.human \
  "name-rev --name-only --refs=refs/heads/*"

$ git human 6507580497bd4ebc1c73373528d16a5608797ad0
master~2^2~1^2

$ git log --oneline --decorate --graph
* e6e8f33 (HEAD, tag: v0.2.5, master) updated to 0.2.5
* 8a3f93b stupid scott. messed up the results text color
* 944de04 Merge remote branch 'rosskaff/master'
|\ \
| * fc0e20c (rosskaff/master) Updgrade to jquery-1.4.2
* | aa55e9b Merge remote branch 'luniki/events'
| \ \
| * | 4f808d2 trigger custom events on showing, and jumping to
| |
* | 0d44711 Merge remote branch 'rick/master'
| \
| * 87df635 (luniki/master) Merge branch 'update_readme' of
| |
| * 6507580 Add links to my osbridge ganeti presentation
* | dd2357b added my talk to README
```

Git Under the Hood

The Git Database

```
$ git commit
```

```
Created commit 77d3001: descriptive commit message  
2 files changed, 4 insertions(+), 2 deletions(-)
```

```
$ git commit
```

```
Created commit 77d3001: descriptive commit message  
2 files changed, 4 insertions(+), 2 deletions(-)
```

77d3001

77d3001a1de6bf8f5e431972fe4d25b01e595c0b

77d3001a1de6bf8f5e431972fe4d25b01e595c0b

77d3001a1de6bf8f5e431972fe4d25b01e595c0b

commit	size
tree	c4ec5
parent	a149e
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	

77d3001a1de6bf8f5e431972fe4d25b01e595c0b

```
tree c4ec543b0322744e55c5efc9b6c4e449d398dbff
parent a149e2160b3f7573768cdc2fce24d0881f3577e1
author Scott Chacon <schacon@gmail.com> 1223402504 -0700
committer Scott Chacon <schacon@gmail.com> 1223402504 -0700
```

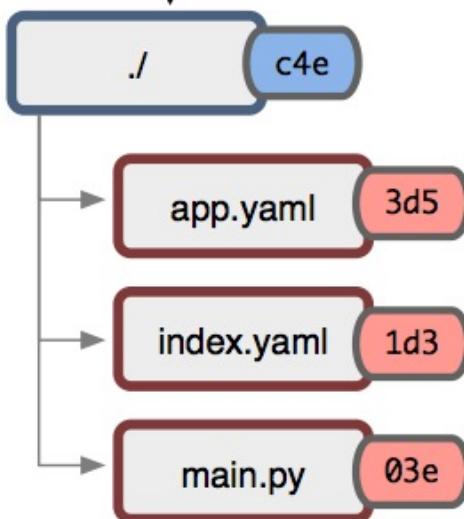
descriptive commit message

77d3001a1de6bf8f5e431972fe4d25b01e595c0b

commit	size
tree	c4ec5
parent	a149e
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	

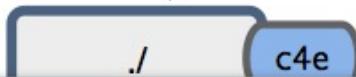
77d3001a1de6bf8f5e431972fe4d25b01e595c0b

commit	size
tree	c4ec5
parent	a149e
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	

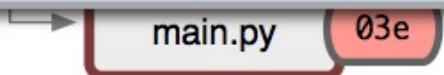


77d3001a1de6bf8f5e431972fe4d25b01e595c0b

commit	size
tree	c4ec5
parent	a149e
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	

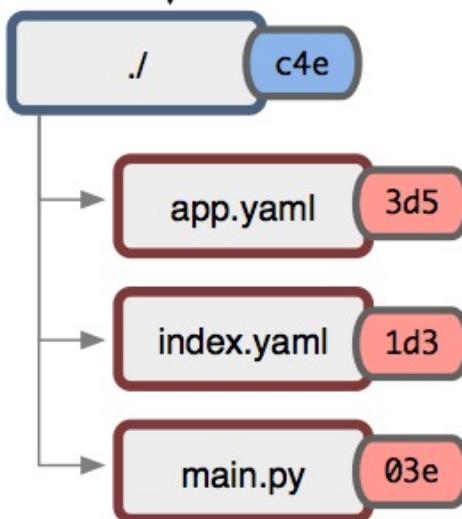


```
100644 blob 3d5cd3e1fc4424472ea247d1bb5fcfc3809aadab app.yaml
100644 blob 1d31bf2dba611ba0de871320b4d73cdc39cc862b index.yaml
100644 blob 03e68c28b73e2650bee34763369faf6e029d5053 main.py
```



77d3001a1de6bf8f5e431972fe4d25b01e595c0b

commit	size
tree	c4ec5
parent	a149e
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	

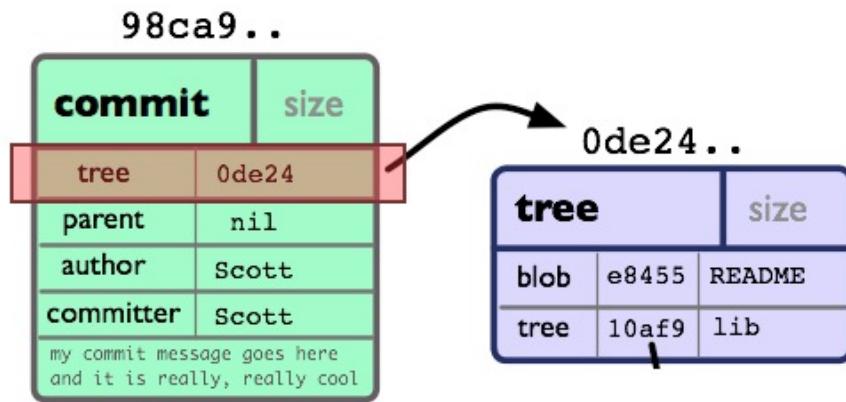


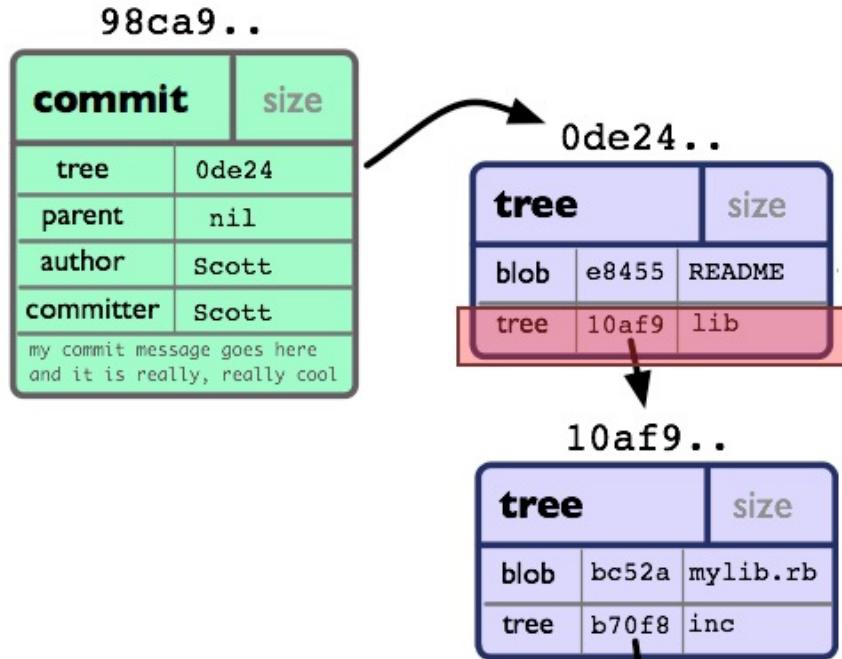
98ca9..

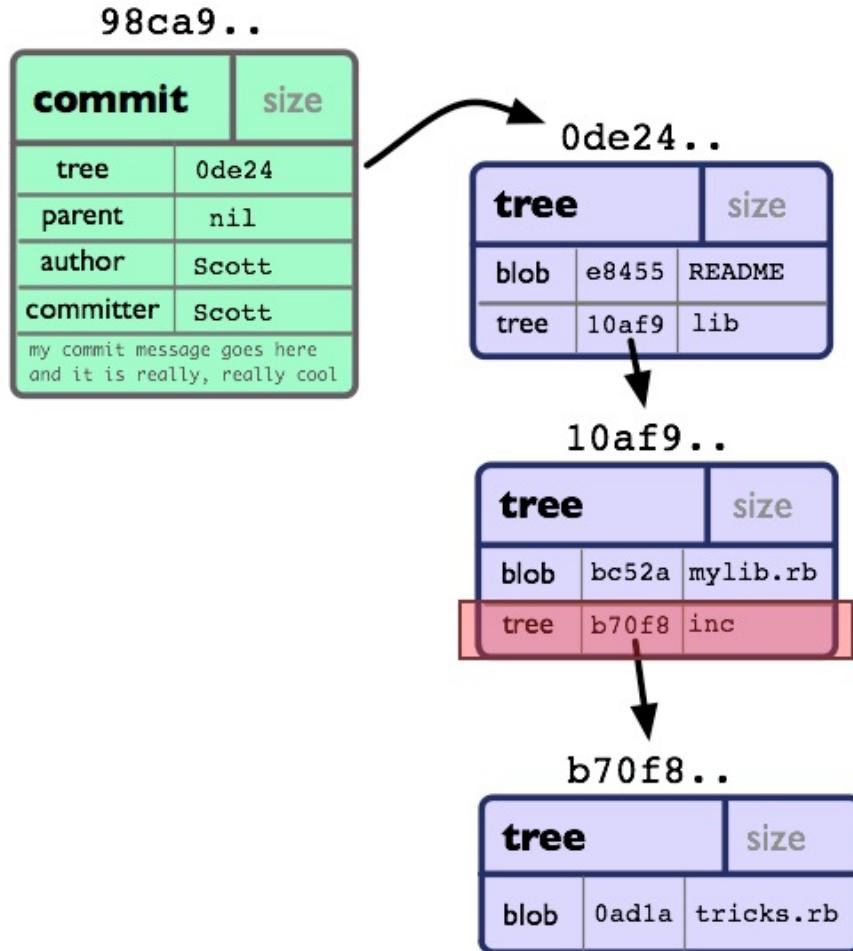
commit	size
tree	0de24
parent	nil
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	

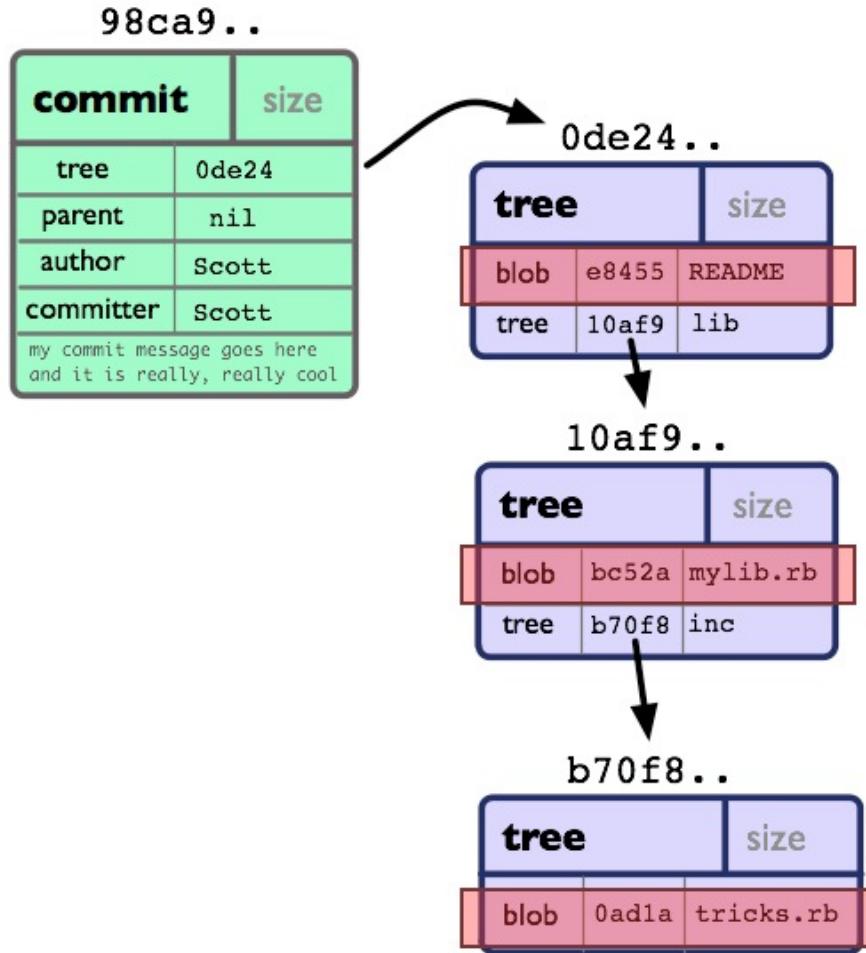
98ca9..

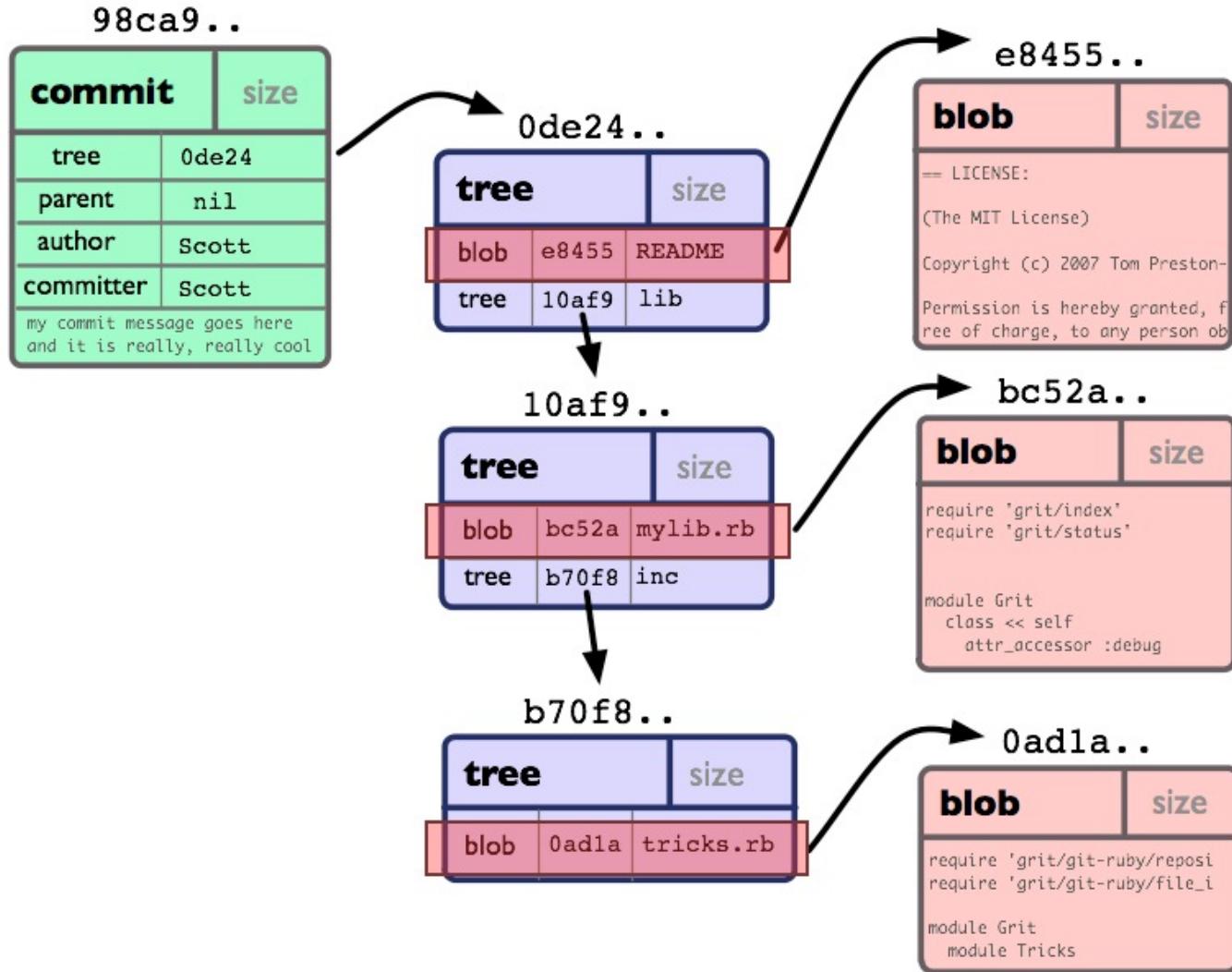
commit	size
tree	0de24
parent	nil
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	

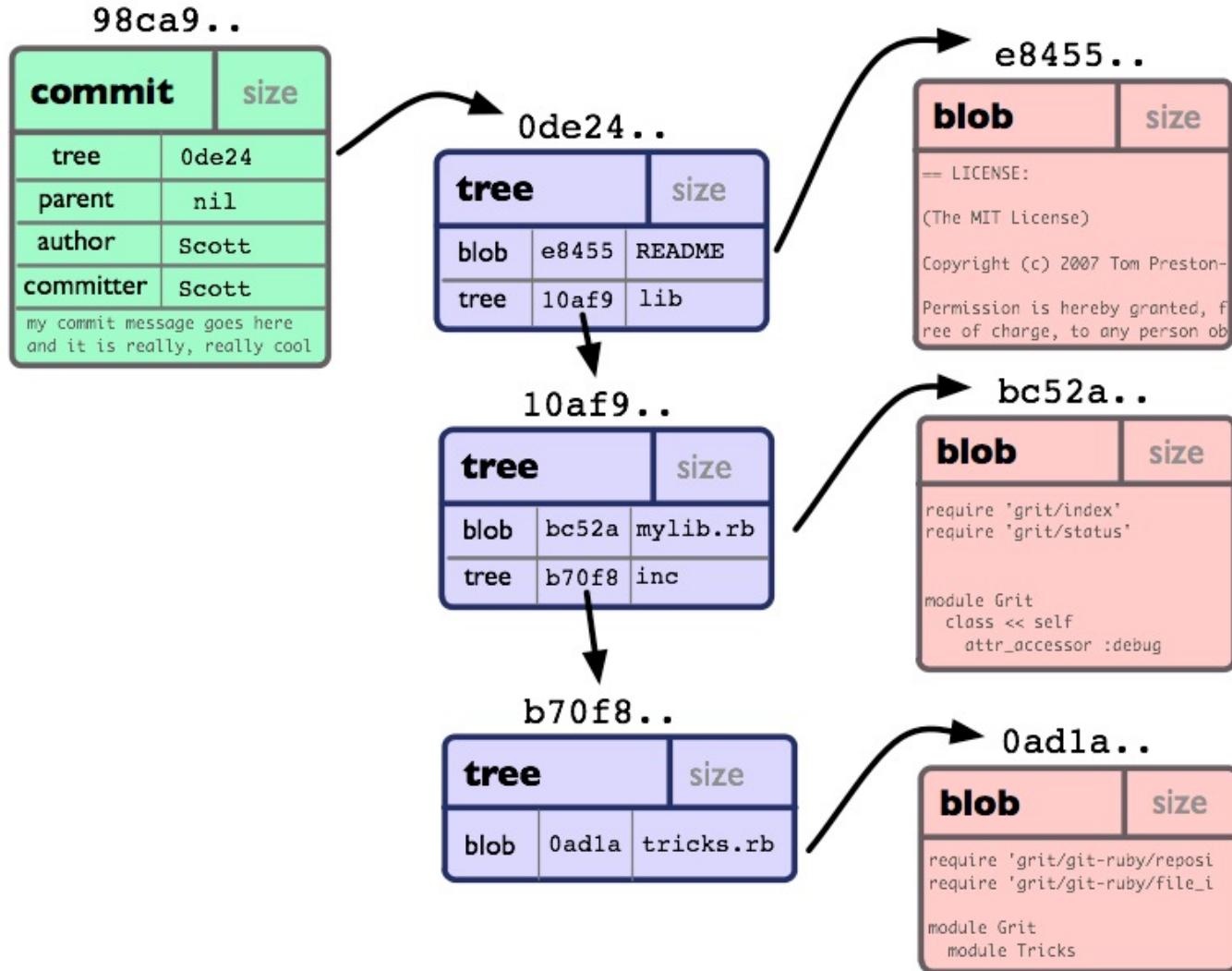






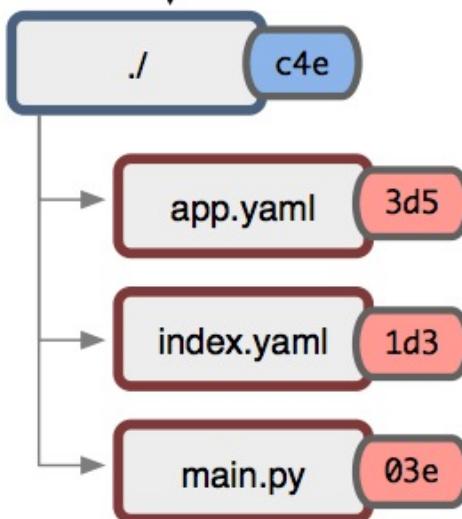






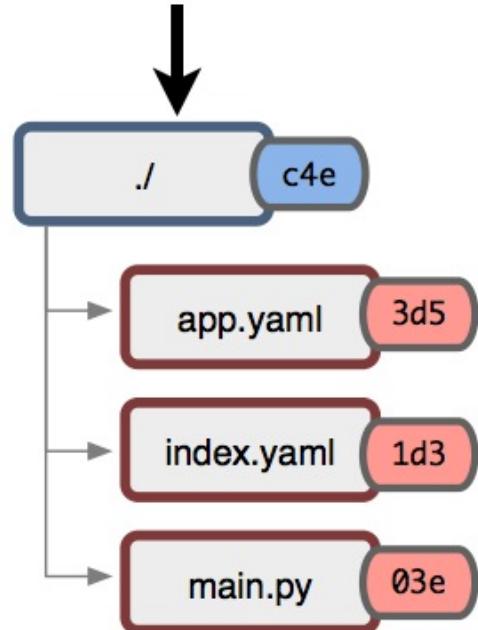
77d3001a1de6bf8f5e431972fe4d25b01e595c0b

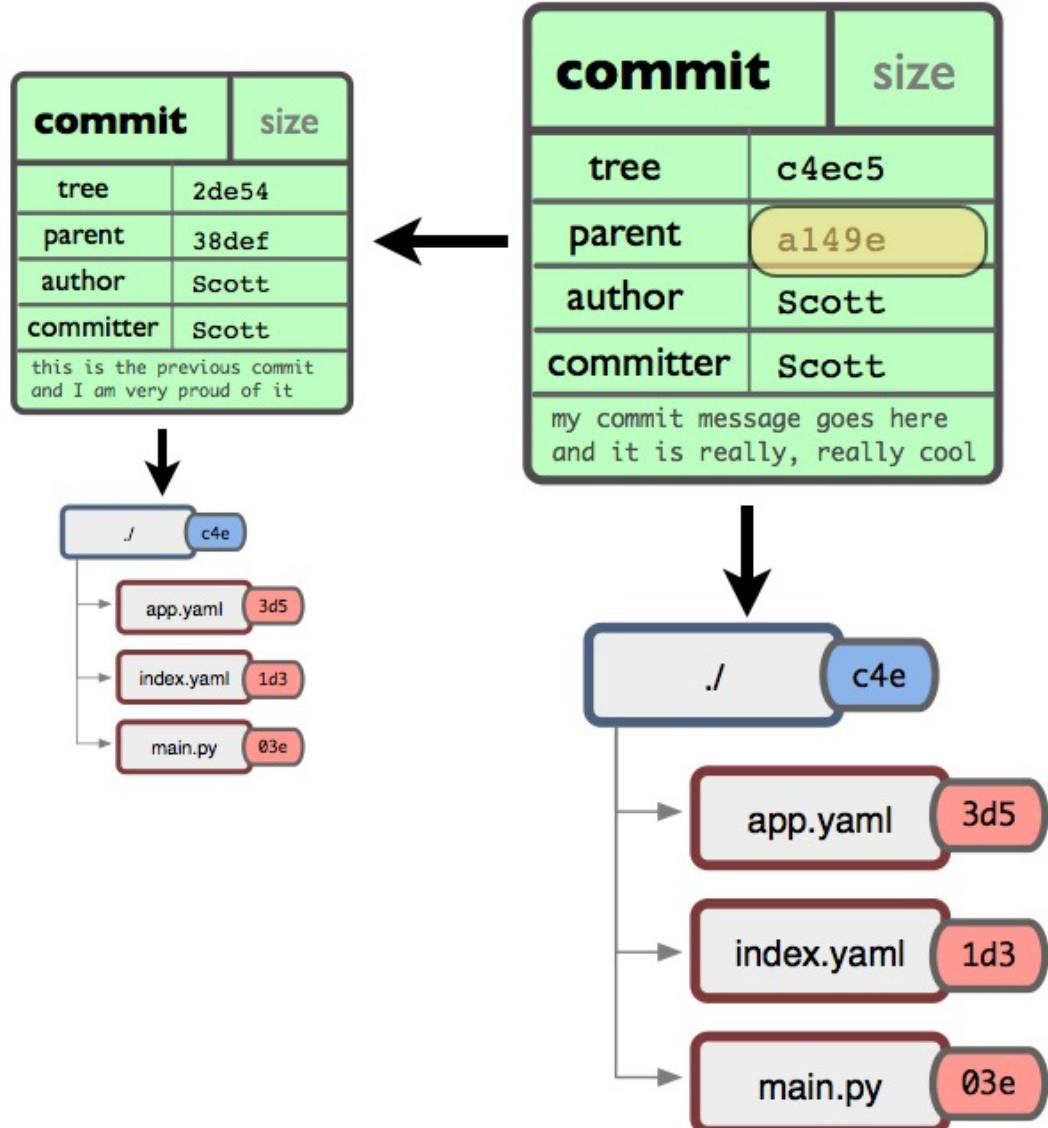
commit	size
tree	c4ec5
parent	a149e
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	

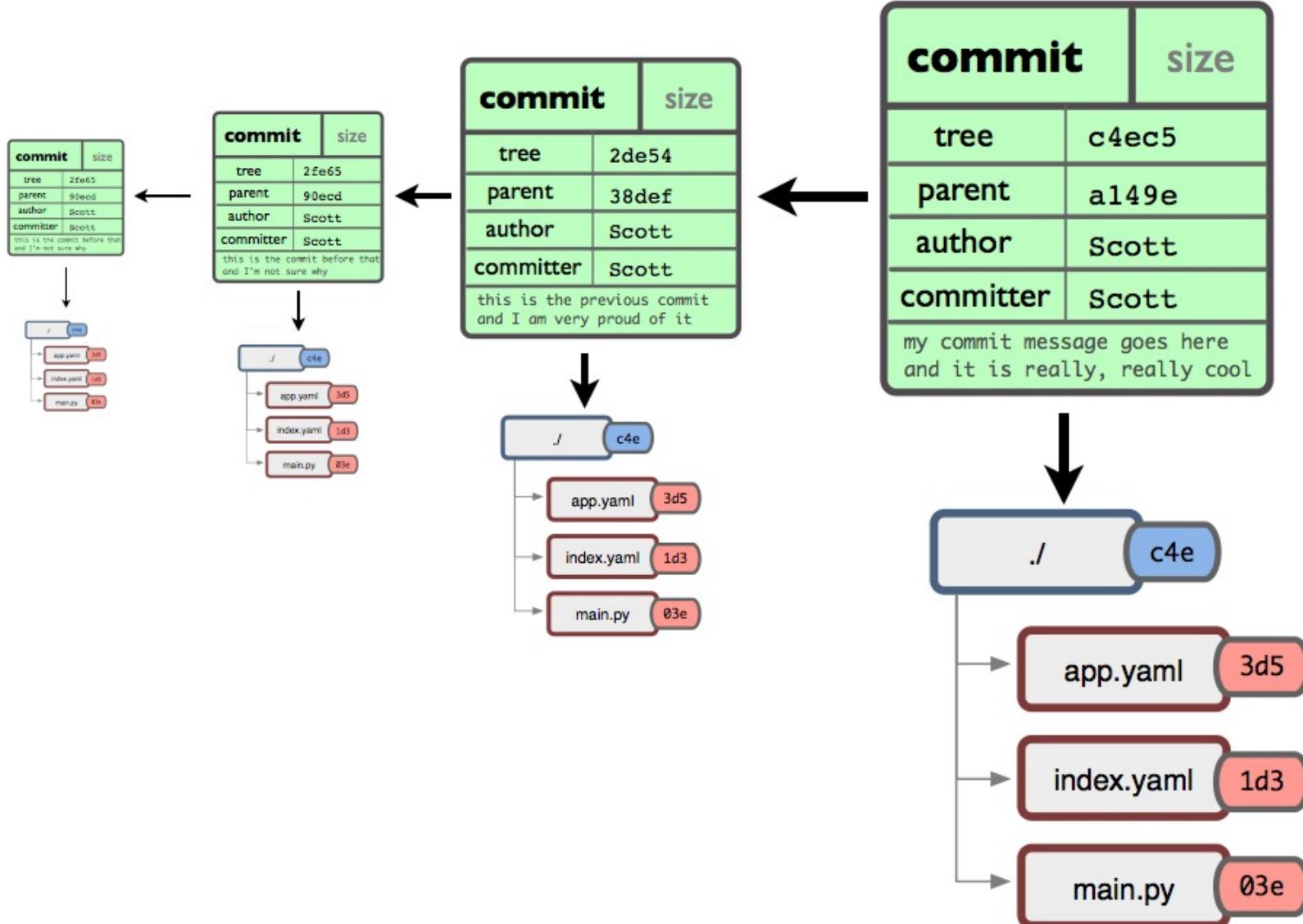


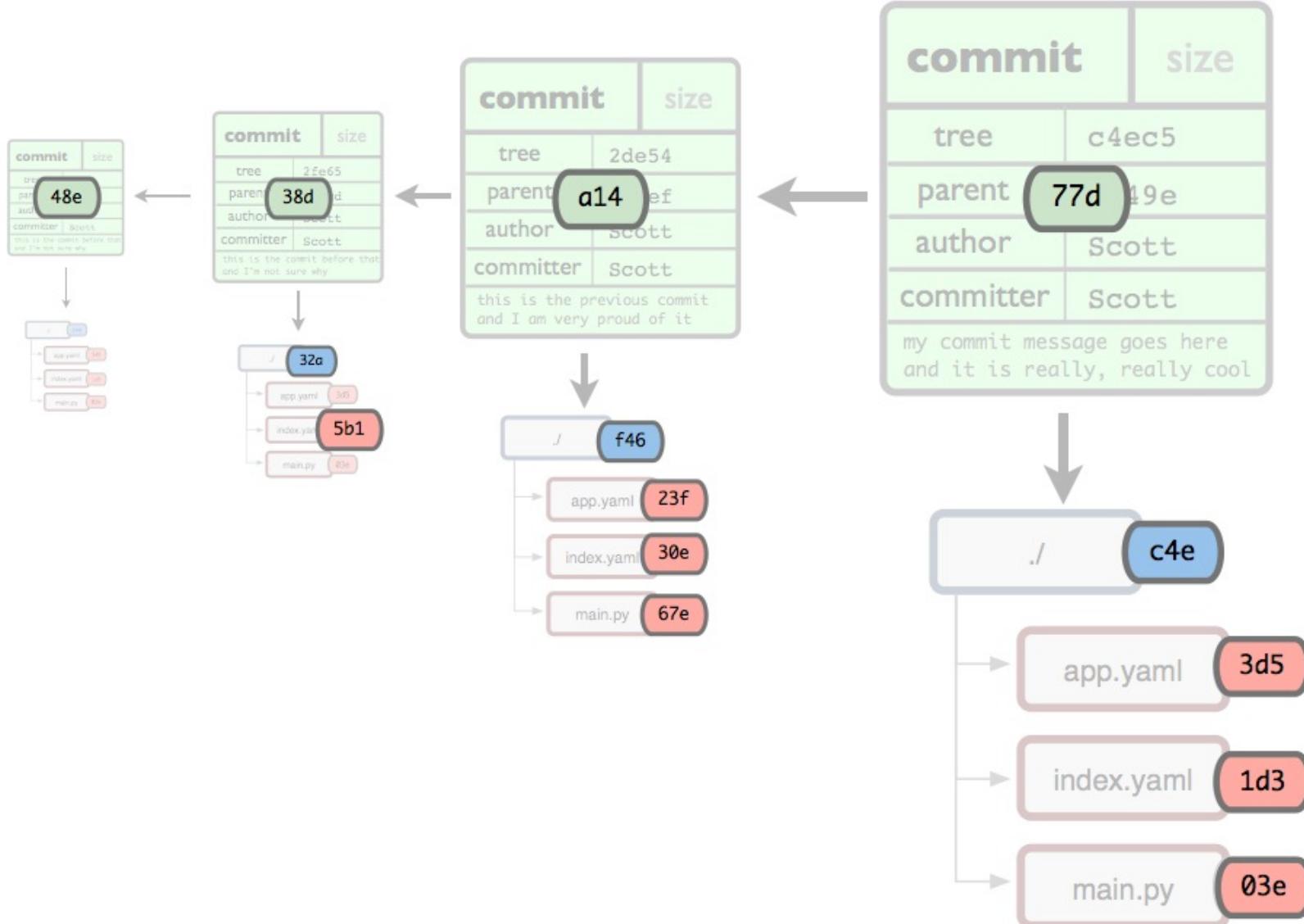
commit		size
tree	2de54	
parent	38def	
author	Scott	
committer	Scott	
this is the previous commit and I am very proud of it		

commit	size
tree	c4ec5
parent	a149e
author	Scott
committer	Scott
my commit message goes here and it is really, really cool	





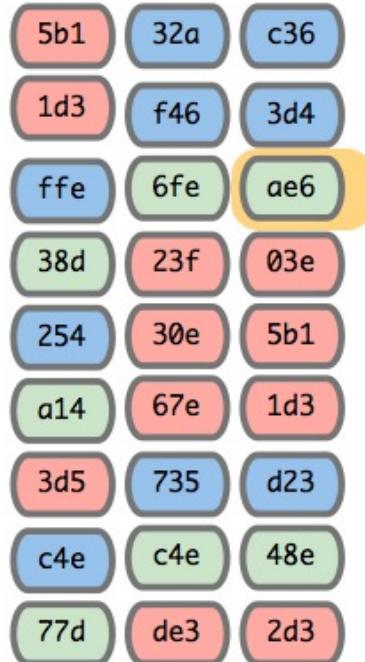




Repository

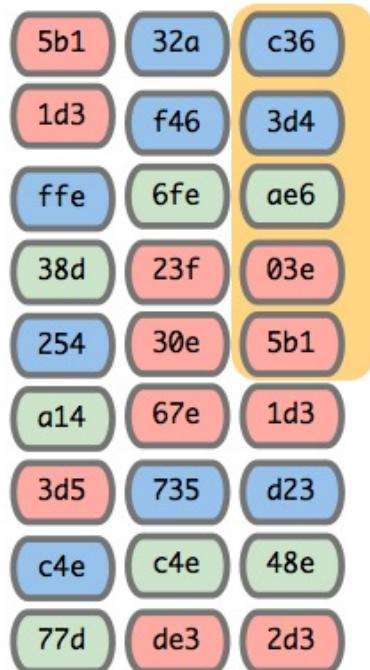
5b1	32a	c36
1d3	f46	3d4
ffe	6fe	ae6
38d	23f	03e
254	30e	5b1
a14	67e	1d3
3d5	735	d23
c4e	c4e	48e
77d	de3	2d3

Repository



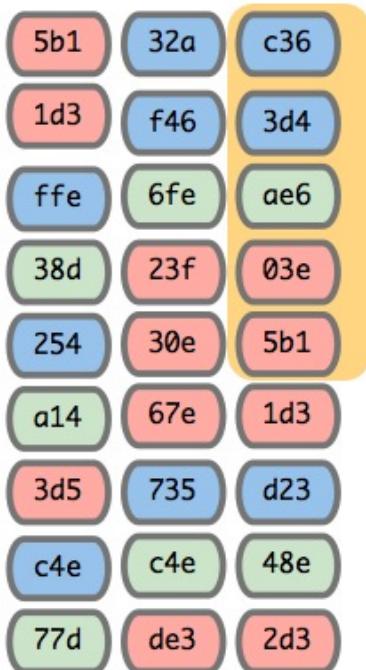
git checkout branch

Repository

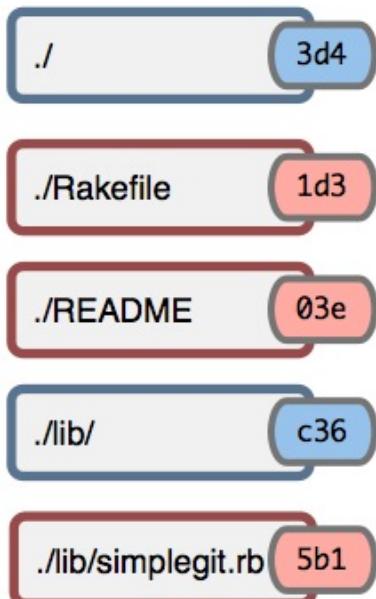


git checkout branch

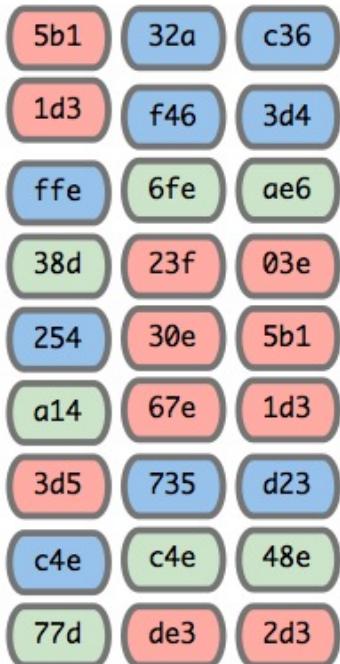
Repository



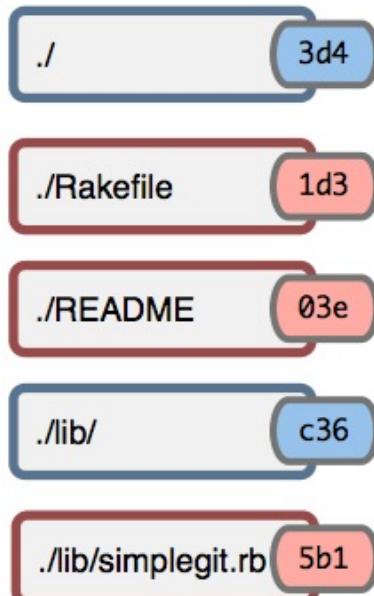
Index



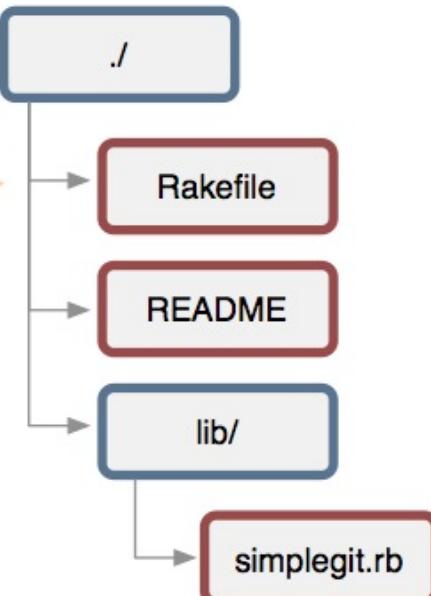
Repository



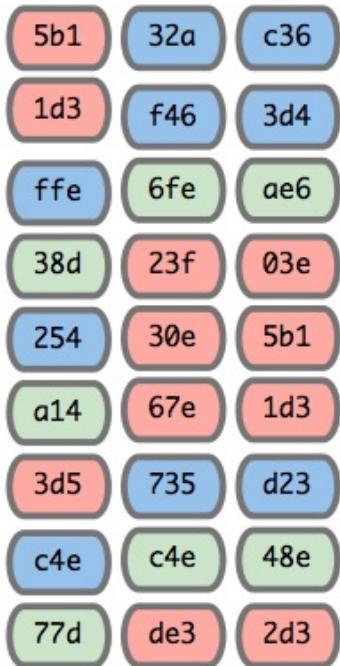
Index



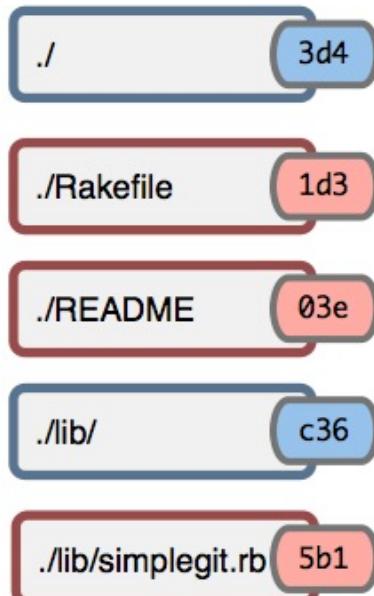
Working Directory



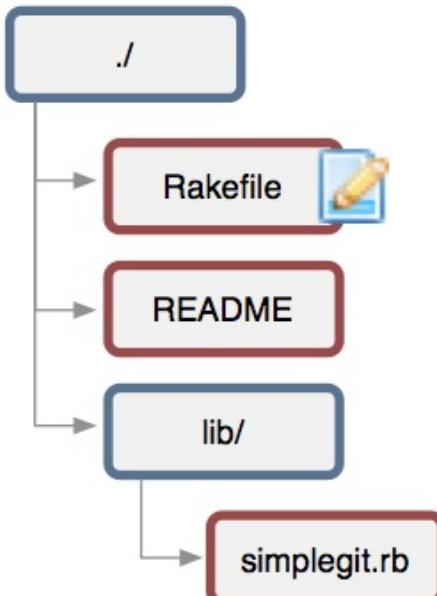
Repository



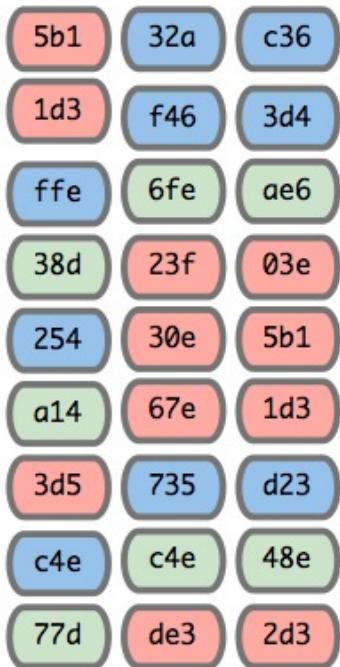
Index



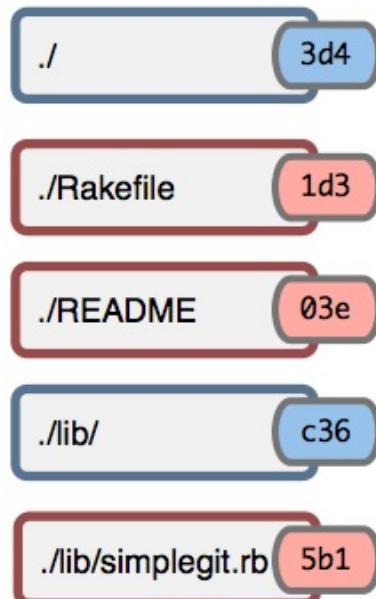
Working Directory



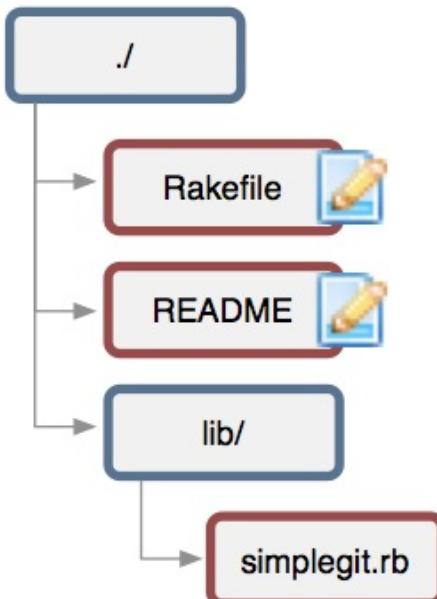
Repository



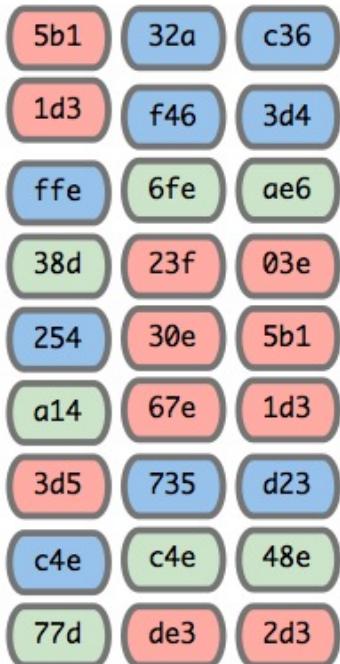
Index



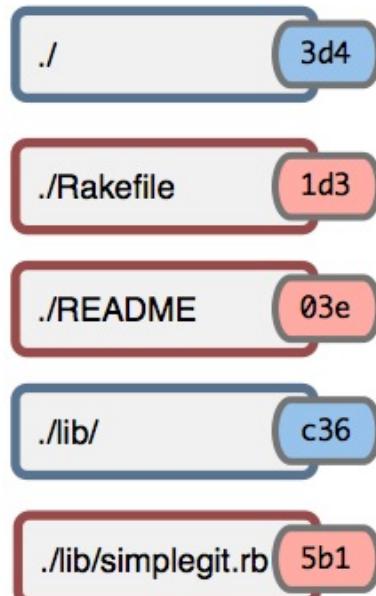
Working Directory



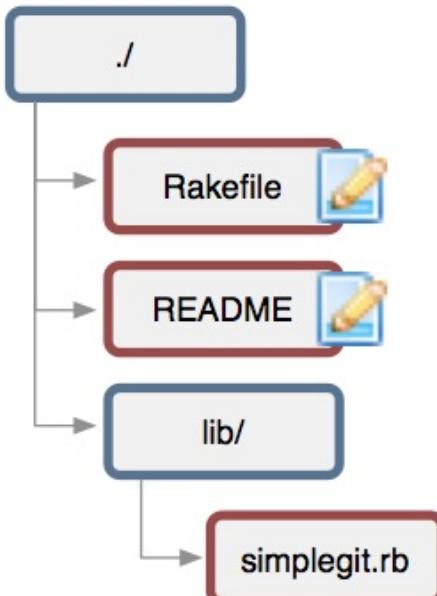
Repository



Index

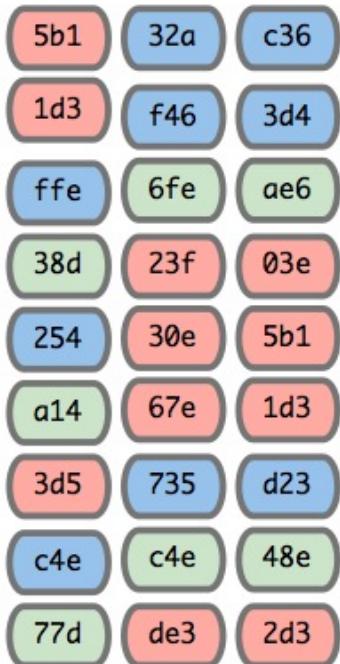


Working Directory

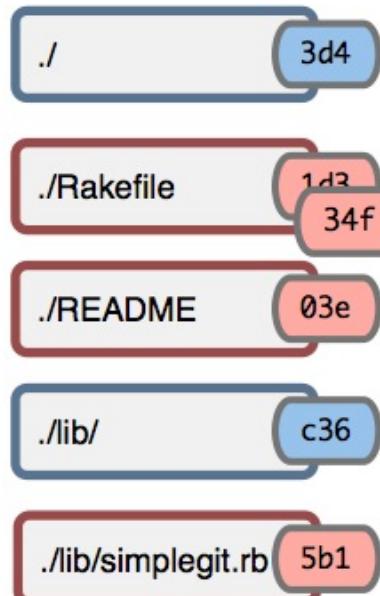


git add

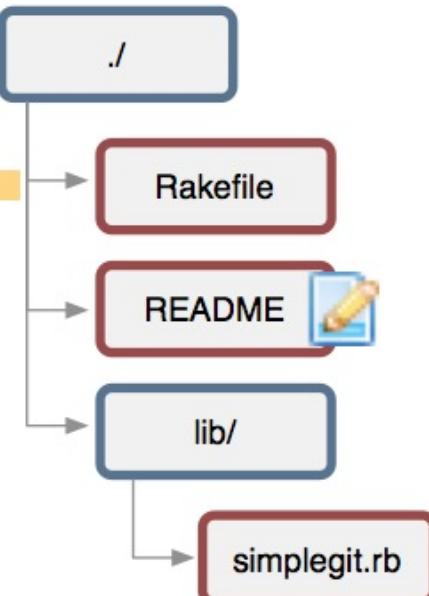
Repository



Index



Working Directory

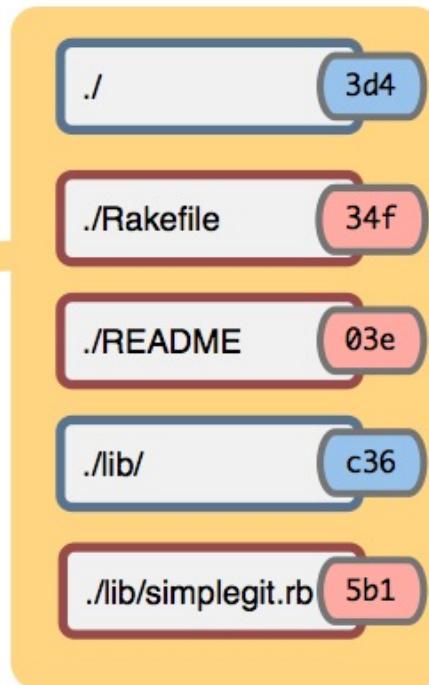


git add

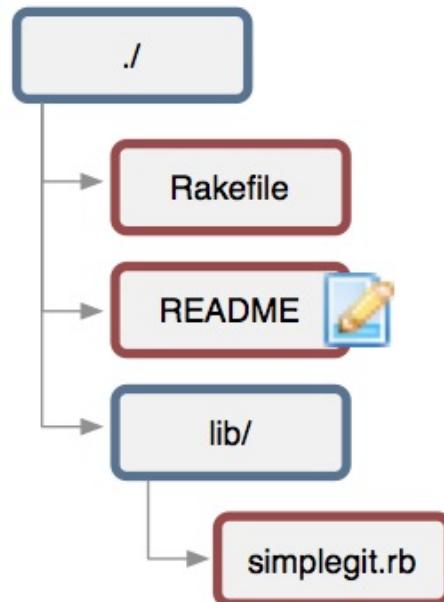
Repository



Index



Working Directory

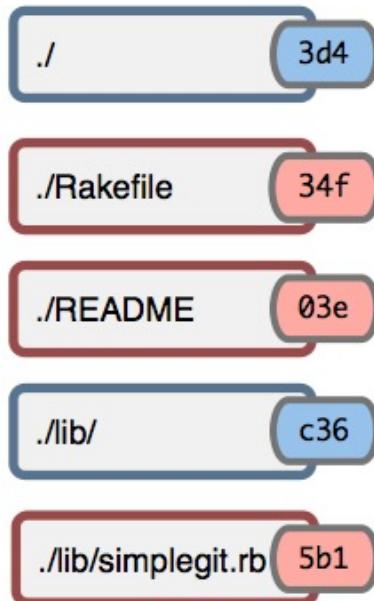


git commit

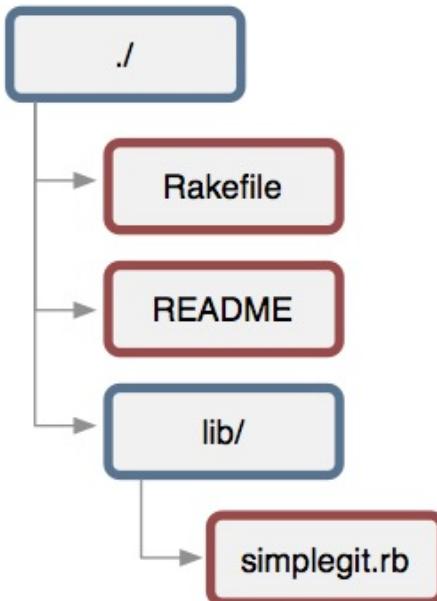
Repository



Index



Working Directory



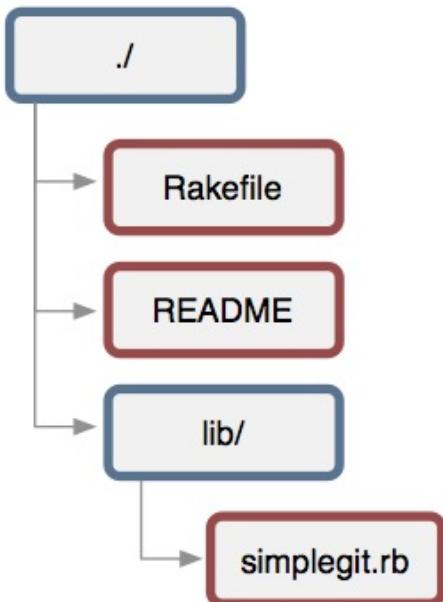
Repository



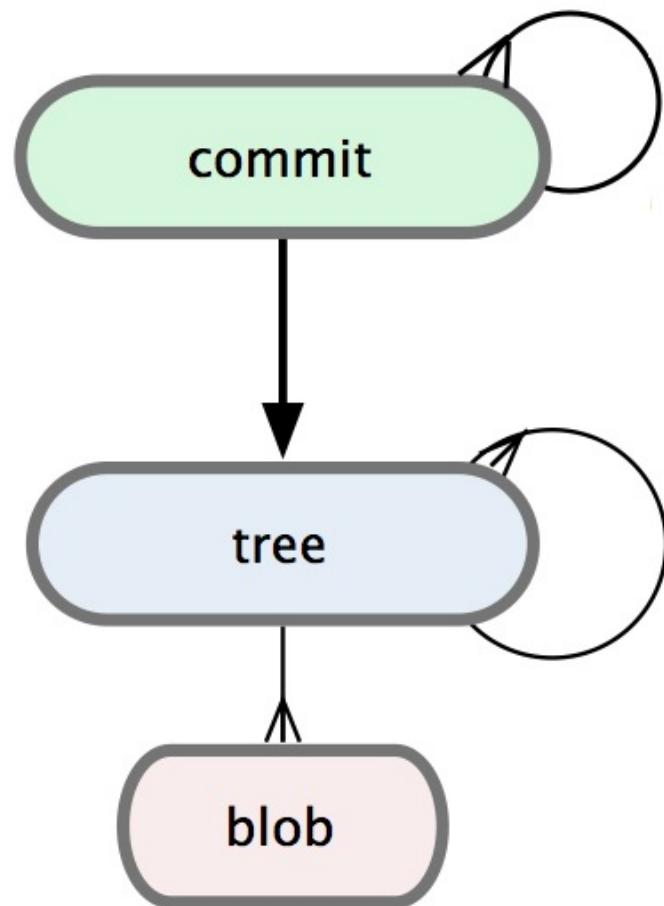
Index



Working Directory



object model

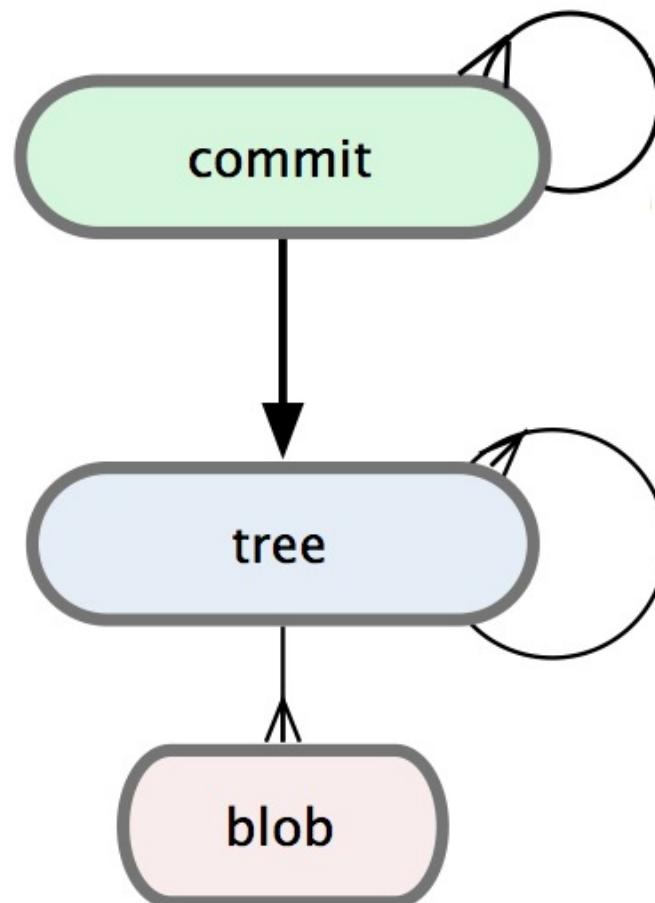


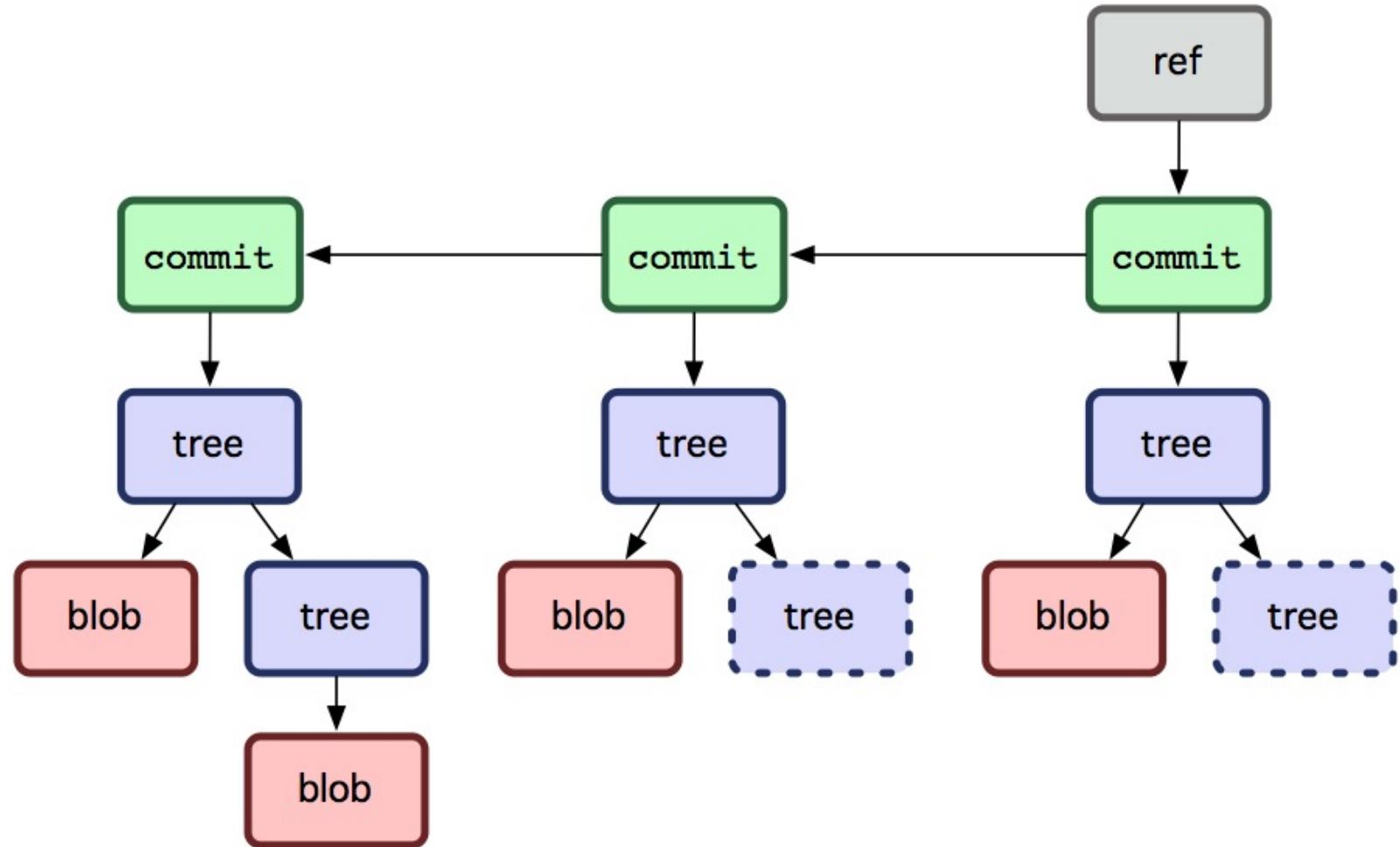
object model

pointer to a
snapshot

directory list

file contents





Inspecting Git Objects

```
git cat-file -p
```

```
$ git cat-file -p HEAD
tree 344ac3ca6eea32f4c517fa8ea97d5093933b892c
parent deed1ee3e0818e833dbafb899020aa2c7f835dd6
author Matt Gauger <matt.gauger@gmail.com> 1269196110 -0500
committer Matt Gauger <matt.gauger@gmail.com> 1269196110 -0500
```

Notes from this attempt at baking them.

```
$ git cat-file -p deed1ee3e0818e833dbafb899020aa2c7f835dd6
tree 33abee470cb69252ec4a41db5a619e15e9b93857
parent 46150e6978cf98ed9ca3e062cac9f7a225ac6e77
author Matt Gauger <matt.gauger@gmail.com> 1269194738 -0500
committer Matt Gauger <matt.gauger@gmail.com> 1269194738 -0500
```

Spelling mistakes >.<

```
$ git cat-file -p 33abee470cb69252ec4a41db5a619e15e9b93857
100644 blob 0d76a05d99ecf4ae2dbf5949      README
100644 blob 0a47f07a73cd6d6a46241348      easier-vegan-scones.md
100644 blob ff2e66e63bc48ff62227fcfd       easy-vegan-scones.txt
100644 blob 6219d4b67e650bad0a160652      pumpkin-oatmeal-cookie
```



```
$ git cat-file -p ff2e66e63bc48ff62227fcfd376 | head -3
Easy Vegan Raisin Scones
Yield: 16 scones
1/2 cup Earth Balance
```

Tutorial

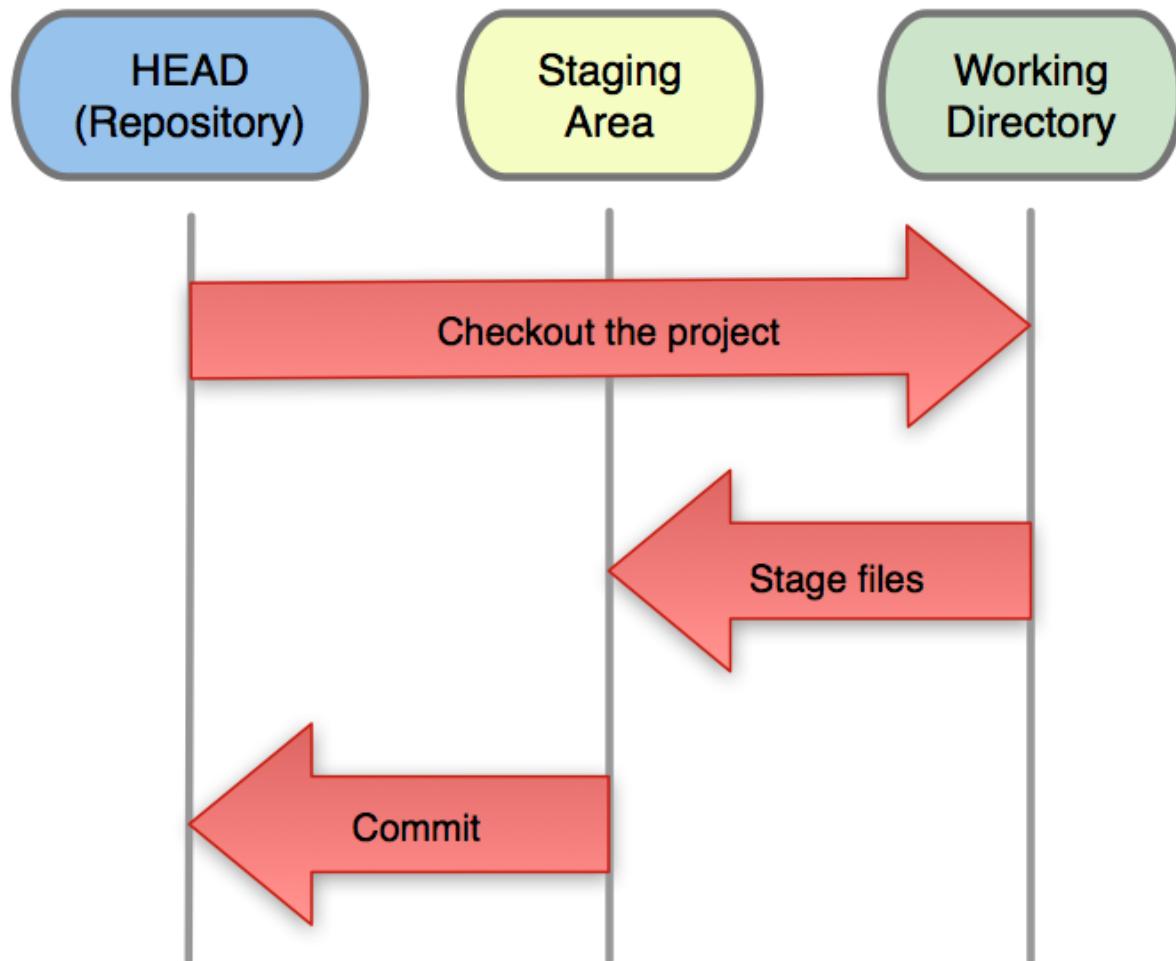
use git cat-file to simulate git log

use git cat-file to see which files changed between
v1.0 and v1.1

The Trees

HEAD, index, work tree

Local Operations



Environment Variables

moving around your git pieces

git directory

index file

working directory

GIT_DIR

```
$ mv .git /opt/repo.git  
$ git --git-dir=/opt/repo.git log  
$ export GIT_DIR=/opt/repo.git  
$ git log
```

GIT_INDEX_FILE

```
$ git status -s
M README
M kidgloves.rb

$ git add kidgloves.rb

$ git status -s
M README
S kidgloves.rb
```

```
$ export GIT_INDEX_FILE=/tmp/index
$ git read-tree HEAD
$ git add README

$ git status -s
S  README
M  kidgloves.rb
```

```
$ unset GIT_INDEX_FILE

$ git status -s
 M README
S  kidgloves.rb

$ export GIT_INDEX_FILE=/tmp/index

$ git status -s
S  README
M kidgloves.rb
```

GIT_WORK_TREE

```
$ git status -s
M README
S kidgloves.rb

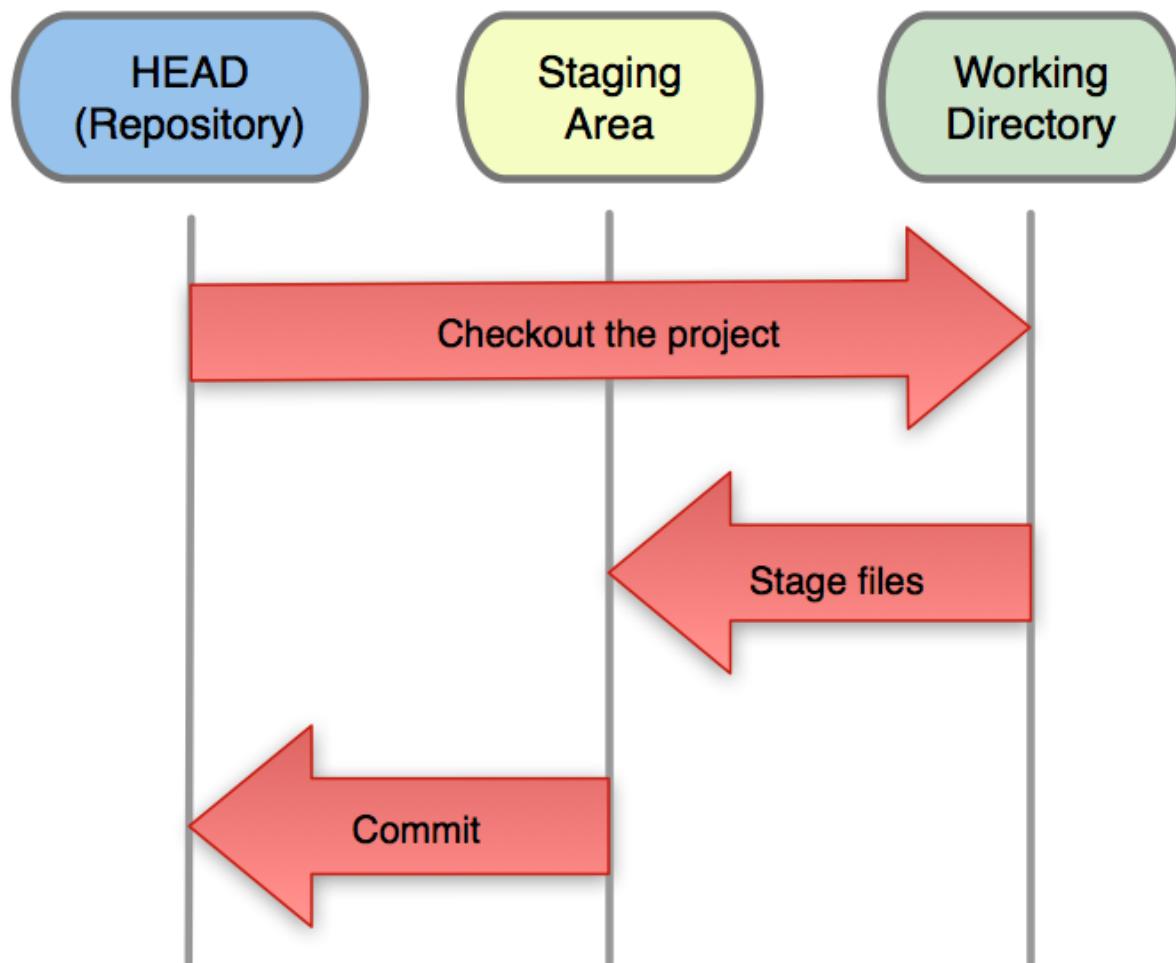
$ export GIT_DIR=$(pwd) /.git
$ export GIT_WORK_TREE=$(pwd)

$ cd /tmp
$ git status -s
M README
S kidgloves.rb
```

Using Reset

Managing your trees

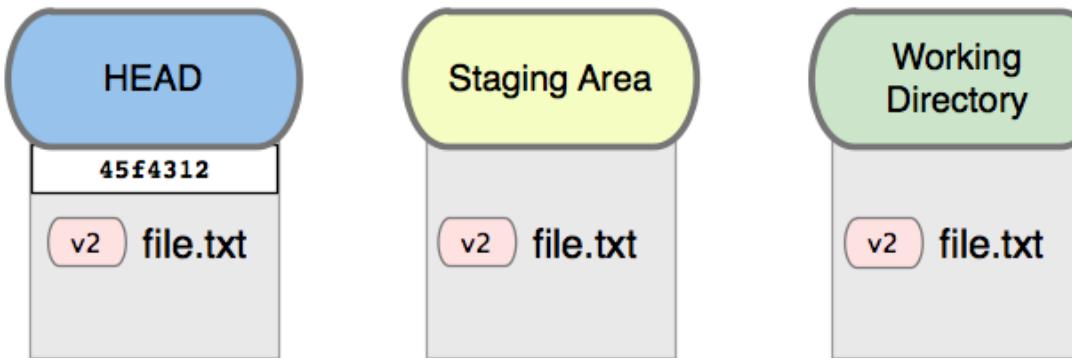
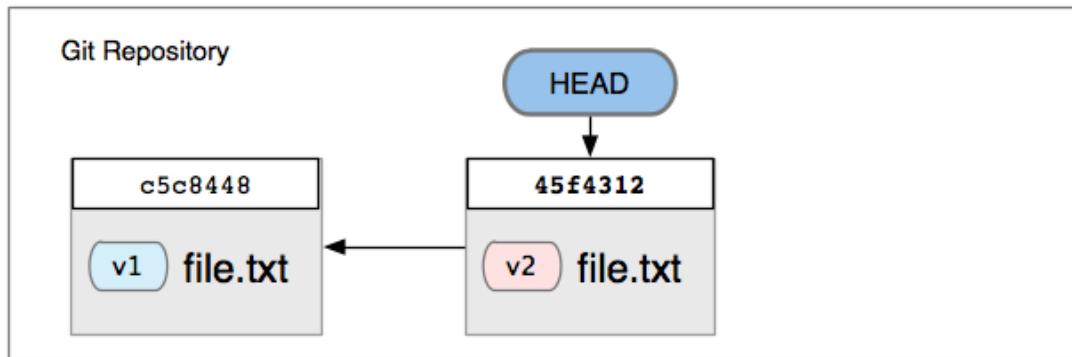
Local Operations

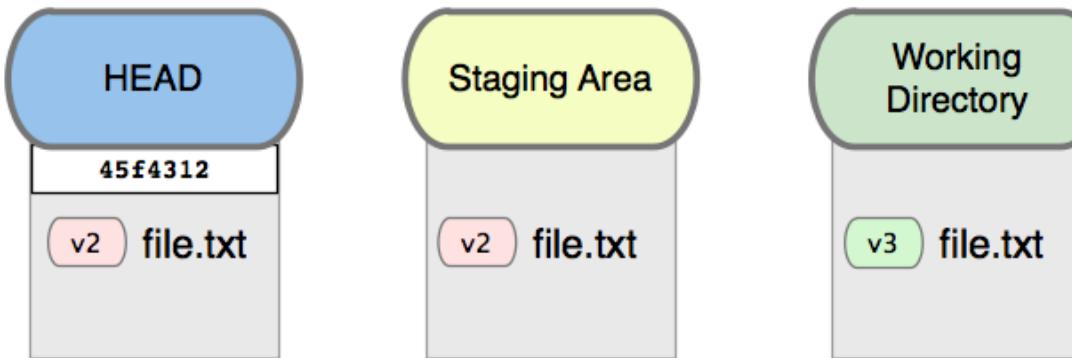
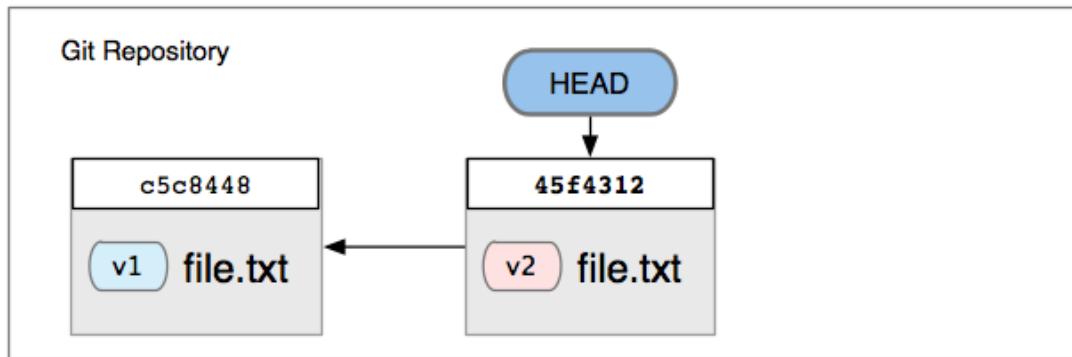


It points the HEAD ref at a new 'target' commit, if you specified one.

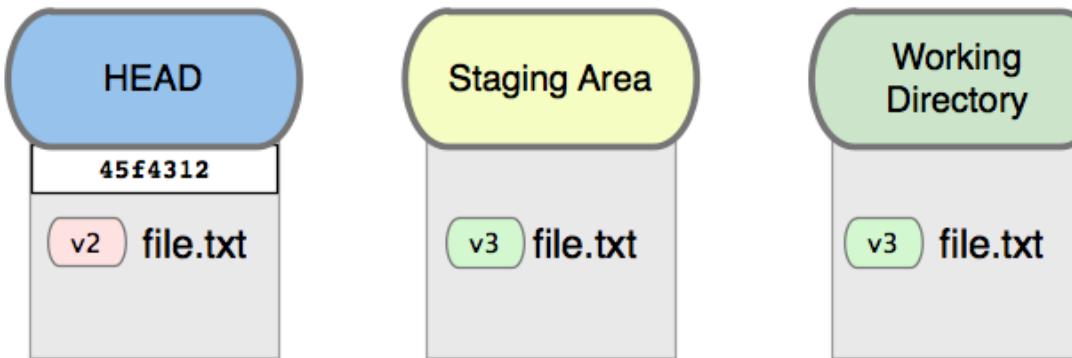
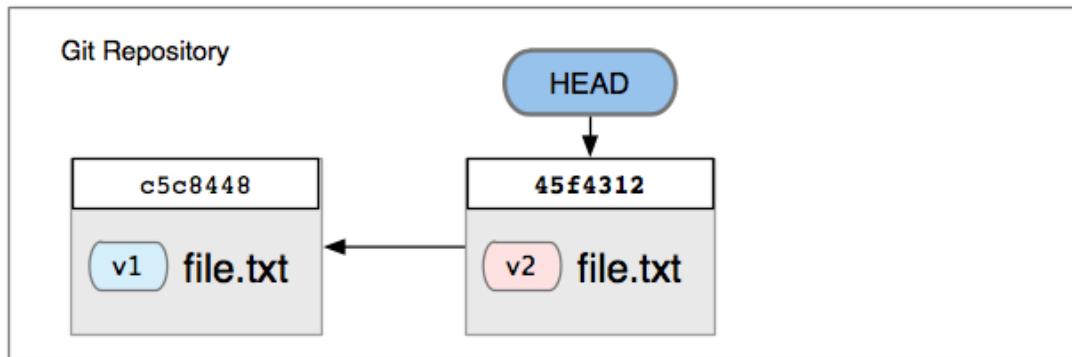
Then it copies the tree of the HEAD commit to the index, unless you said --soft.

Finally, it copies the contents of the index to the working tree, if you said --hard.

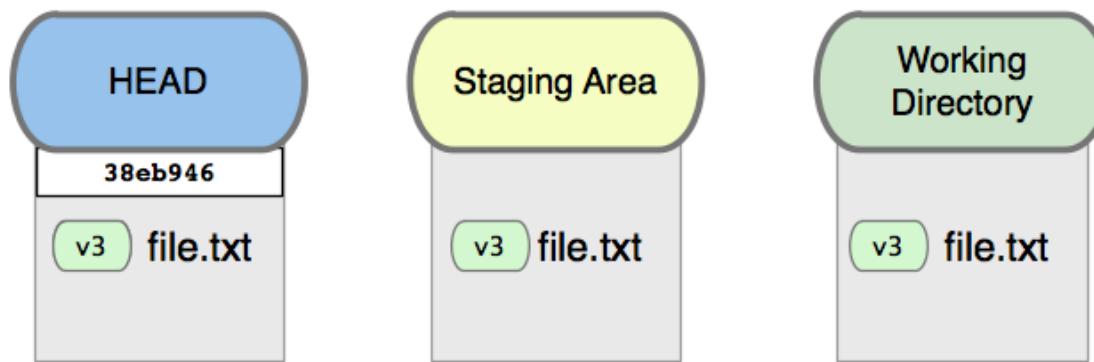
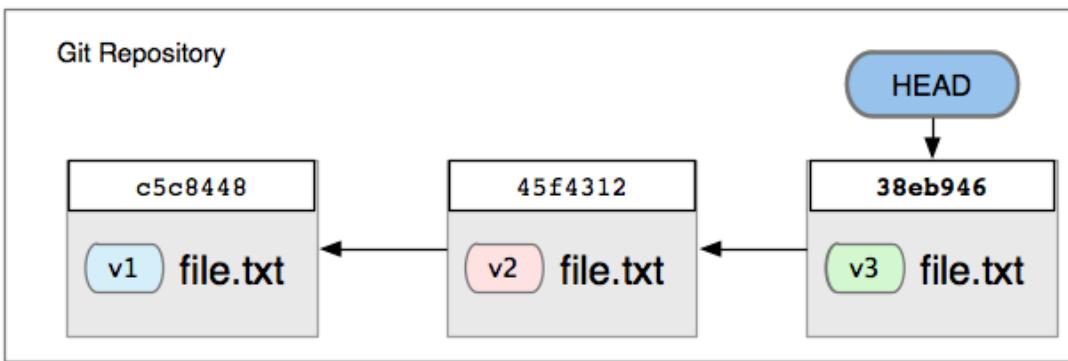




edit files

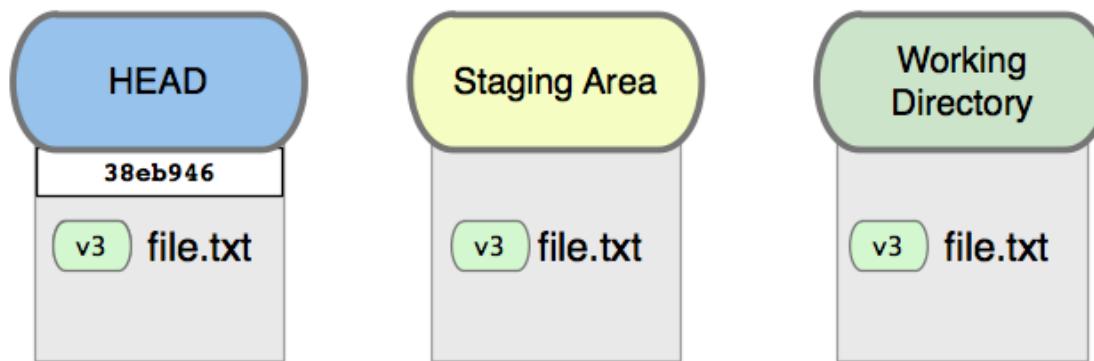
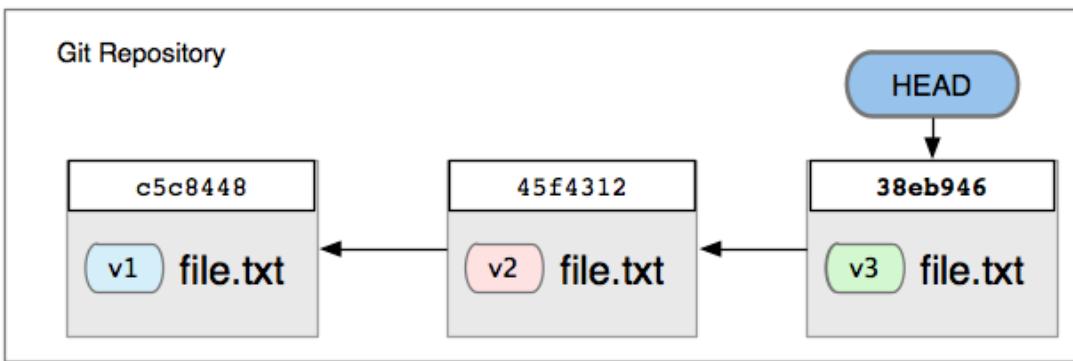


git add

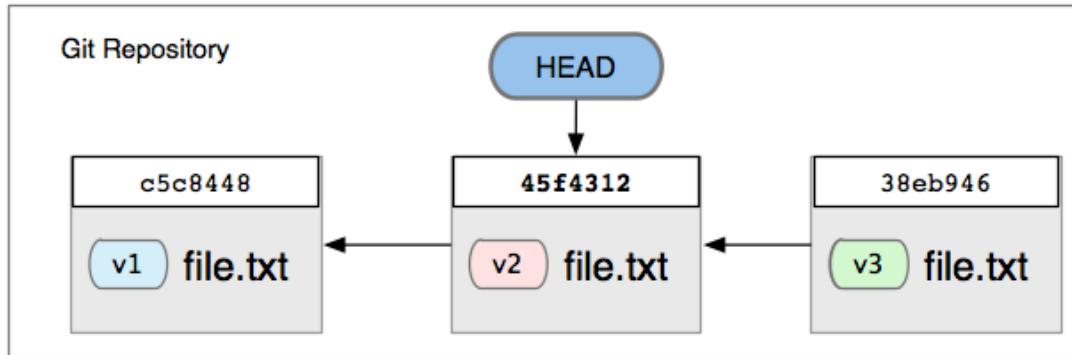


git commit

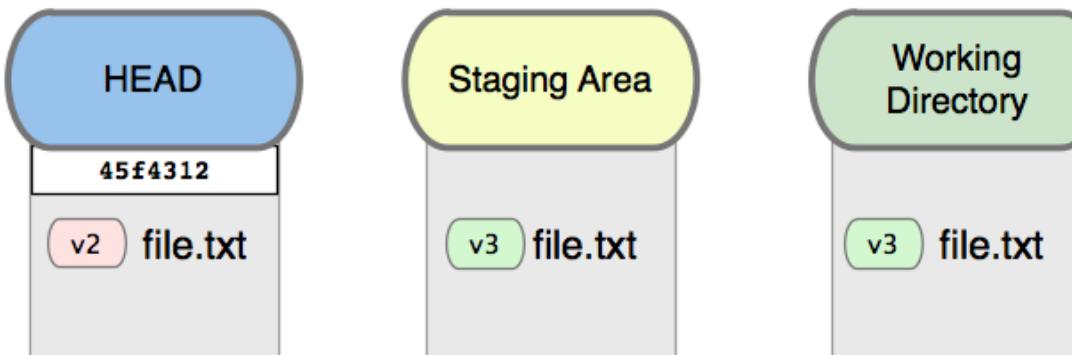
git reset

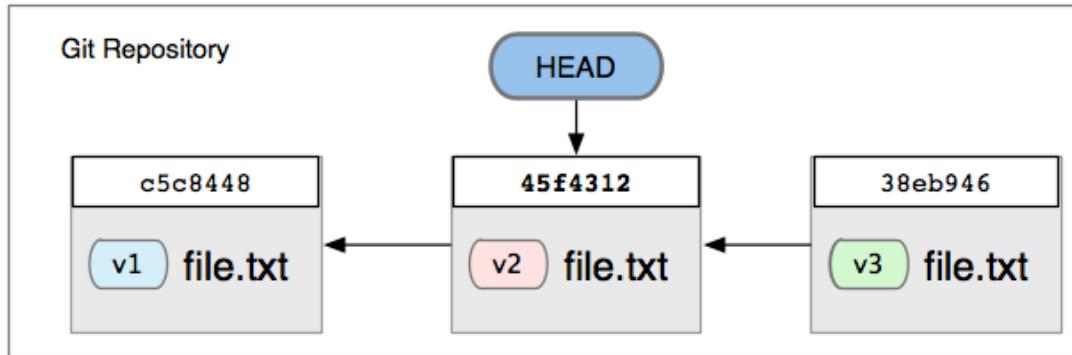


git commit

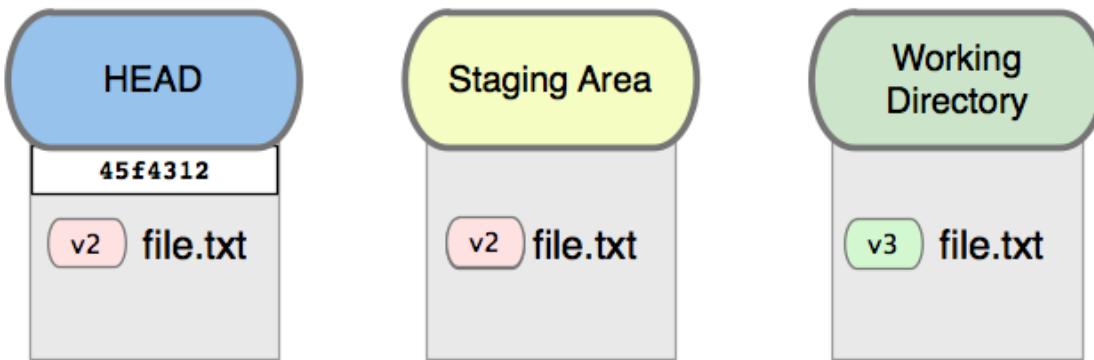


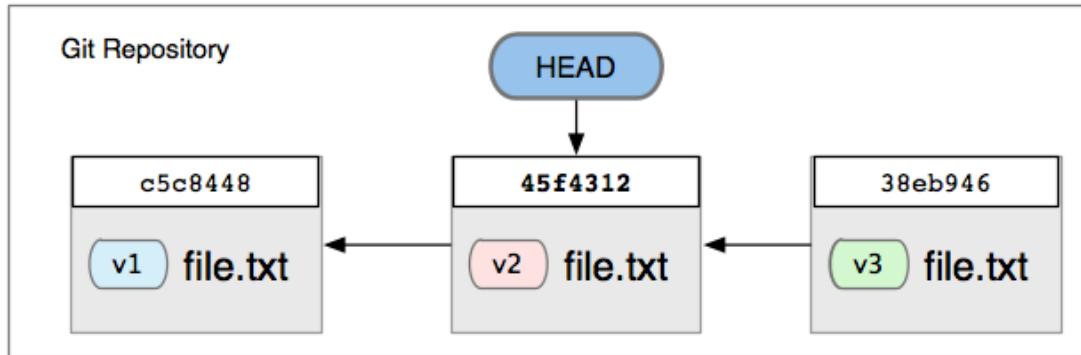
`git reset --soft HEAD^`



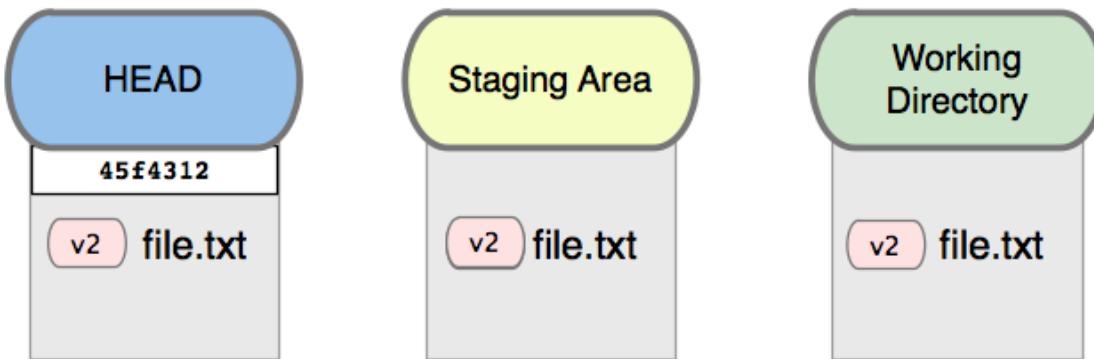


git reset [--mixed] HEAD^





git reset --hard HEAD^



examples

unstaging changes

```
git reset [--mixed] HEAD -- file
```

sets index to HEAD

undo last commit

```
git reset [--mixed] HEAD~
```

moves HEAD back and moves index back

squash the last 2 commits into
a new one

```
git reset --soft HEAD~2
```

```
git commit
```

moves HEAD back, keeps index

Tutorial

edit two files, stage one

run git reset HEAD **to undo the stage**

re-stage the file and commit

run git reset --soft HEAD~ **to undo the commit but keep the staged files**

commit again

run git reset HEAD~ **to undo the commit and all staging; commit again**

run git reset --hard HEAD~ **to lose the commit and all that work**

Stashing and Cleaning

Stashing

git stash

saving work in progress

```
$ git status -s
M kidgloves.rb

$ git stash
Saved working directory and index state WIP on master: acca3c0
HEAD is now at acca3c0 oops. resolved that conflict somewhat v

$ git status
# On branch master
# Your branch is behind 'origin/master' by 3 commits, and can
#
nothing to commit (working directory clean)

$ git stash apply
# On branch master
# Your branch is behind 'origin/master' by 3 commits, and can
#
# Changed but not updated:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in wor
#
#       modified:   kidgloves.rb
#
no changes added to commit (use "git add" and/or "git commit"
```

```
git stash --keep-index
```

```
$ git status -s
MM kidgloves.rb

$ git stash --keep-index
Saved working directory and index state WIP on master: 686d28b
HEAD is now at 686d28b allow calling server start, accept, std

$ git status -s
M  kidgloves.rb

$ git stash apply
Auto-merging kidgloves.rb
# On branch master
# Your branch is behind 'origin/master' by 2 commits, and can
#
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   kidgloves.rb
#
# Changed but not updated:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in w
#
#       modified:   kidgloves.rb
#
```

```
$ git stash save 'my really good stash'
Saved working directory and index state On master: my really good stash
HEAD is now at 686d28b allow calling server start, accept, std::string
$ git stash list
stash@{0}: On master: my really good stash
stash@{1}: WIP on master: 686d28b allow calling server start, accept, std::string
stash@{2}: WIP on master: acca3c0 oops. resolved that conflict
$ git stash show --stat stash@{0}
kidgloves.rb |    46 +-----
1 files changed, 2 insertions(+), 44 deletions(-)

$ git stash drop stash@{1}
Dropped stash@{1} (a4e6c90d6282675e2f71091e1331b5679f967f00)

$ git stash list
stash@{0}: On master: my really good stash
stash@{1}: WIP on master: acca3c0 oops. resolved that conflict
```

save

apply

list

show

drop

branch

Tutorial

edit a file and stash the changes to that file naming
the stash “my first file change”

edit another file and stash the changes naming it
“my second file change”

apply the “second file change” stash and commit it

drop the first change

Tutorial

edit two files and stage one of them

stash the unstaged changes

edit another file and stash part of it

commit

apply both stashes and commit them

Cleaning

```
git clean -fdx
```

- f

actually run the thing

-d

do directories too

- X

ignore the ignores

-X

only stuff in .gitignore

```
$ git status -s
## master
?? build.o
?? mistake_file.c

$ git clean -fdx
Removing build.o
Removing mistake_file.c
Removing stupid.swp
```

```
$ git status -s
## master
?? build.o
?? mistake_file.c

$ git clean -fX
Removing stupid.swp
```

Tutorial

create a .gitignore file with a *.o pattern

create a build.o file with any content

create a test.c file with any content

use git clean to remove just the .o files

use git clean to remove all untracked files

Data Mining

Reflog

```
$ git reflog
6b490d2 HEAD@{0}: checkout: moving from master to locale
53a33b7 HEAD@{1}: checkout: moving from local to master
6b490d2 HEAD@{2}: commit: updated almost all of the explore an
2a062e7 HEAD@{3}: commit (merge): Merge remote branch 'origin/
d4409b8 HEAD@{4}: commit: started on the meta pages
ce14759 HEAD@{5}: commit: plans page i18nd
79bbe8a HEAD@{6}: commit: home and login pages i18nd
0b9c5fa HEAD@{7}: commit: read-only/write descriptions i18nd
5ab5182 HEAD@{8}: commit: i think forkqueue is all done
ae20364 HEAD@{9}: checkout: moving from master to local
53a33b7 HEAD@{10}: checkout: moving from local to master
ae20364 HEAD@{11}: commit: forkqueue main page done
732f689 HEAD@{12}: commit: some setup screen translations
6a32edb HEAD@{13}: commit: i think issues is basically i18nd
5ee4b9d HEAD@{14}: commit: moved rails translation files into
c83a3d9 HEAD@{15}: commit (amend): replaced create issue button
189cd67 HEAD@{16}: commit (amend): replaced create issue button
```

```
$ git reflog show http_proxy
cbelaad http_proxy@{0}: commit: copied slummin to github for c
8dded6d http_proxy@{1}: commit: really add the benchmark
6e0bd39 http_proxy@{2}: commit: svn proxy benchmarking
ab98772 http_proxy@{3}: commit: oops. wrong port
f626f6e http_proxy@{4}: commit: pid file is here
48077c4 http_proxy@{5}: commit (amend): pulling in Grit change
3169e43 http_proxy@{6}: commit: pulling in changes from slummi
d7e3c45 http_proxy@{7}: merge origin/master: Merge made by rec
d7b2b6a http_proxy@{8}: commit: updated to all the config file
0dd2e03 http_proxy@{9}: commit: changed port
0ea0f22 http_proxy@{10}: commit: nginx config that works with
b65a812 http_proxy@{11}: commit: for svn clients that are redo
3eff356 http_proxy@{12}: commit (amend): sending proper header
1a8e1da http_proxy@{13}: commit (amend): sending proper header
```

```
$ git log -g http_proxy
commit cbelaad3efd172abf4ea05affdeb69052609c8a4
Reflog: http_proxy@{0} (Scott Chacon <schacon@gmail.com>)
Reflog message: commit: copied slummin to github for deps and
Author: Scott Chacon <schacon@gmail.com>
Date: Tue Jan 12 16:29:06 2010 -0800

        copied slummin to github for deps and cap reuse

commit 8dded6d16910c1497bb2810b43c097cf908f84f
Reflog: http_proxy@{1} (Scott Chacon <schacon@gmail.com>)
Reflog message: commit: really add the benchmark
Author: Scott Chacon <schacon@gmail.com>
Date: Wed Dec 30 14:39:13 2009 -0800

        really add the benchmark

commit 6e0bd393322fa5a91214a9a1f0cb77ee59ac69d8
Reflog: http_proxy@{2} (Scott Chacon <schacon@gmail.com>)
Reflog message: commit: svn proxy benchmarking
Author: Scott Chacon <schacon@gmail.com>
Date: Wed Dec 30 14:17:50 2009 -0800

        svn proxy benchmarking
```

```
$ git show http_proxy@{3}
commit ab98772c61d25d26e2f5713bbf28ba0bc06c1d79
Author: Scott Chacon <schacon@gmail.com>
Date:   Wed Dec 30 11:34:17 2009 -0800
```

oops. wrong port

```
$ git show http_proxy@{1.week.ago}
commit 8dded6d16910c1497bb2810b43c097cf908f84f
Author: Scott Chacon <schacon@gmail.com>
Date:   Wed Dec 30 14:39:13 2009 -0800
```

really add the benchmark

Advanced Log

Log Formatting

```
git log --oneline
```

```
$ git log --oneline
b809d9c Git 1.6.6-rc1
c0ecb07 git-pull.sh: Fix call to git-merge for new command for
28044ba Prepare for 1.6.5.4
ce9d823 merge: do not add standard message when message is given
76bf488 Do not misidentify "git merge foo HEAD" as an old-style
c86485d Update draft release notes to 1.6.6 before -rc1
b81e00a git-merge: a deprecation notice of the ancient command
92f676f get_ref_states: strdup entries and free util in stale
af6fbf9 help: Do not unnecessarily look for a repository
3c652d1 Documentation: Fix a few i.e./e.g. mix-ups
87e573f gitweb: Add link to other blame implementation in blame
e627e50 gitweb: Make linking to actions requiring JavaScript a
db9bc00 Documentation: Document --branch option in git clone so
e2ced7d builtin-merge: show user-friendly error messages for failing
264b774 merge-recursive: make the error-message generation an
e160da7 t/README: Document GIT_TEST_INSTALLED and GIT_TEST_EXECUTABLE
5d59a40 t3409 t4107 t7406 t9150: use dashless commands
ed87465 builtin-merge.c: call exclude_cmds() correctly.
```

```
git log --graph
```

```
$ git log --graph
* commit c0ecb07048ce2123589a2f077d296e8cf29a9570
| Author: Horst H. von Brand <vonbrand@inf.utfsm.cl>
| Date:   Tue Dec 1 19:44:11 2009 -0300

|     git-pull.sh: Fix call to git-merge for new command forma

|     Now "git merge <msg> HEAD" is officially deprecated, we
|     clean our own use as well.

|     Signed-off-by: Horst H. von Brand <vonbrand@inf.utfsm.cl>
|     Signed-off-by: Junio C Hamano <gitster@pobox.com>

* commit 0748494e866041034605aaaf177f29a61bdc25951
| \ Merge: c86485d 28044ba
| | Author: Junio C Hamano <gitster@pobox.com>
| | Date:   Wed Dec 2 10:30:12 2009 -0800

| |     Merge branch 'maint'

| |     * maint:
| |     Prepare for 1.6.5.4
| |     merge: do not add standard message when message is o
```

```
git log --decorate
```

```
$ git log --oneline --graph --decorate
* b809d9c (HEAD, tag: v1.6.6-rc1, master) Git 1.6.6-rc1
* c0ecb07 git-pull.sh: Fix call to git-merge for new command f
* 0748494 Merge branch 'maint'
|\ \
| * 28044ba Prepare for 1.6.5.4
| * ce9d823 merge: do not add standard message when message is
| * 76bf488 Do not misidentify "git merge foo HEAD" as an old-
* | c86485d Update draft release notes to 1.6.6 before -rc1
* | 32ef08f Merge branch 'maint'
|\ \
| |
| * af6fbf9 help: Do not unnecessarily look for a repository
| * 3c652d1 Documentation: Fix a few i.e./e.g. mix-ups
| * db9bc00 Documentation: Document --branch option in git cl
* | 36a83f3 Merge branch 'jc/deprecate-old-syntax-from-merge'
|\ \
| * | b81e00a git-merge: a deprecation notice of the ancient d
* | | 4a27759 Merge branch 'bw/remote-get-ref-states-fix'
|\ \
| |
```

Log Filtering

```
--author  
  
--since, --after  
  
--until, --before  
  
--grep  
  
--all-match  
  
--no-merges  
  
--all  
  
-- (path)
```

```
$ git log --format="%h - %s"  
--author=gitster  
--since="2008-10-01"  
--before="2008-11-01" --no-merges -- t/  
5610e3b - Fix testcase failure when extended  
acd3b9e - Enhance hold_lock_file_for_{update,  
f563754 - demonstrate breakage of detached ch  
d1a43f2 - reset --hard/read-tree --reset -u:  
51a94af - Fix "checkout --track -b newbranch"  
b0ad11e - pull: allow "git pull origin $somet
```

Pickaxe

```
$ git log -S[search]
```

```
$ git log -Sp4merge --oneline master
785c58e Update draft release notes to 1.6.6
c8998b4 mergetool--lib: add p4merge as a pre-c
48c74a5 git-gui: Support more merge tools.
```

Graph Subsets

```
git log branchA ^branchB
```

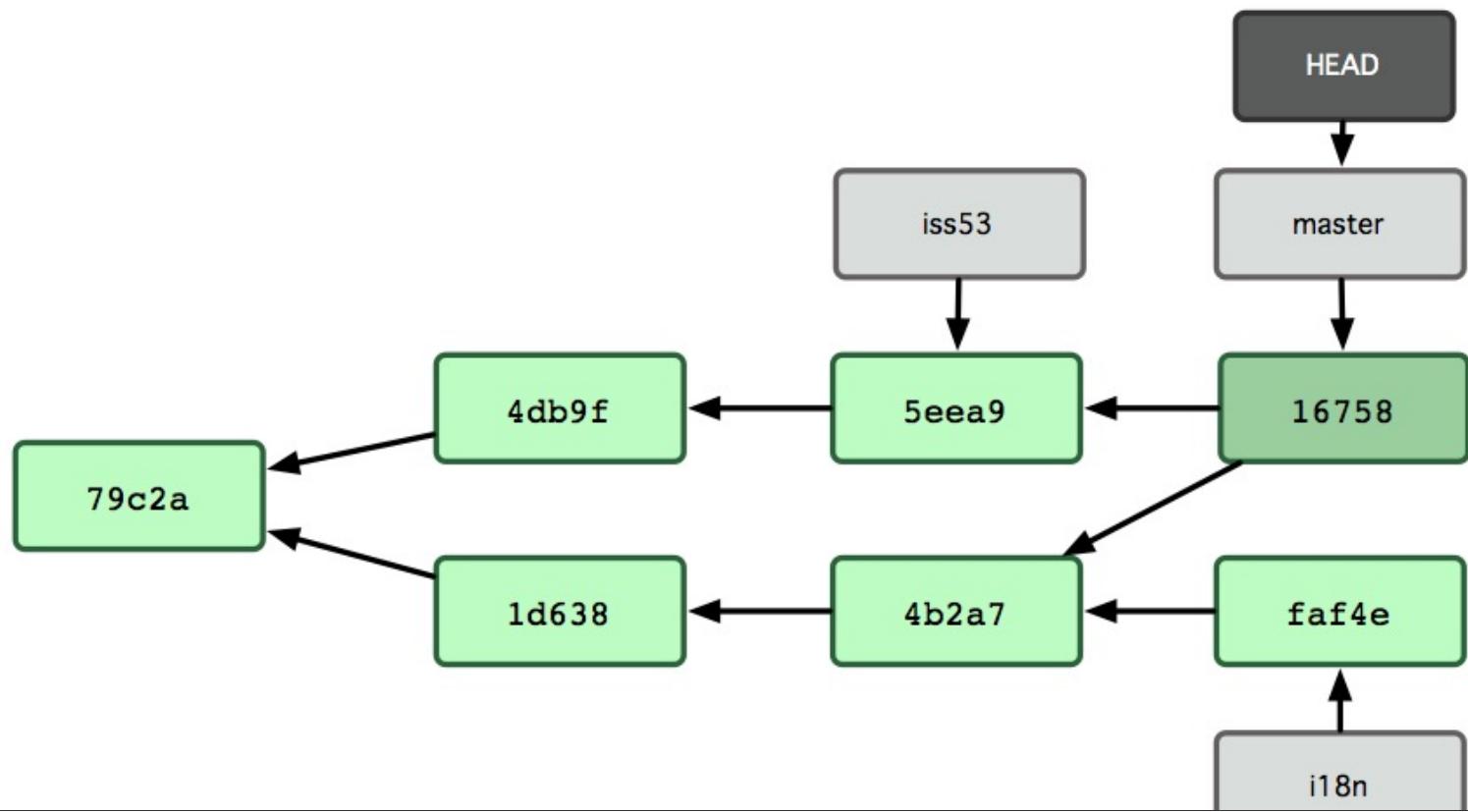
```
git log branchA ^branchB
```

show me commits reachable by branchA
that are not reachable by branchB

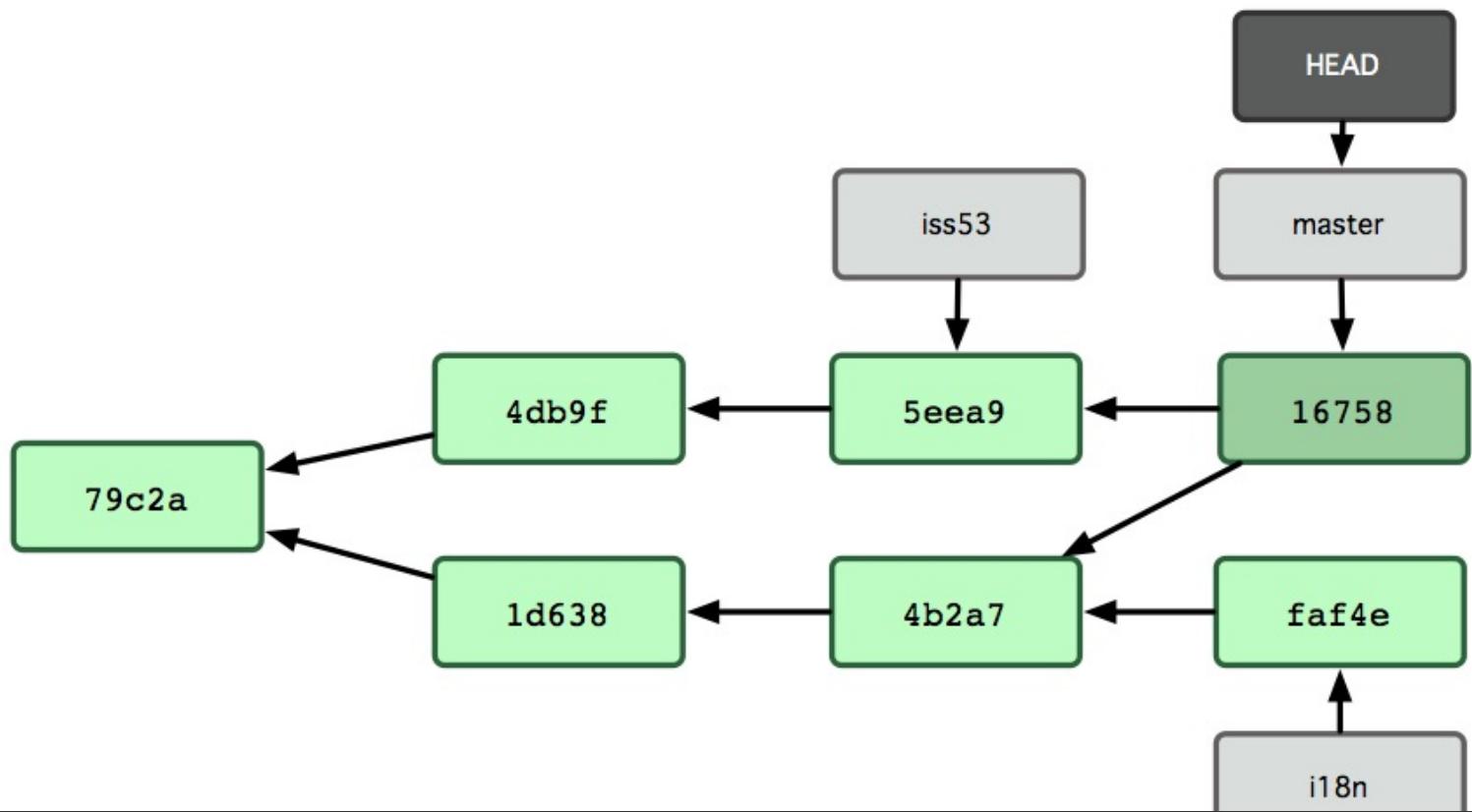
git log branchA ^branchB

```
git log branchA ^branchB
```

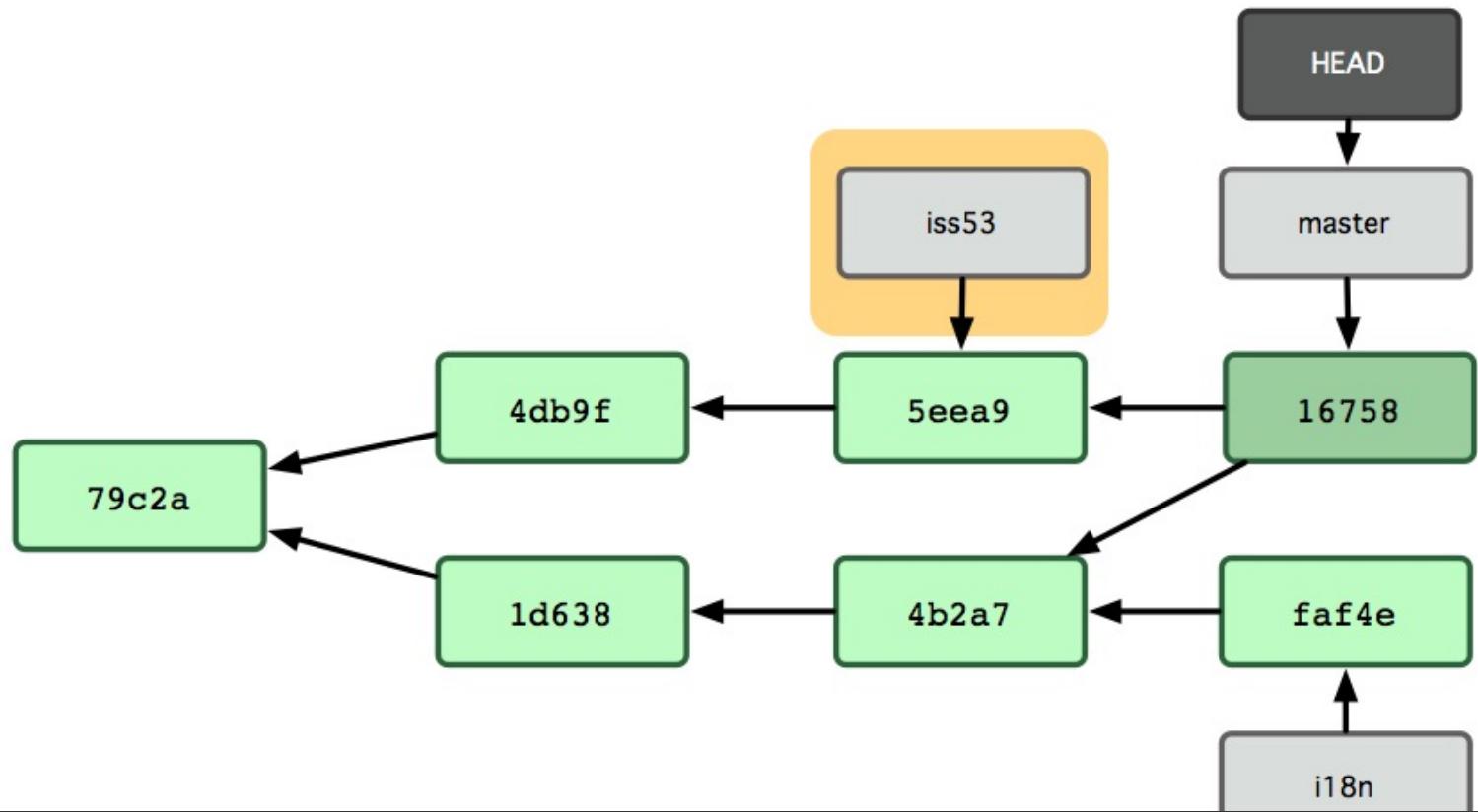
not reachable



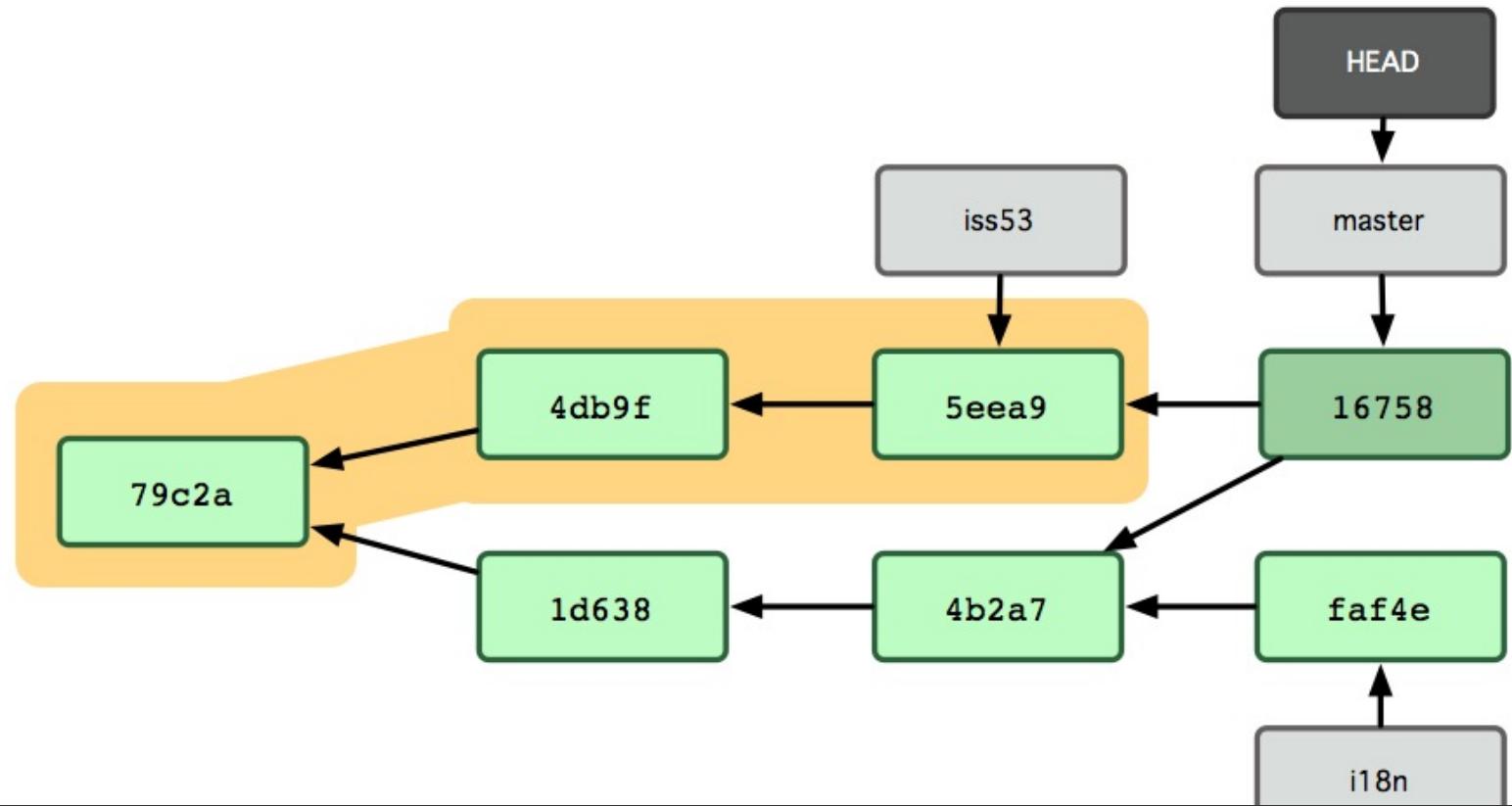
iss53 ^i18n



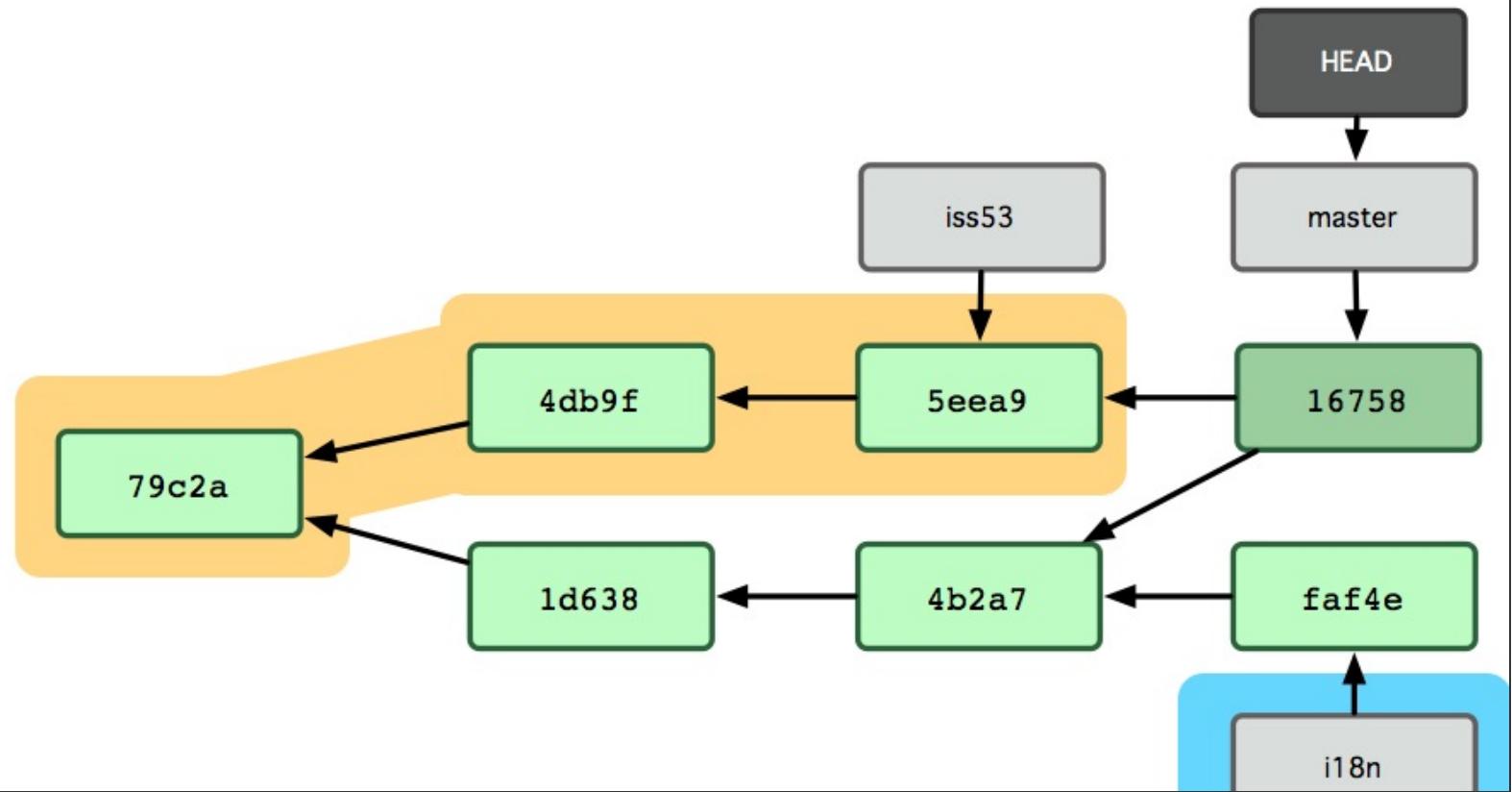
iss53 ^i18n



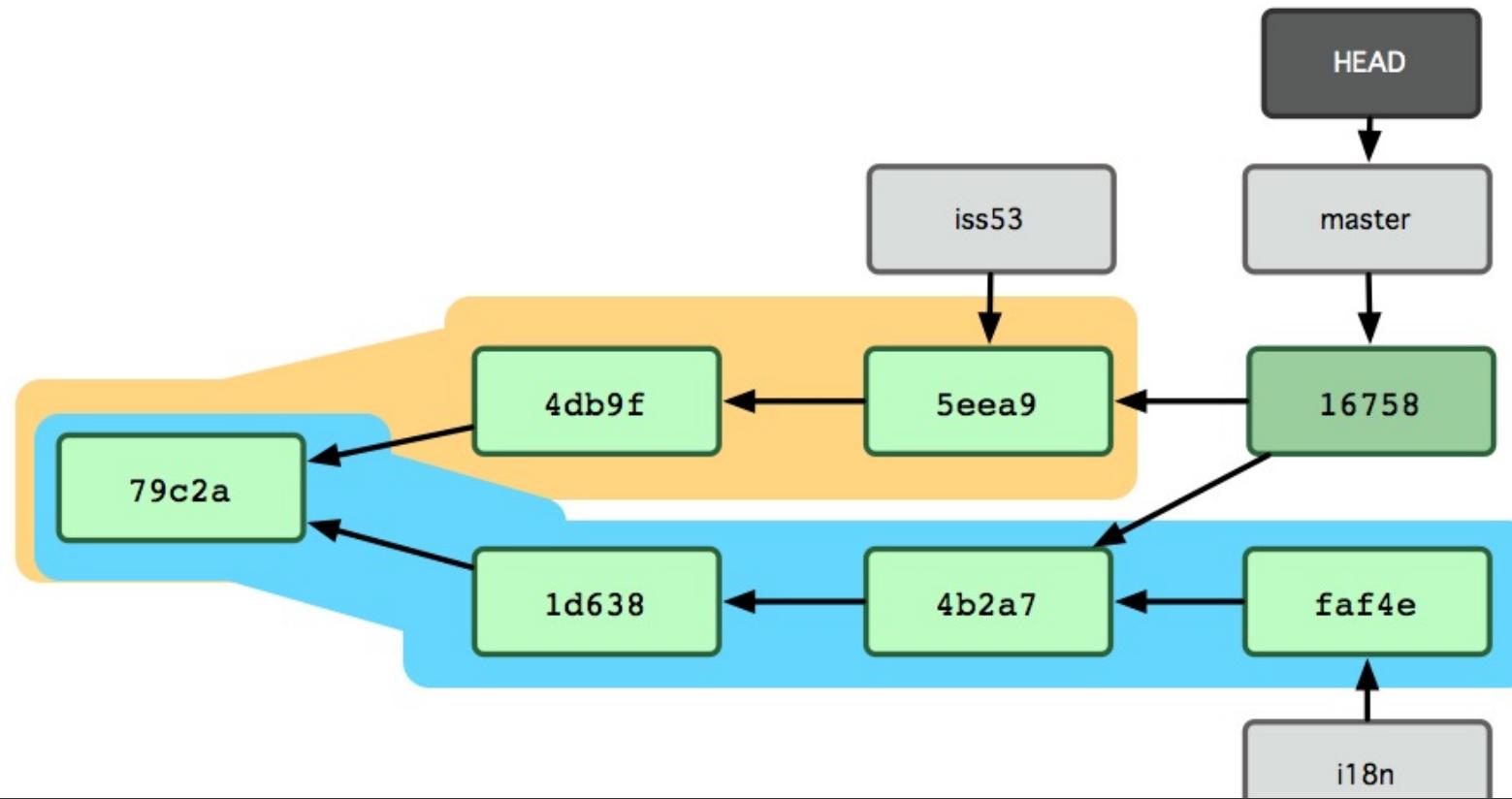
iss53 ^i18n



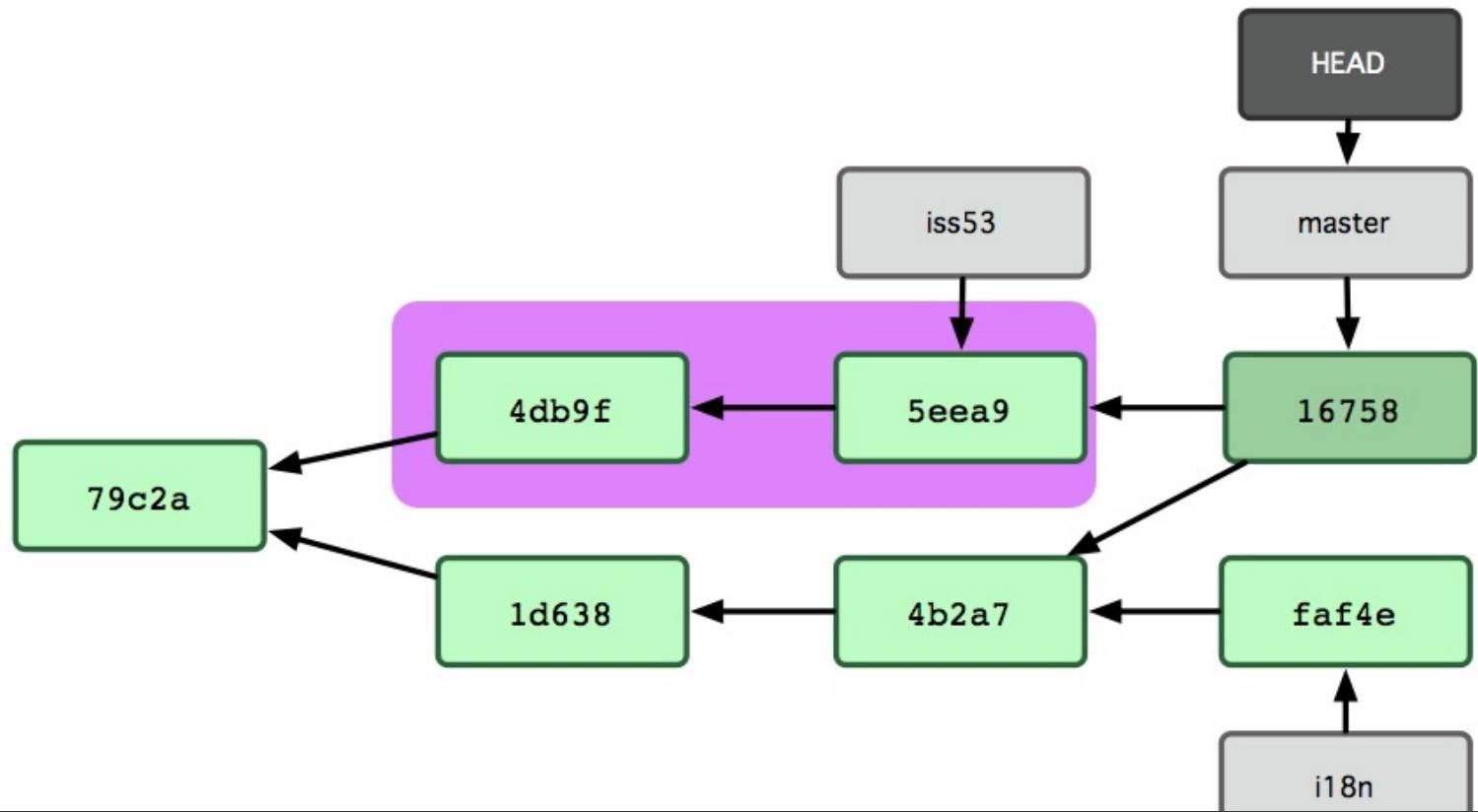
iss53 ^i18n



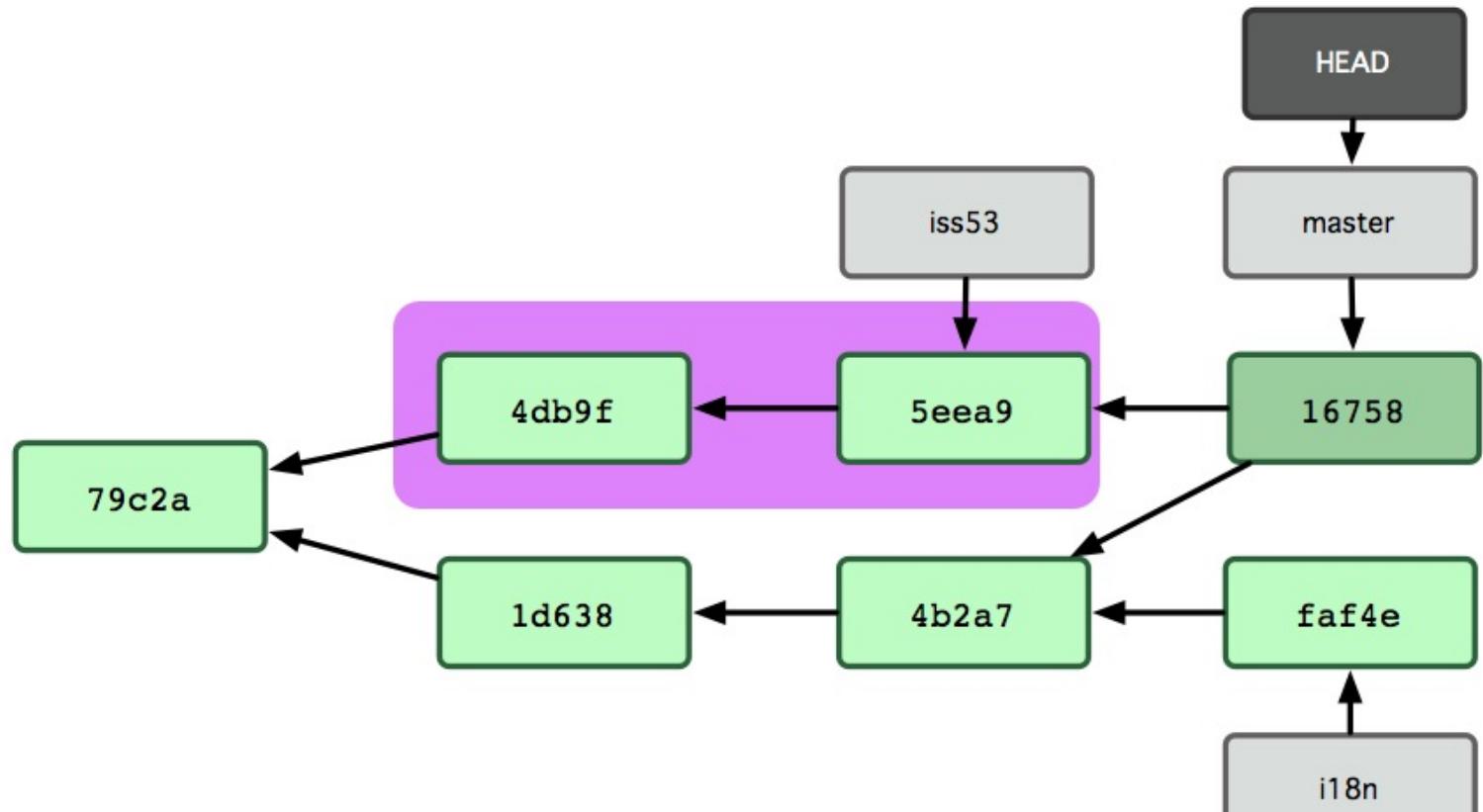
iss53 ^i18n



iss53 ^i18n

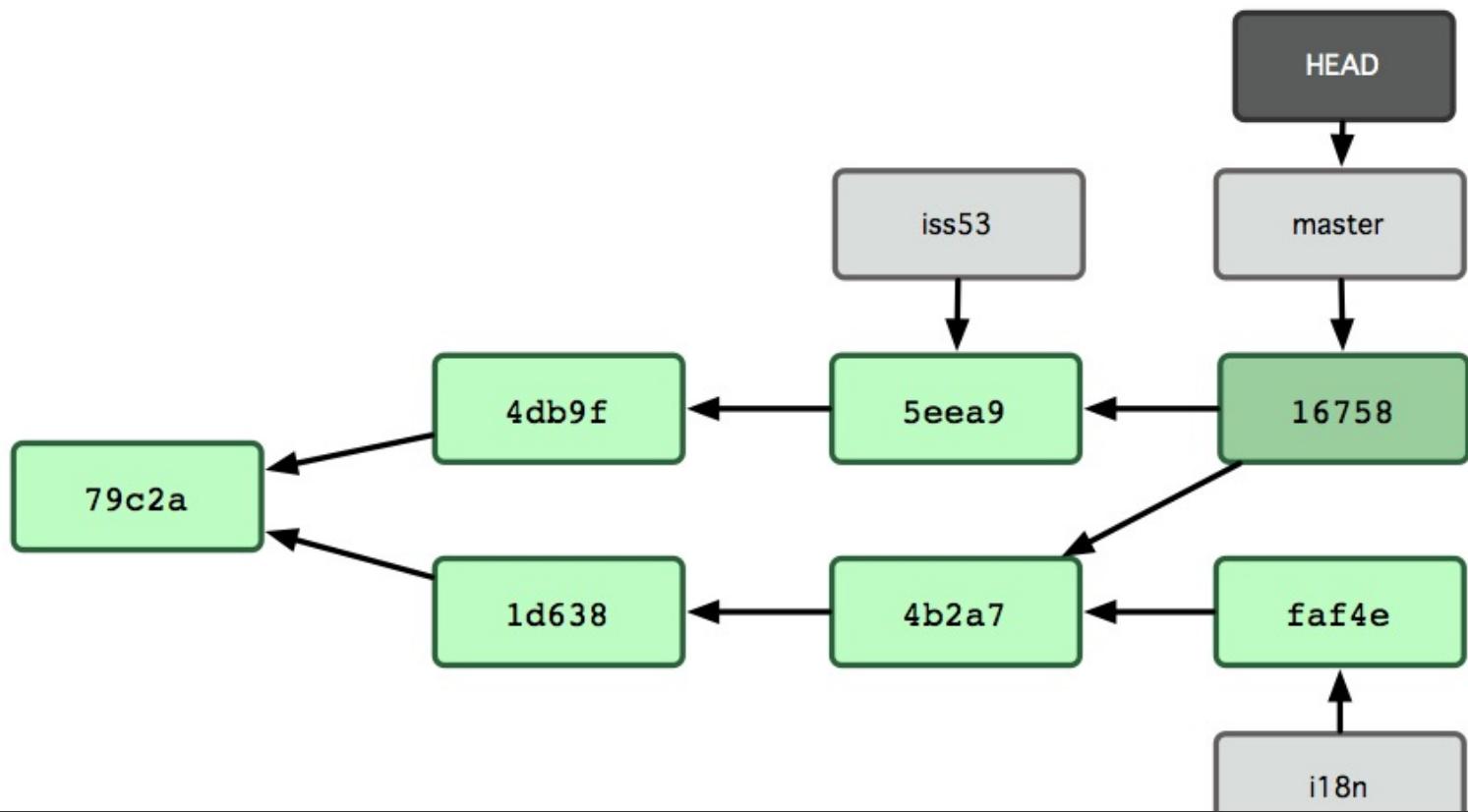


```
$ git lol iss53 ^i18n
* 5eea9cf (iss53) documented issue file
* 4db9f5c added issue file
```

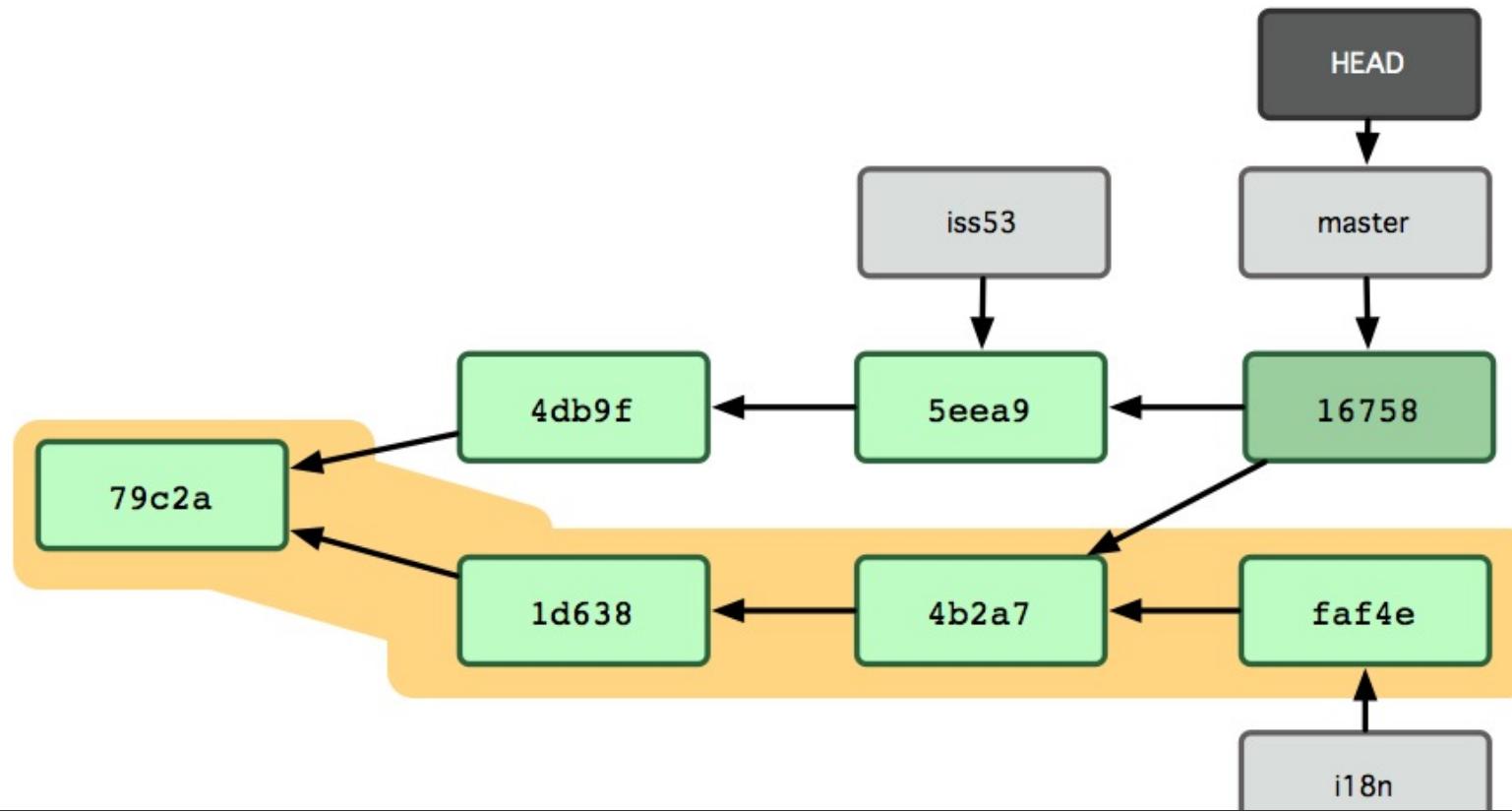


**what work is in our i18n
branch that is not
merged into master yet?**

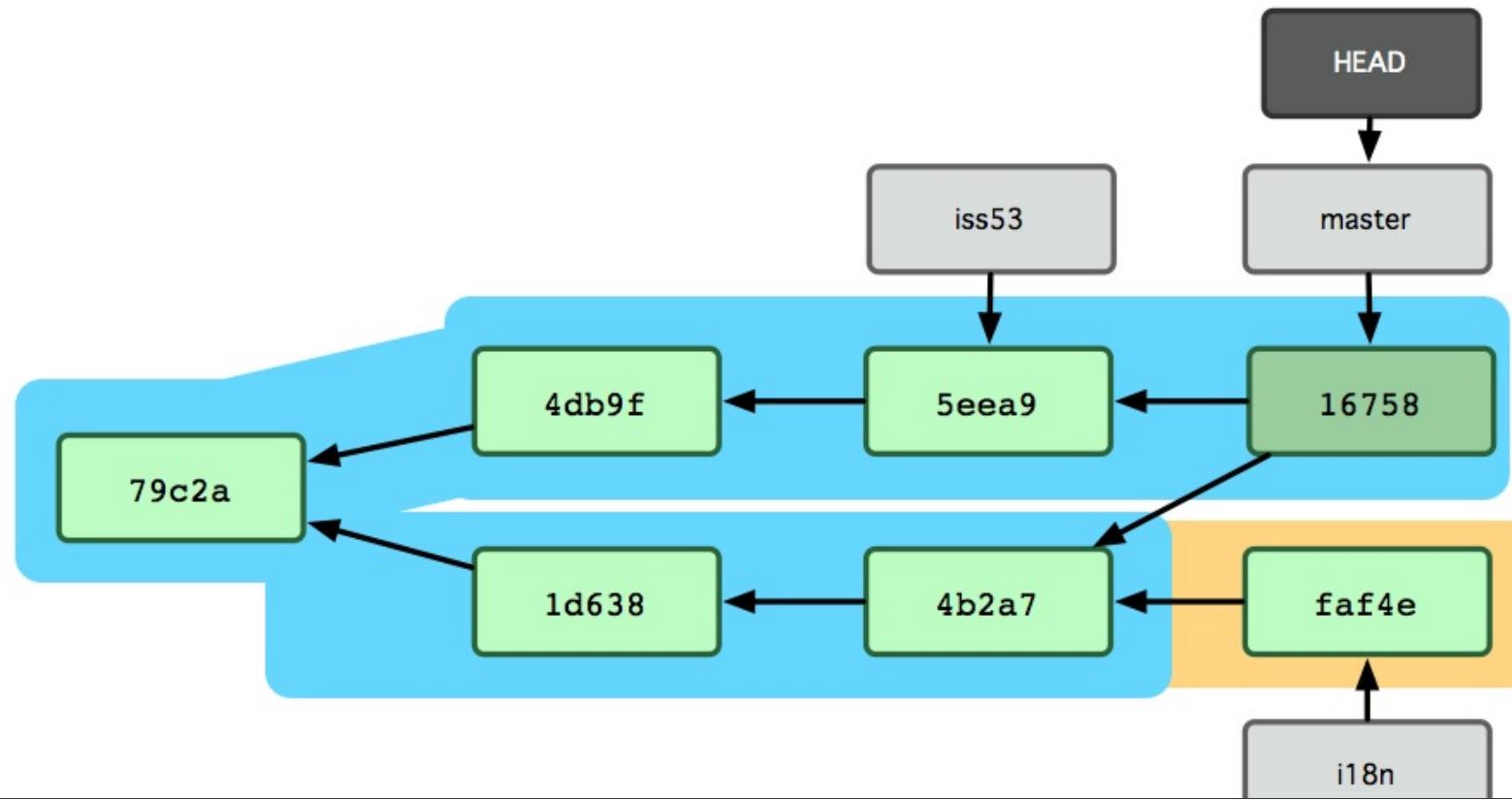
i18n ^master



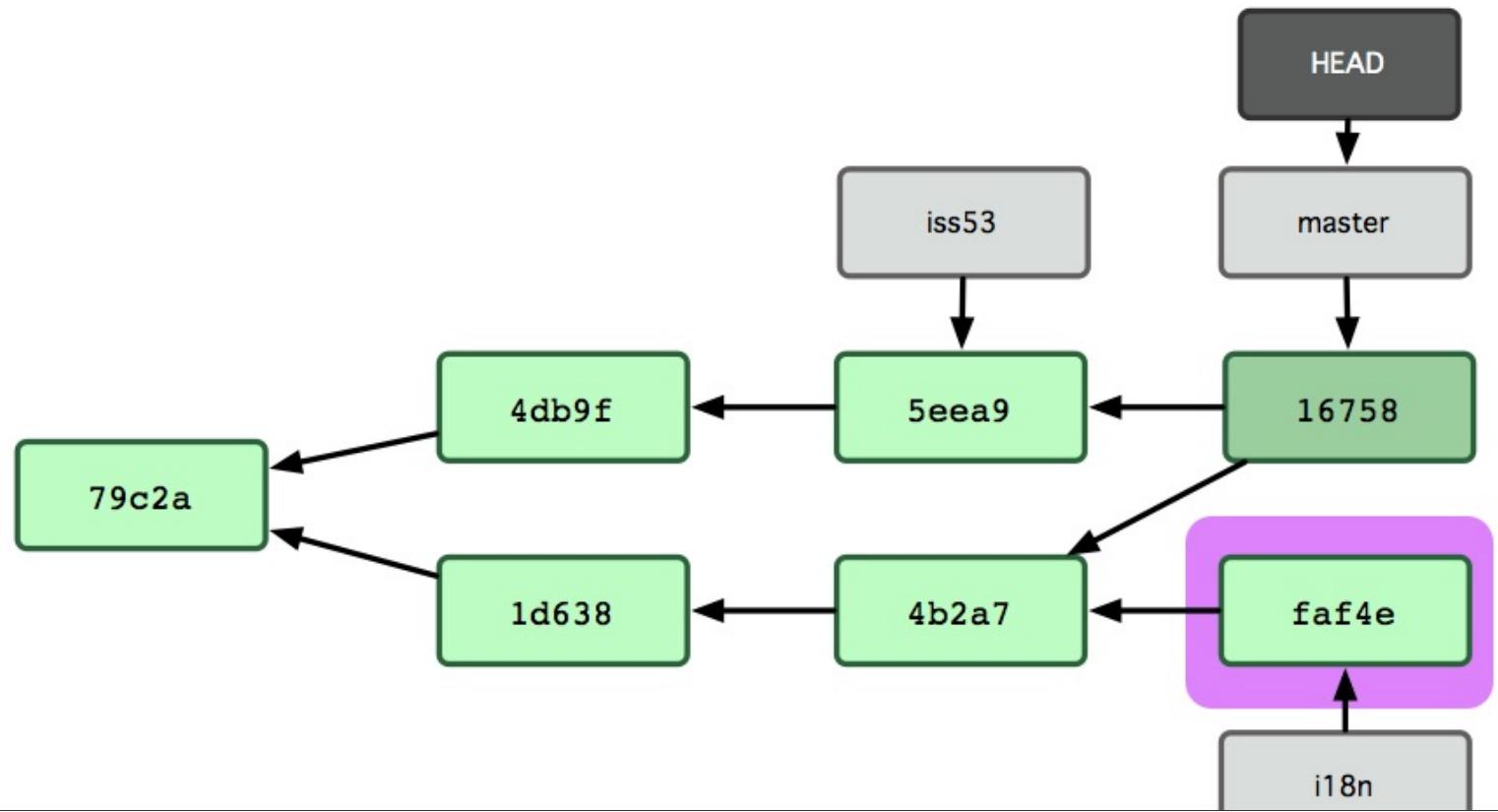
i18n ^master



i18n ^master

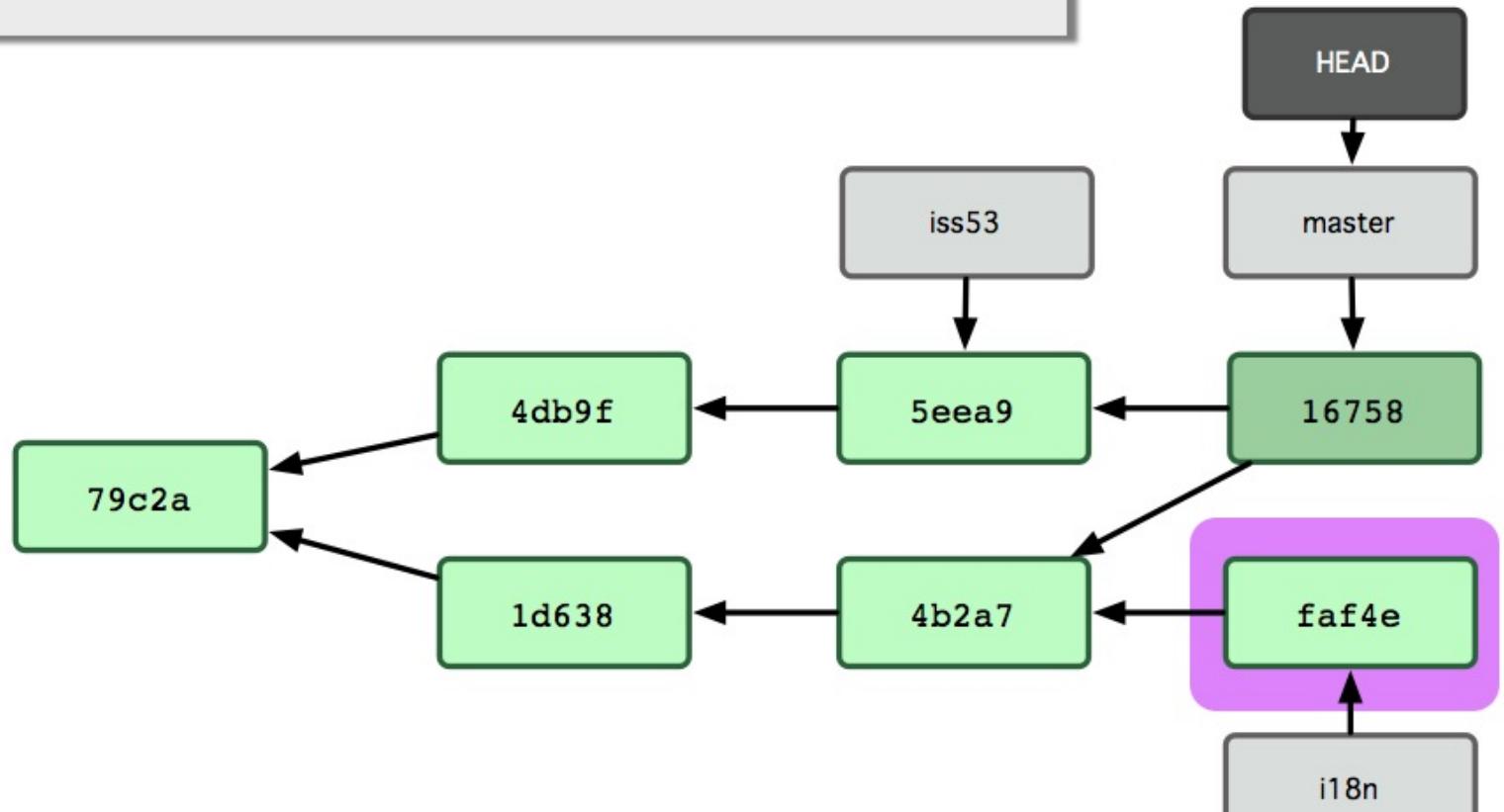


i18n ^master



```
$ git log i18n ^master
commitfaf4ecee82715967b07ff98e88796a311c8761fe
Author: Scott Chacon <schacon@gmail.com>
Date:   Wed Jul 28 16:30:00 2010 -0700

    update README to be more specific
$
```



quiz

```
git log origin/master ^master
```

```
git log master ^origin/master
```

Branch Diffing

```
$ git log --graph --oneline --decorate master experiment
* 420eac9 (experiment) Added a method for getting the current
| * 30e367c (HEAD, master) timeout code and tests
| * 5a09431 add timeout protection to grit
| * e1193f8 support for heads with slashes in them
|/
* d6016bc require time for xmldata
* 11d191e Merge branch 'defunkt' into local
```

```
$ git show --stat 420eac9
commit 420eac97a826bfac8724b6b0eef35c20922124b7
Author: Dustin Sallings <dustin@spy.net>
Date:   Tue Apr 1 10:52:03 2008 -0700
```

Added a method for getting the current branch.

```
lib/grit/head.rb |    16 ++++++-----
lib/grit/repo.rb |      9 ++++++-
2 files changed, 22 insertions(+), 3 deletions(-)
```

git diff master experiment

```
$ git show --stat 420eac9
...
lib/grit/head.rb |    16 ++++++-----
lib/grit/repo.rb |     9 ++++++-
2 files changed, 22 insertions(+), 3 deletions(-)

$ git diff --stat master experiment
lib/grit.rb           |    2 -
lib/grit/git.rb       |   54 +-----+
lib/grit/head.rb      |   19 ++++++-----
lib/grit/repo.rb      |    9 ++++++-
test/fixtures/for_each_ref | Bin 126 -> 58 bytes
test/test_git.rb       |   31 -----
test/test_head.rb      |   13 +-----+
7 files changed, 34 insertions(+), 94 deletions(-)
```

```
$ git log --graph --oneline --decorate master experiment
* 420eac9 (experiment) Added a method for getting the current
| * 30e367c (HEAD, master) timeout code and tests
| * 5a09431 add timeout protection to grit
| * e1193f8 support for heads with slashes in them
|/
* d6016bc require time for xschema
* 11d191e Merge branch 'defunkt' into local
```

```
git diff master...experiment
```

```
git diff master...experiment  
git diff $(git merge-base master experiment) experiment
```

```
$ git diff --stat master...experiment
 lib/grit/head.rb |    16 ++++++-----
 lib/grit/repo.rb |     9 ++++++-
 2 files changed, 22 insertions(+), 3 deletions(-)
```

```
$ git show --stat experiment
commit 420eac97a826bfac8724b6b0eef35c20922124b7
Author: Dustin Sallings <dustin@spy.net>
Date:   Tue Apr 1 10:52:03 2008 -0700
```

Added a method for getting the current branch.

```
lib/grit/head.rb |    16 ++++++-----
 lib/grit/repo.rb |     9 ++++++-
 2 files changed, 22 insertions(+), 3 deletions(-)
```

Tutorial

find which commits introduced lines with the
phrase 'raisins' in them

how many commits were made in the month of
march?

how many commits were made by matt?

Tutorial

find the difference in the sweet_potatoes branch
from the master branch

find the difference in the master branch from the
sweet_potatoes branch

find the commits in the sweet_potatoes branch
not in the master branch

find the commits in the master branch not in the
sweet_potatoes branch

Bundles

```
$ git log
commit 9a466c572fe88b195efd356c3f2bbeccdb504102
Author: Scott Chacon <schacon@gmail.com>
Date:   Wed Mar 10 07:34:10 2010 -0800
```

second commit

```
commit b1ec3248f39900d2a406049d762aa68e9641be25
Author: Scott Chacon <schacon@gmail.com>
Date:   Wed Mar 10 07:34:01 2010 -0800
```

first commit

```
$ git bundle create repo.bundle master
Counting objects: 6, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (6/6), 441 bytes, done.
Total 6 (delta 0), reused 0 (delta 0)

$ ls -lh repo.bundle
-rw-r--r-- 1 schacon schacon 6.4K Aug 12 09:45 repo.bundle
```

email repo.bundle to someone

```
$ git ls-remote repo.bundle
9a466c572fe88b195efd356c3f2bbeccdb504102      refs/heads/master

$ git clone repo.bundle -b master repo
Cloning into repo...

$ cd repo
$ git log --oneline
9a466c5 second commit
b1ec324 first commit
```

```
$ git log --oneline
```

```
71b84da last commit - second repo
c99cf5b fourth commit - second repo
7011d3d third commit - second repo
9a466c5 second commit
blec324 first commit
```

```
$ git log --oneline master ^origin/master
```

```
71b84da last commit - second repo
c99cf5b fourth commit - second repo
7011d3d third commit - second repo
```

```
$ git bundle create commits.bundle master ^origin/master
```

```
Counting objects: 11, done.
```

```
Delta compression using up to 2 threads.
```

```
Compressing objects: 100% (3/3), done.
```

```
Writing objects: 100% (9/9), 775 bytes, done.
```

```
Total 9 (delta 0), reused 0 (delta 0)
```

```
$ ls -lh commits.bundle
```

```
-rw-r--r-- 1 maddog maddog 16.2K Aug 13 12:32 commits.bund
```

email commits.bundle to
original author

```
$ git bundle verify ../commits.bundle
The bundle contains 1 ref
71b84daaf49abed142a373b6e5c59a22dc6560dc refs/heads/master
The bundle requires these 1 ref
9a466c572fe88b195efd356c3f2bbeccdb504102 second commit
../commits.bundle is okay
```

```
$ git bundle verify ../commits-bad.bundle
error: Repository lacks these prerequisite commits:
error: 7011d3d8fc200abe0ad561c011c3852a4b7bbe95 third commit
```

```
$ git fetch ../commits.bundle master:other-master
From ../commits.bundle
 * [new branch]          master      -> other-master

$ git log --oneline --decorate --graph --all
* 8255d41 (HEAD, master) third commit - first repo
| * 71b84da (other-master) last commit - second repo
| * c99cf5b fourth commit - second repo
| * 7011d3d third commit - second repo
| /
* 9a466c5 second commit
* b1ec324 first commit
```

Branch Management

```
$ git branch
  android
  iss53
  iss182
* master
  zindex

$ git branch --merged
  iss53
* master
  zindex

$ git branch --no-merged
  android
  iss182
```

```
$ git branch --contains 7e4a8bd2b  
iss53
```

Tutorial

find which branches in the recipes repo are
already merged

find which branches are not yet merged

The Refspec

```
git remote add origin user@server:repo.git
```

```
.git/config
```

```
[remote "origin"]
  url = user@server:repo.git
  fetch = +refs/heads/*:refs/remotes/origin/*
```

```
$ tree .git/refs
.git/refs/
├── heads
│   ├── experiment
│   ├── master
│   └── testing
└── remotes
    ├── personal
    │   └── master
    └── origin
        ├── experiment
        └── master
└── tags
    └── v1.0

$ cat .git/refs/heads/master
e65062eeb3f54a2265a03b0a196480bd5266d5ea
```

master

heads/master

refs/heads/master

`origin/master`

`remotes/origin/master`

`refs/remotes/origin/master`

```
.git/config
```

```
[remote "origin"]
  url = user@server:repo.git
  fetch = +refs/heads/*:refs/remotes/origin/*
          +[src] : [dest]
```

```
$ tree .git/refs
.git/refs/
├── heads
│   ├── experiment
│   ├── master
│   └── testing
└── remotes
    ├── personal
    │   └── master
    └── origin
        ├── experiment
        └── master
└── tags
    └── v1.0
```

'+' updates the reference even if it is not a fast-forward

```
.git/config

[remote "origin"]
  url = user@server:repo.git
  fetch = +refs/heads/*:refs/remotes/origin/*

[remote "personal"]
  url = user@chacon.com:myrepo.git
  fetch = +refs/heads/master:refs/remotes/origin/my_master

[remote "github"]
  url = git@github.com:schacon/repo.git
  fetch = +refs/heads/*:refs/remotes/github/br/*
```

```
git fetch origin +master:refs/remotes/origin/master
```

```
$ git fetch origin master:refs/remotes/origin/mymaster \
  topic:refs/remotes/origin/topic
From git@github.com:schacon/simplegit
 ! [rejected]           master      -> origin/mymaster (non fast
* [new branch]          topic       -> origin/topic

$ cat .git/config
[remote "origin"]
  url = git@github.com:schacon/repo.git
  fetch = +refs/heads/master:refs/remotes/origin/master
  fetch = +refs/heads/topic:refs/remotes/origin/topic
```

Pushing Refspecs

```
$ git push origin master:another_name
Total 0 (delta 0), reused 0 (delta 0)
To git@github.com:schacon/repo.git
 * [new branch]      master -> another_name

$ git push origin :another_name
To git@github.com:schacon/repo.git
 - [deleted]          another_name

$ cat .git/config
[remote "origin"]
  url = user@server:repo.git
  fetch = +refs/heads/*:refs/remotes/origin/*
  push = +refs/heads/master:refs/heads/another_name
[remote "github"]
  url = git@github.com:schacon/repo.git
  fetch = +refs/heads/*:refs/remotes/origin/*
  push = refs/heads/*:refs/heads/*
```

Tutorial

fetch from the server all the refs in refs/secret

Revision Selection

Revision Selection

full sha-1

partial sha-1

branch or tag name

caret parent

tilde spec

blob spec

Full SHA1

6e453f523fa1da50ecb04431101112b3611c6a4d

Partial SHA1

6e453f523fa1da50ecb04431101112b3611c6a4d

6e453f523fa1da50

6e453

Branch, Remote or Tag Name

v1.0

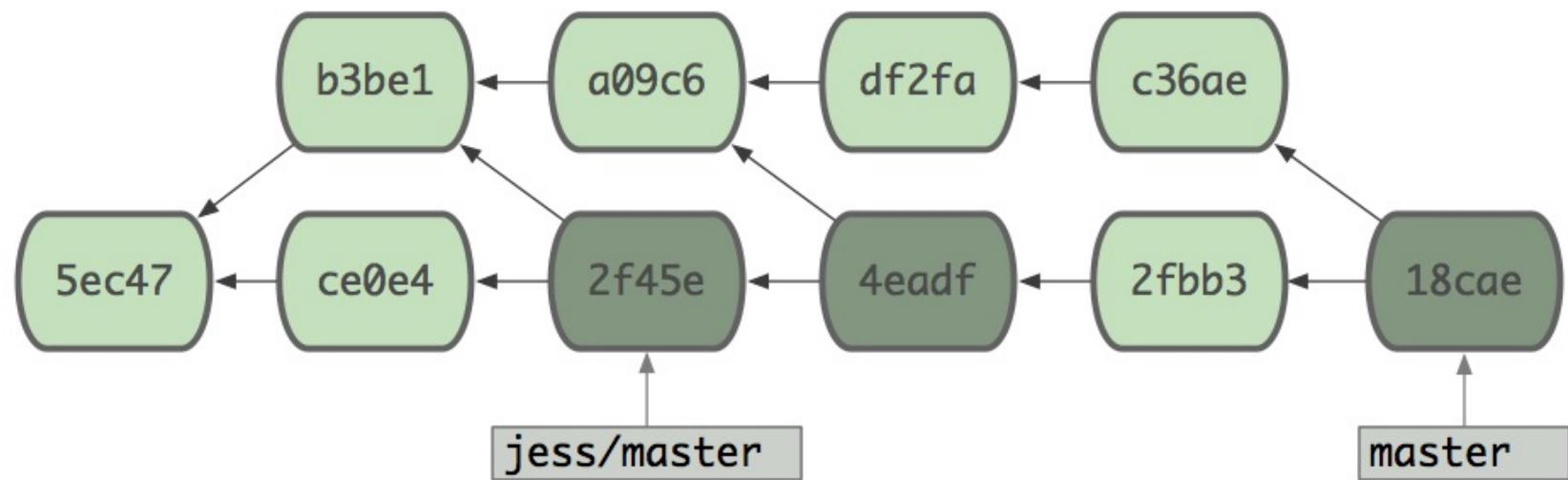
default

m/cupcake

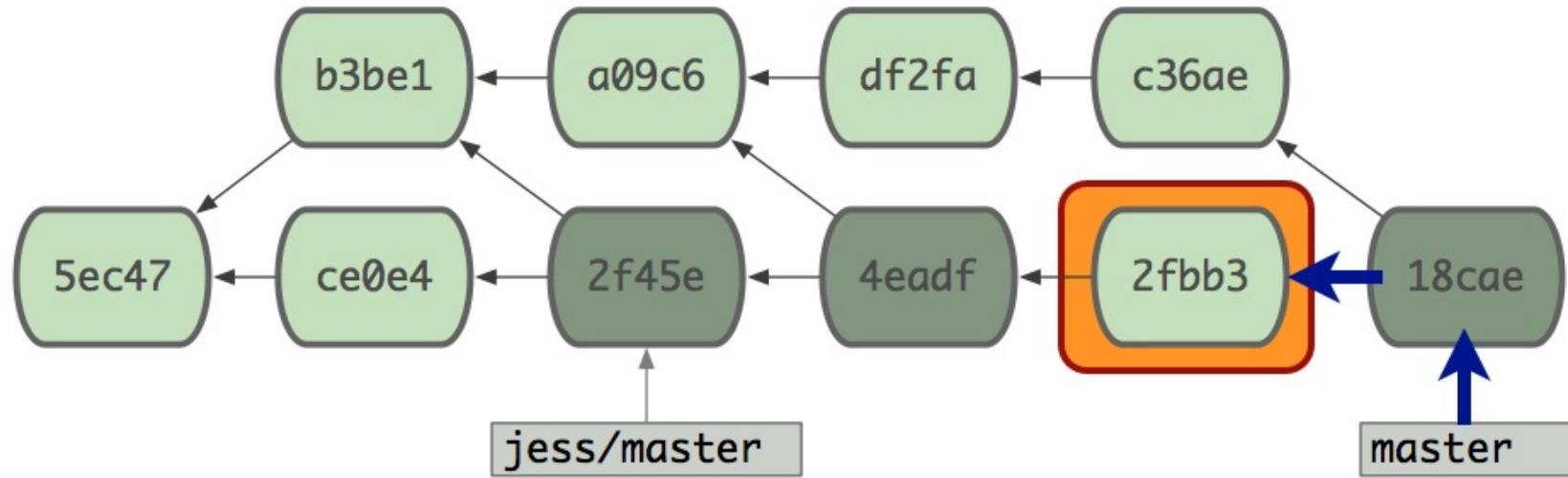
Caret Parent

default^{^2}

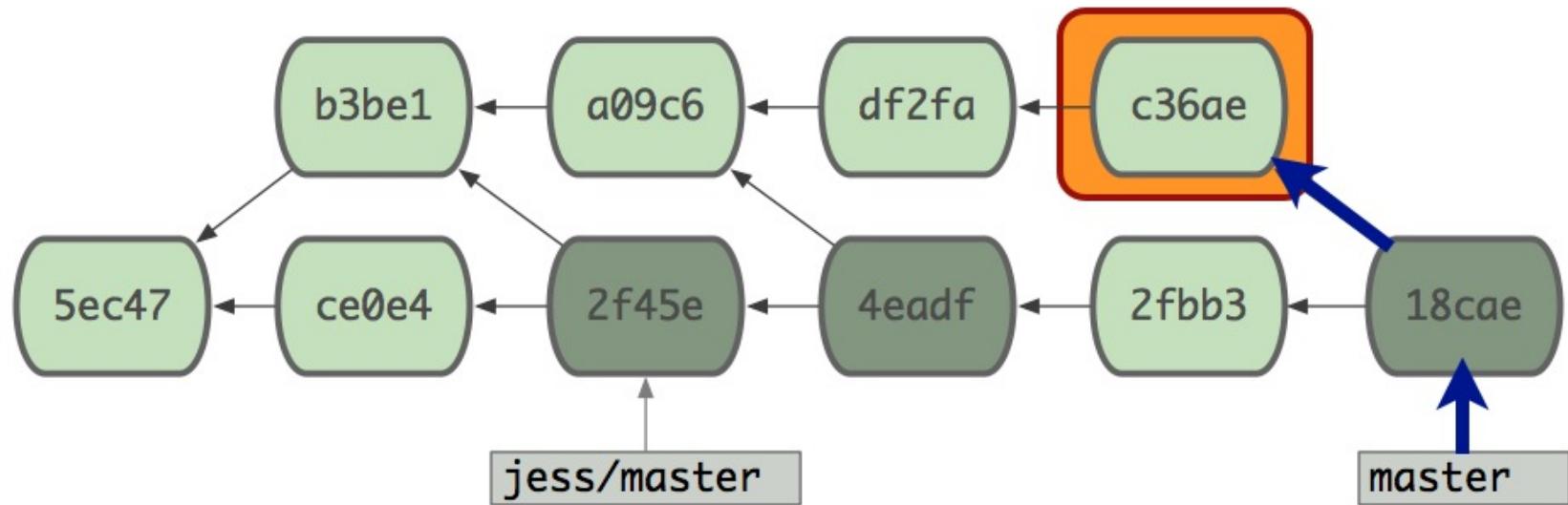
2nd parent of 'default'



master^Λ



master[^]2

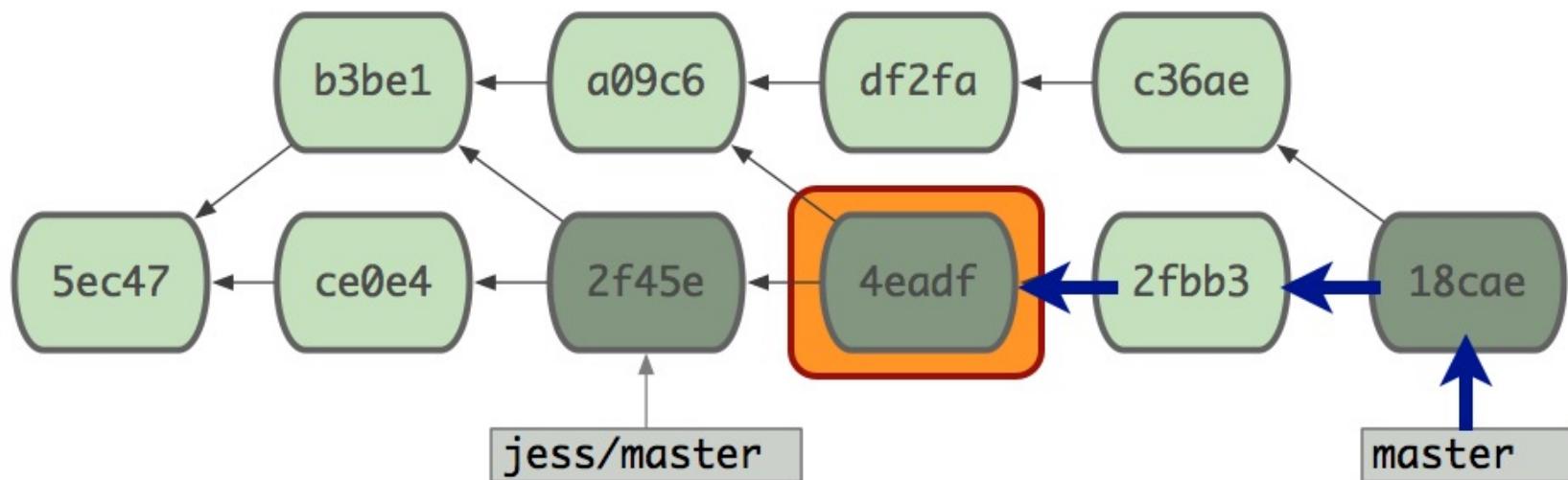


Tilde Spec

default~2

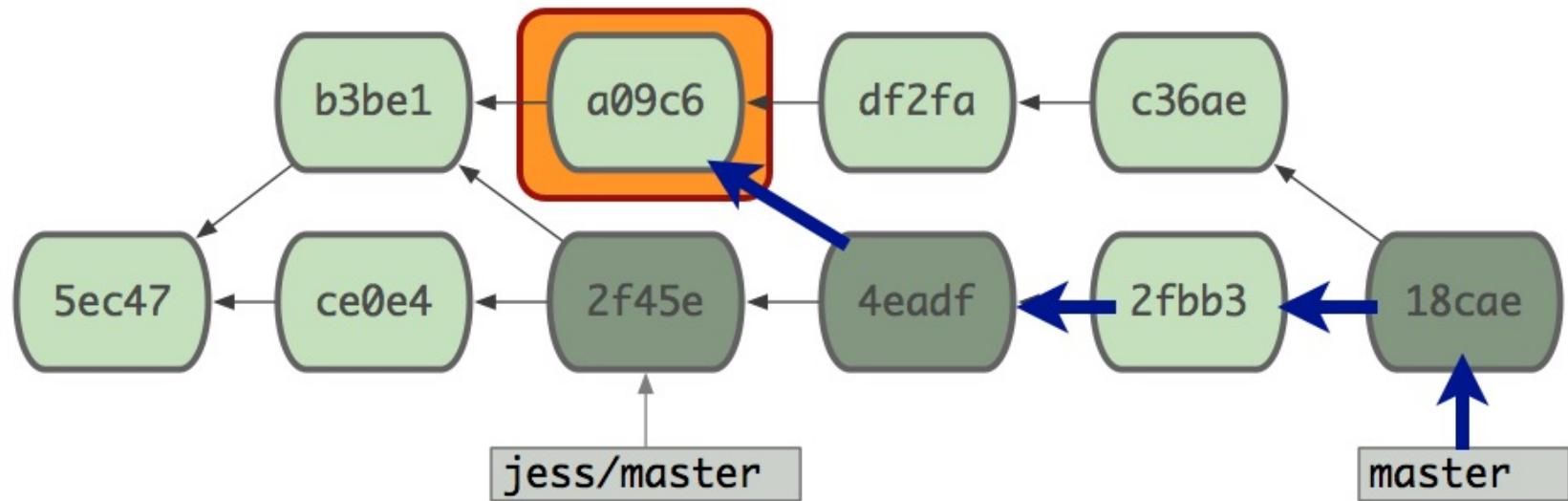
grandparent of ‘default’
(parent of the parent)

master~2

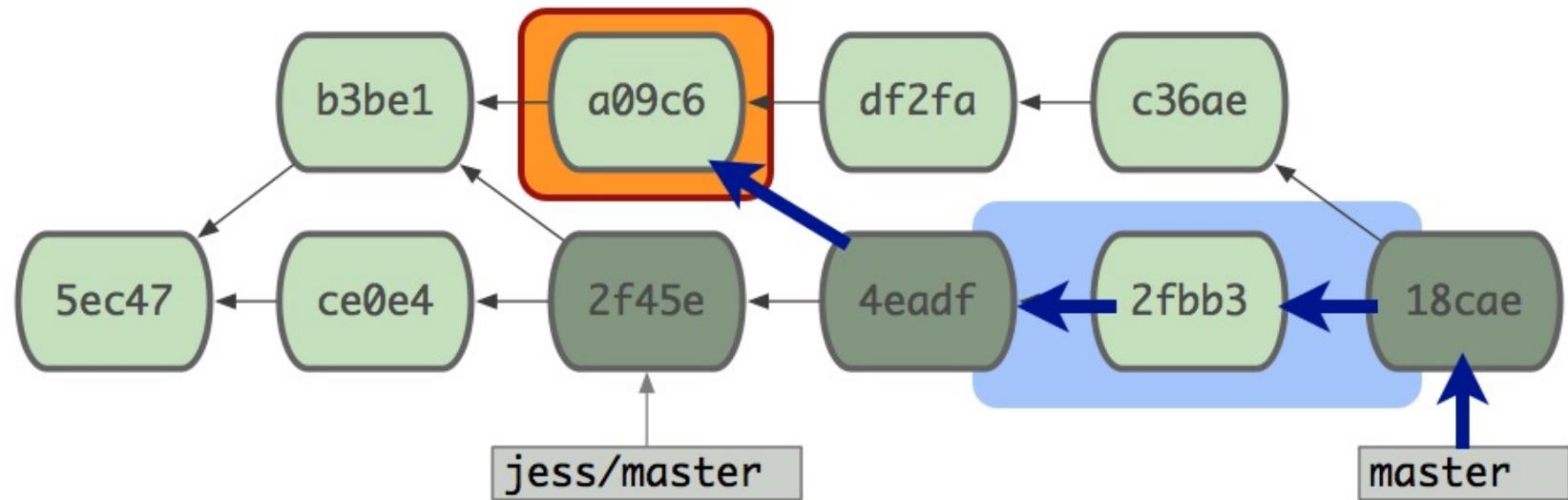


master^{^2}

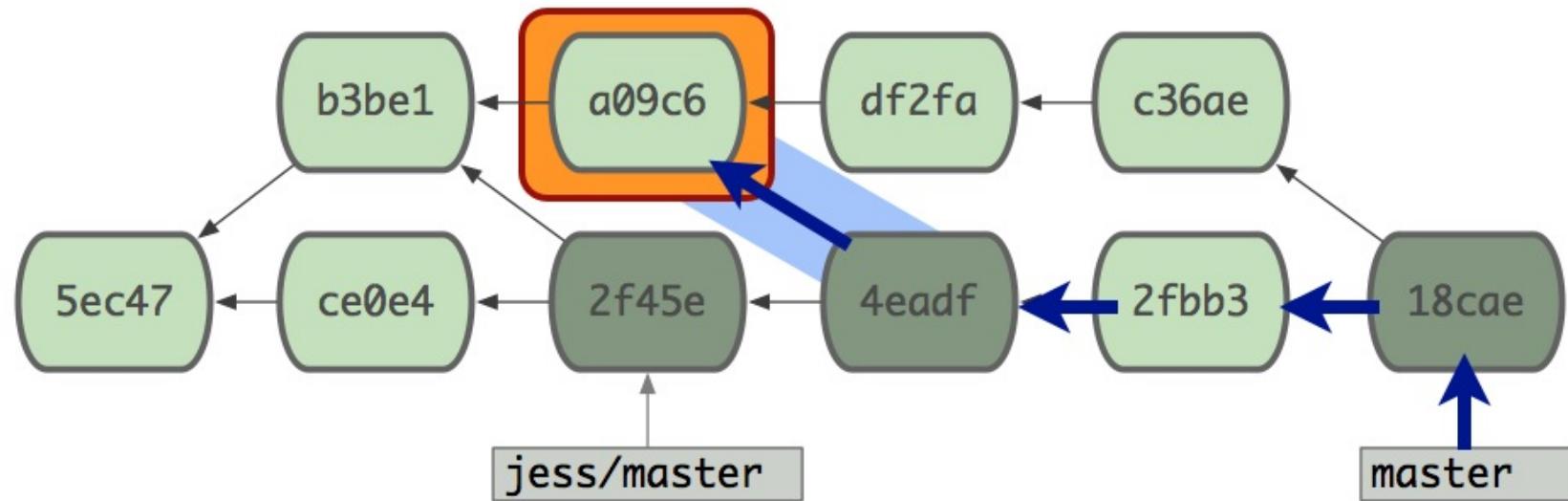
master~2^2



master^{^2}
master~2^2



master^{^2}
master~2^{^2}



Blob Spec

v1.1:README.txt

Relative Spec

master@{1 month 2 weeks 3 days 1 hour ago}

Reflog Spec

master@{5}

fifth prior value of master

Colon Syntax

```
$ git show :/fix
```

```
# shows the last commit which has the  
word "fix" in its message
```

```
$ git show :/love
commit 1eee16988ed737b1805a5ed022bfa3f37dce8da5
Author: rick <technoweenie@gmail.com>
Date:   Wed Aug 11 20:43:57 2010 -0700

    love ie8

diff --git a/public/javascripts/github/editbar.js b/public/javascripts/github/editbar.js
index d461822..d6c2ad4 100644
--- a/public/javascripts/github/editbar.js
+++ b/public/javascripts/github/editbar.js
@@ -170,7 +170,7 @@ $(function(){
    var classes = $(this).attr('class').split(' ')
    var name = classes[0]
    var format = $('#guides .write select#wiki_format option')
-   if (classes.indexOf('gollum') == -1) {
+   if ($(this).hasClass('gollum')) {
        $('#editbar .sections .page.' + name + '.' + format).addClass('current')
    } else {
        $('#editbar .sections .page.' + name).addClass('current')
```

```
$ git show :/^Merge  
# shows the last merge commit
```

```
$ git show :/^Merge
commit 1549652ce43f07ad53baaa2ce4898a7df1a3d727
Merge: ec4b82b 3e92ceb
Author: Ryan Tomayko <rtomayko@gmail.com>
Date:   Thu Aug 12 04:05:27 2010 -0700

Merge branch 'locale-whole-hash-cache'
```

man git-rev-parse

progit.org/book/ch6-1.html

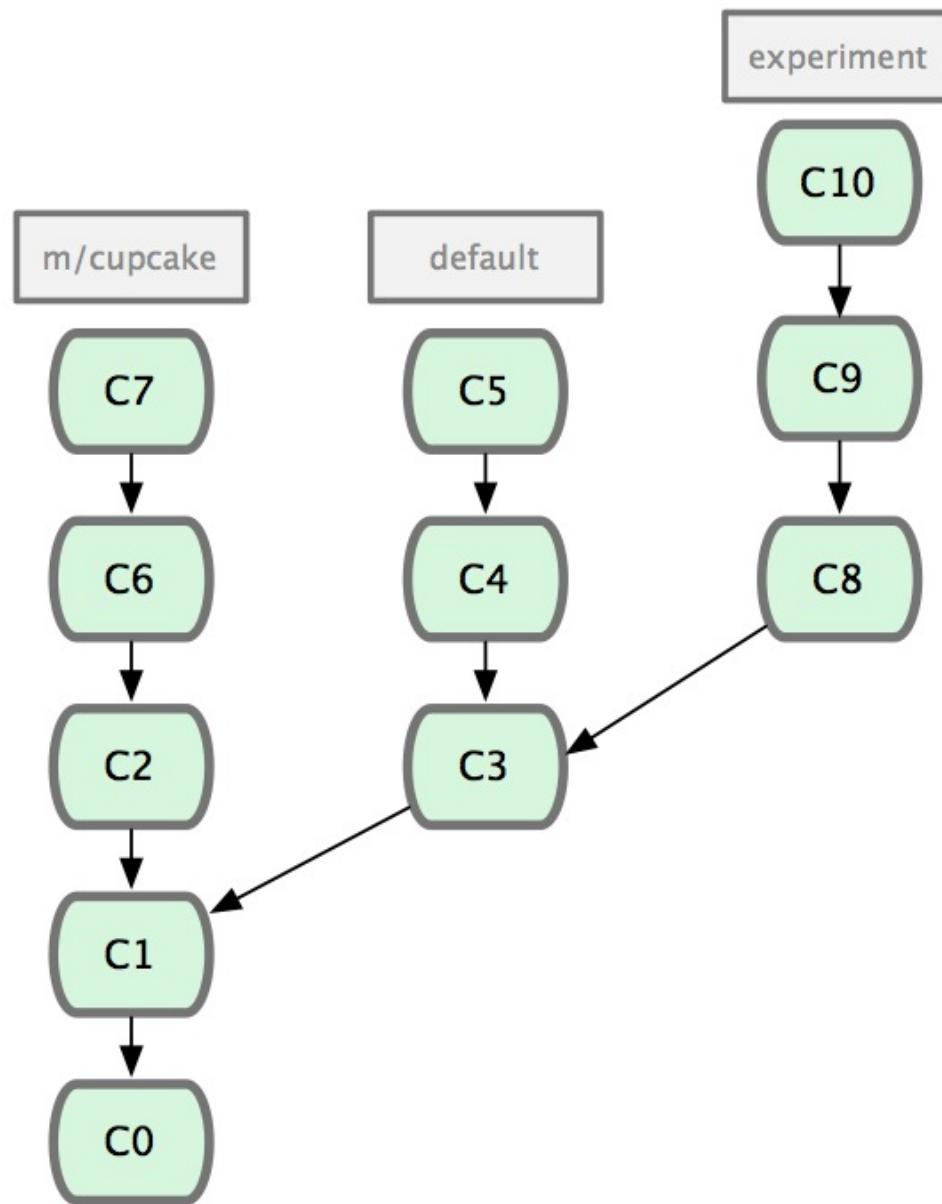
Rewriting History

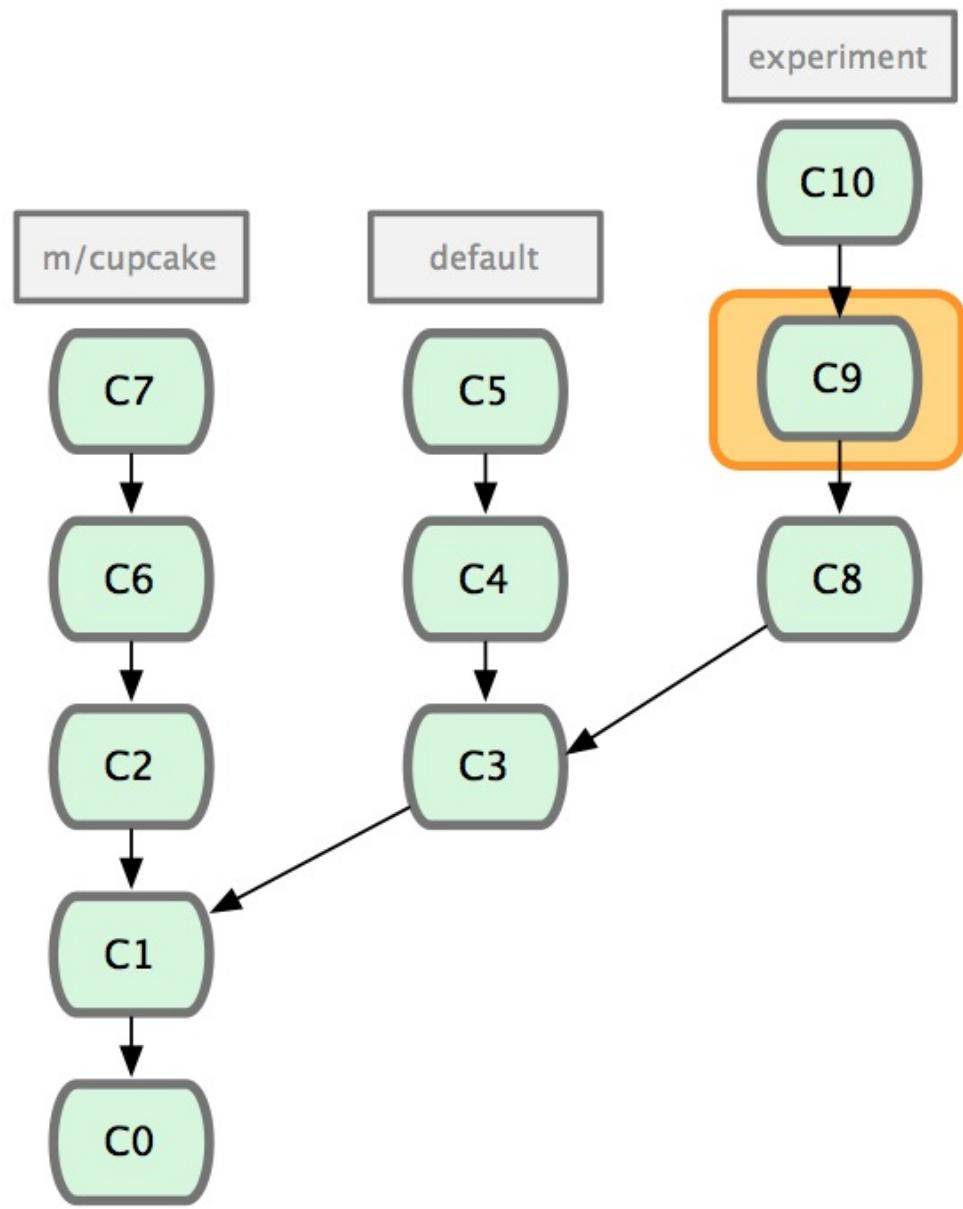
Commit Amending

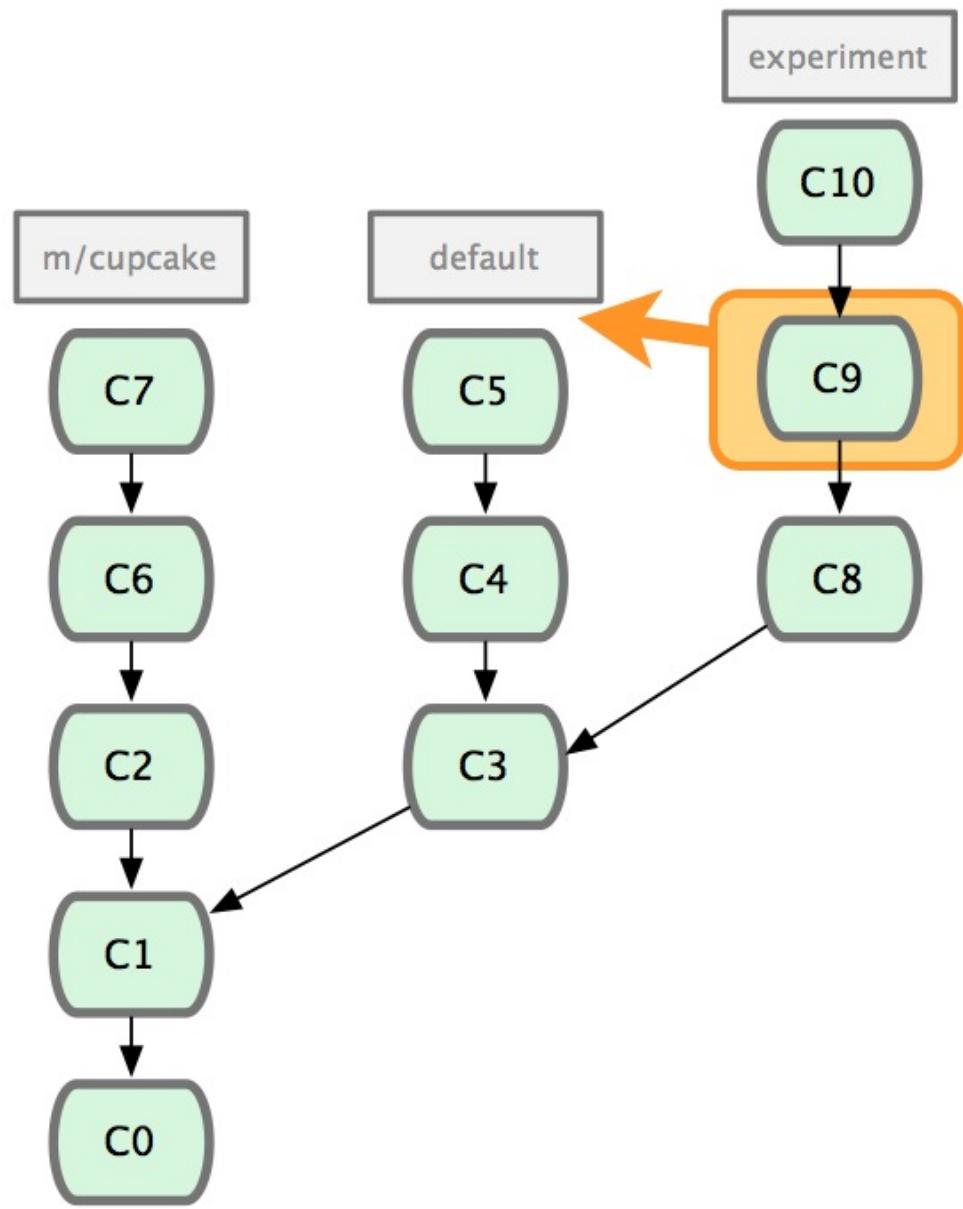
```
git commit --amend
```

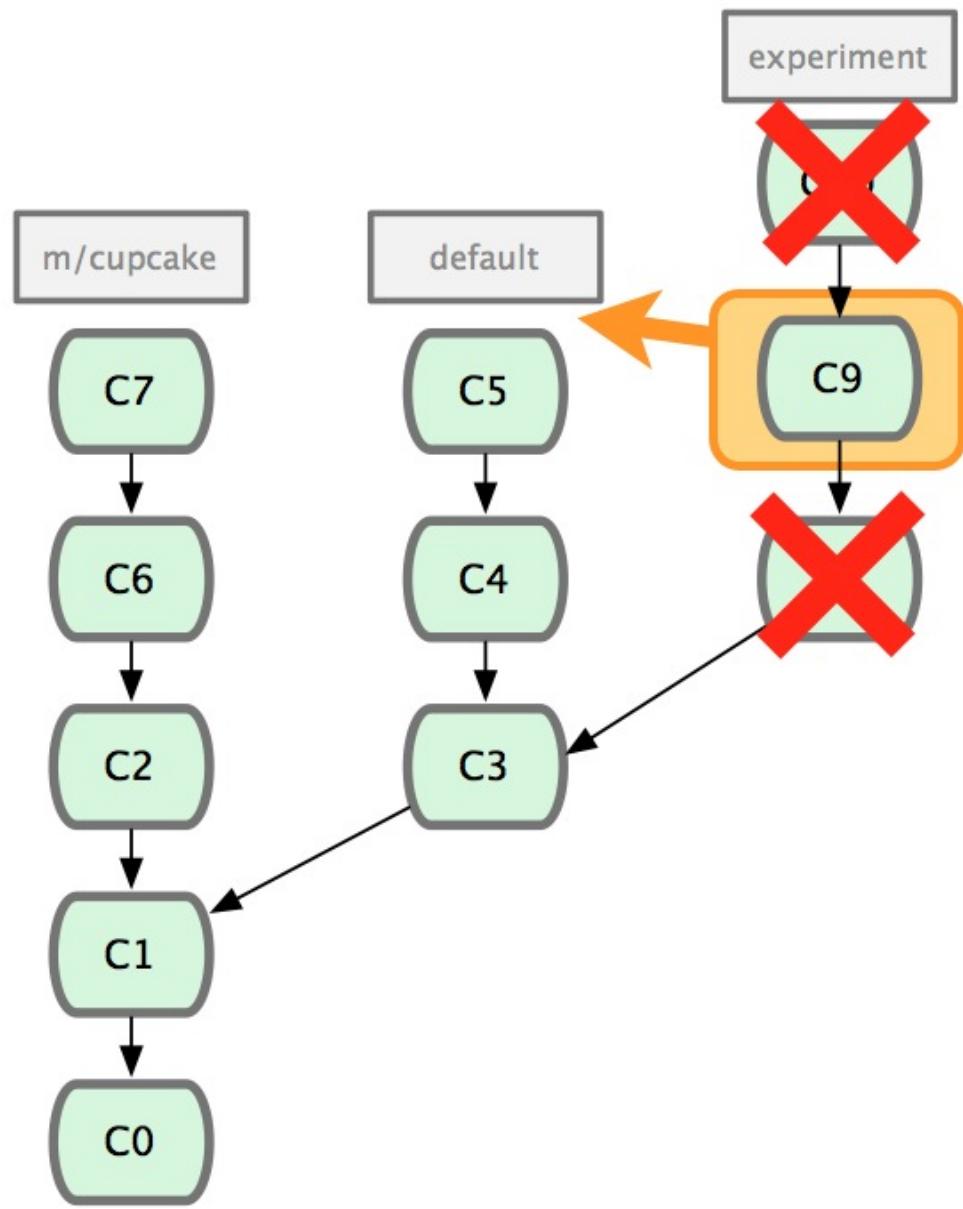
```
git reset --soft HEAD~  
git commit
```

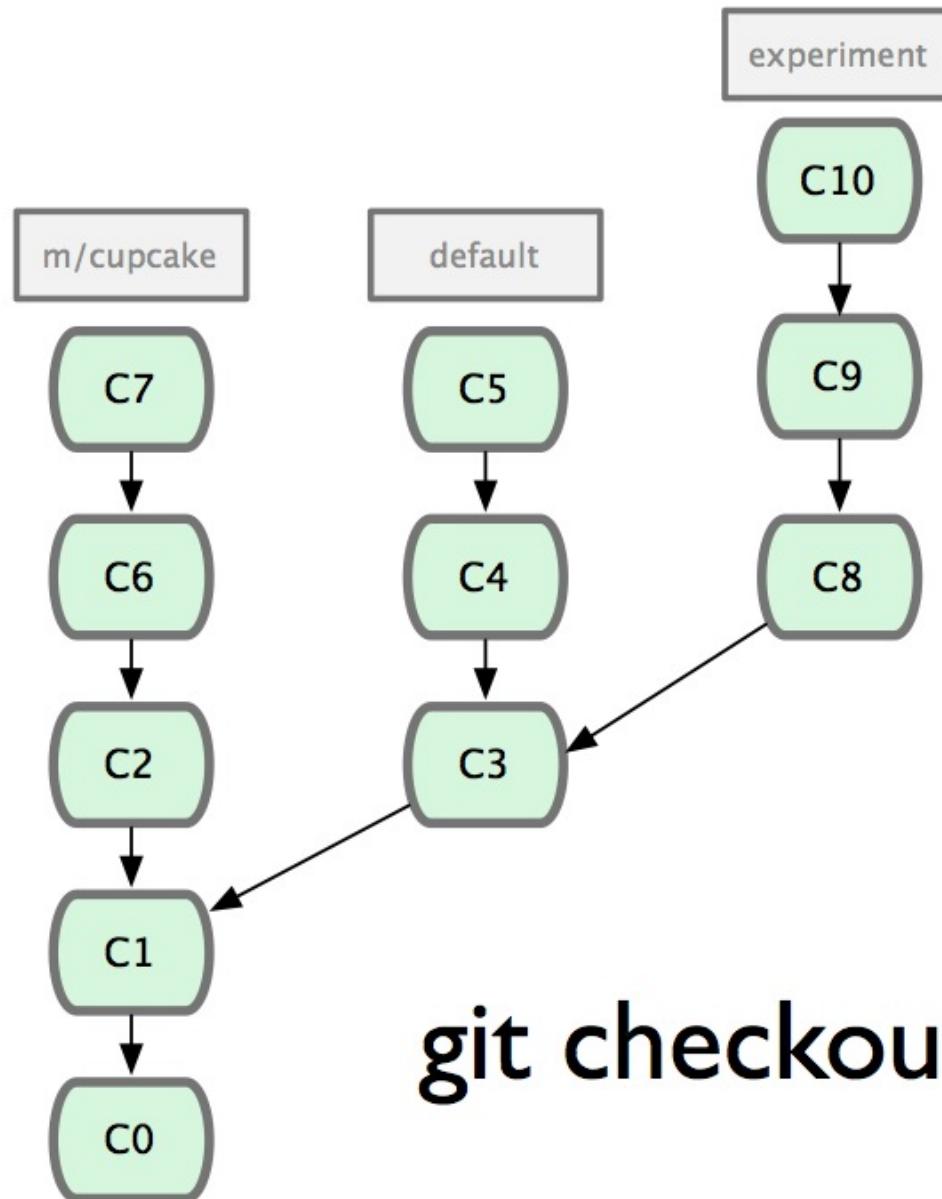
Cherry Picking



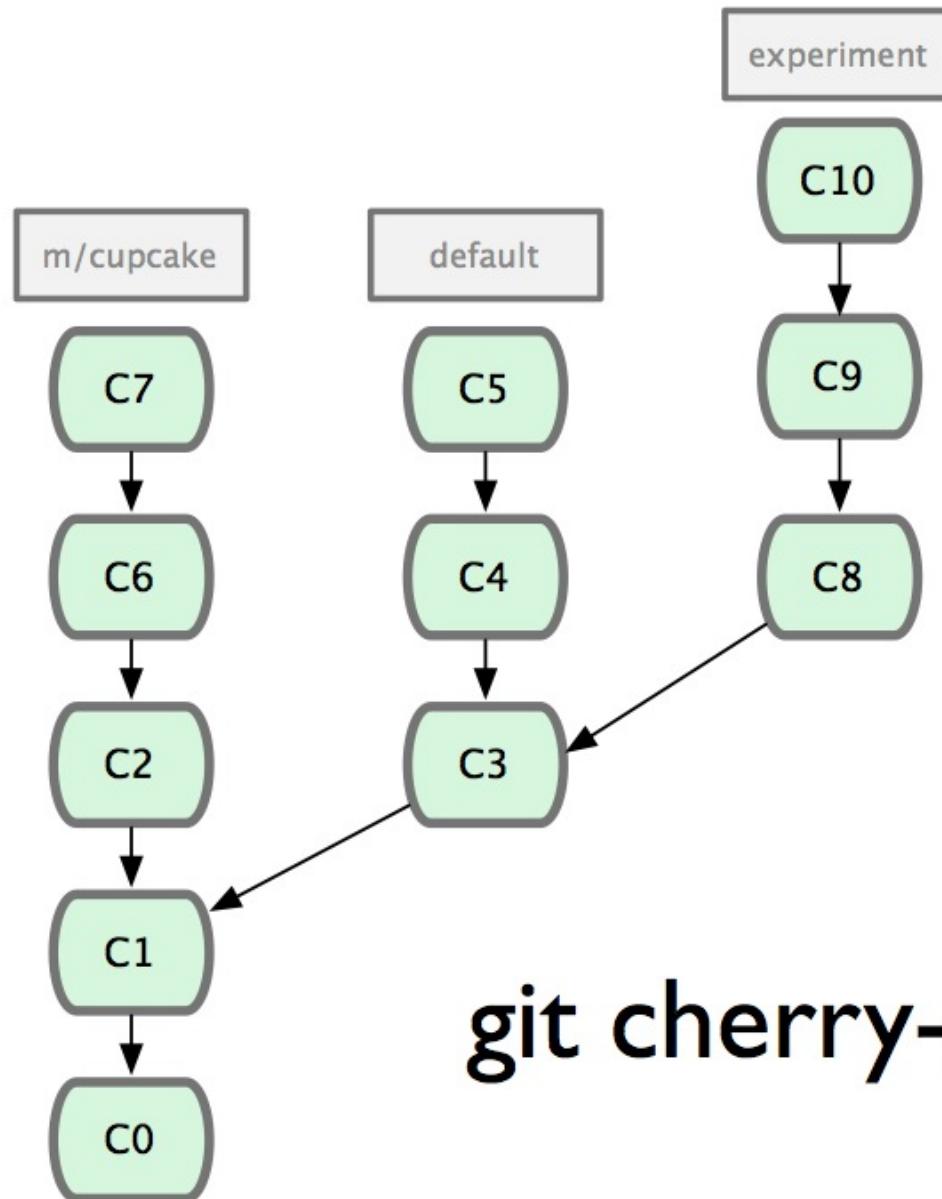




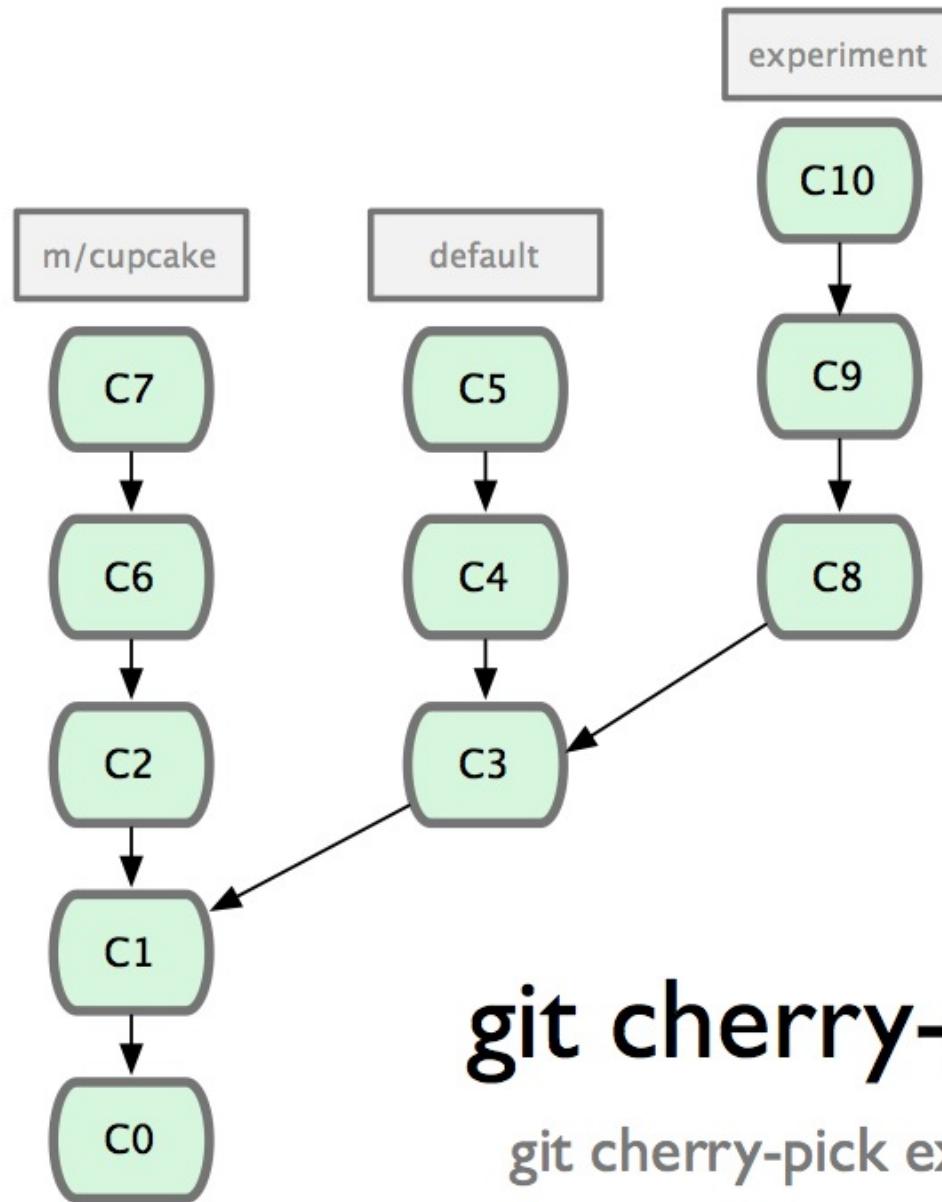




git checkout default

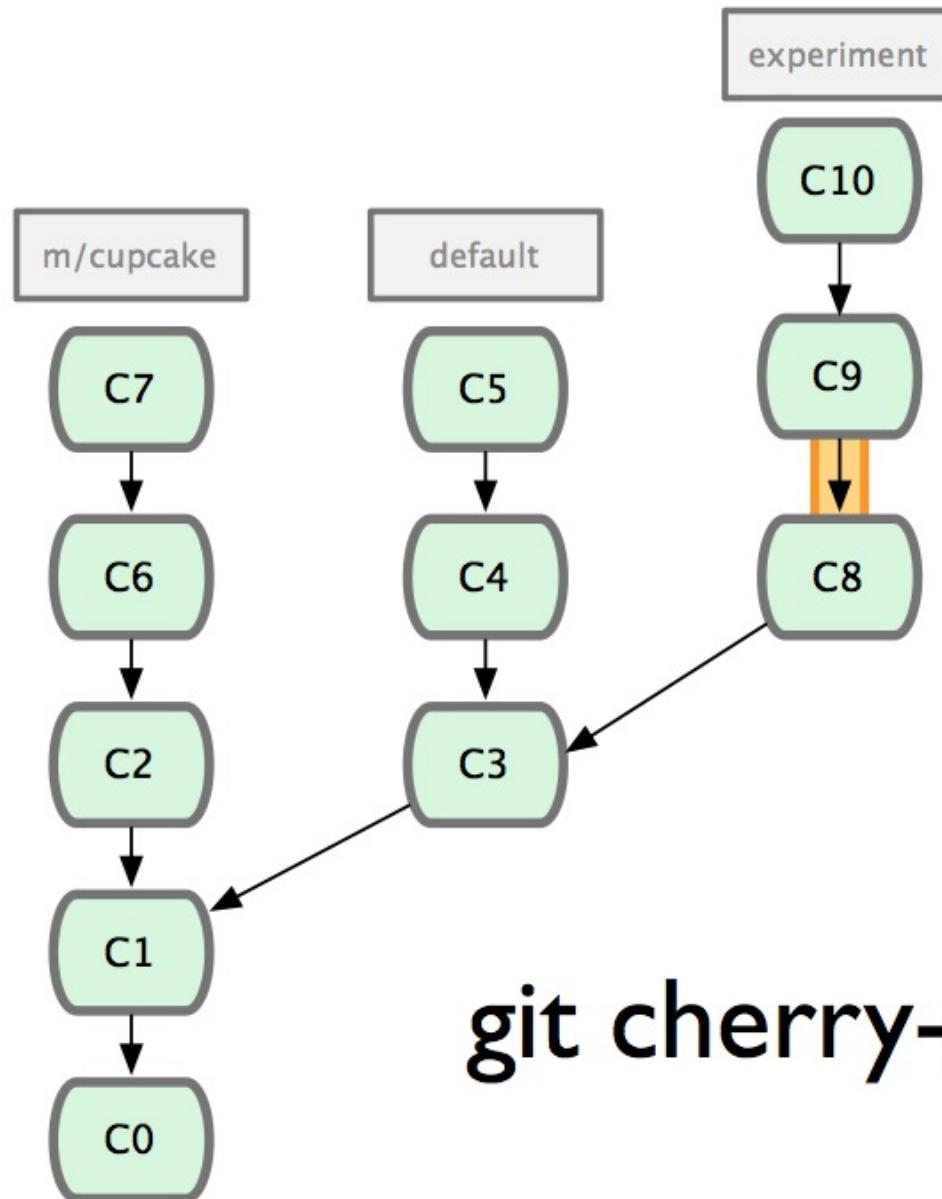


git cherry-pick c9

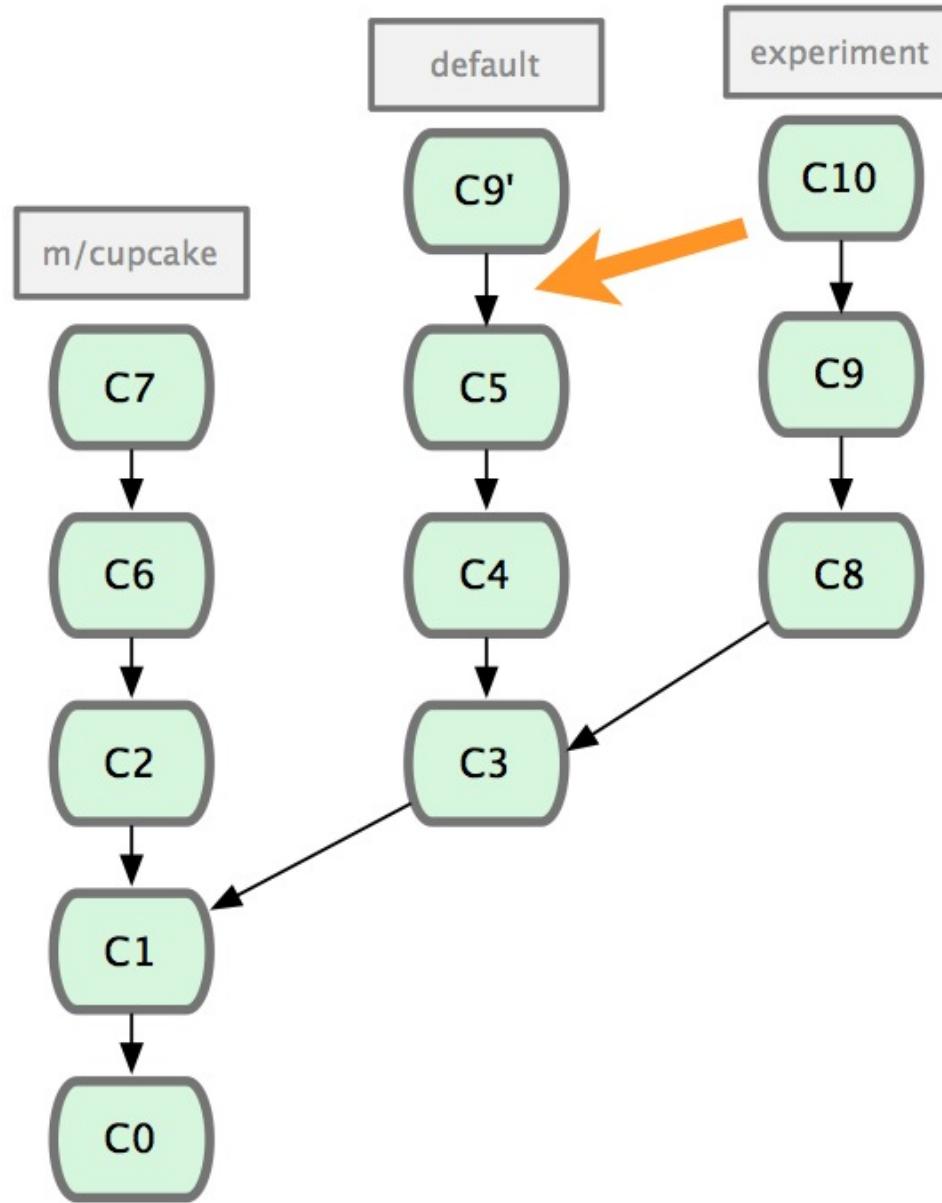


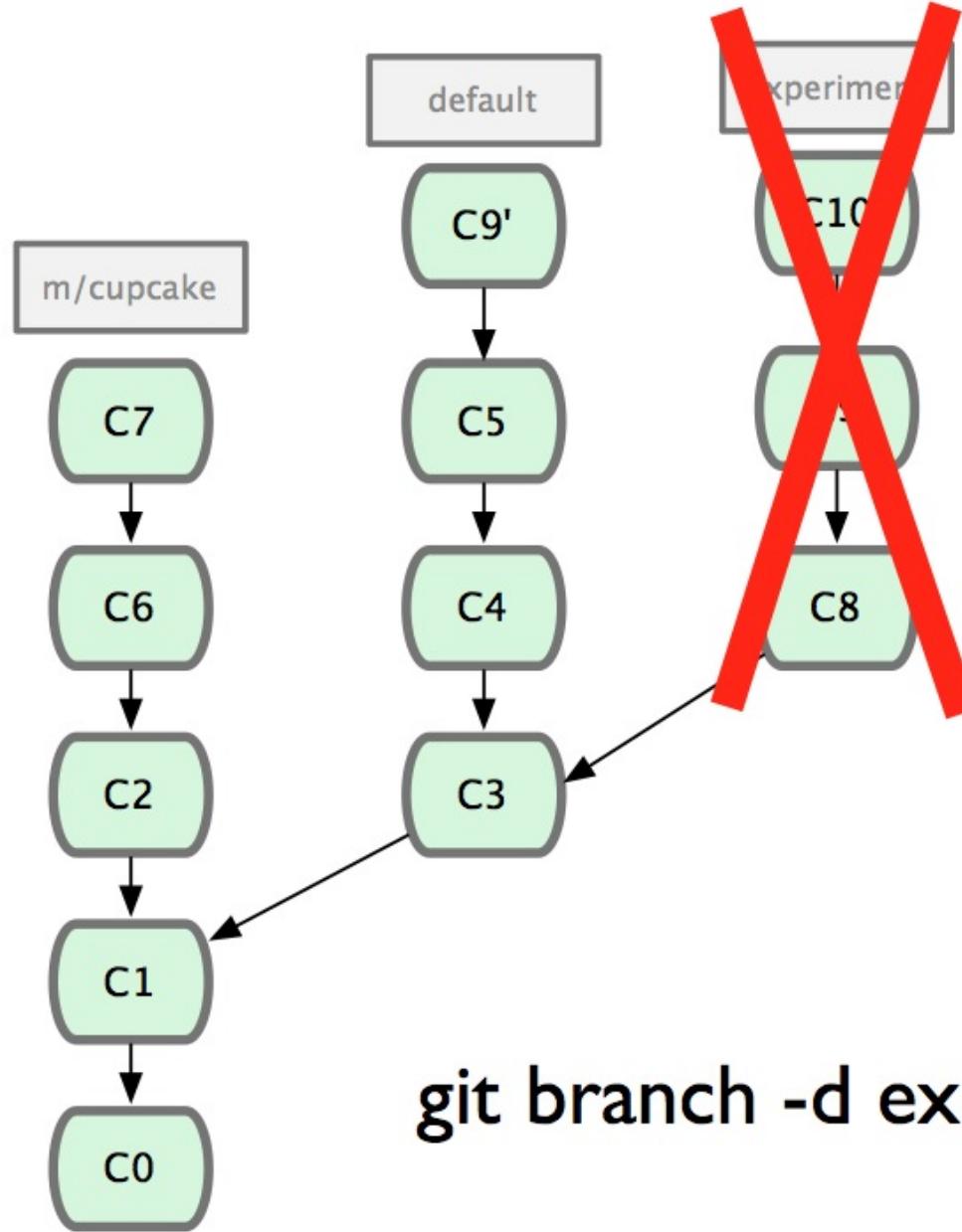
git cherry-pick c9

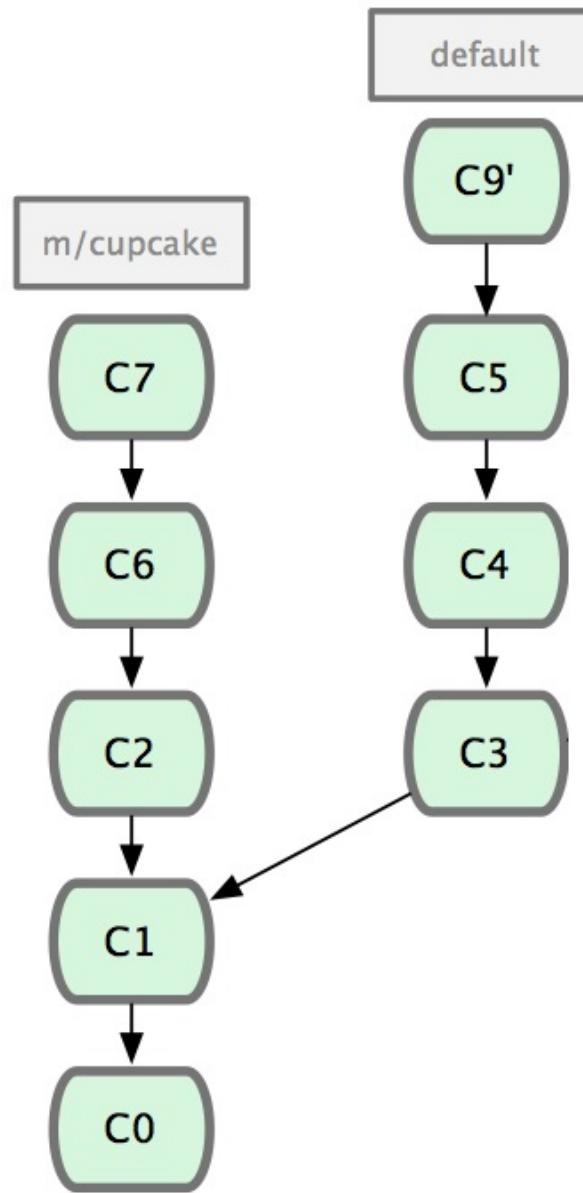
git cherry-pick experiment[^]



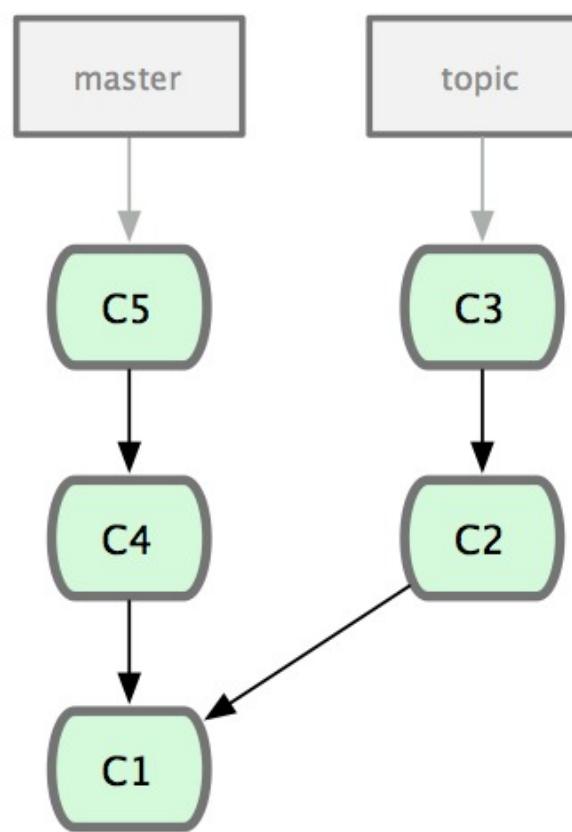
git cherry-pick c9

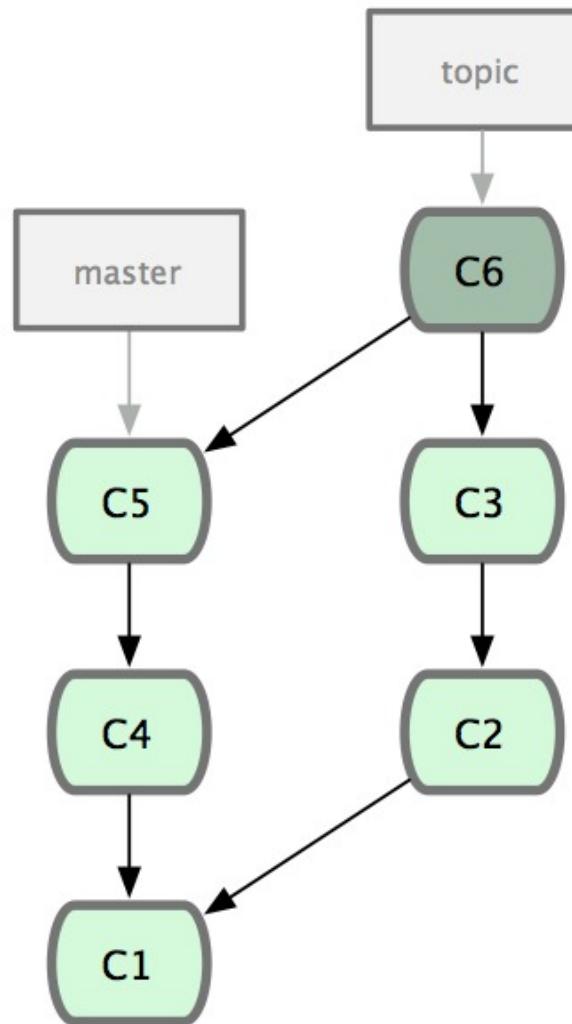




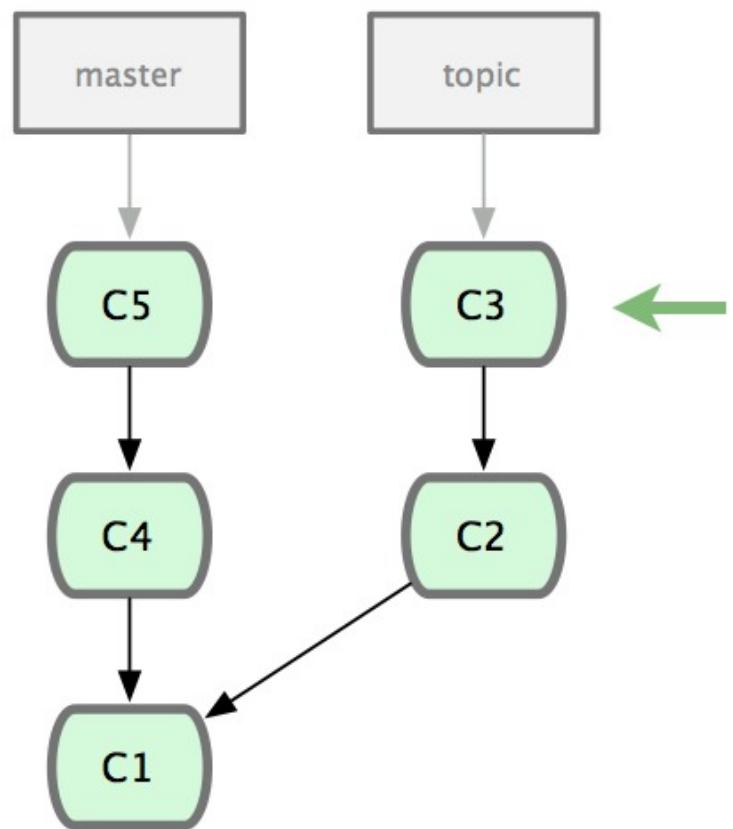


Rebasing

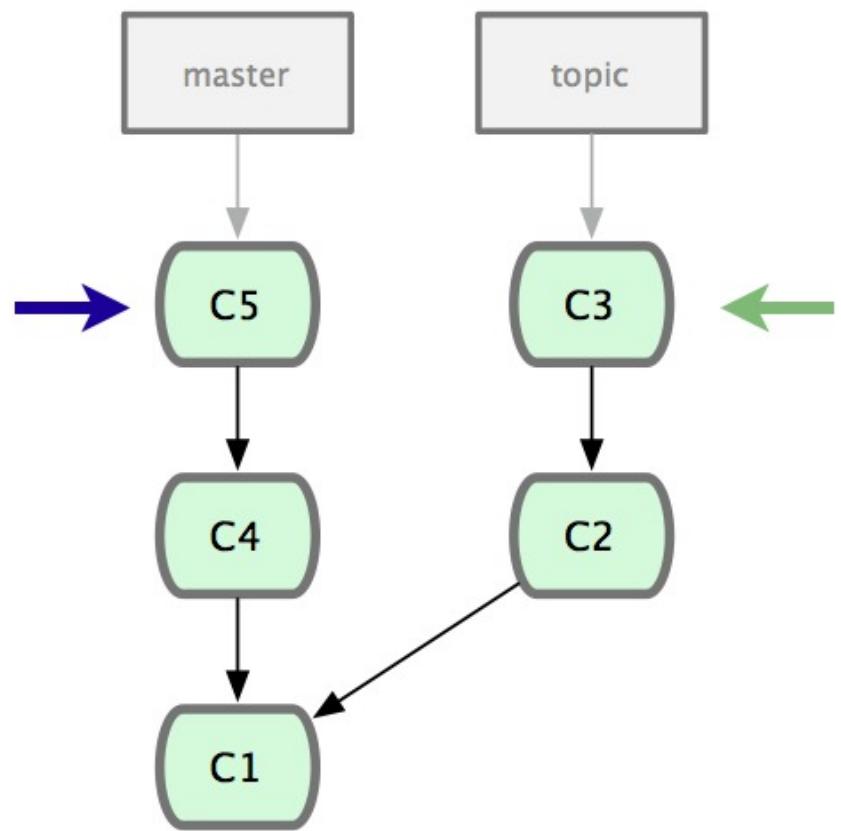




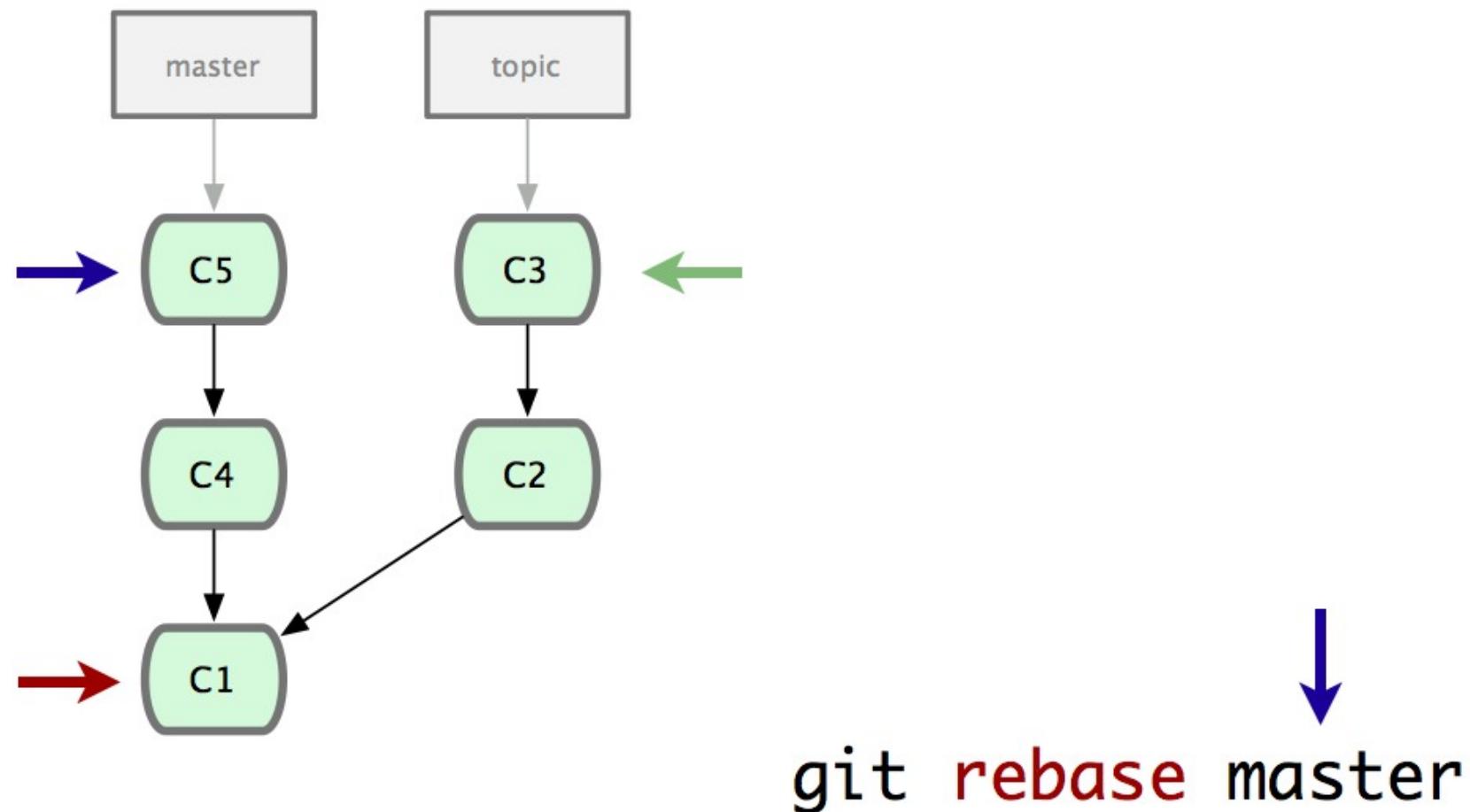
git merge master

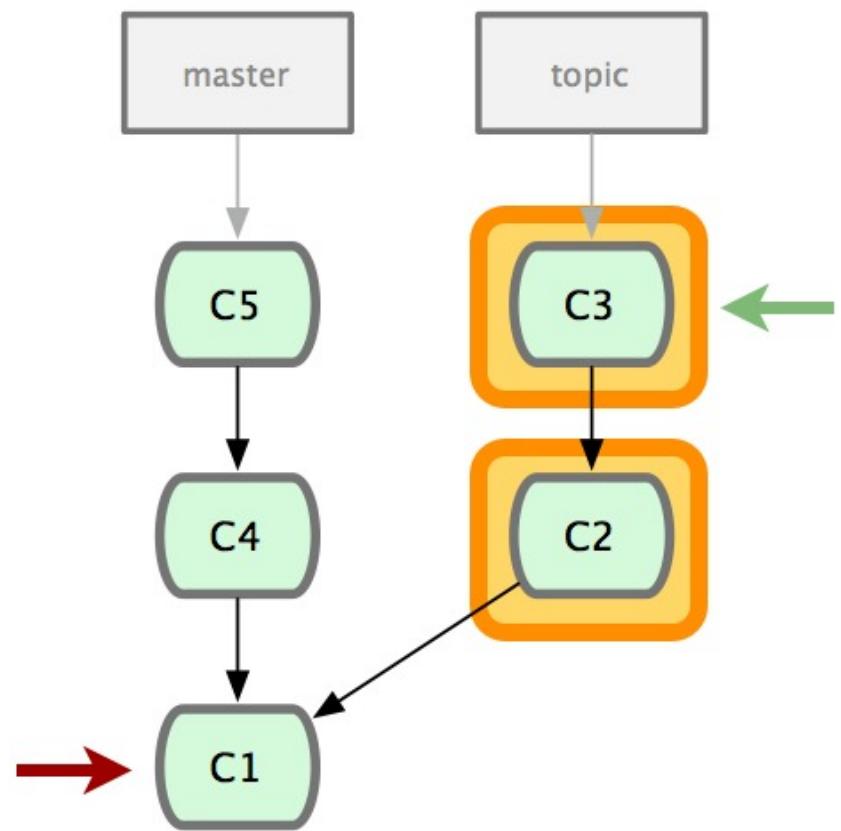


git rebase master

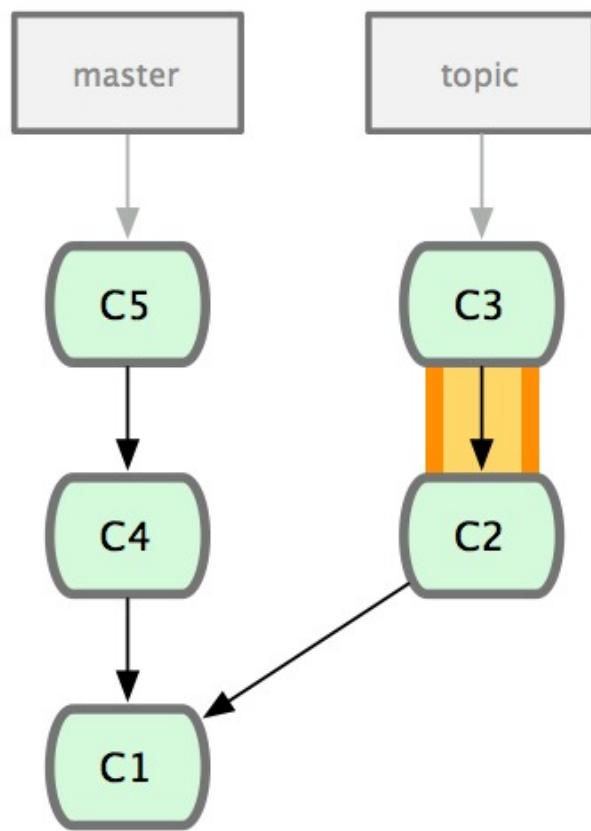


git rebase master



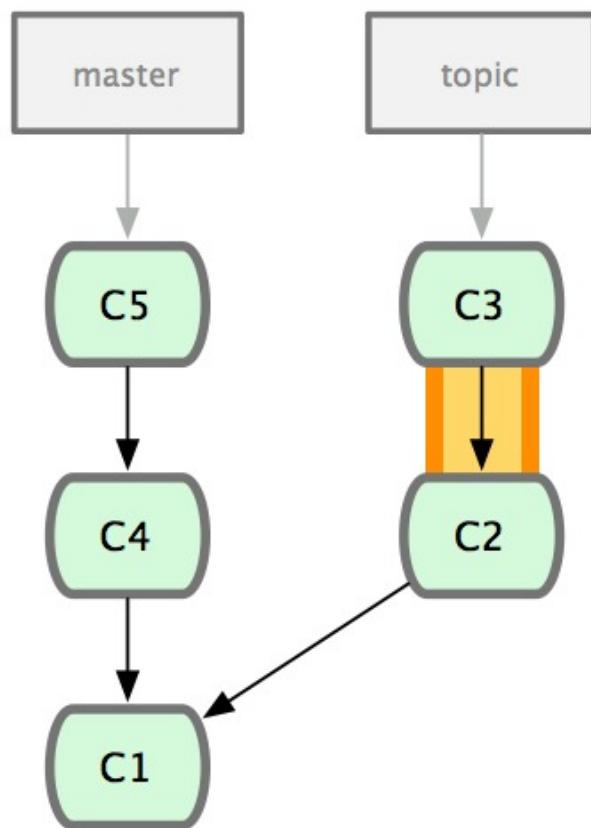


git rebase master



`git diff c2 c3 > 2-3.patch`

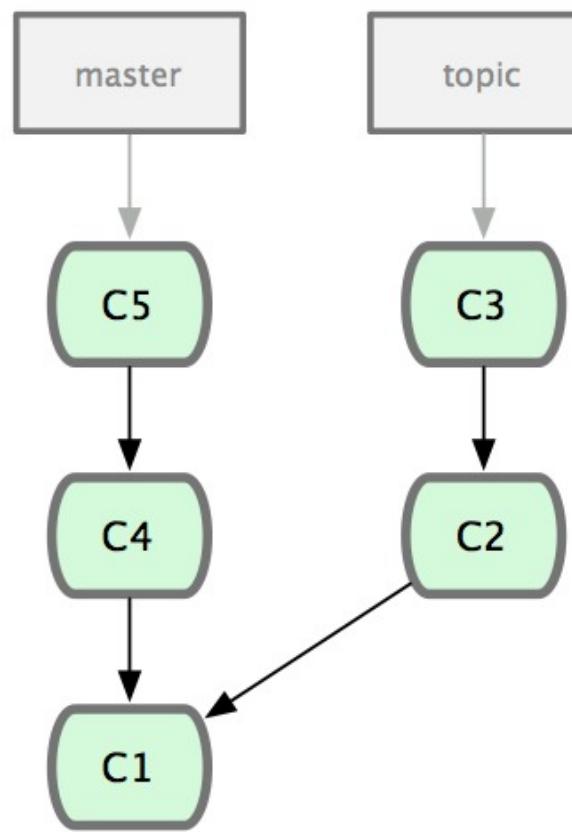
`git rebase master`

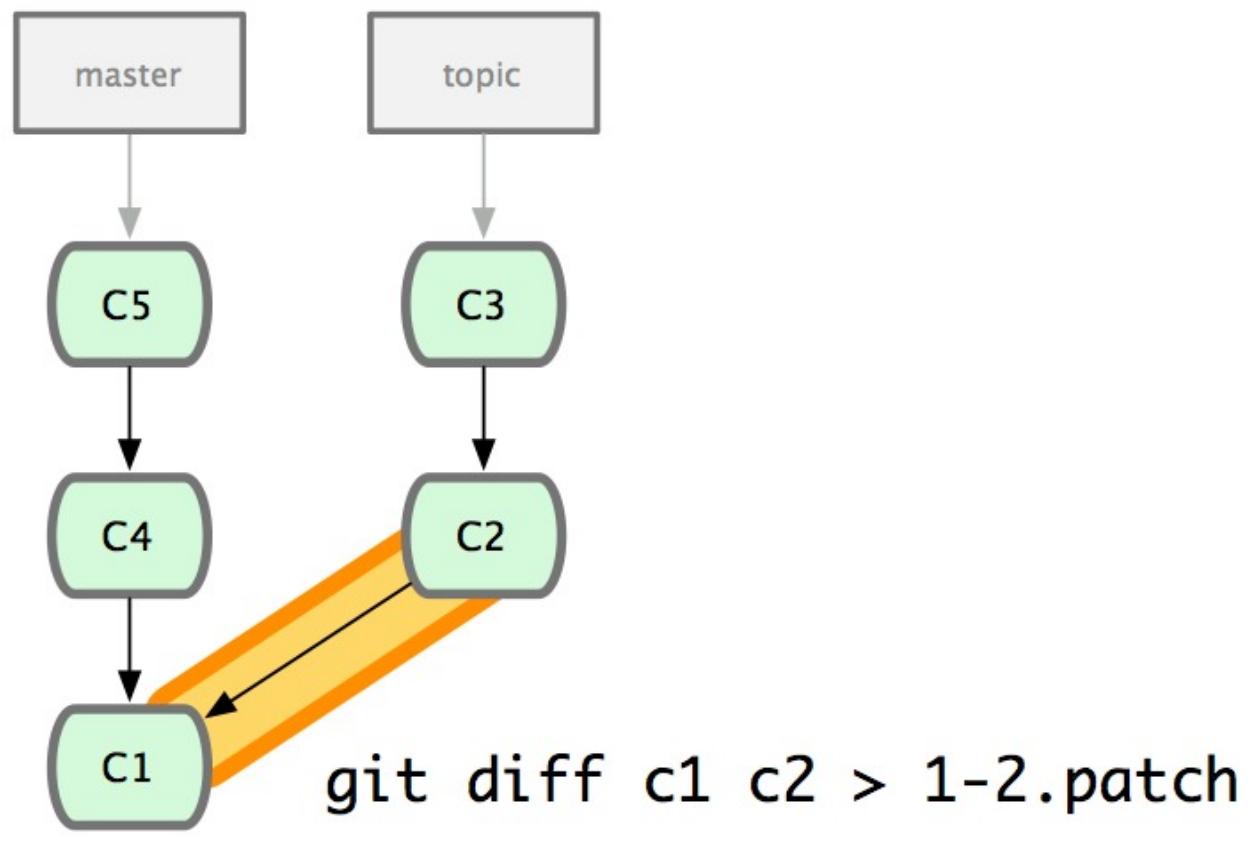


`git diff c2 c3 > 2-3.patch`

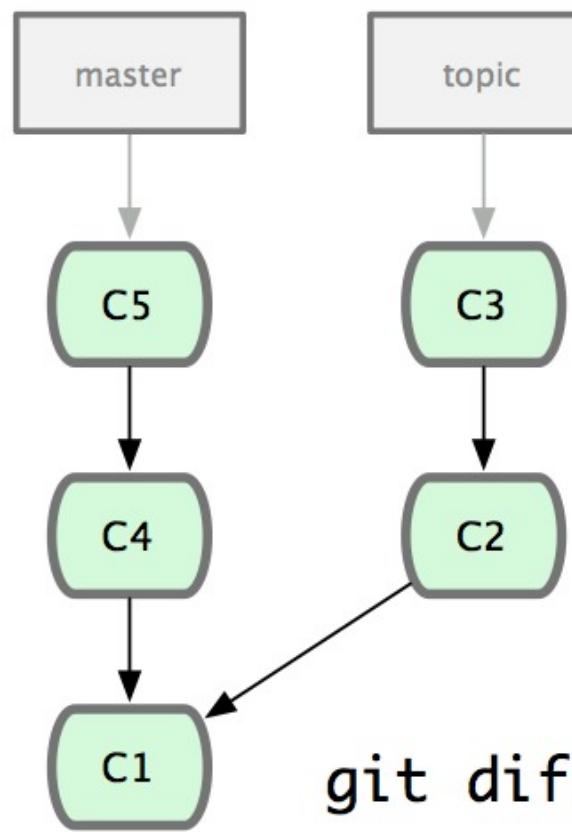
```
diff --git a/test b/test
index 2eadcec..bd8c6c9 100644
--- a/test
+++ b/test
@@ -1,2 +1,3 @@
version one
version four
+version five
```

`git rebase master`



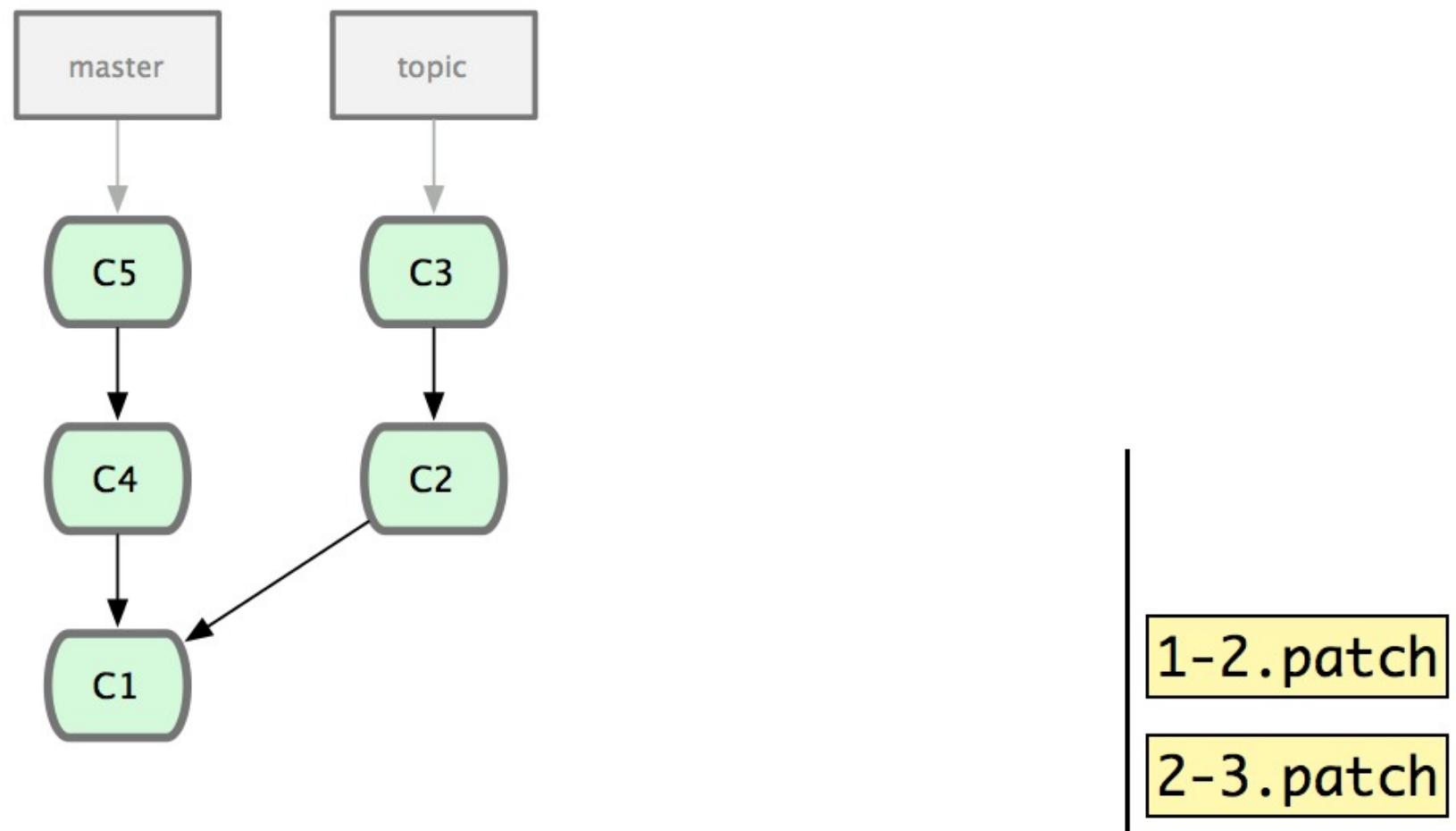


2-3.patch

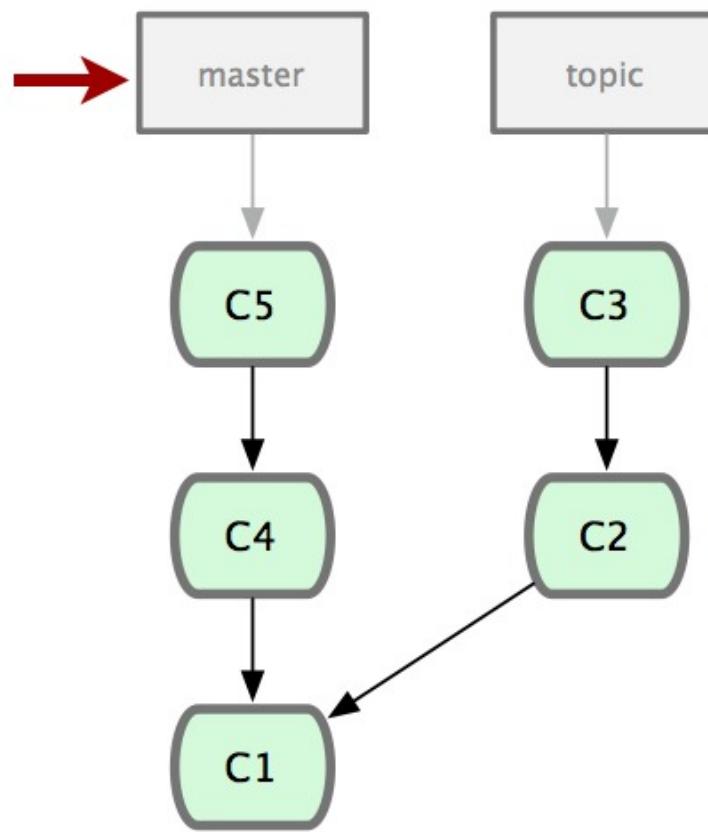


`git diff c1 c2 > 1-2.patch`

`2-3.patch`

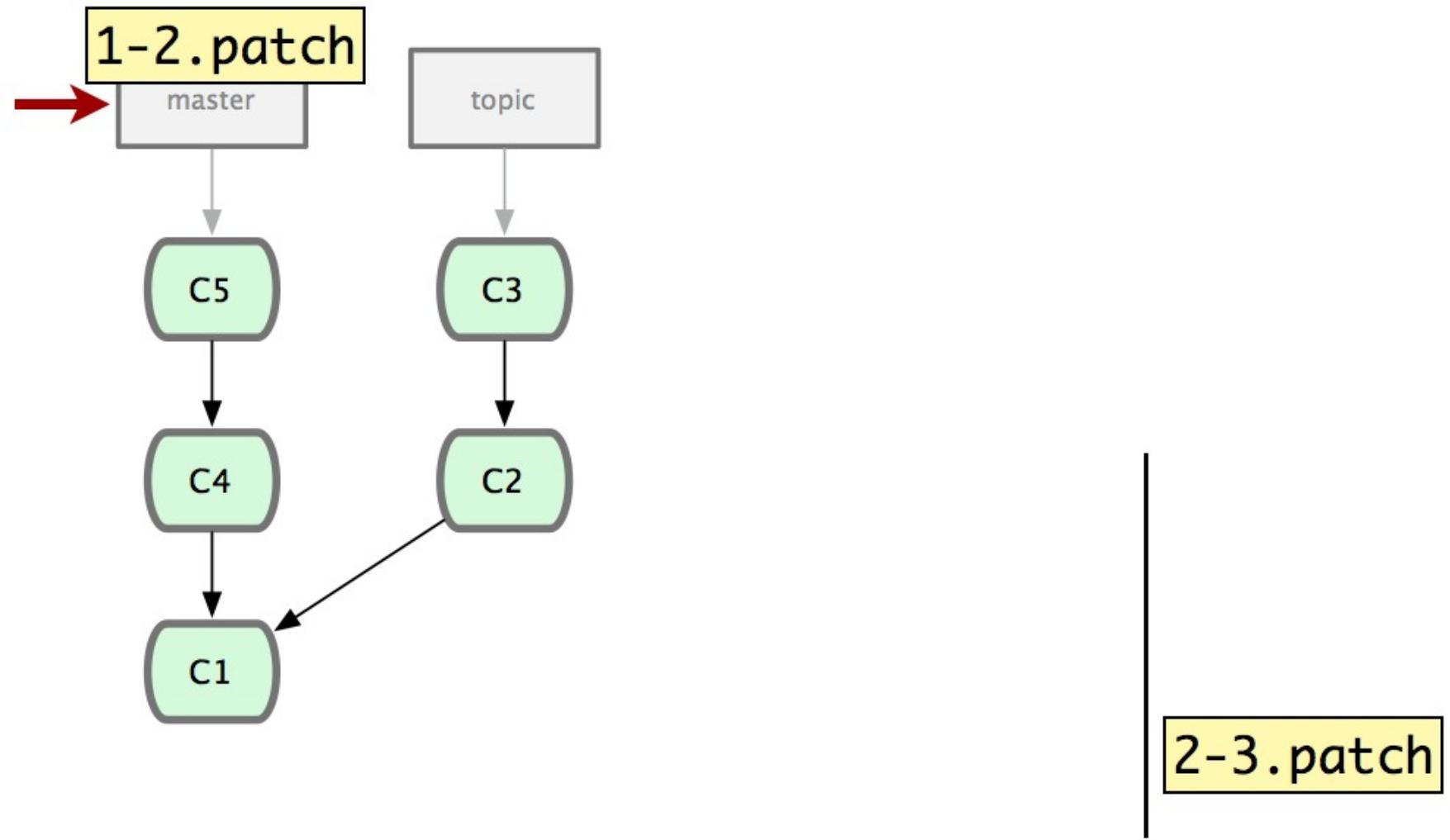


git rebase master

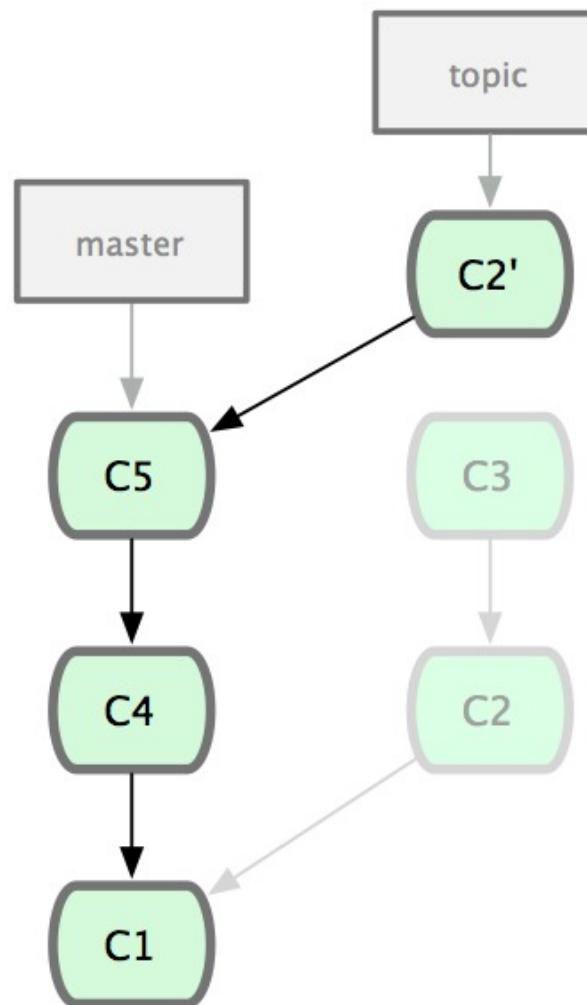


1-2.patch
2-3.patch

git rebase master

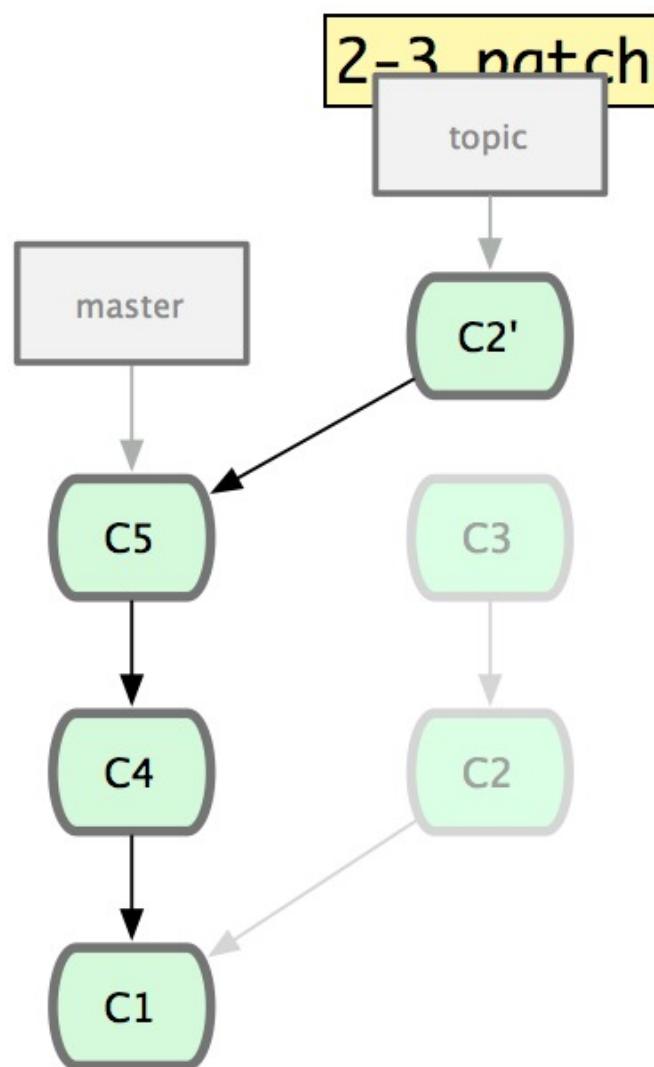


git rebase master

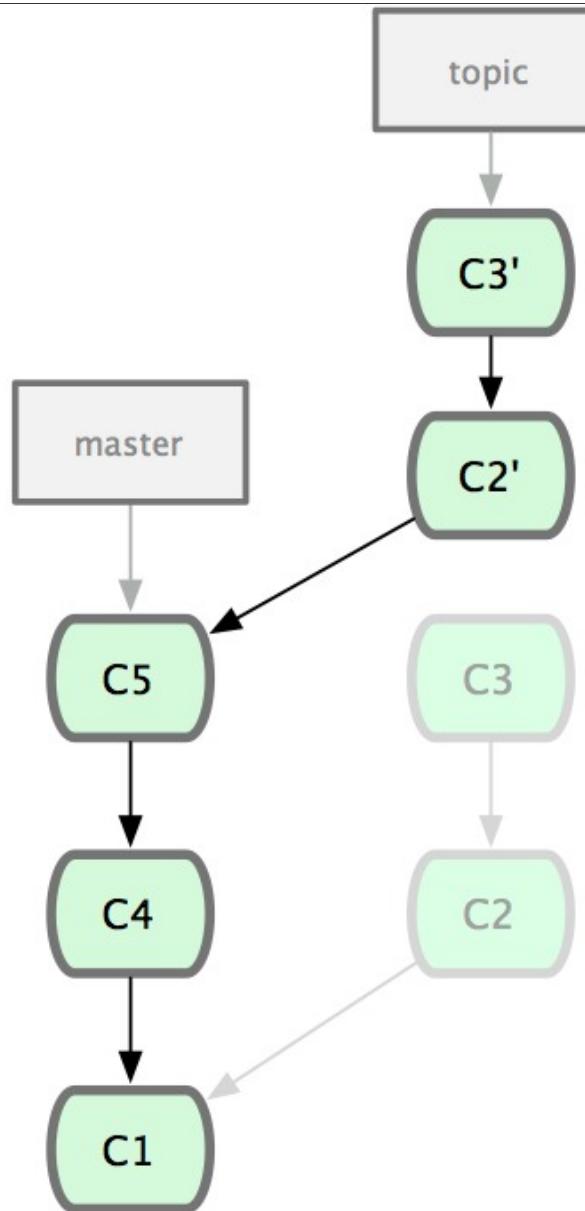


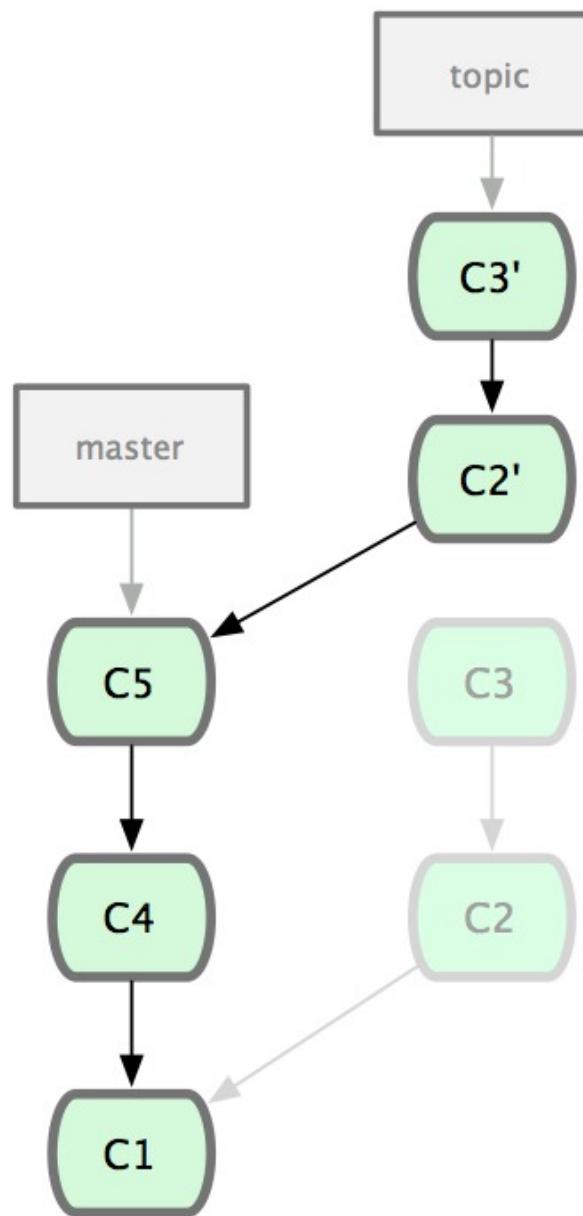
2-3.patch

git rebase master

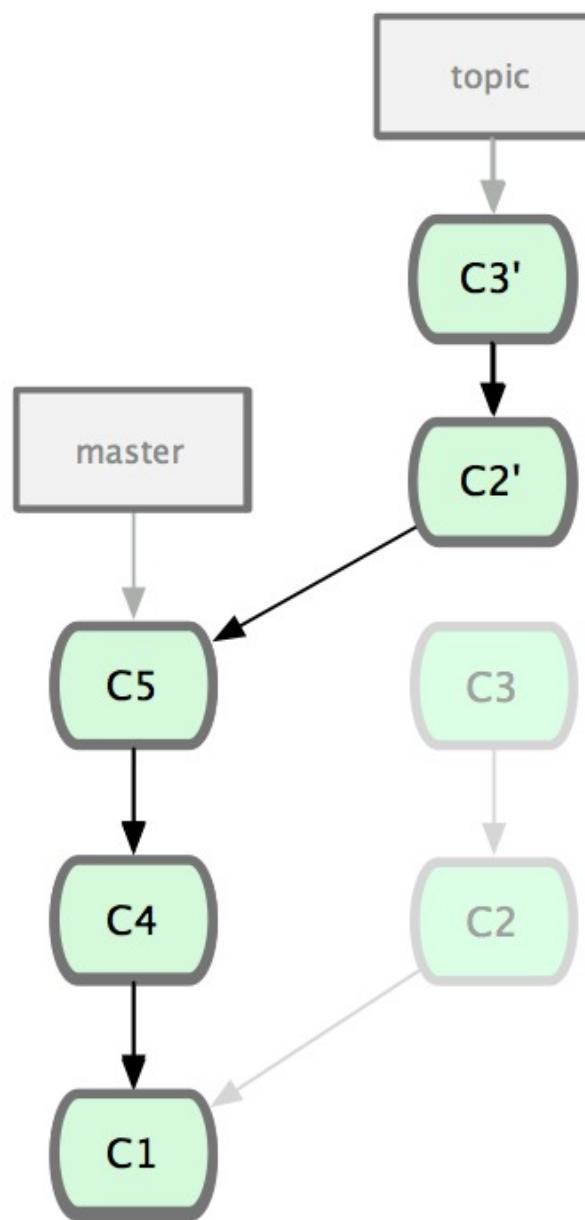


git rebase master

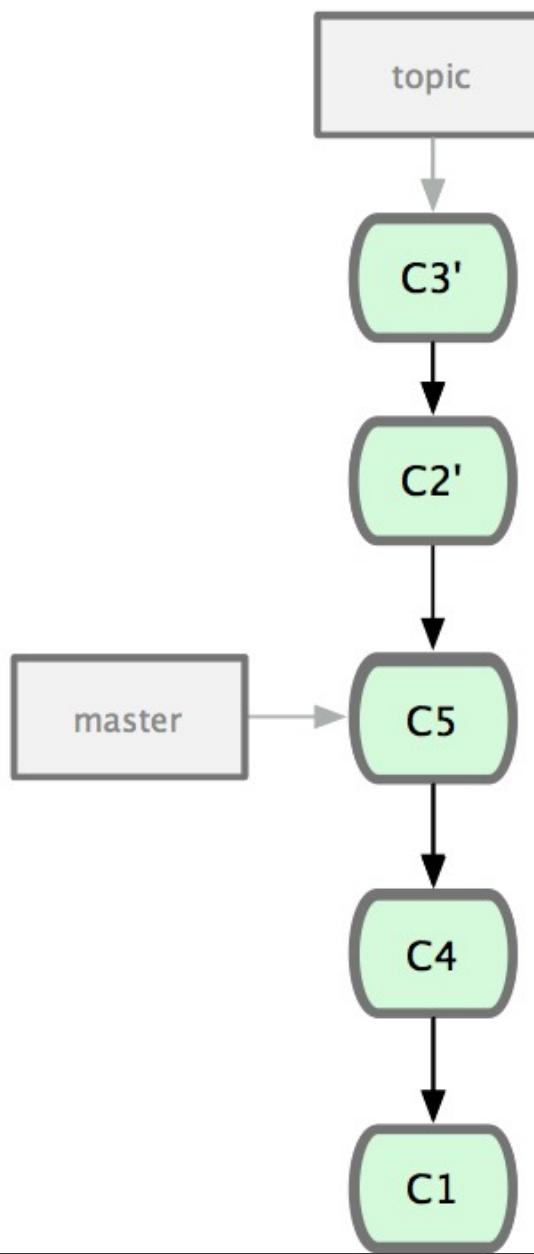




git rebase master



git **rebase** master

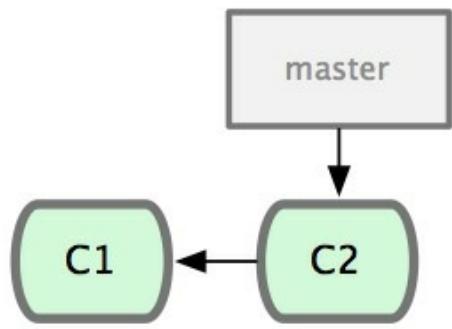


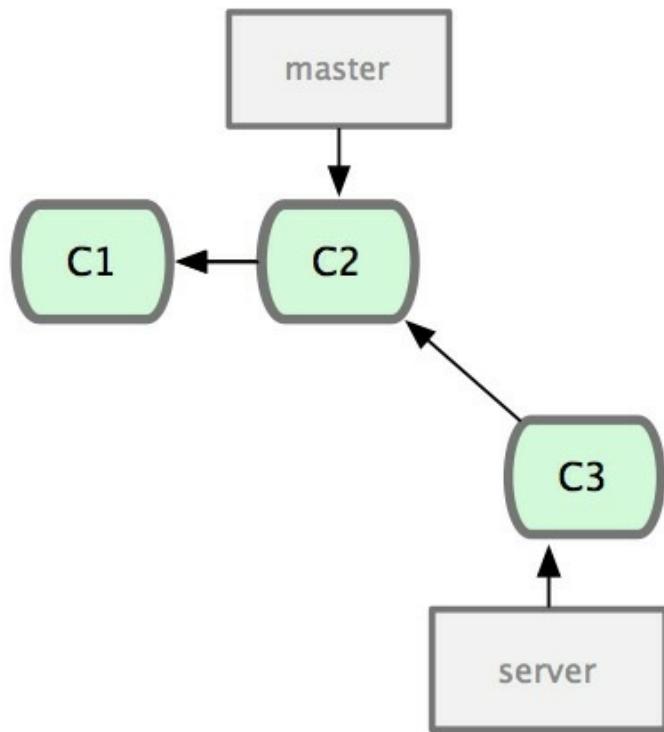
git rebase master

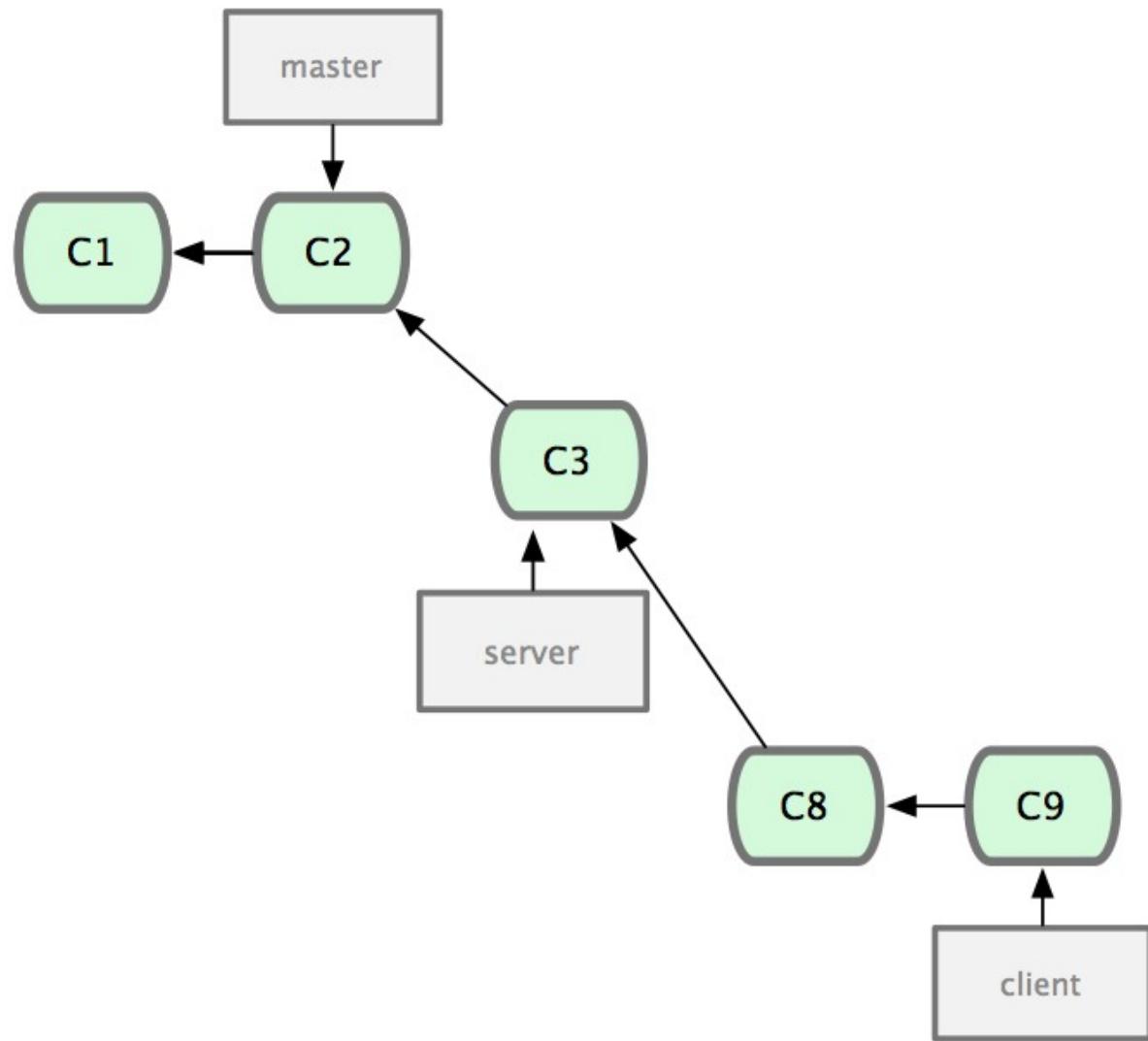
Fun with Rebasing

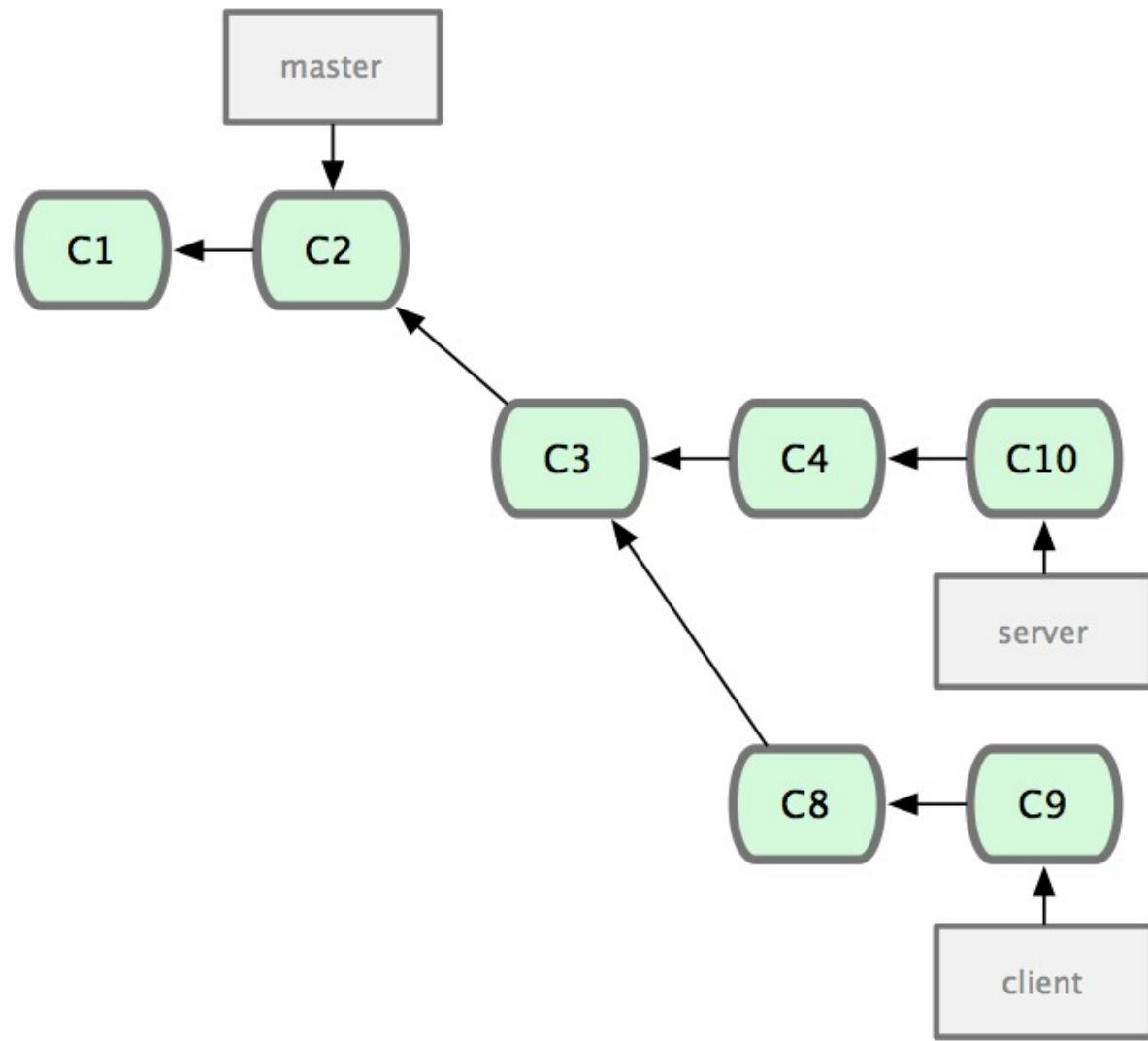
```
git rebase --onto
```

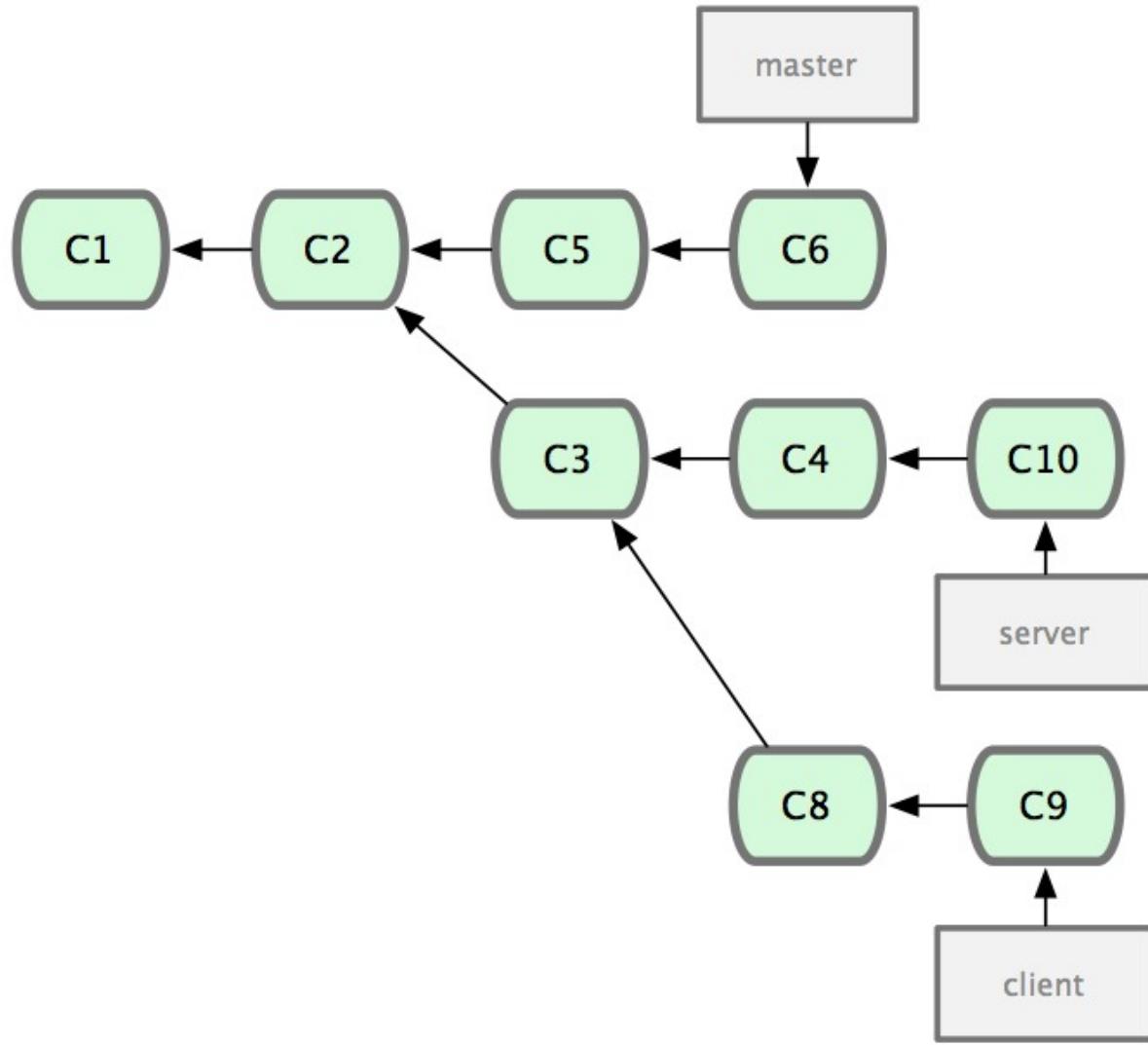
Transplanting Topic Branches



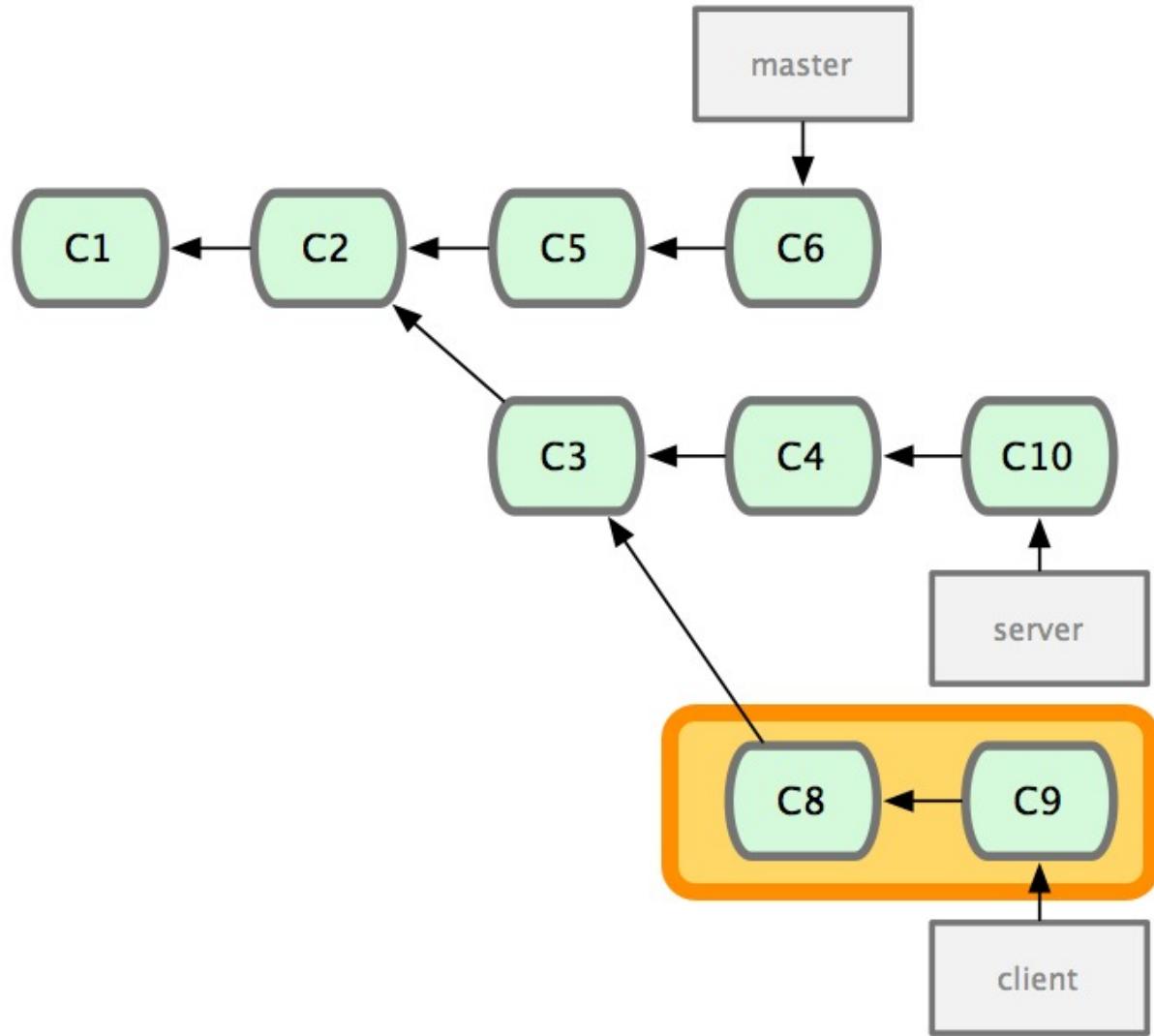


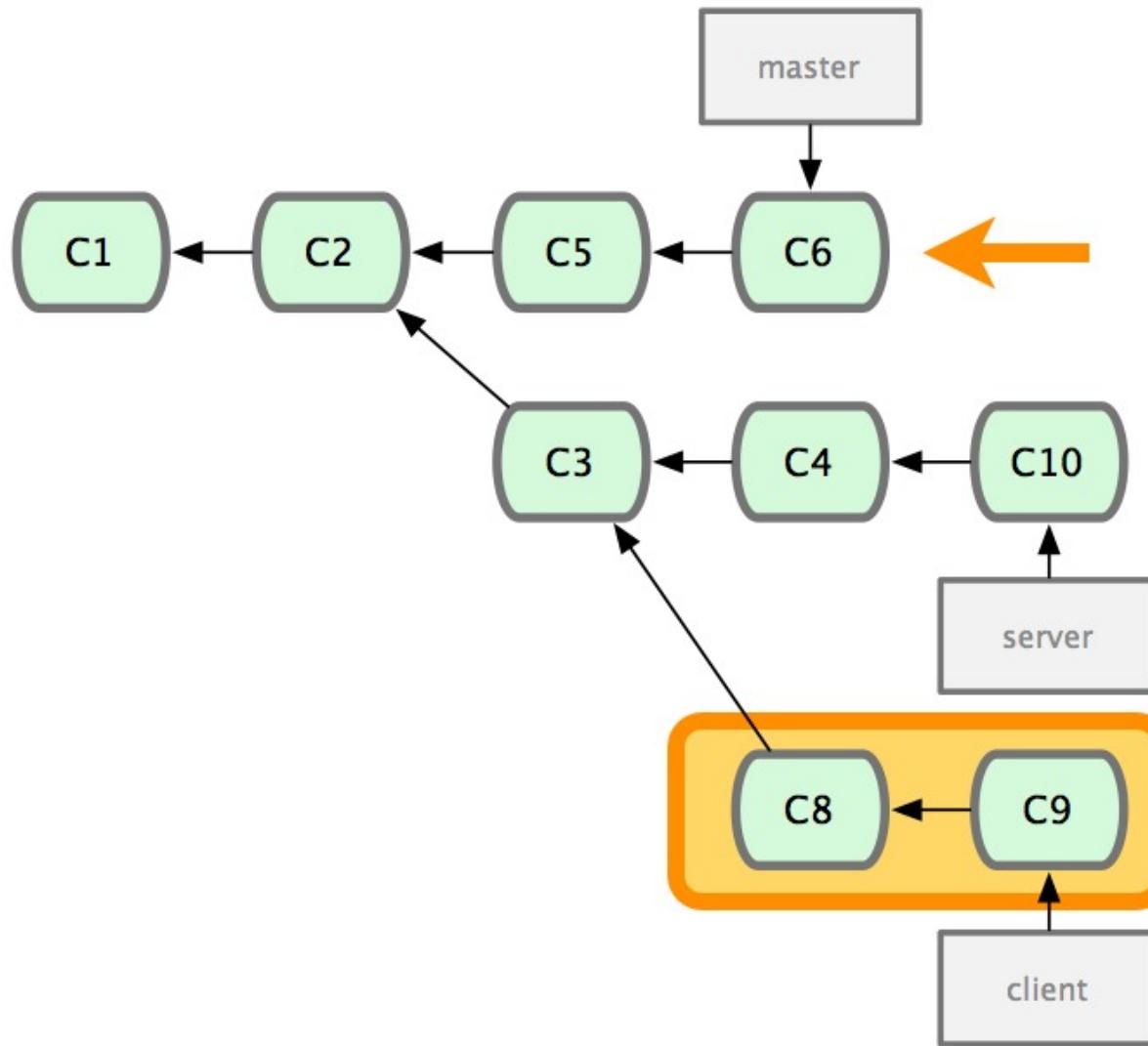


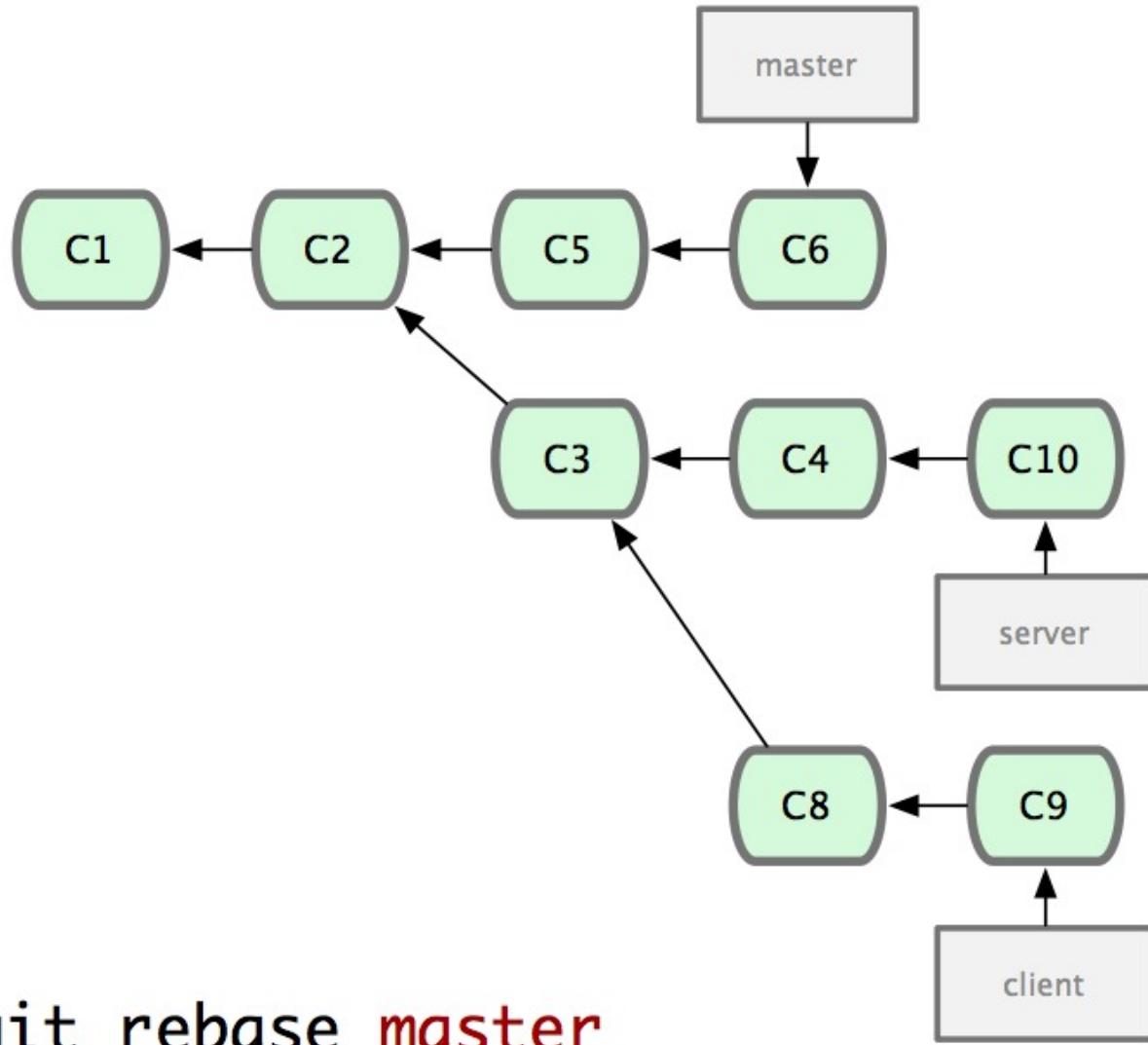




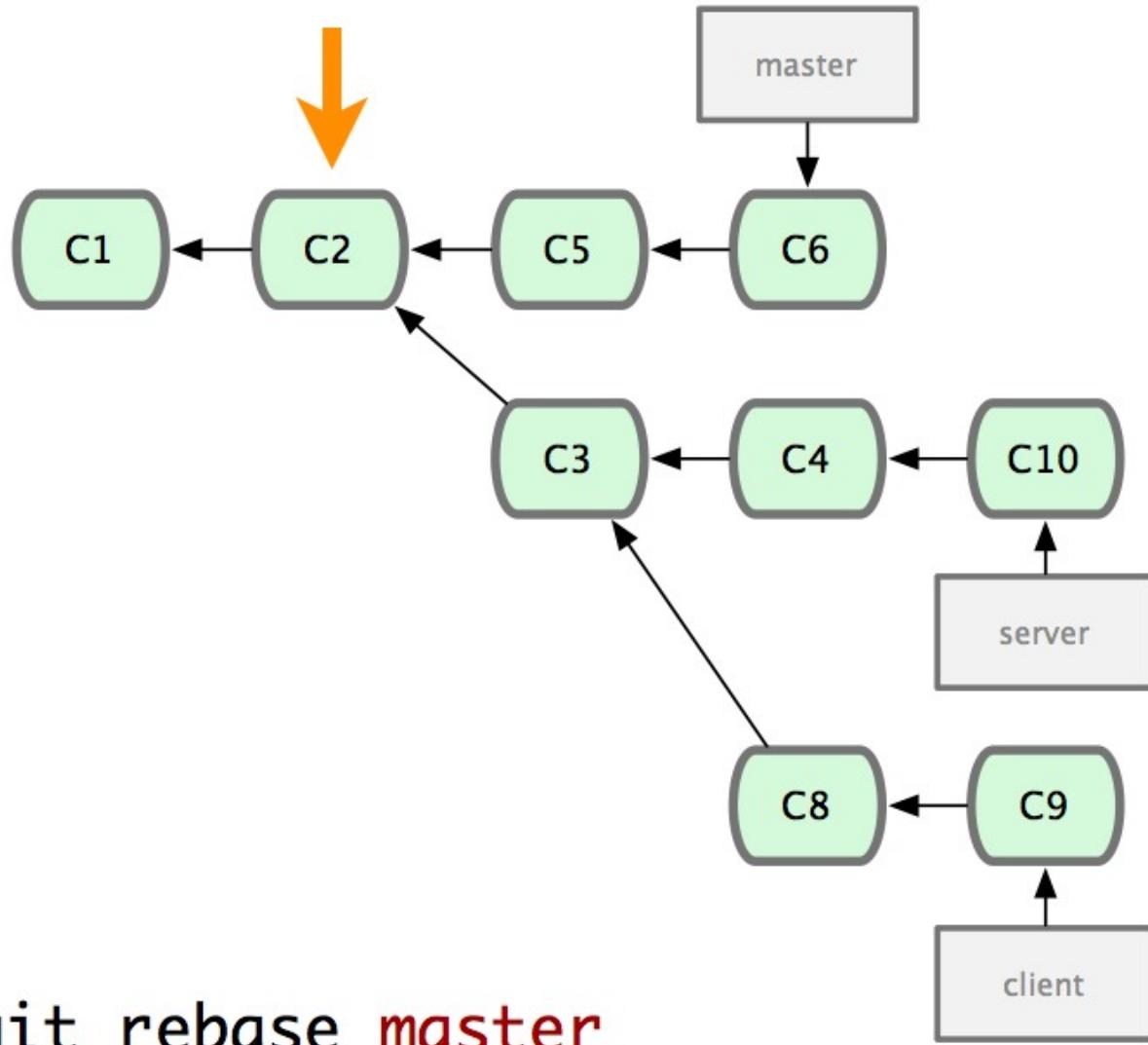
move your ‘client’
branch work to your
‘master’ branch

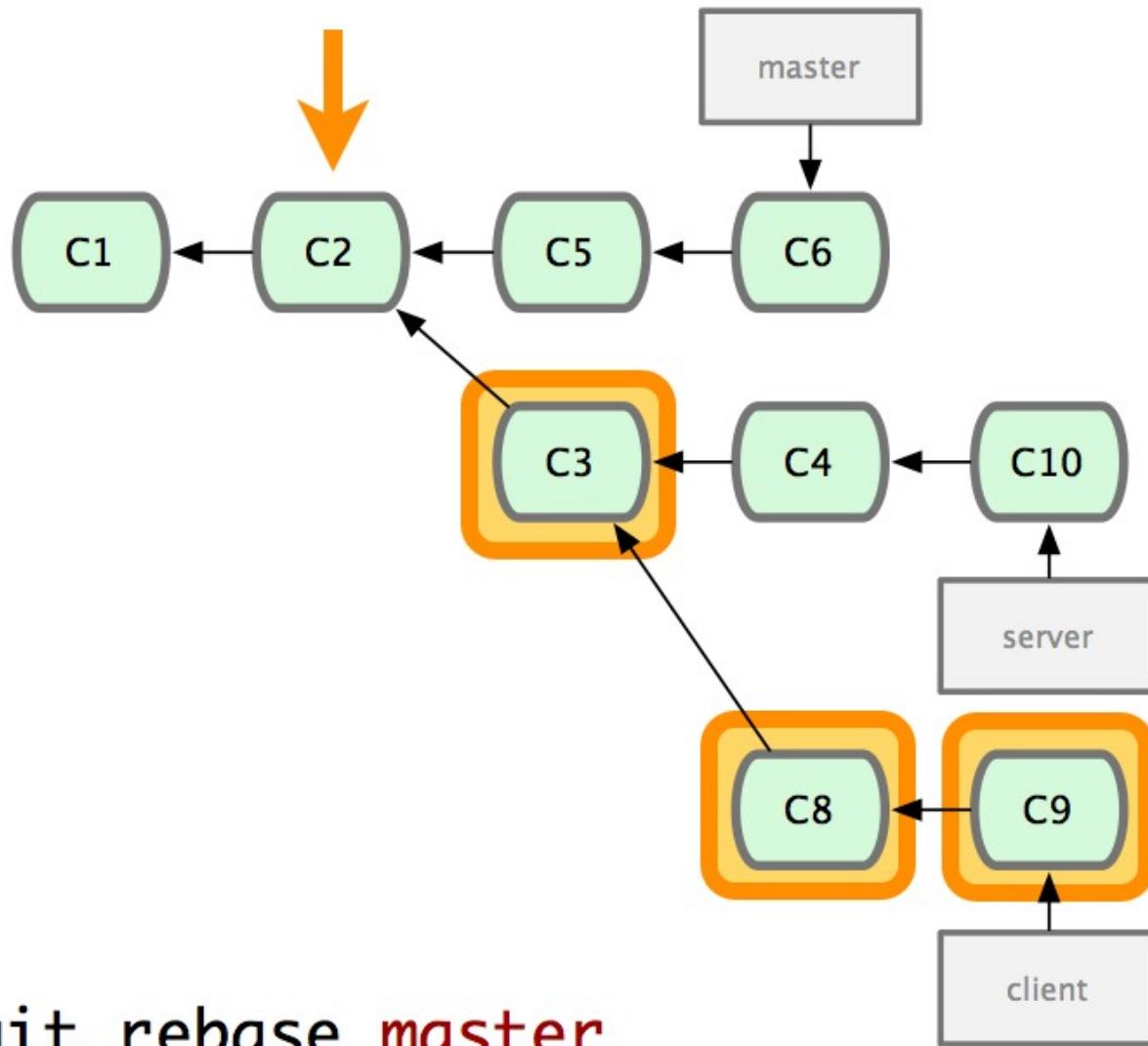


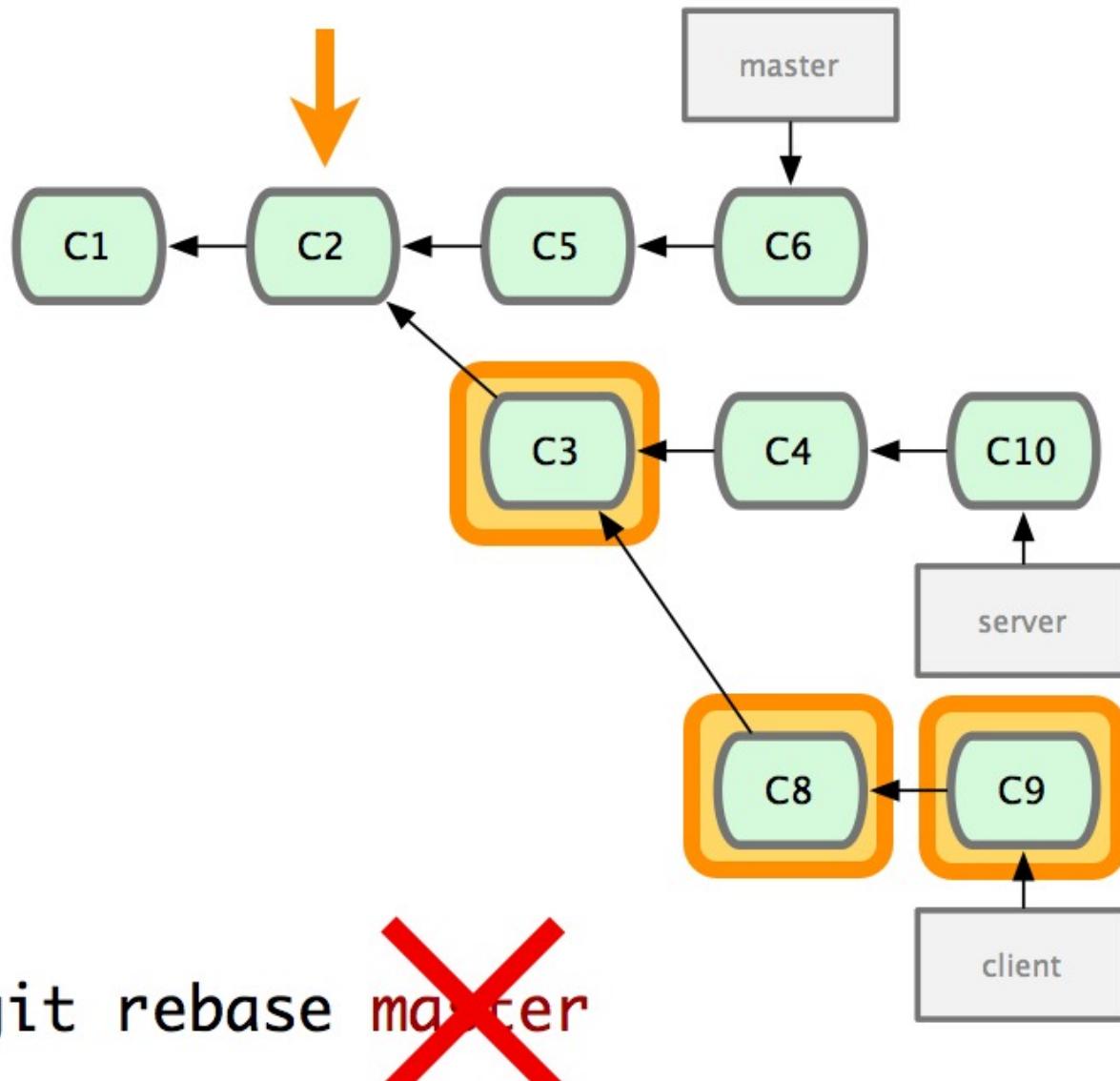


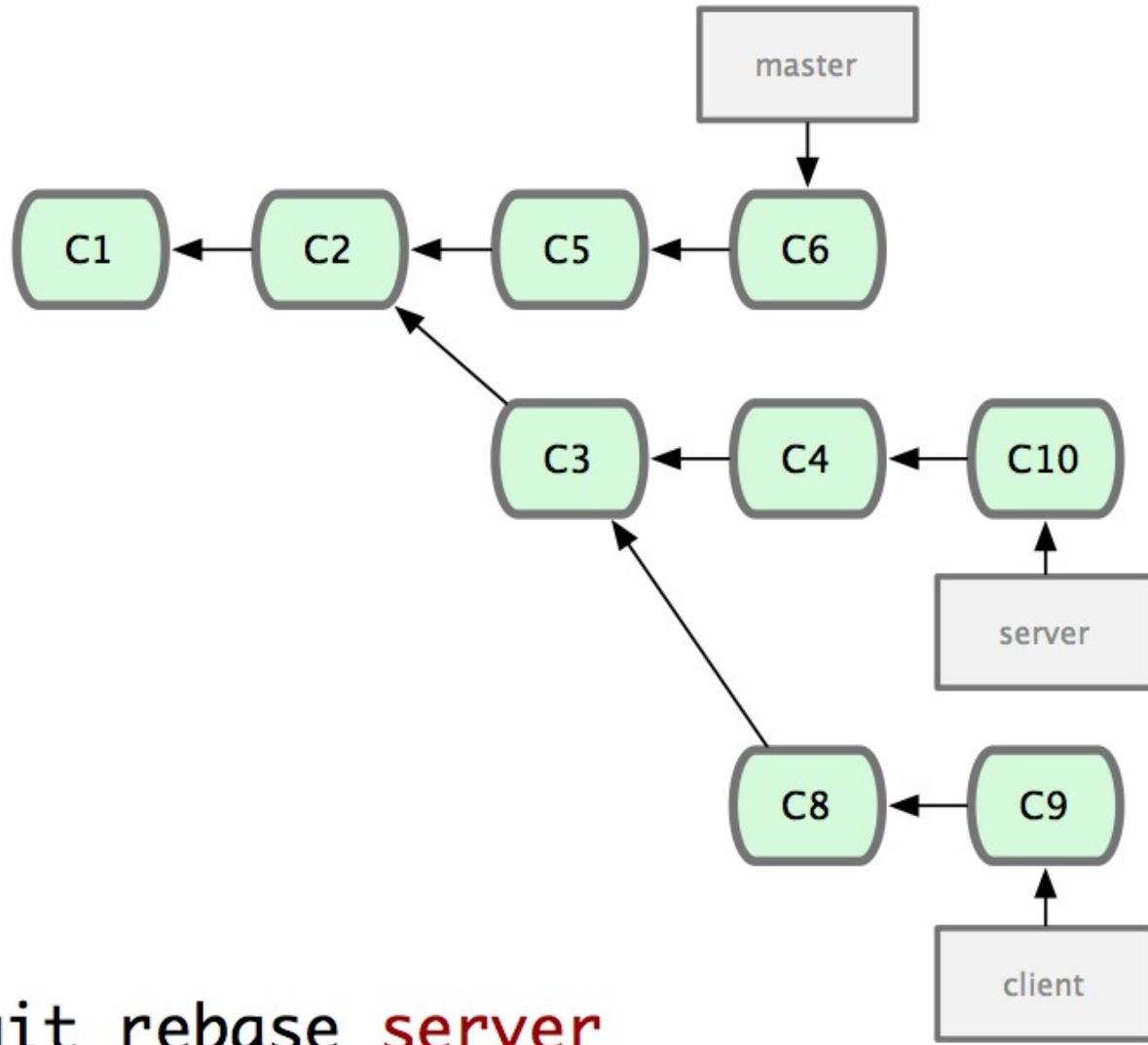


git rebase master

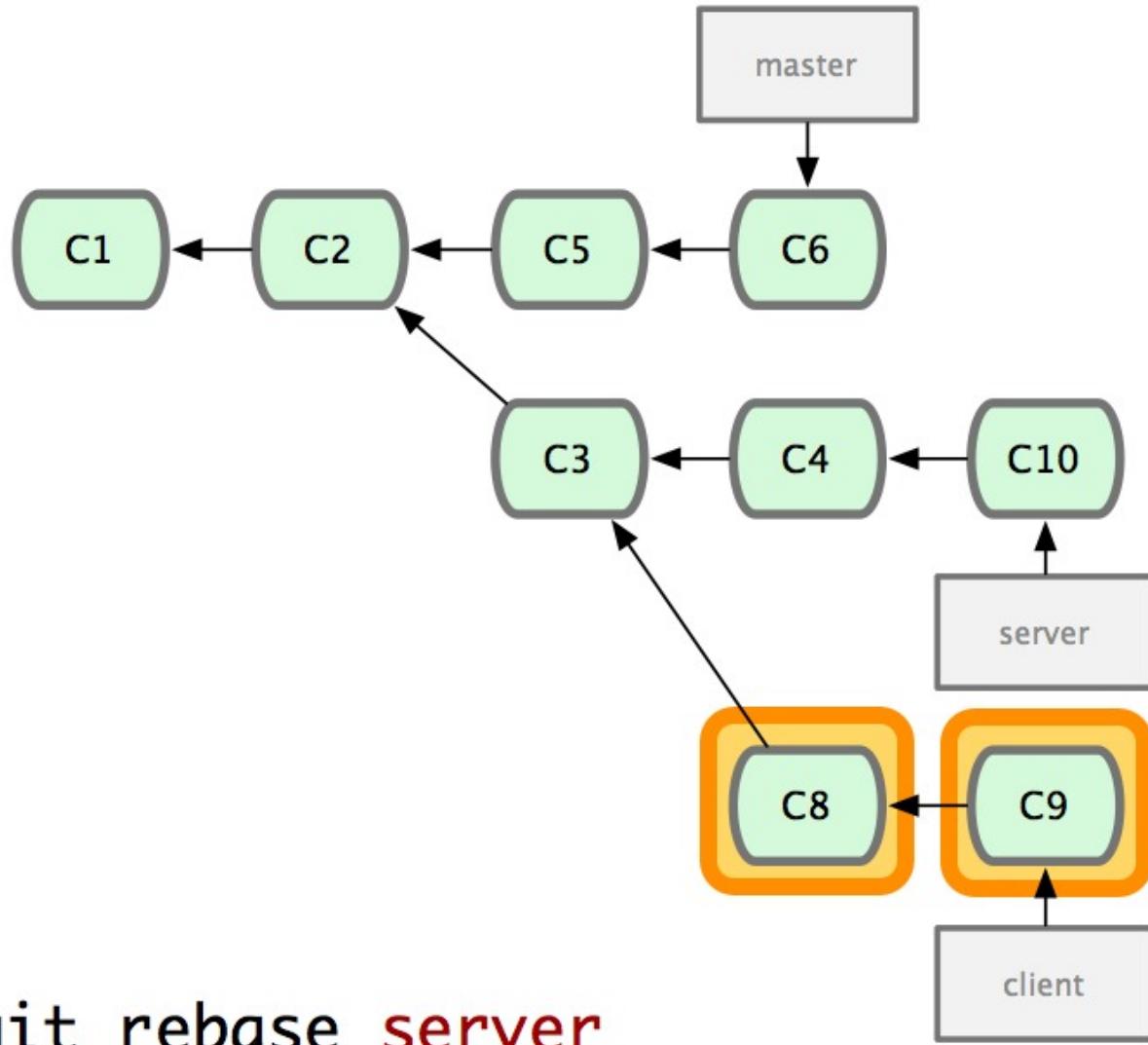




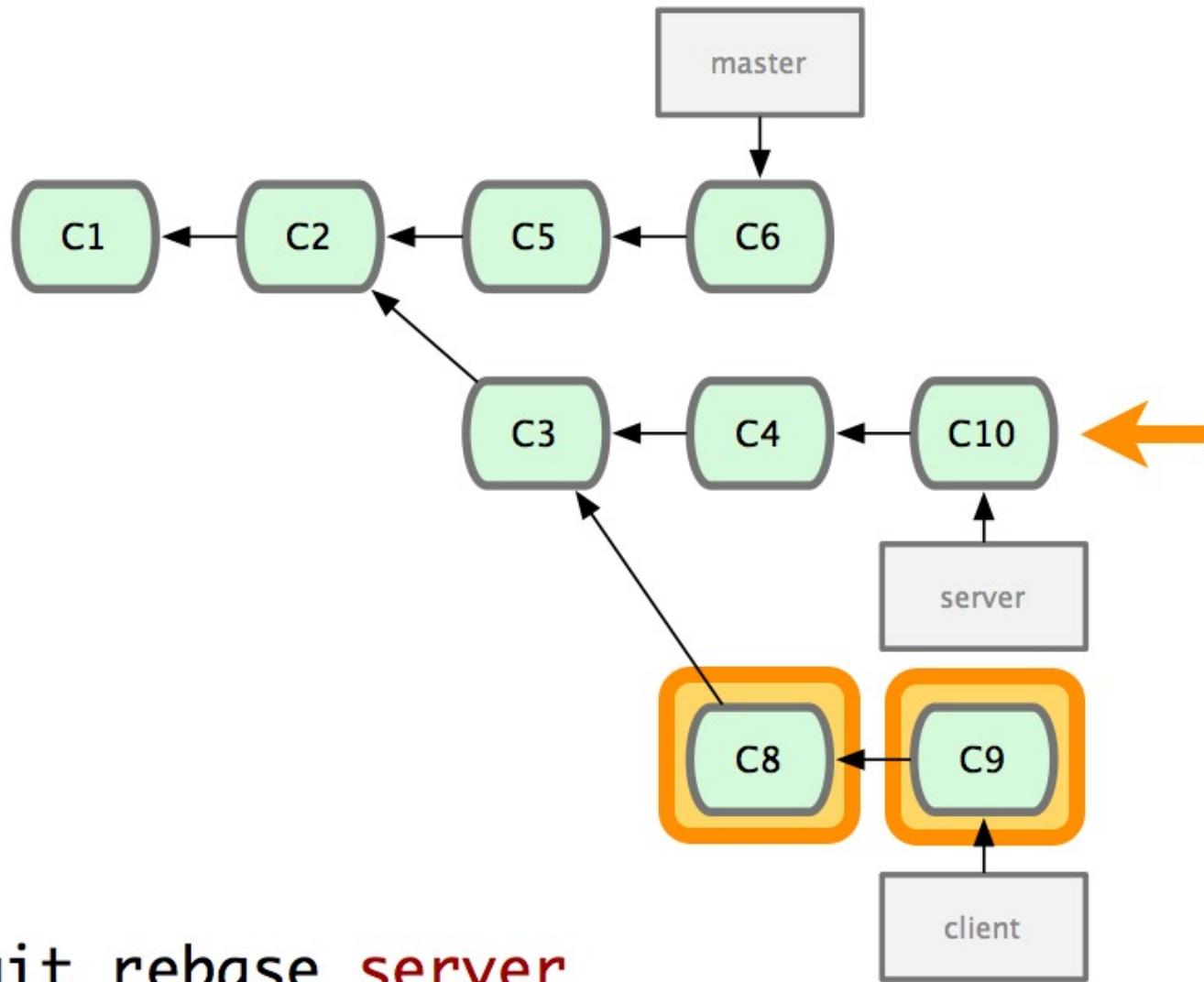


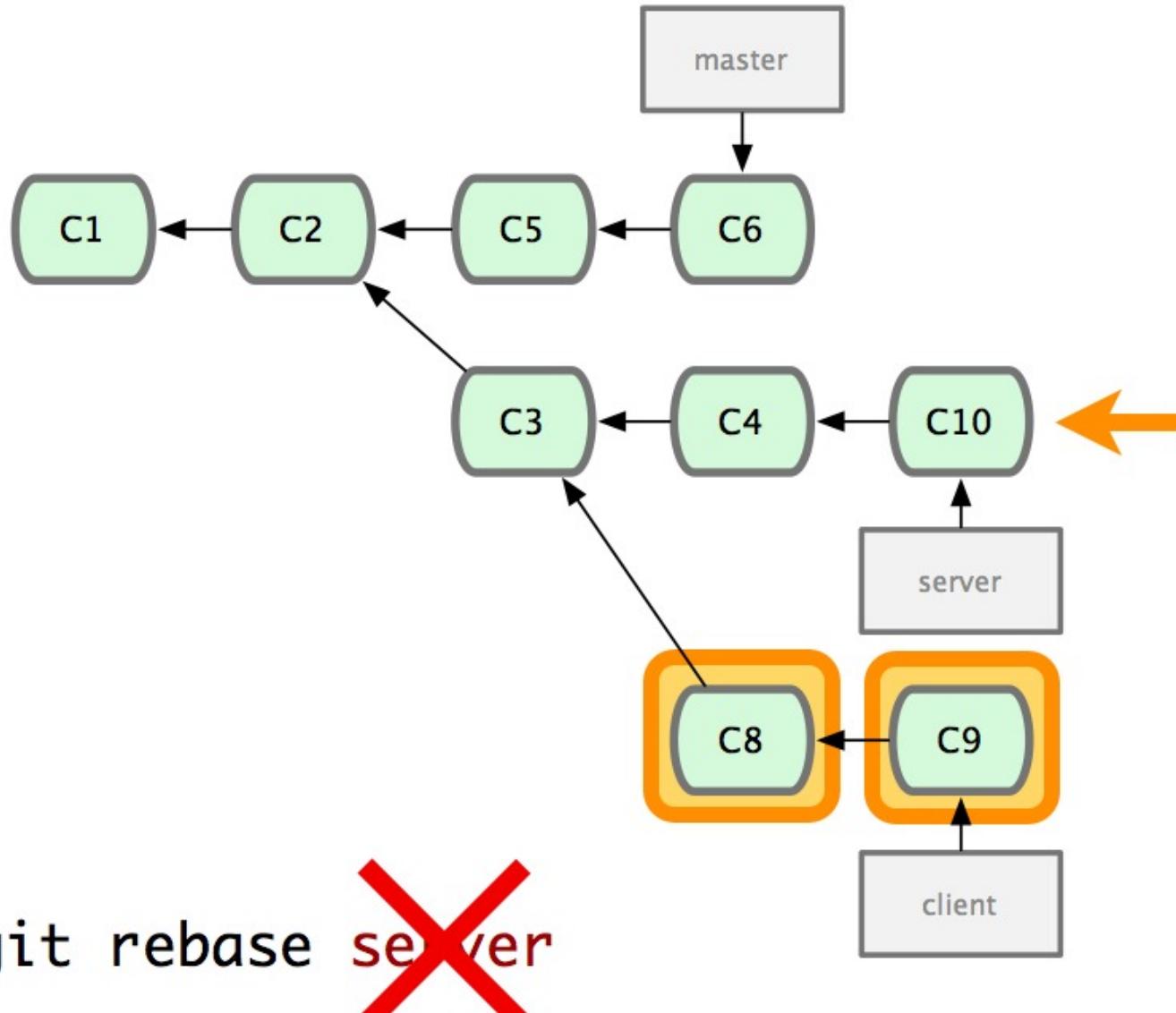


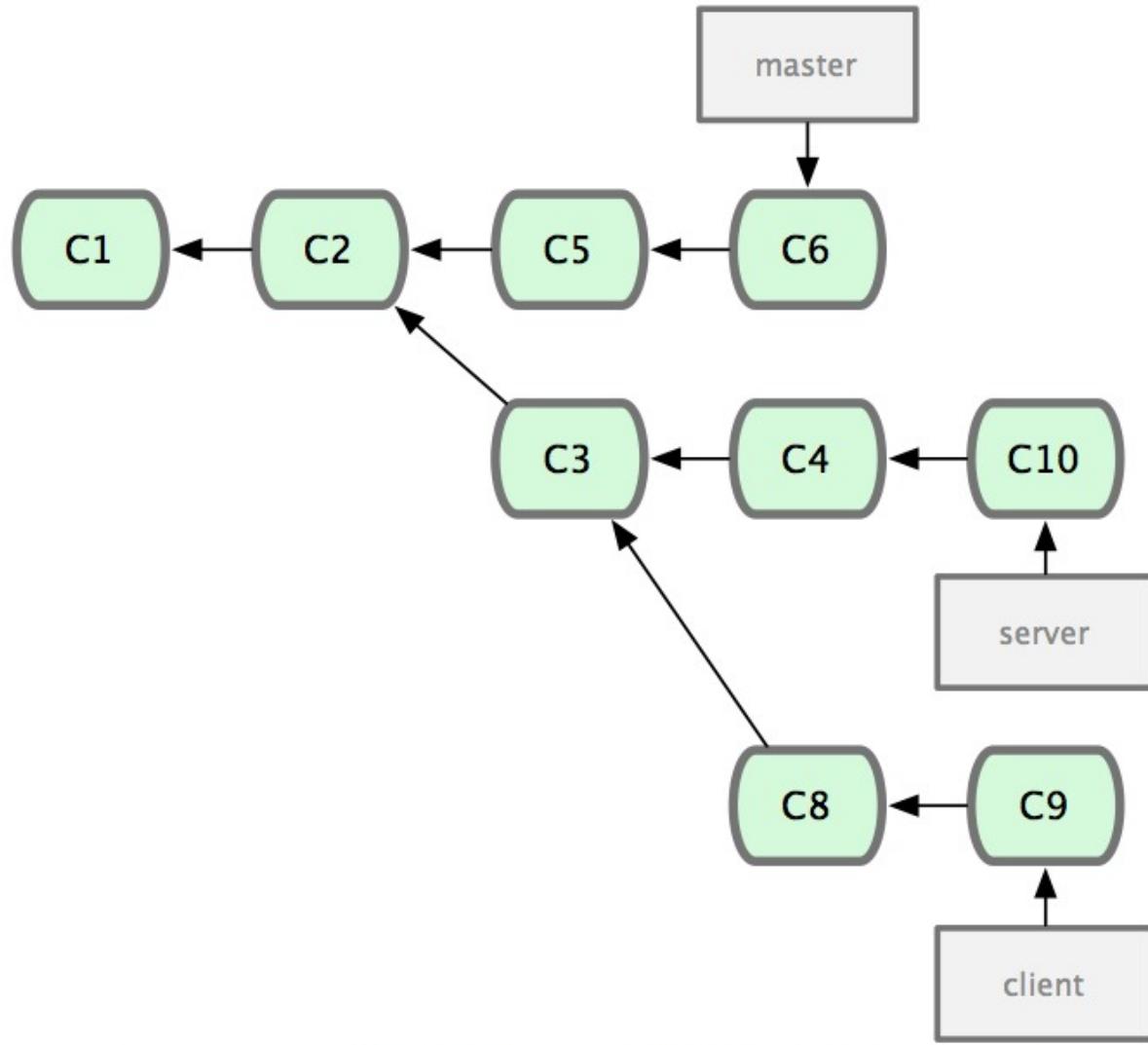
git rebase server



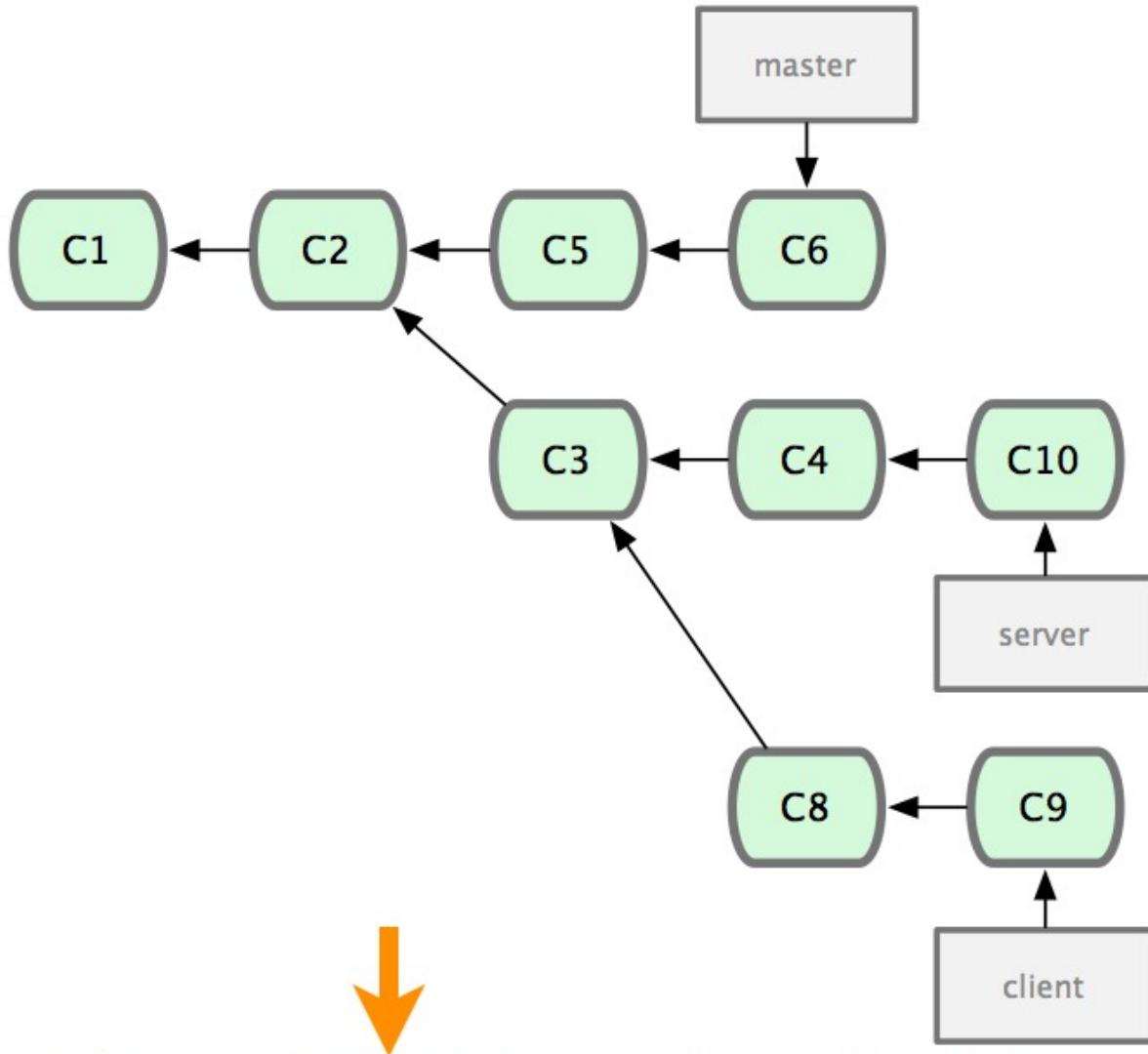
git rebase server



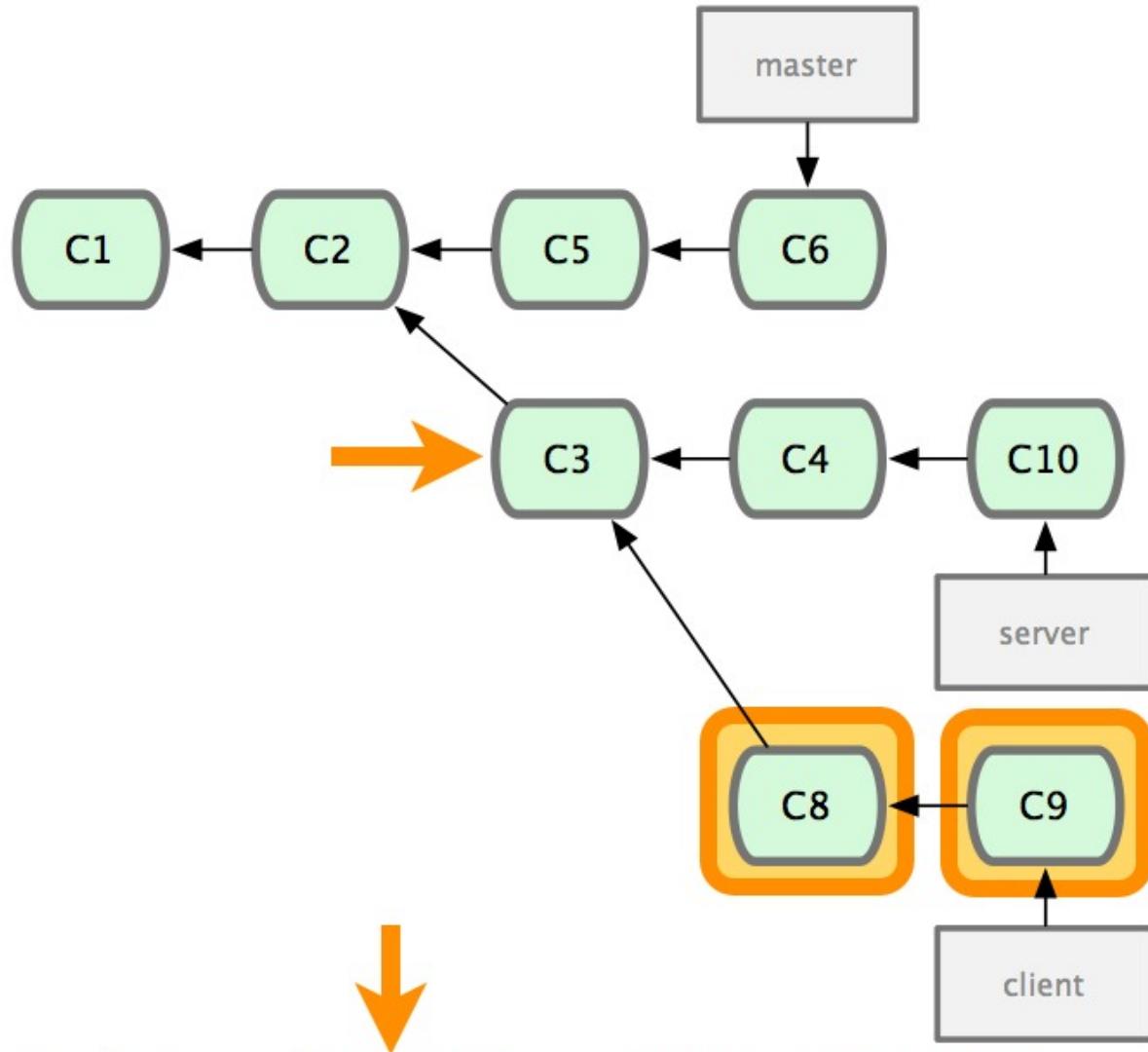




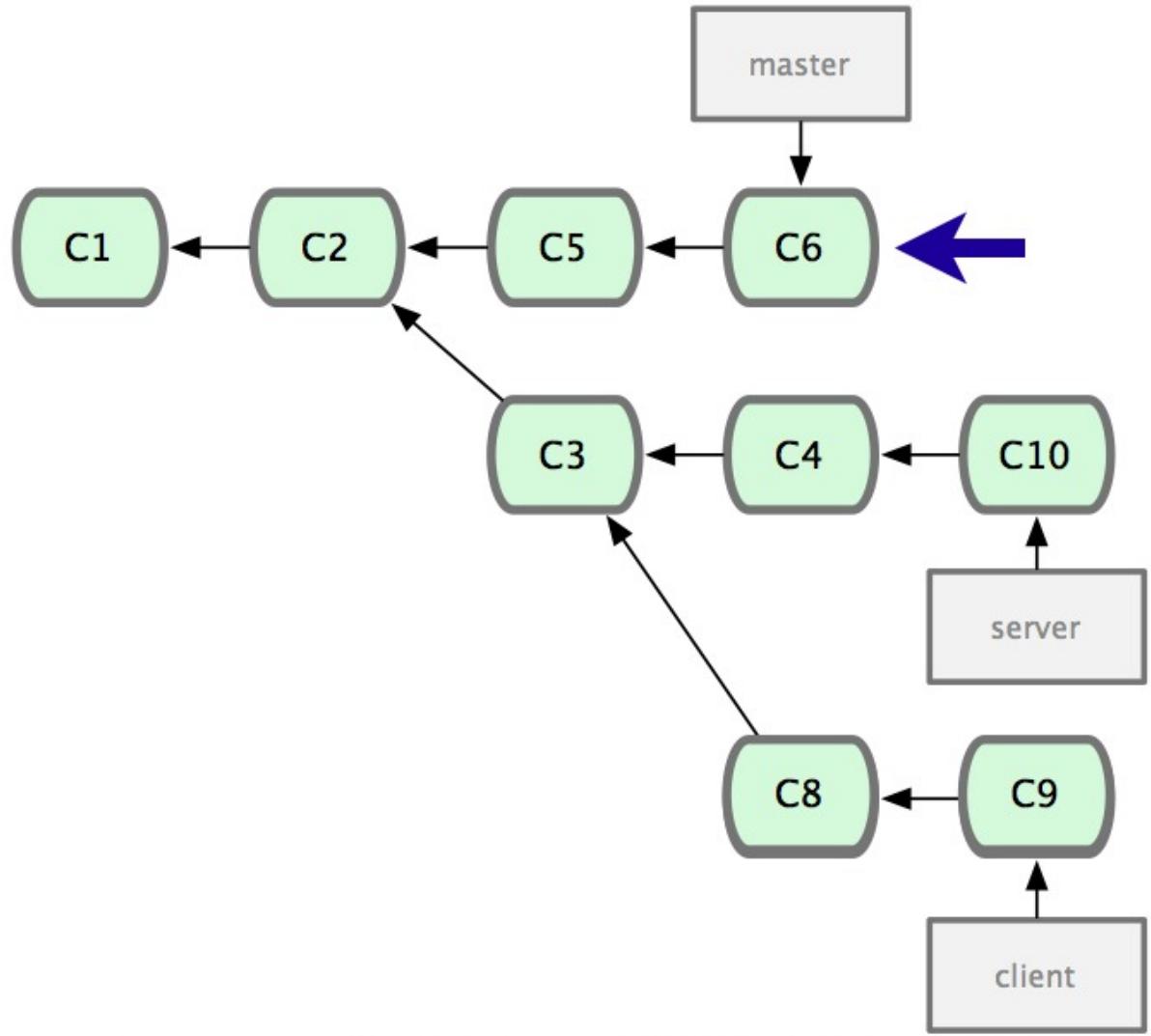
git rebase server --onto master



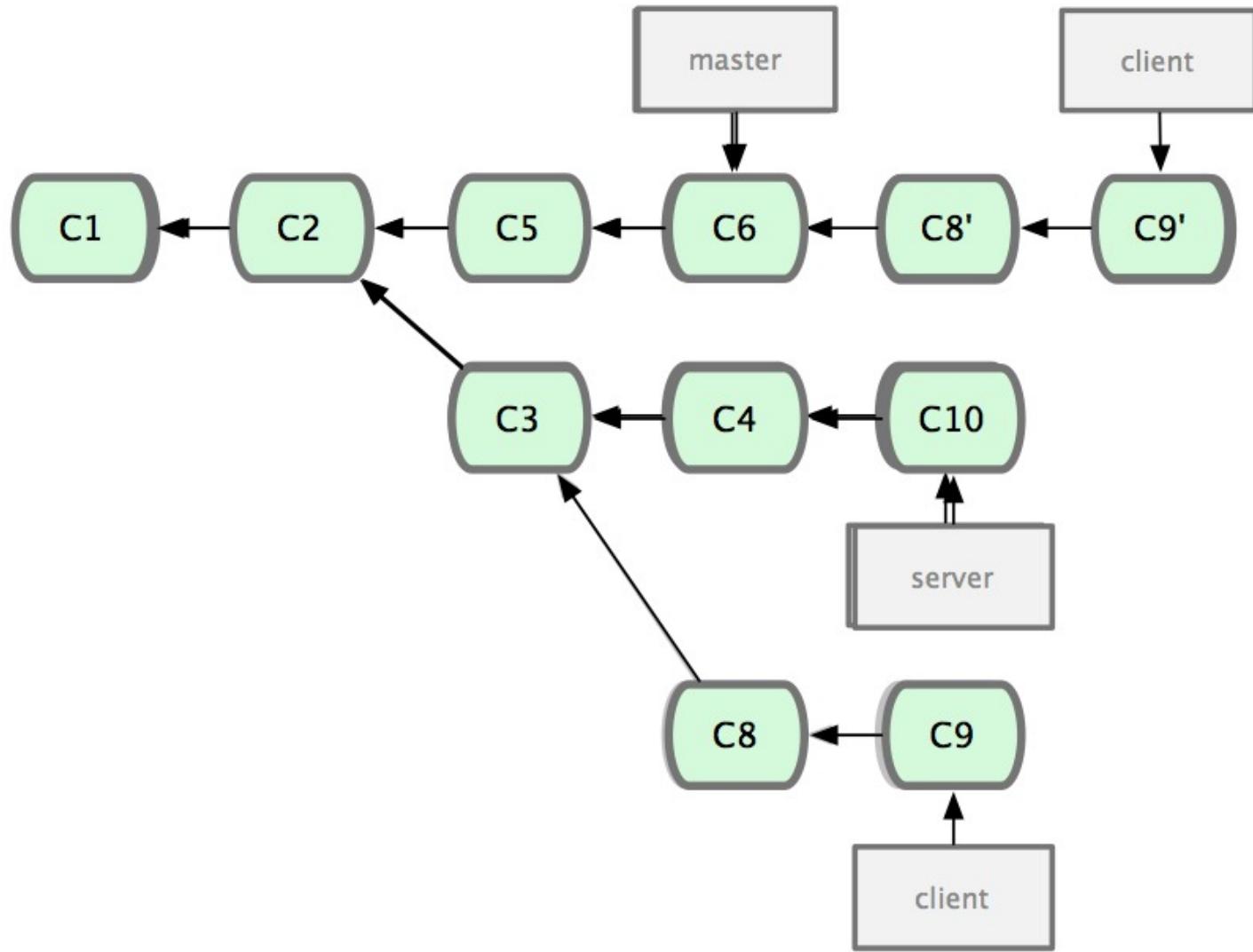
`git rebase server --onto master`

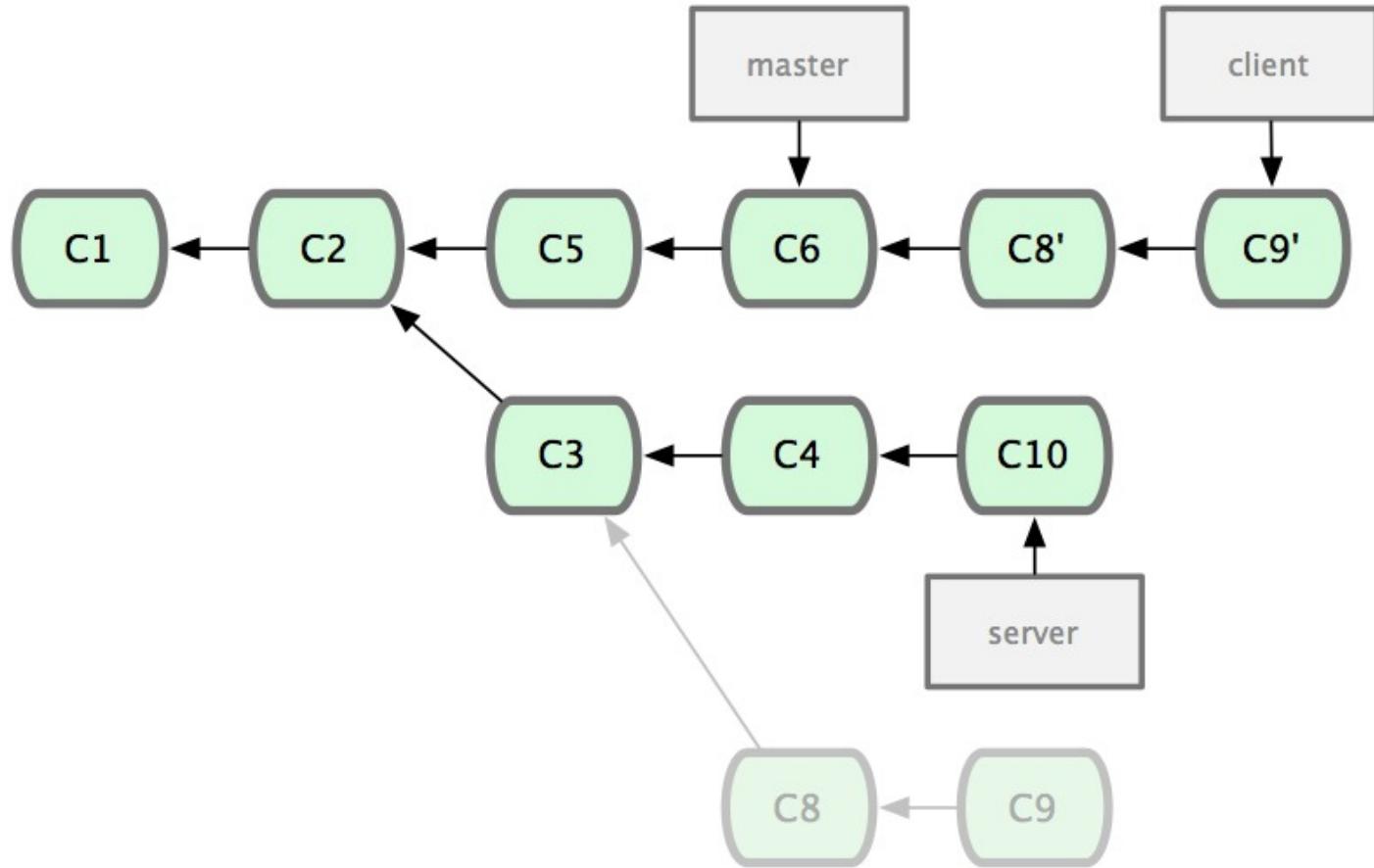


`git rebase server --onto master`

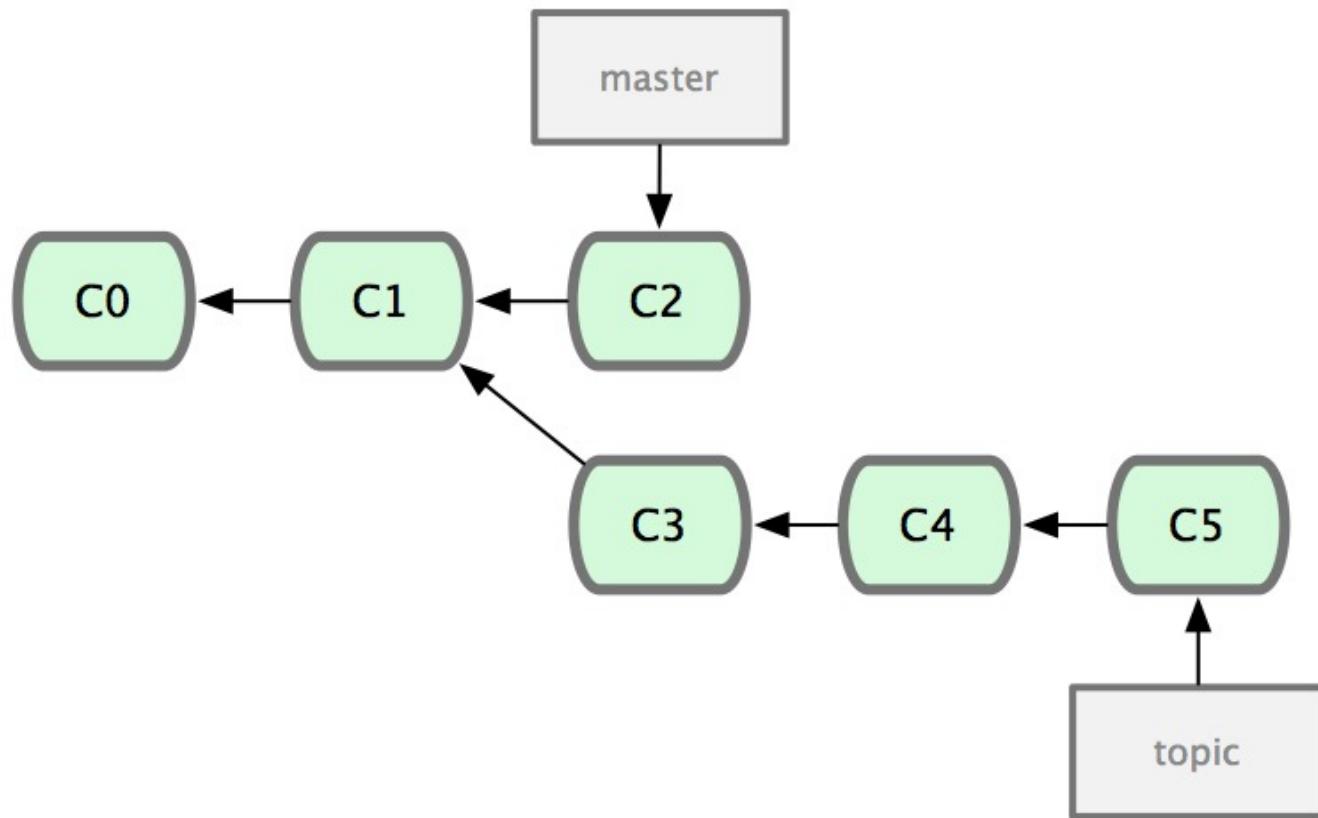


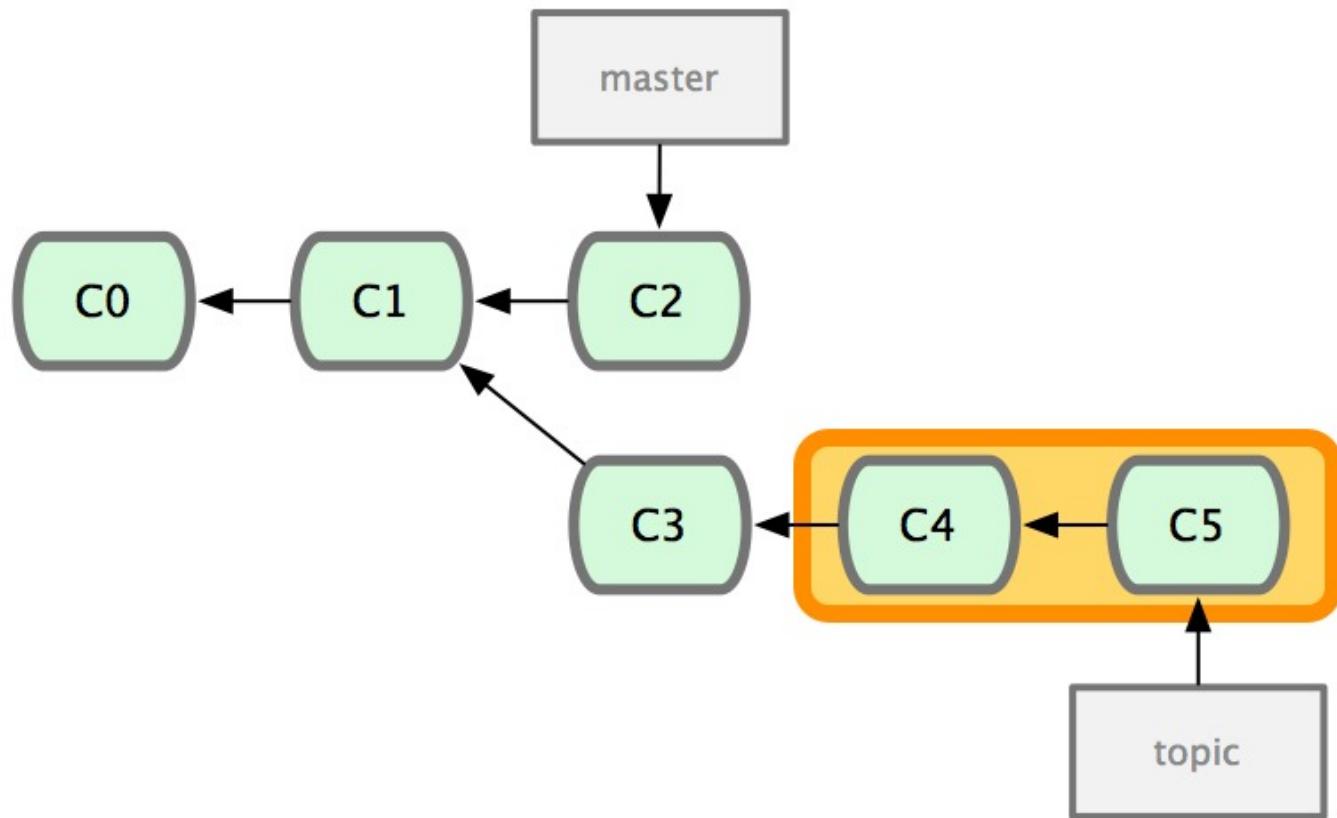
git rebase server --onto master

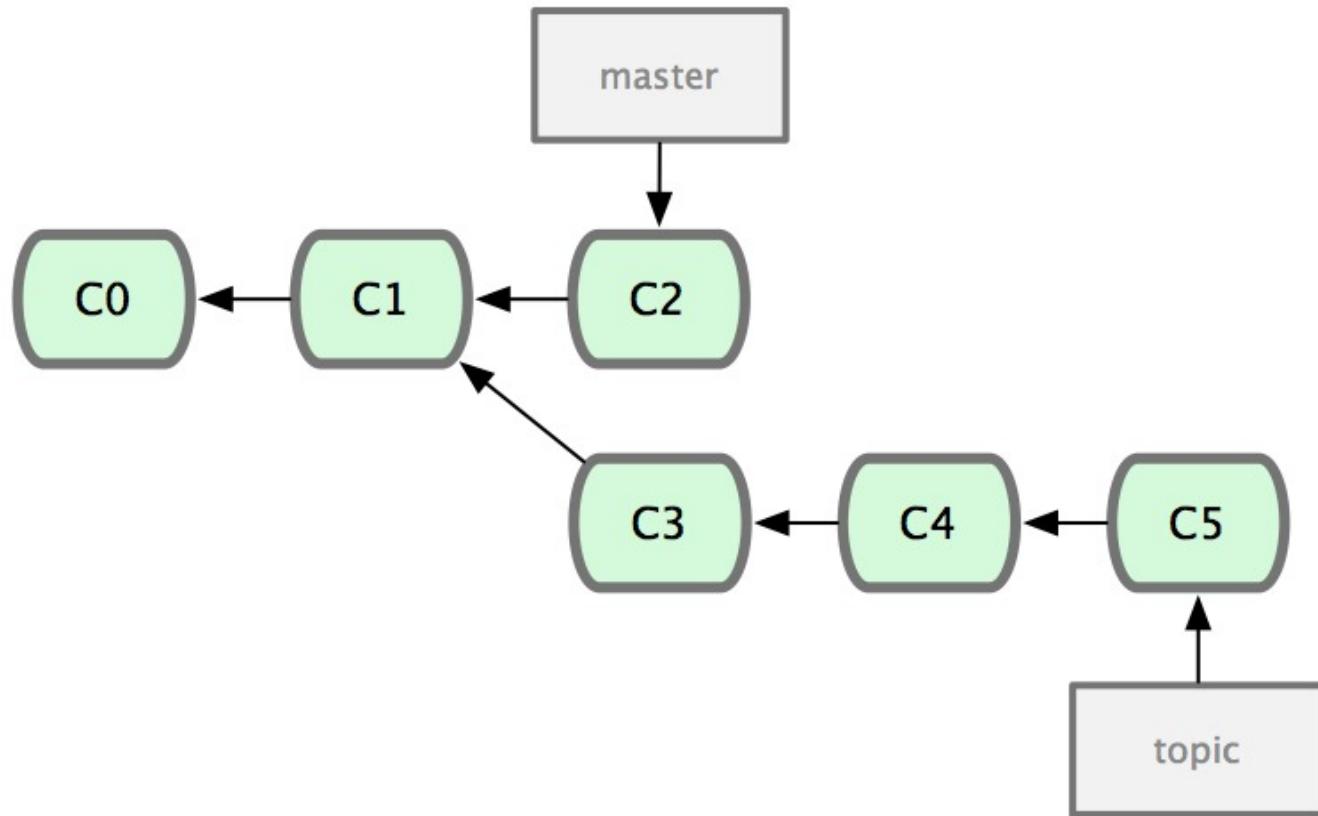




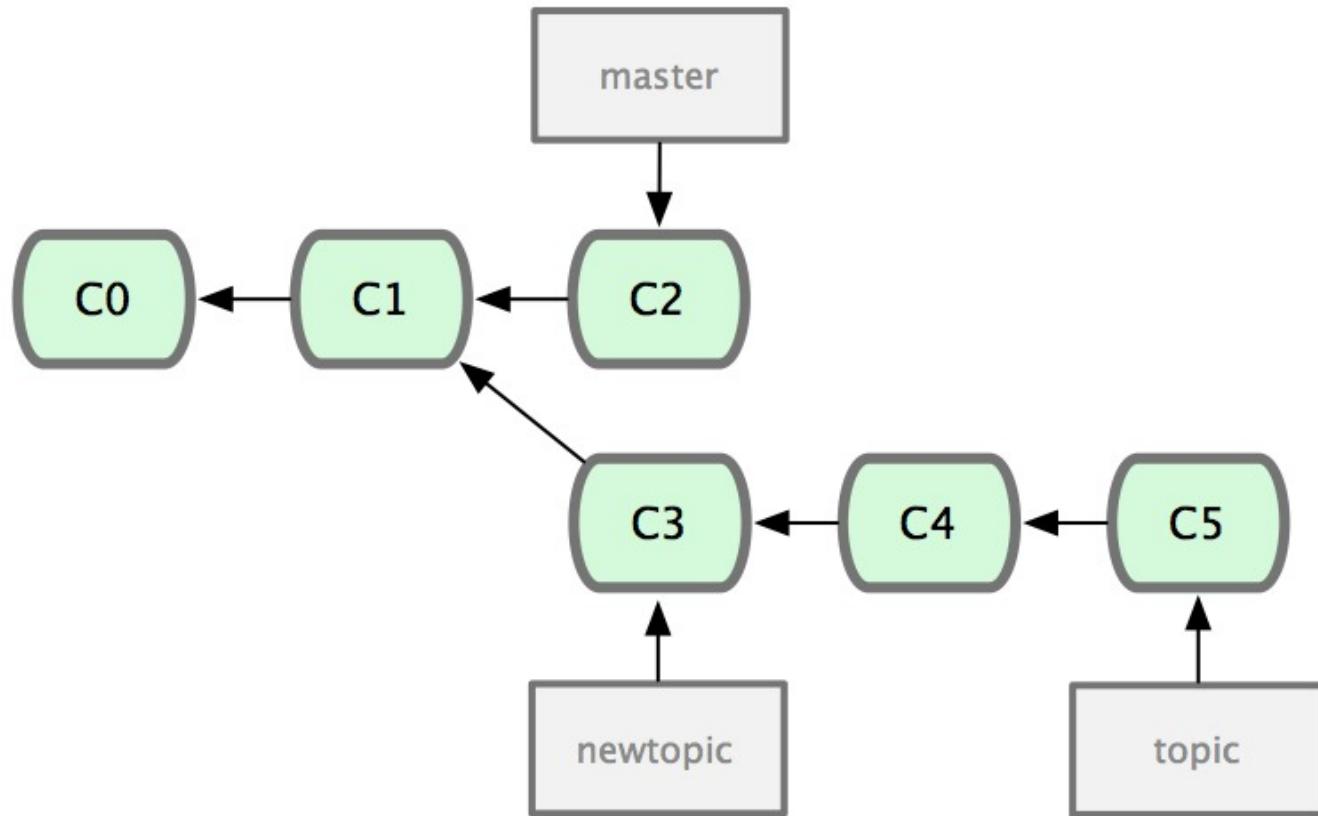
**Transplant some of a topic
branch**



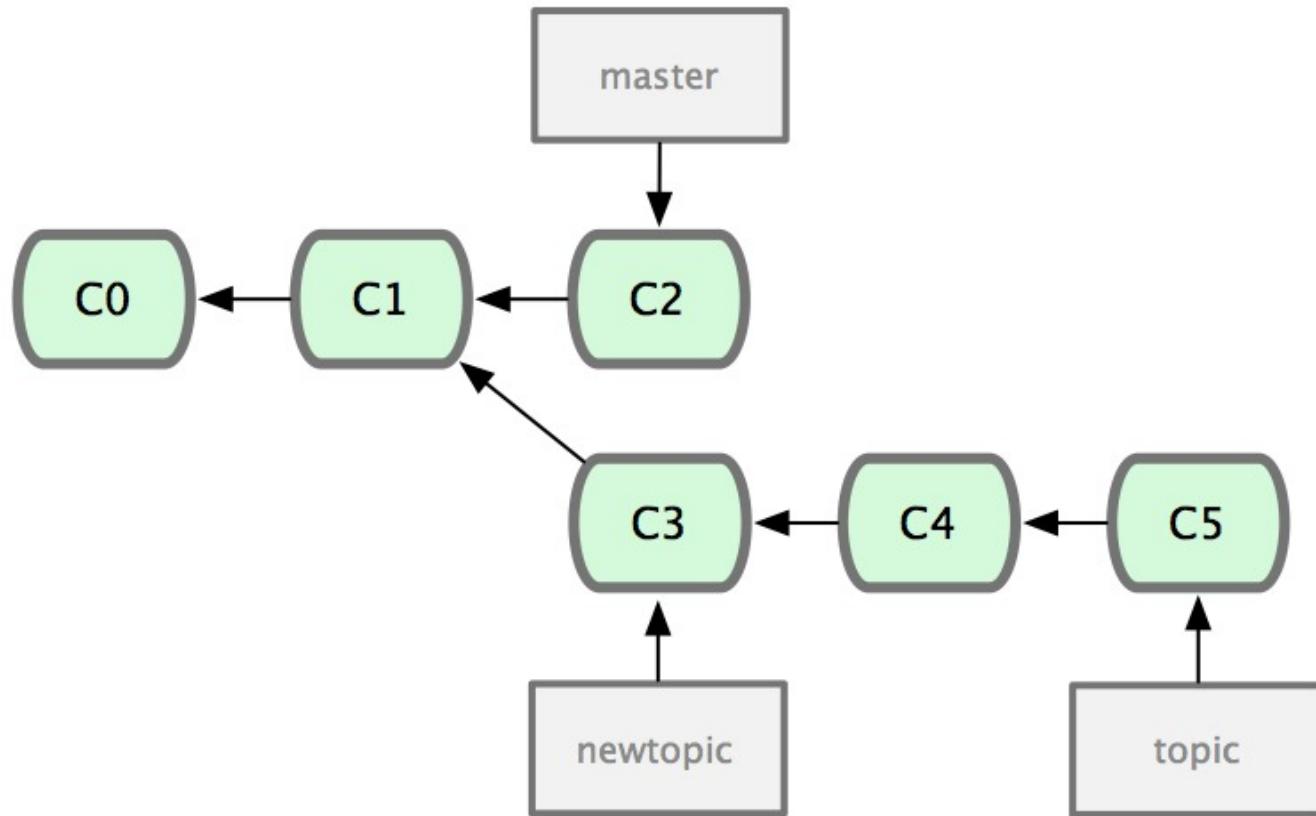




`git branch newtopic C3`

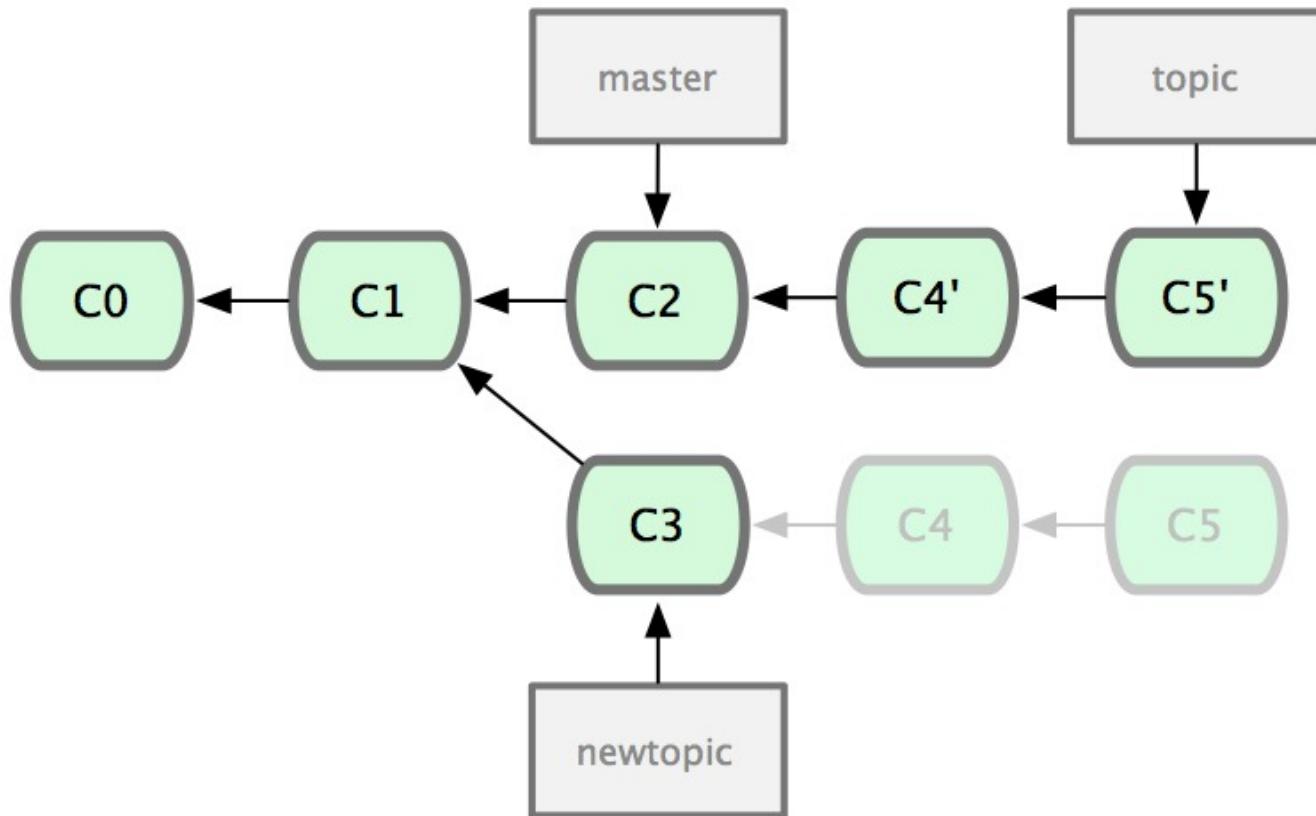


`git branch newtopic C3`



```
git branch newtopic C3
```

```
git rebase newtopic --onto master
```



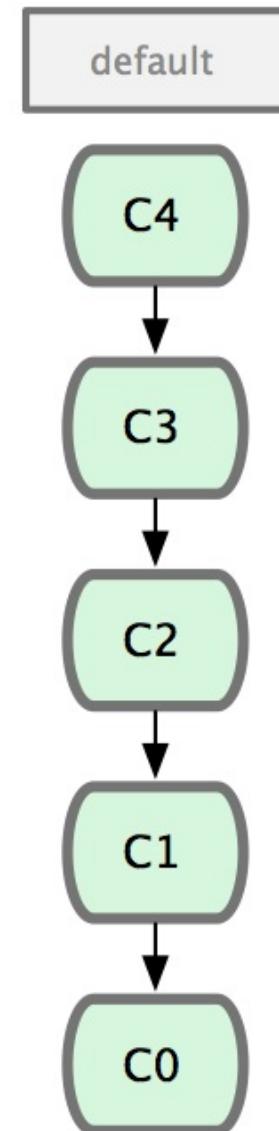
```
git branch newtopic C3
```

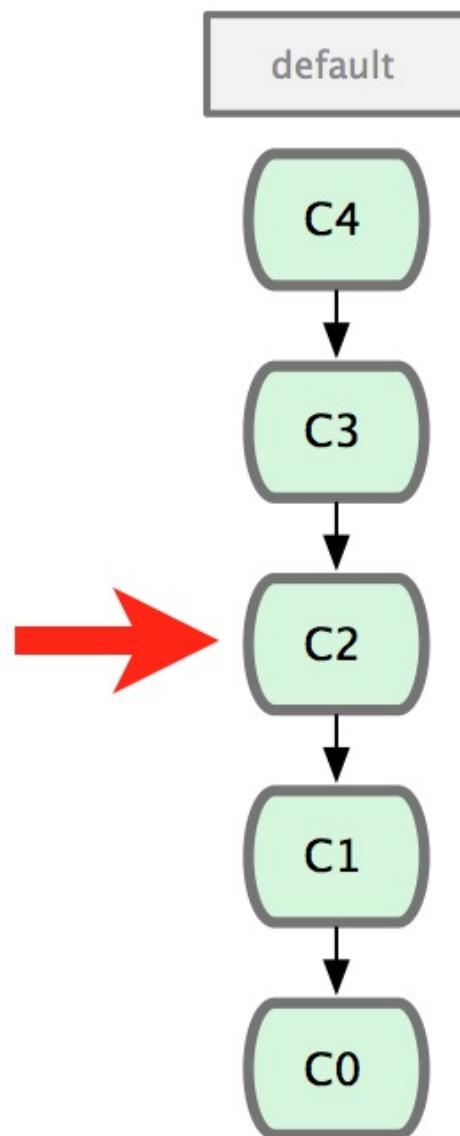
```
git rebase newtopic --onto master
```

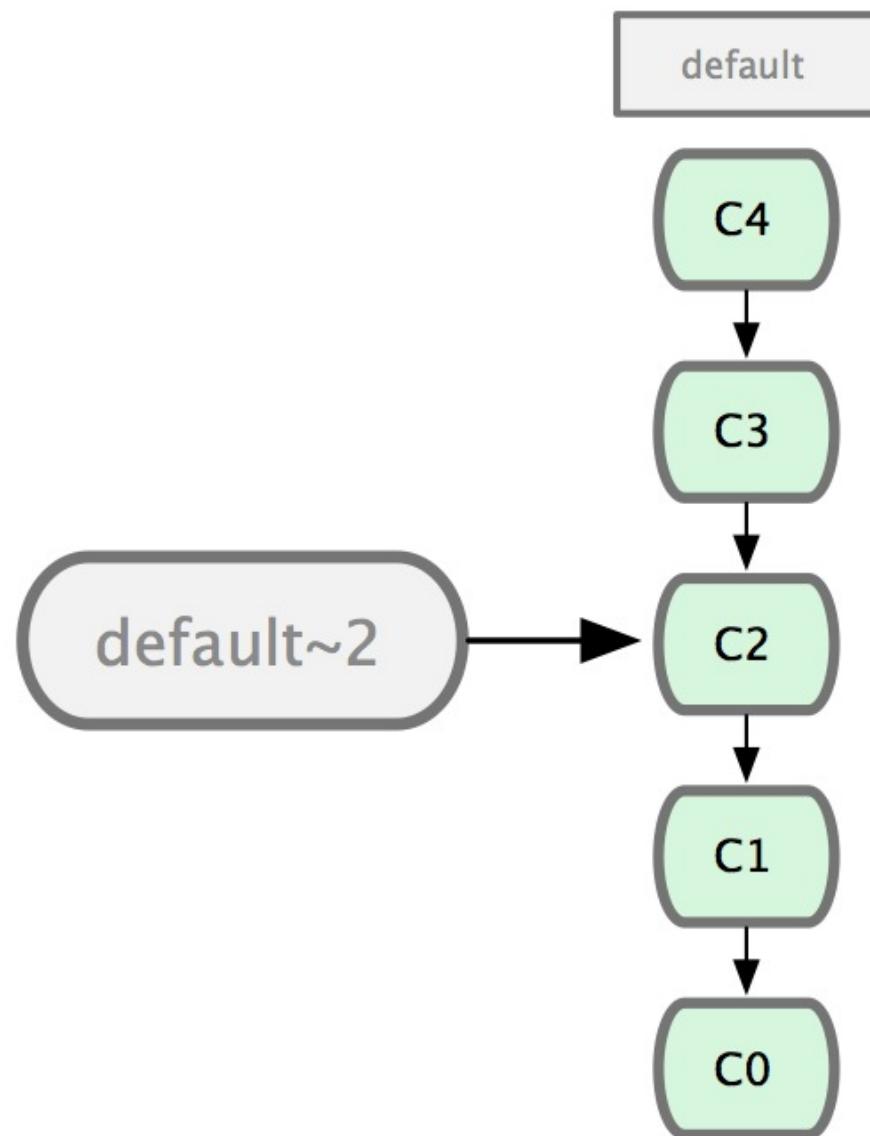
Fixing a commit several back

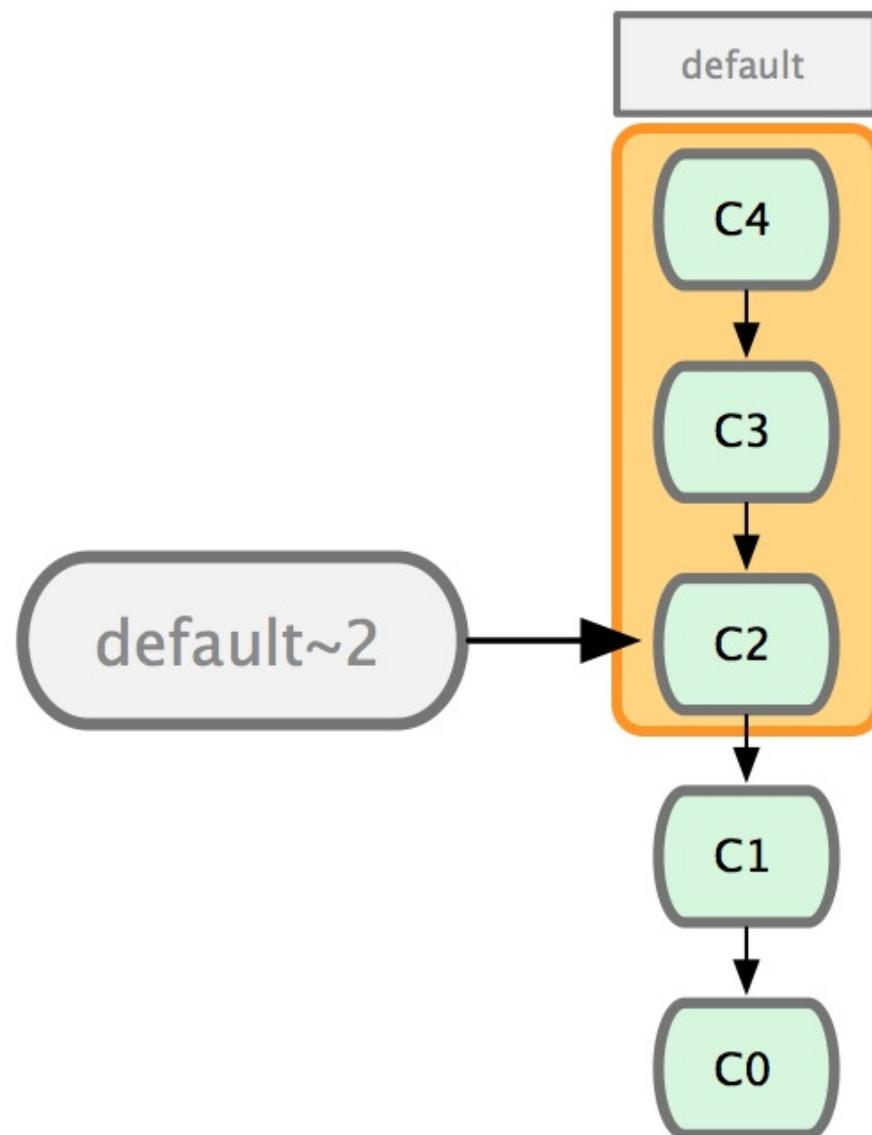
git rebase -i

git rebase --interactive





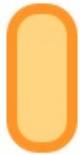




```
git rebase -i default~2^
```

```
git rebase -i default~2^
```

```
git rebase -i HEAD~3
```



```
pick 969c877 git apply --directory broken for new files
pick b75271d git diff <tree>{3,}: do not reverse order of args
pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
C2 pick 969c877 git apply --directory broken for new files
C3 pick b75271d git diff <tree>{3,}: do not reverse order of args
C4 pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
pick 969c877 git apply --directory broken for new files
pick b75271d git diff <tree>{3,}: do not reverse order of args
pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
# p, pick = use commit
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
pick 969c877 git apply --directory broken for new files
pick b75271d git diff <tree>{3,}: do not reverse order of args
pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
pick 969c877 git apply --directory broken for new files
pick b75271d git diff <tree>{3,}: do not reverse order of args
pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
pick 969c877 git apply --directory broken for new files
pick b75271d git diff <tree>{3,}: do not reverse order of args
pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
pick 969c877 git apply --directory broken for new files
pick b75271d git diff <tree>{3,}: do not reverse order of args
pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
edit 969c877 git apply --directory broken for new files
pick b75271d git diff <tree>{3,}: do not reverse order of args
pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
$ git rebase -i default~2^
Stopped at 969c877... git apply --directory broken for new files
You can amend the commit now, with
```

```
    git commit --amend
```

Once you are satisfied with your changes, run

```
    git rebase --continue
```

```
$ _
```

```
$ git rebase -i default~2^  
Stopped at 969c877... git apply --directory broken for new files  
You can amend the commit now, with
```

```
    git commit --amend
```

Once you are satisfied with your changes, run

```
    git rebase --continue
```

```
$_
```

edit files

```
$ git rebase -i default~2^  
Stopped at 969c877... git apply --directory broken for new files  
You can amend the commit now, with
```

```
    git commit --amend
```

Once you are satisfied with your changes, run

```
    git rebase --continue
```

```
$_
```

**edit files
git add**

```
$ git rebase -i default~2^  
Stopped at 969c877... git apply --directory broken for new files  
You can amend the commit now, with
```

 **git commit --amend**

Once you are satisfied with your changes, run

```
git rebase --continue
```

```
$_
```

edit files
git add
git commit --amend

```
$ git rebase -i default~2^  
Stopped at 969c877... git apply --directory broken for new files  
You can amend the commit now, with
```

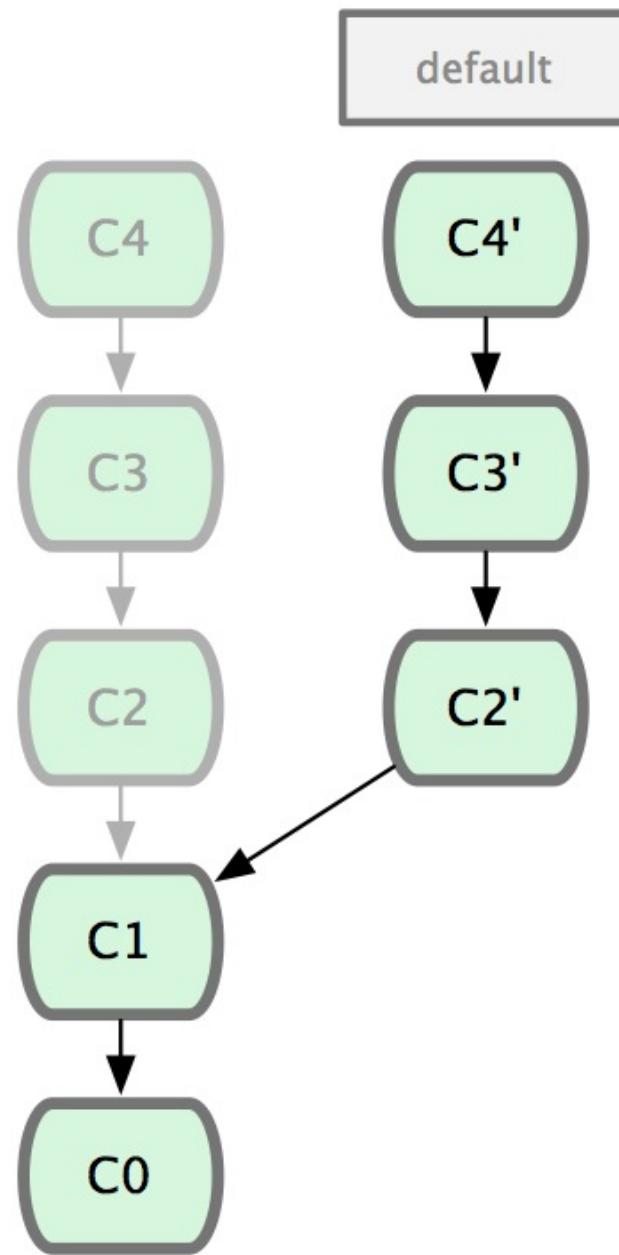
```
    git commit --amend
```

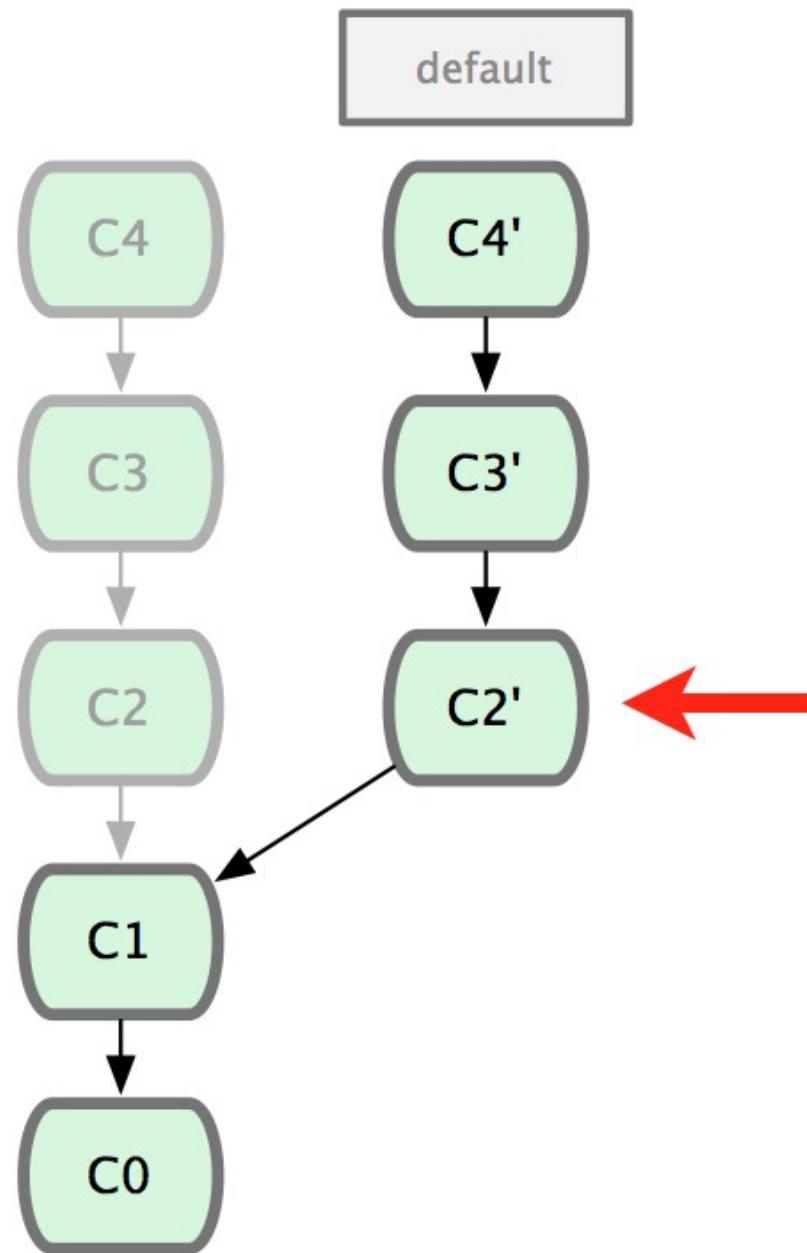
Once you are satisfied with your changes, run

 **git rebase --continue**

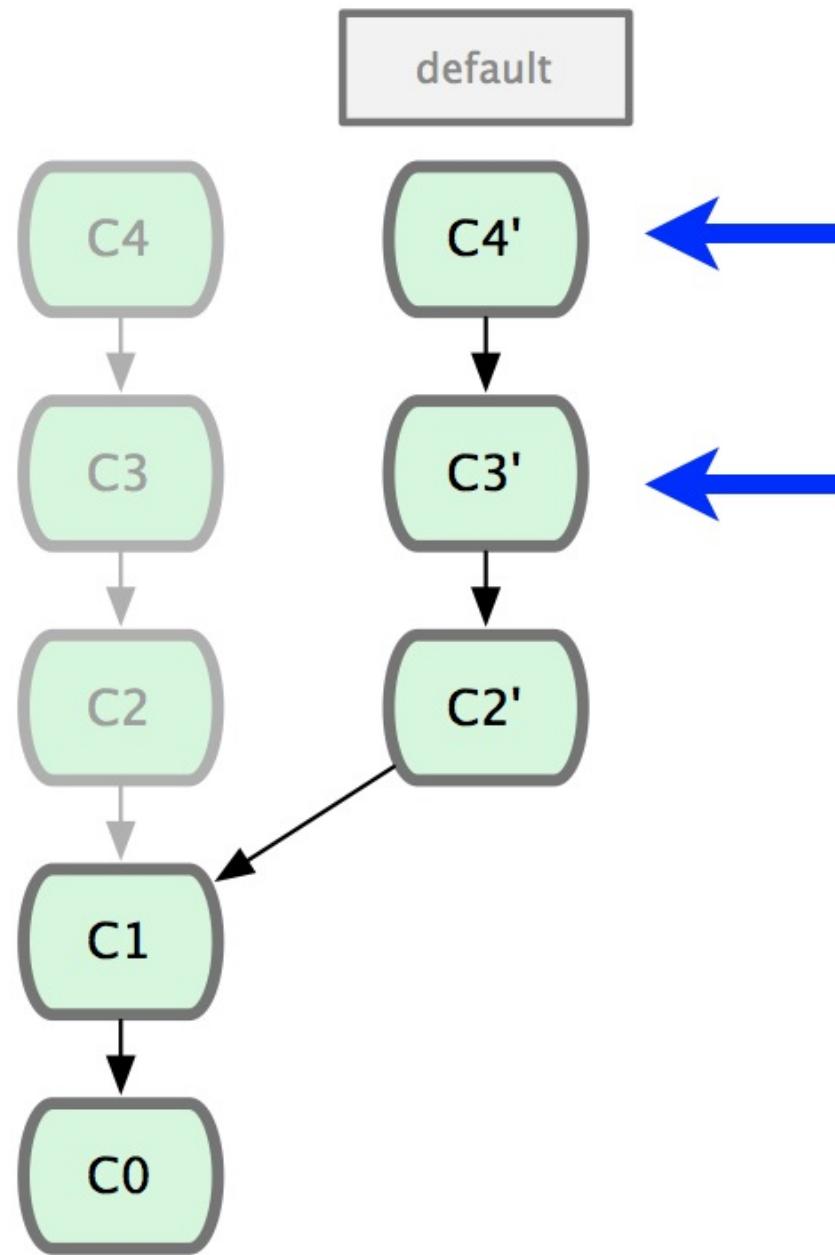
```
$_
```

edit files
git add
git commit --amend
git rebase --continue





default



Squashing commits together

```
pick 969c877 git apply --directory broken for new files
pick b75271d git diff <tree>{3,}: do not reverse order of args
pick 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
pick 969c877 git apply --directory broken for new files
squash b75271d git diff <tree>{3,}: do not reverse order of args
squash 72d404d test-lib: fix broken printf

# Rebase f285a2d..5c283eb onto f285a2d
#
# Commands:
#   p, pick = use commit
#   e, edit = use commit, but stop for amending
#   s, squash = use commit, but meld into previous commit
#
# If you remove a line here THAT COMMIT WILL BE LOST.
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
~
~

"~/projects/git/.git/rebase-merge/git-rebase-todo" 14L, 472C
```

```
# This is a combination of 3 commits.
# The first commit's message is:
git apply --directory broken for new files

# This is the 2nd commit message:

git diff <tree>{3,}: do not reverse order of args

# This is the 3rd commit message:

test-lib: fix broken printf

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Author:    Jeff King <peff@peff.net>
#
# Not currently on any branch.
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
# modified:  builtin-apply.c
# modified:  builtin-diff.c
# modified:  t/t4013-diff-various.sh
# new file:  t/t4013/diff.diff_master_master^_side
# modified:  t/t4128-apply-root.sh
# modified:  t/test-lib.sh
#
~
~
".git/COMMIT_EDITMSG" 39L, 1454C
```

```
# This is a combination of 3 commits.
# The first commit's message is:
git apply --directory broken for new files

# This is the 2nd commit message:

git diff <tree>{3,}: do not reverse order of args

# This is the 3rd commit message:

test-lib: fix broken printf

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Author:    Jeff King <peff@peff.net>
#
# Not currently on any branch.
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
# modified:  builtin-apply.c
# modified:  builtin-diff.c
# modified:  t/t4013-diff-various.sh
# new file:  t/t4013/diff.diff_master_master^_side
# modified:  t/t4128-apply-root.sh
# modified:  t/test-lib.sh
#
~
~

".git/COMMIT_EDITMSG" 39L, 1454C
```

```
# This is a combination of 3 commits.
# The first commit's message is:
git apply --directory broken for new files

# This is the 2nd commit message:

git diff <tree>{3,}: do not reverse order of args

# This is the 3rd commit message:

test-lib: fix broken printf

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Author:    Jeff King <peff@peff.net>
#
# Not currently on any branch.
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
# modified:  builtin-apply.c
# modified:  builtin-diff.c
# modified:  t/t4013-diff-various.sh
# new file:  t/t4013/diff.diff_master_master^_side
# modified:  t/t4128-apply-root.sh
# modified:  t/test-lib.sh
#
~
~

".git/COMMIT_EDITMSG" 39L, 1454C
```

```
# This is a combination of 3 commits.
# The first commit's message is:
git apply --directory broken for new files

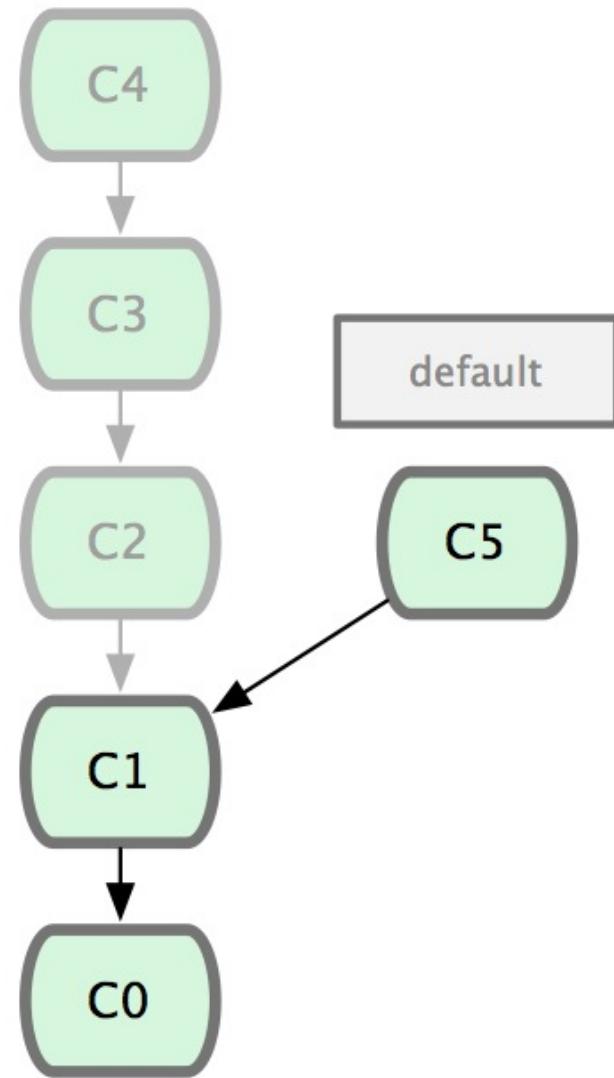
# This is the 2nd commit message:

git diff <tree>{3,}: do not reverse order of args

# This is the 3rd commit message:

test-lib: fix broken printf

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Author:    Jeff King <peff@peff.net>
#
# Not currently on any branch.
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
# modified:  builtin-apply.c
# modified:  builtin-diff.c
# modified:  t/t4013-diff-various.sh
# new file:  t/t4013/diff.diff_master_master^_side
# modified:  t/t4128-apply-root.sh
# modified:  t/test-lib.sh
#
~
~
".git/COMMIT_EDITMSG" 39L, 1454C
```

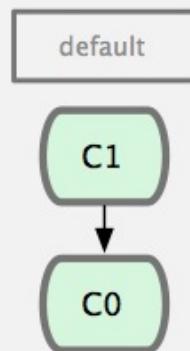


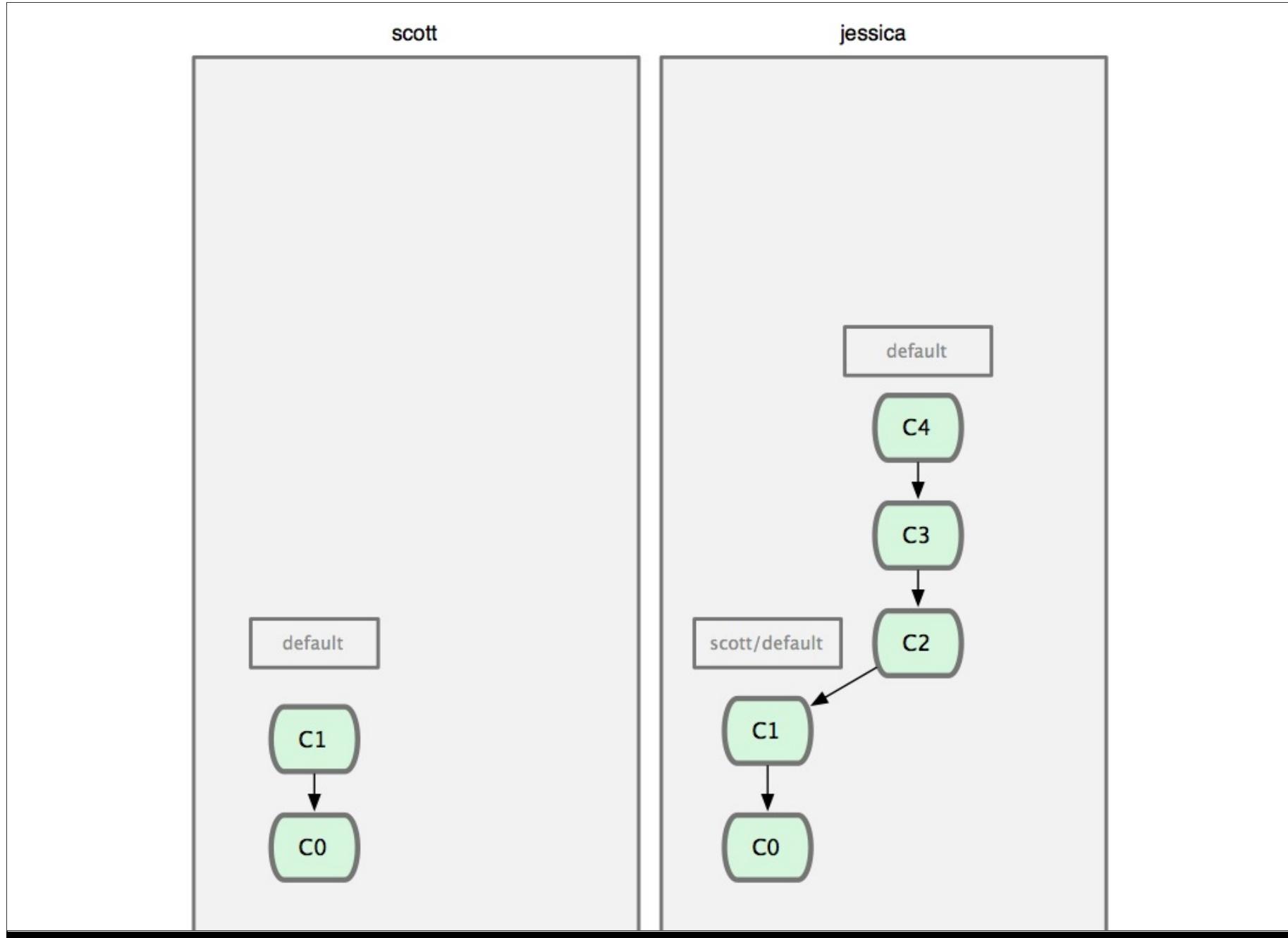
Splitting a Commit

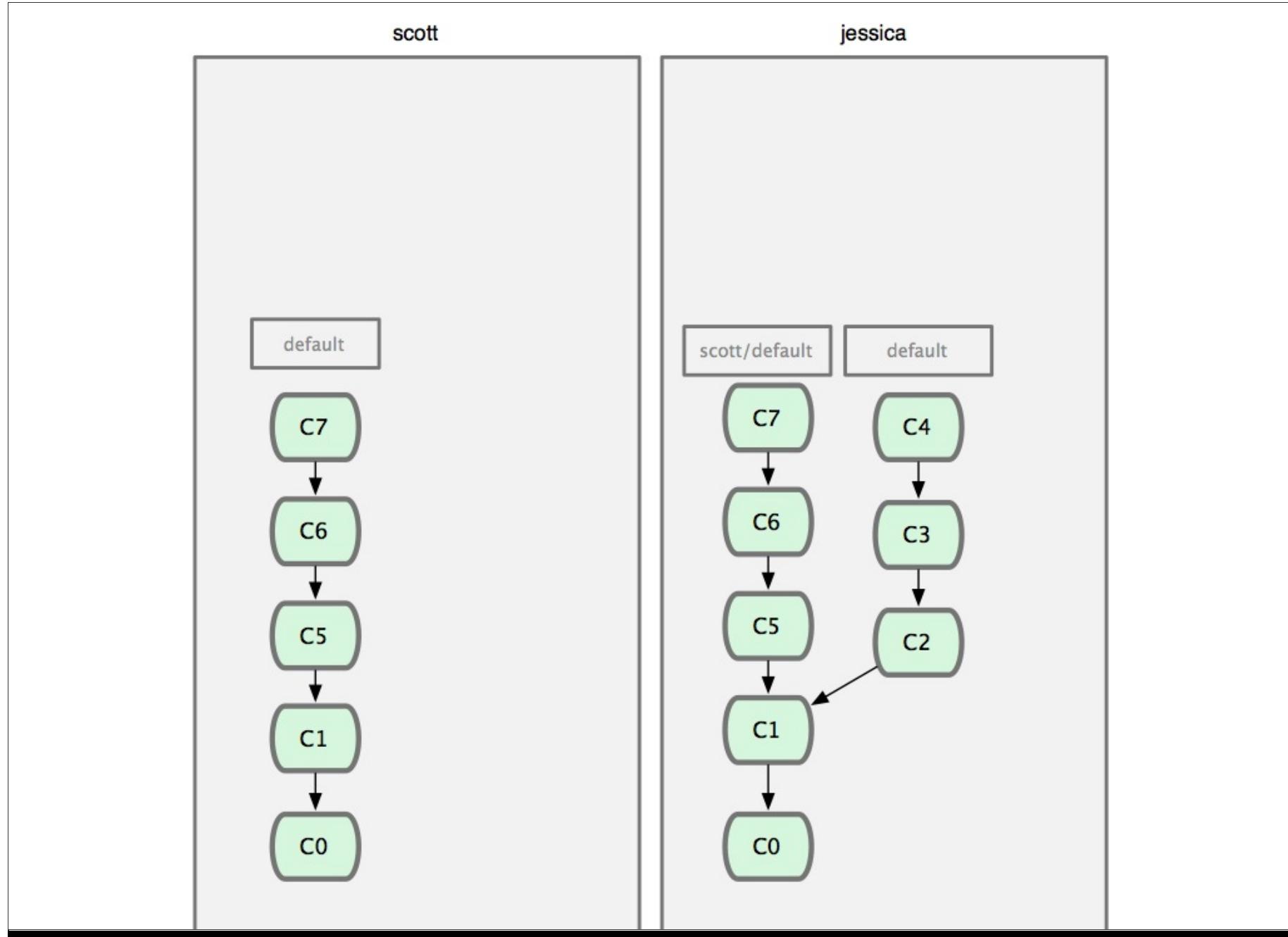
The Perils

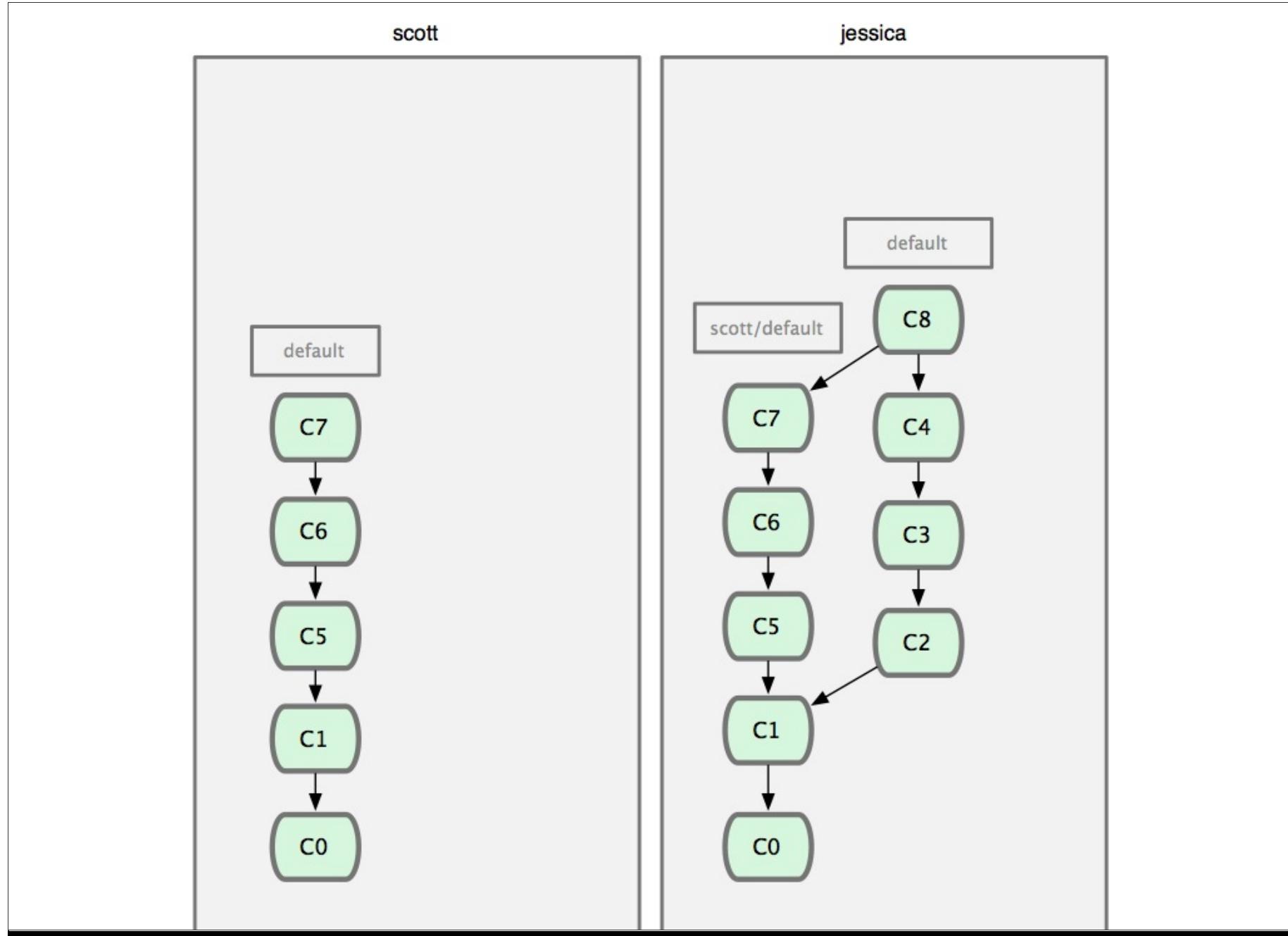
scott

jessica

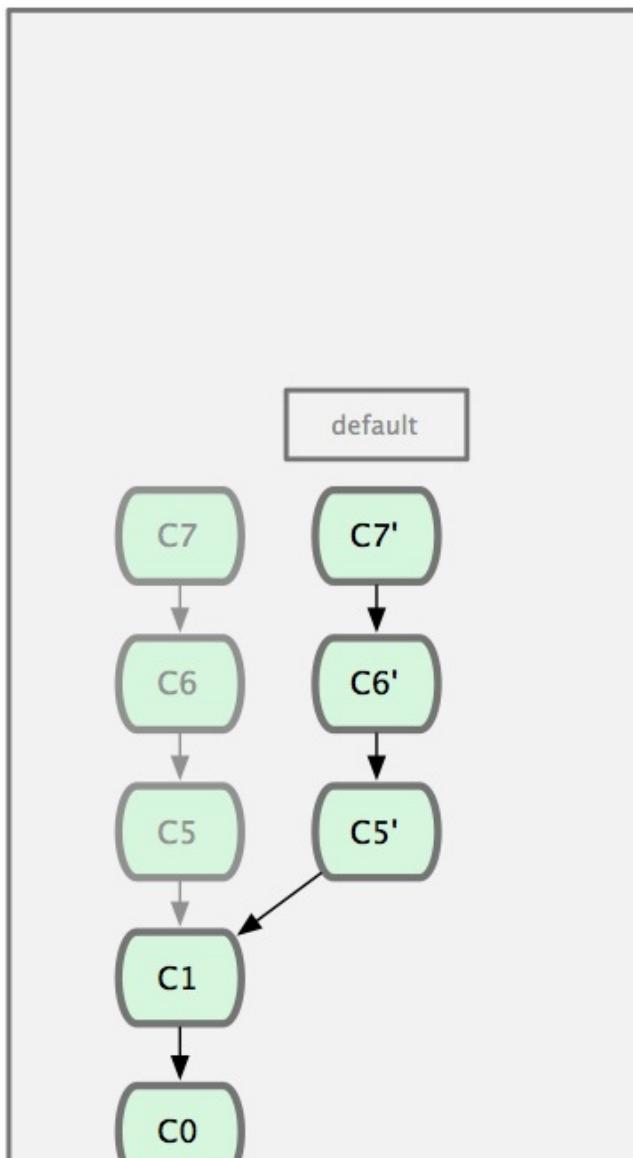




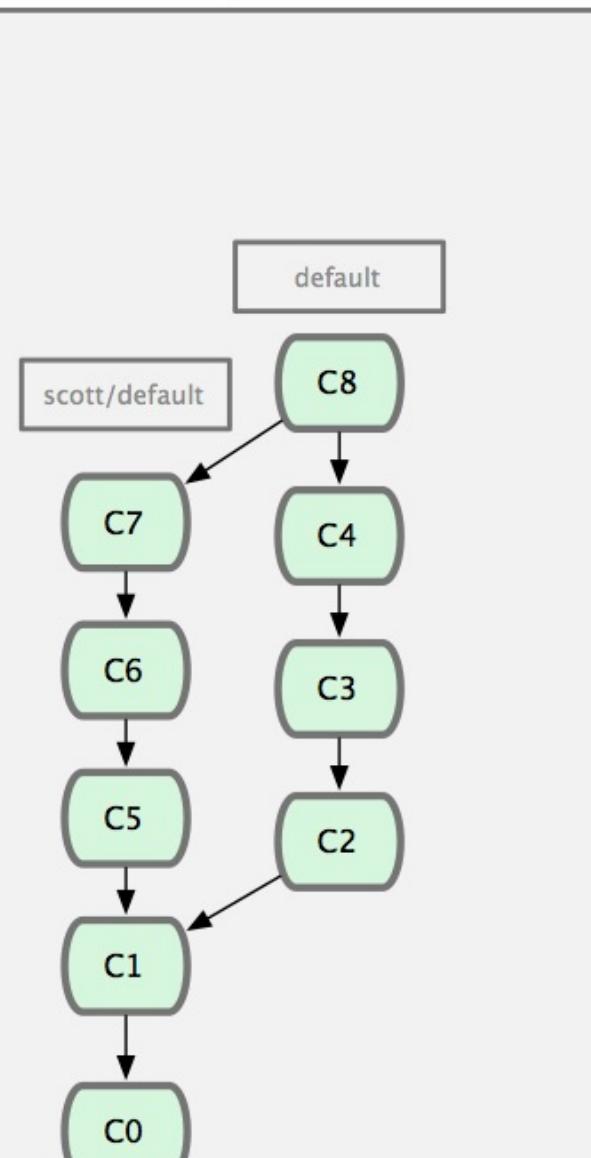


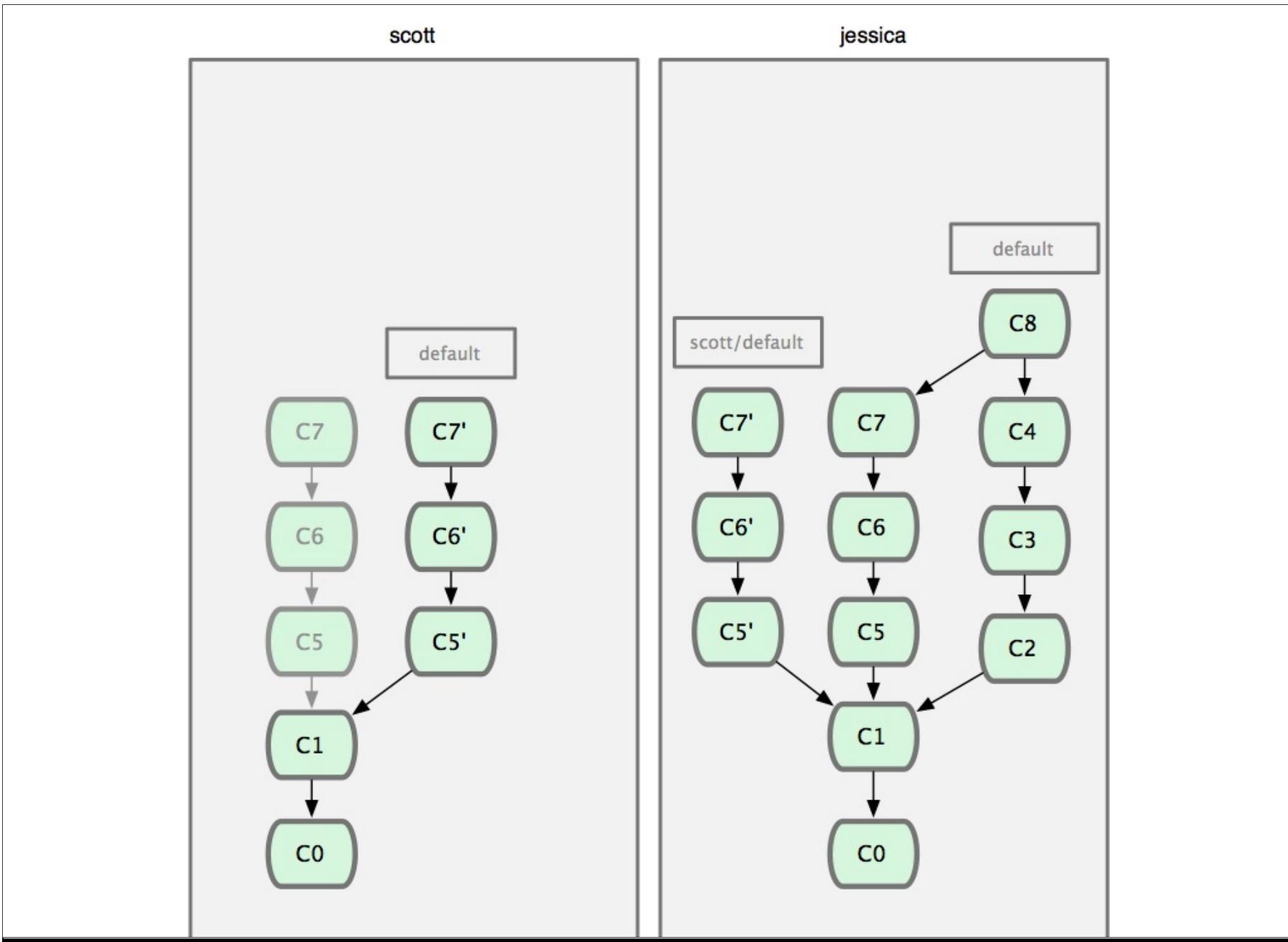


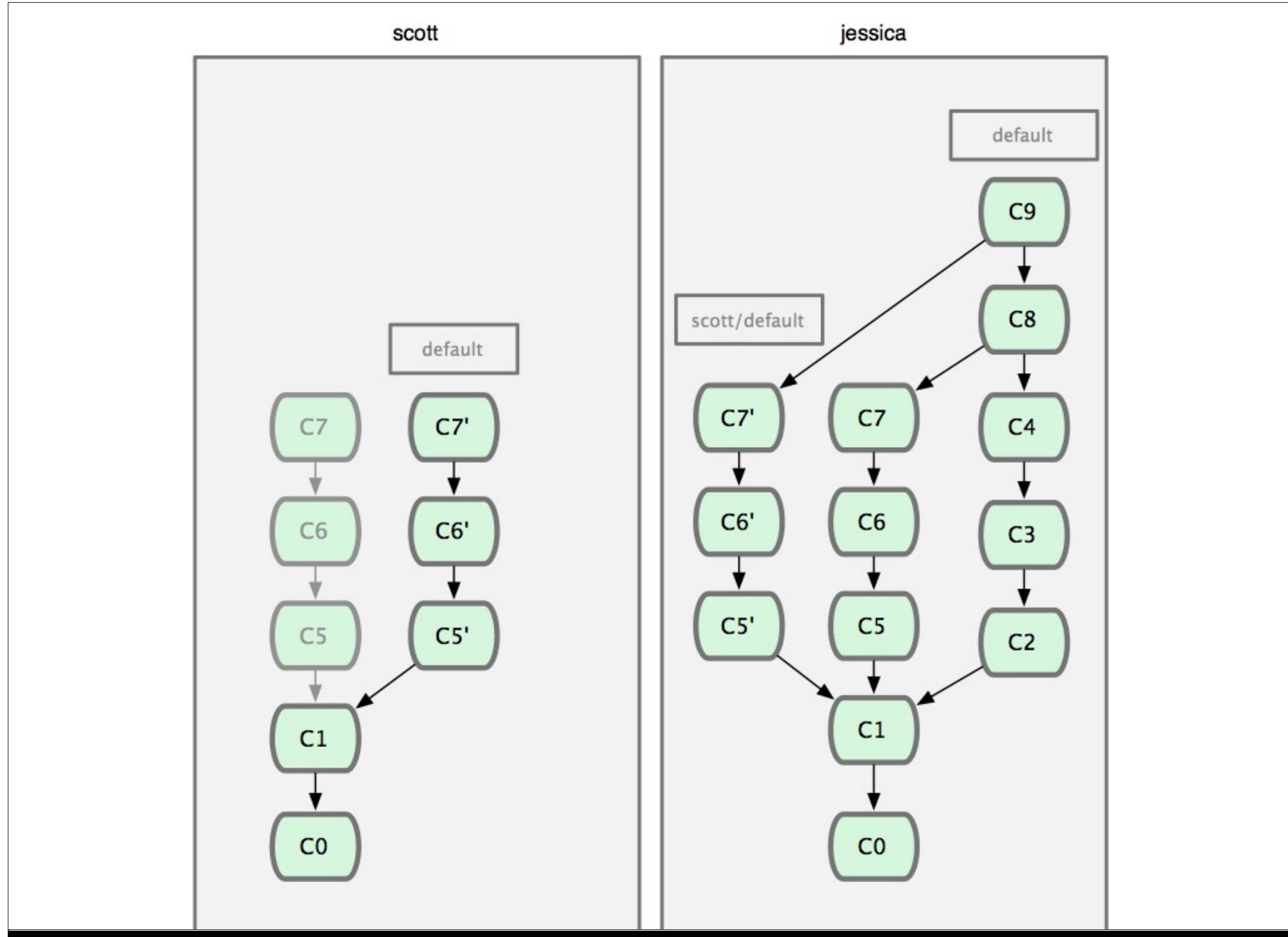
scott



jessica







Tutorial

checkout a new branch called myrebase

modify three files, commit each separately

squash the second and third commits together
using `git rebase -i`

rebase the two new commits on top of the
`sweet_potatoes` branch

Cherry

```
$ git log --oneline origin/gsoc-index ^HEAD
b0941f9 Add support for inserting empty entries on the index
8c870fc Add support for extended index entries
b376cb1 Add tests to t0601 to cover the TREE extension
edc438e Move test resources to a common directory
933cb11 Add support for the TREE index extension
6018dbf Add unit tests for index manipulation

$ git cherry -v HEAD origin/gsoc-index
+ 6018dbfa702b818590ae682bf9bbe716ef290e Add unit tests for
+ 933cb118437ee8a4422e956197a8a4f09fd7e9df Add support for the
+ edc438eedf6854c51e1a0d7954a6849046f5a4f6 Move test resources
+ b376cb120f0113294b80e7fd540d5dc197797764 Add tests to t0601
+ 8c870fcebb8f625a8e172a49a44153af8f37c8b7 Add support for ext
+ b0941f9c70ffe67f0387a827b338e64ecf3190f0 Add support for ins
```

```
$ git cherry-pick 6018dbfa70
Finished one cherry-pick.
[test e144b96] Add unit tests for index manipulation
Author: Vicent Marti <tanoku@gmail.com>
 4 files changed, 209 insertions(+), 0 deletions(-)
$ git cherry-pick edc438eedf
(output)
$ git cherry-pick b376cb1
(output)

$ git cherry -v HEAD origin/gsoc-index
- 6018dbfa702b818590ae682bf9bbe716ef290e Add unit tests for
+ 933cb118437ee8a4422e956197a8a4f09fd7e9df Add support for the
- edc438eedf6854c51e1a0d7954a6849046f5a4f6 Move test resources
- b376cb120f0113294b80e7fd540d5dc197797764 Add tests to t0601
+ 8c870fcebb8f625a8e172a49a44153af8f37c8b7 Add support for ext
+ b0941f9c70ffe67f0387a827b338e64ecf3190f0 Add support for ins
```

Advanced Merging

Merge Conflict Resolution

```
$ git merge topic
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the resu
```

```
$ git diff
diff --cc file.txt
index 802992c,2b60207..0000000
--- a/file.txt
+++ b/file.txt
@@@ -1,1 -1,1 +1,5 @@@
++<<<<< HEAD:file.txt
+Hello world
=====
+ Goodbye
++>>>>> 77976da35a11db4580b80ae27e8d65caf5208086:file.txt
```

```
$ git show :1:file.txt
# the file in a common ancestor of both branches

$ git show :2:file.txt
# the version from HEAD.

$ git show :3:file.txt
# the version from MERGE_HEAD.
```

After Fixing

```
$ git diff
diff --cc file.txt
index 802992c,2b60207..0000000
--- a/file.txt
+++ b/file.txt
@@@ -1,1 -1,1 +1,1 @@@
- Hello world
-Goodbye
++Goodbye world
```

```
git log --merge
```

only commits that touch an unmerged file

git add [file]

gets it out of the way

```
git checkout --ours -- [file]
```

```
git checkout --theirs -- [file]
```

```
git checkout -m -- [file]
```

```
git checkout --conflict=diff3 (file)
```

```
$ cat conflict_file.txt
<<<<< HEAD
2 cups white rice
3 cups water
=====
1/3 cup uncooked white rice
2 cups water
>>>>> conflict
```

```
$ git checkout --conflict=diff3 conflict_file.txt
<<<<< ours
2 cups white rice
3 cups water
||||||| base
2/3 cup uncooked white rice
1 cup water
=====
1/3 cup uncooked white rice
2 cups water
>>>>> theirs
```

```
git config --global merge.conflictstyle diff3
```

```
git merge-file mine common theirs
```

manually re-do a file merge

Fixing Whitespace Merge Hell

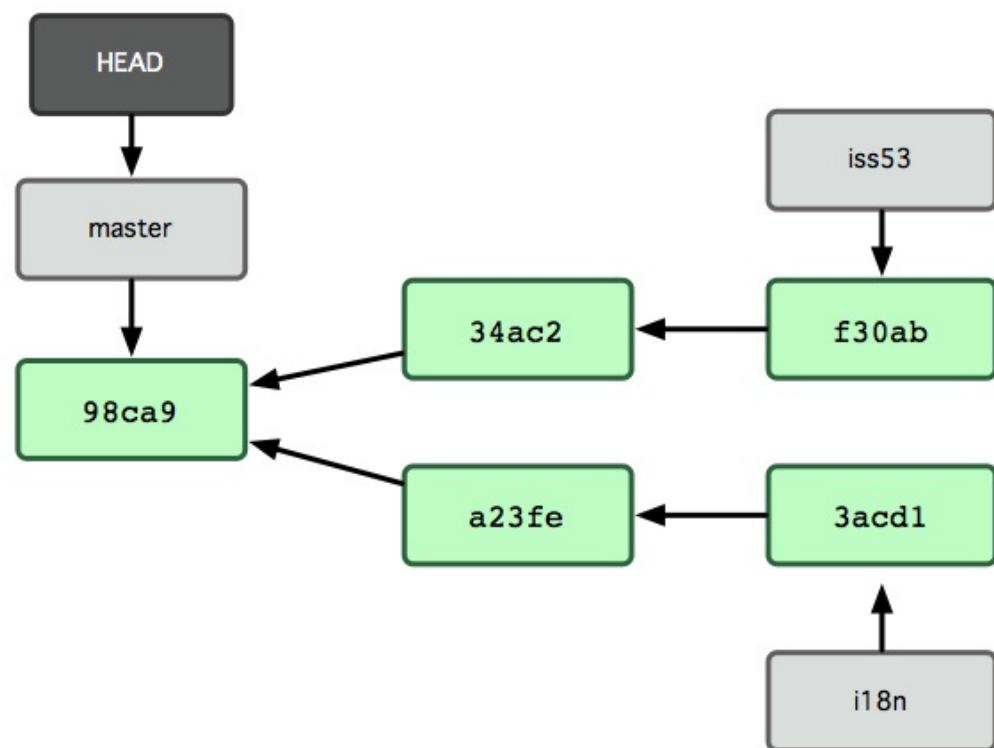
```
$ git show :1:file.txt > common  
$ git show :2:file.txt > mine  
$ git show :3:file.txt > theirs  
  
$ dos2unix theirs  
  
$ git merge-file mine common theirs  
  
$ cp mine file.txt  
$ git add file.txt
```

abandon a merge

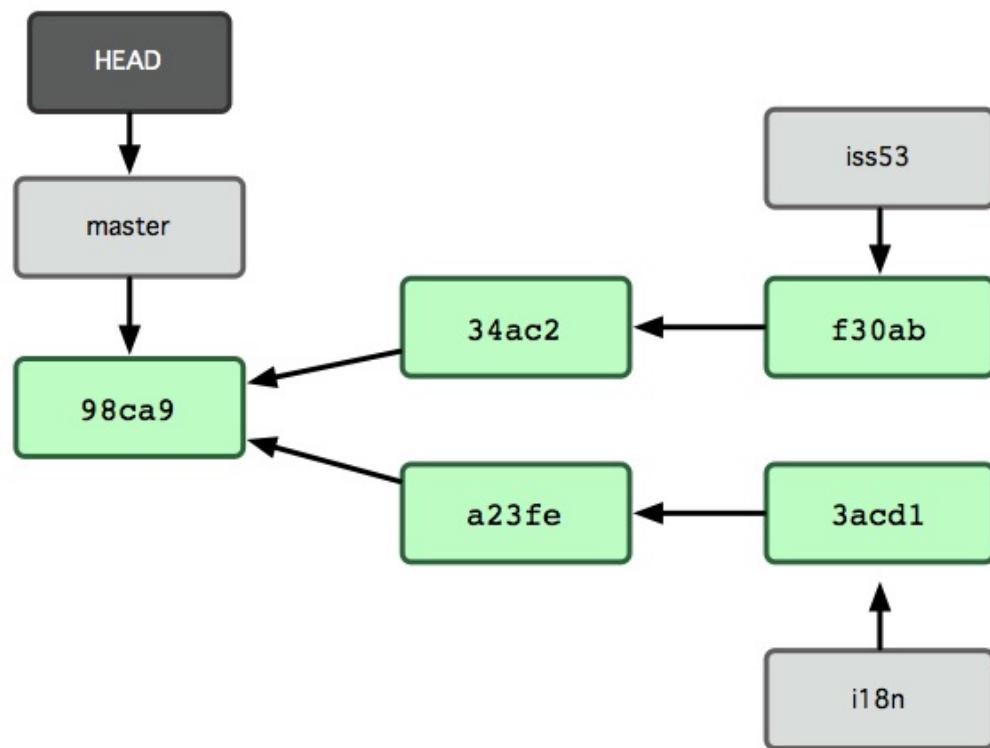
```
git reset --hard HEAD
```

other merging types

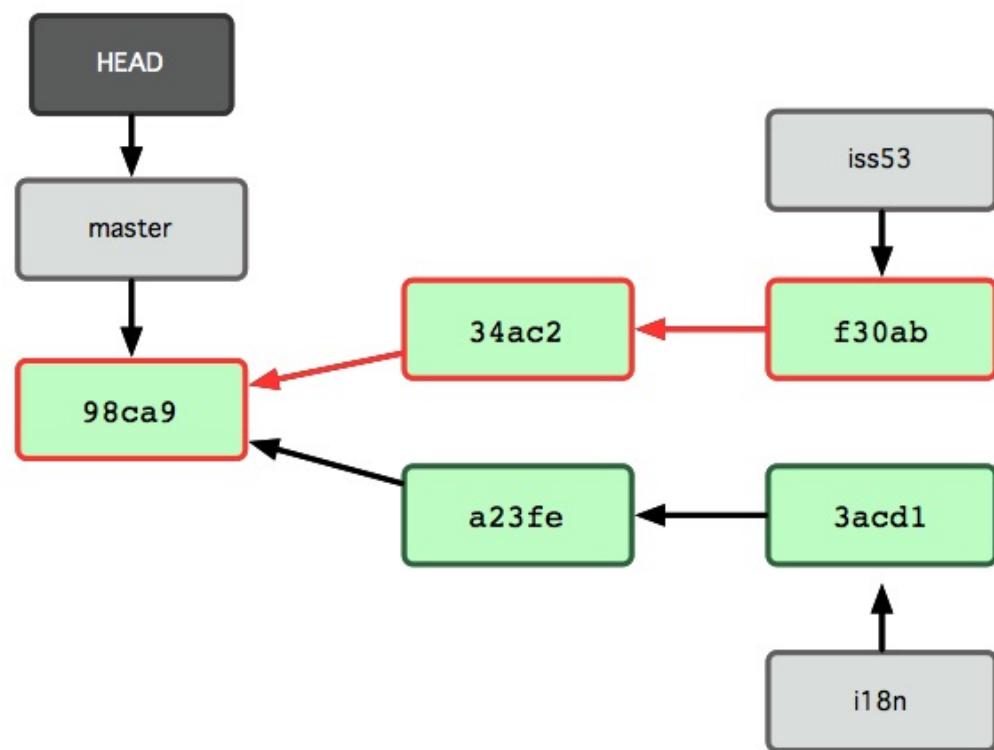
no fast forwarding



git merge iss53

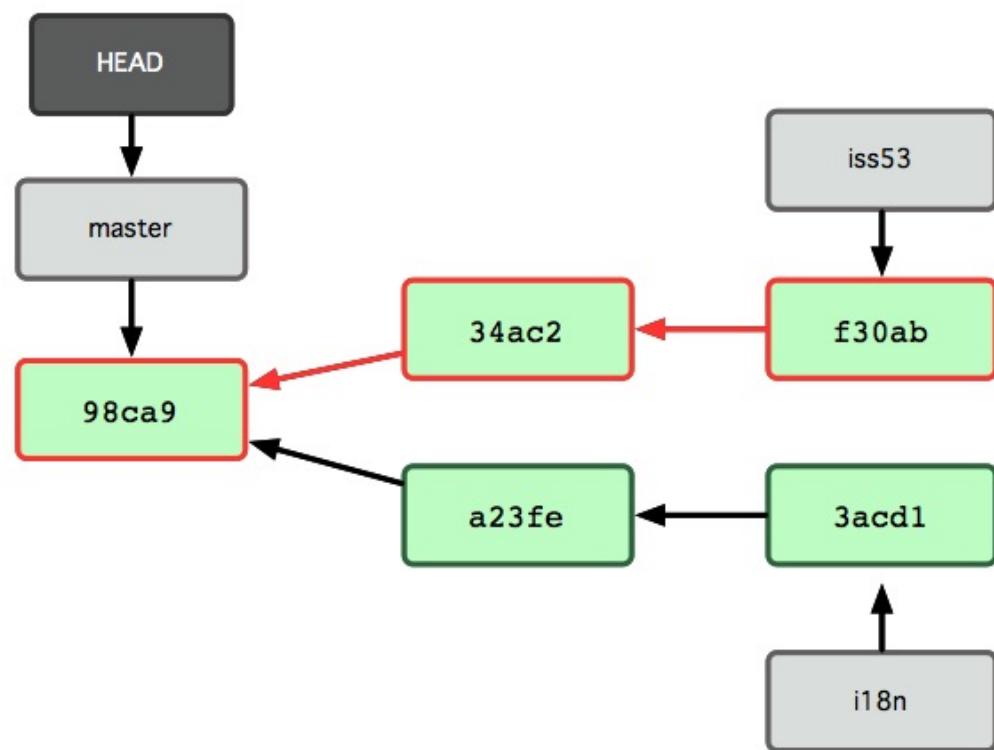


git merge iss53

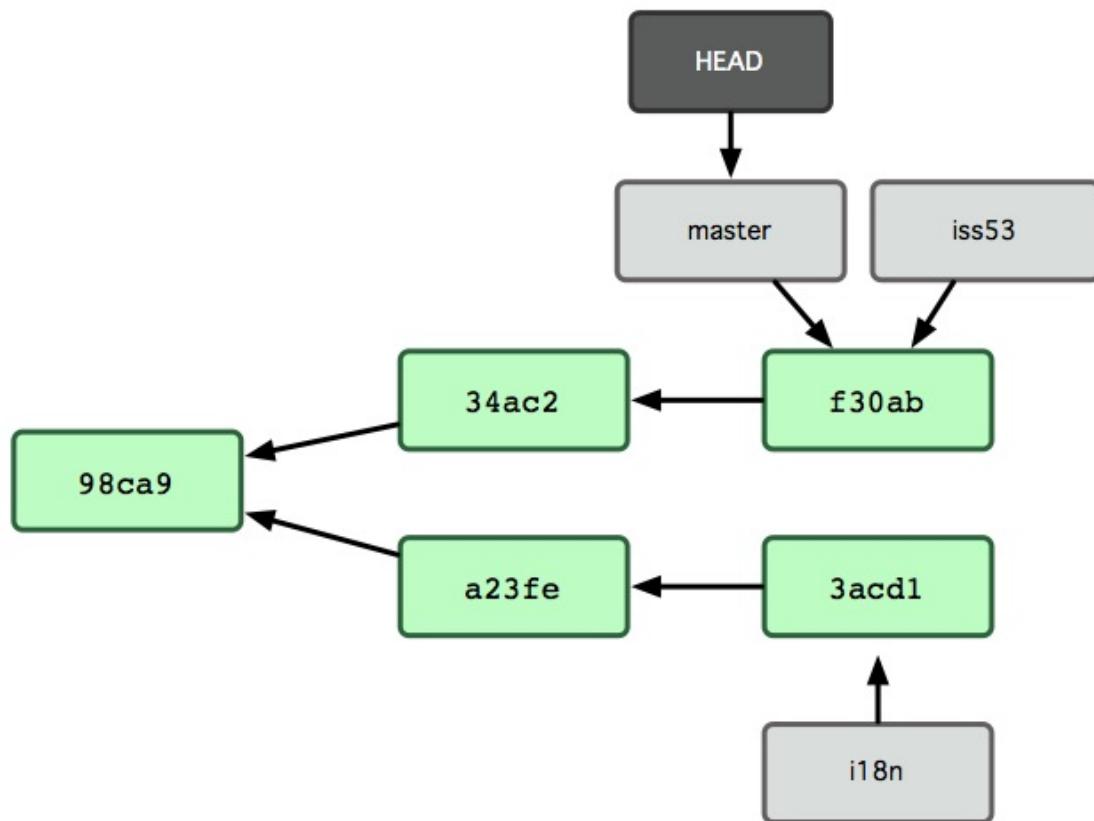


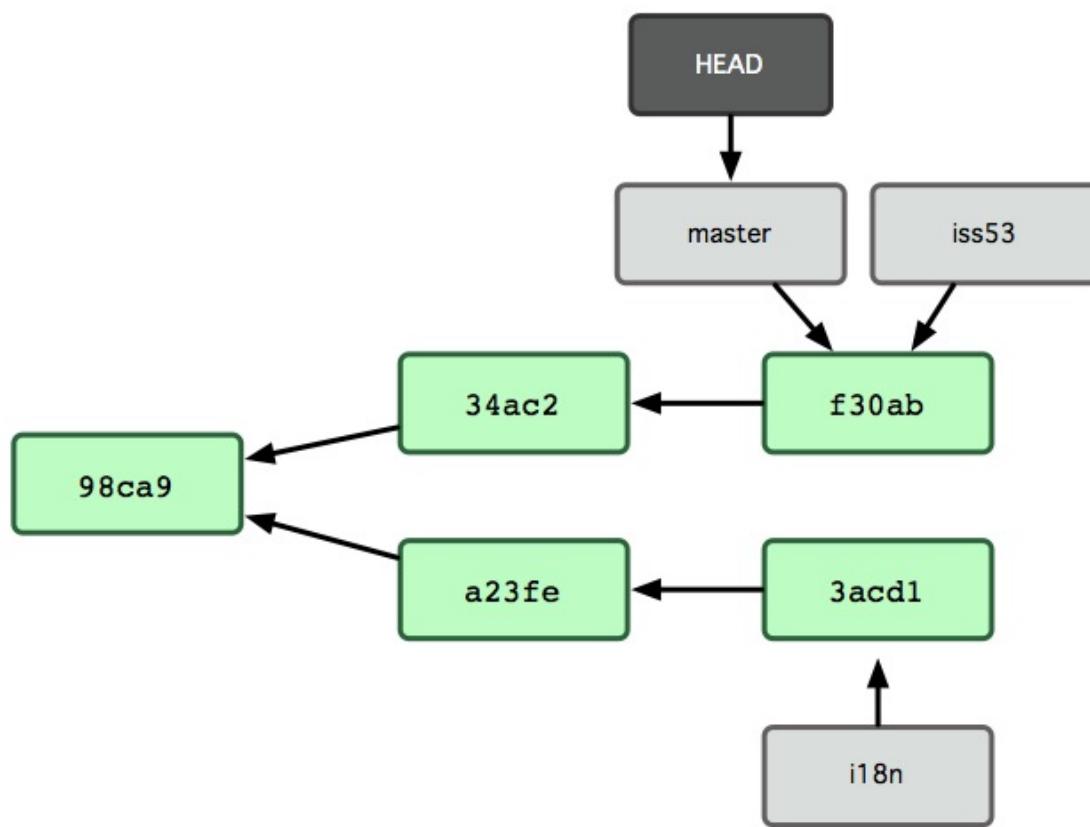
fast-forward merge

git merge iss53

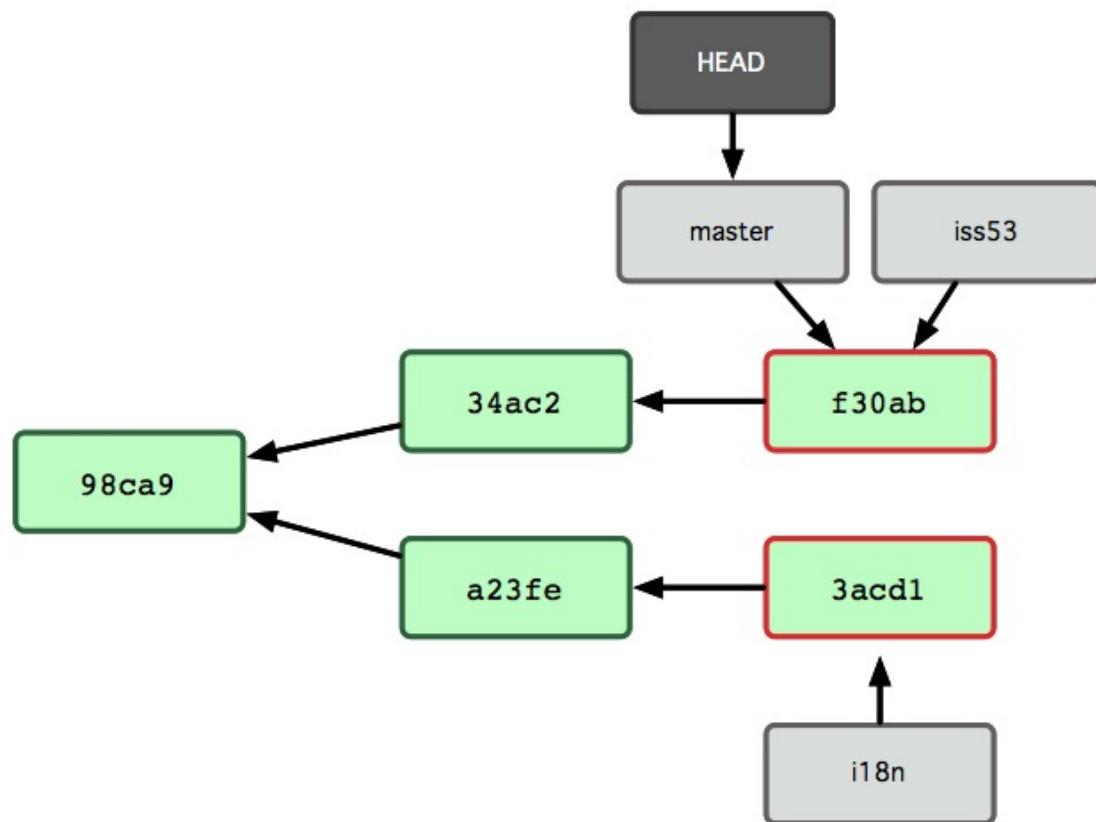


git merge iss53



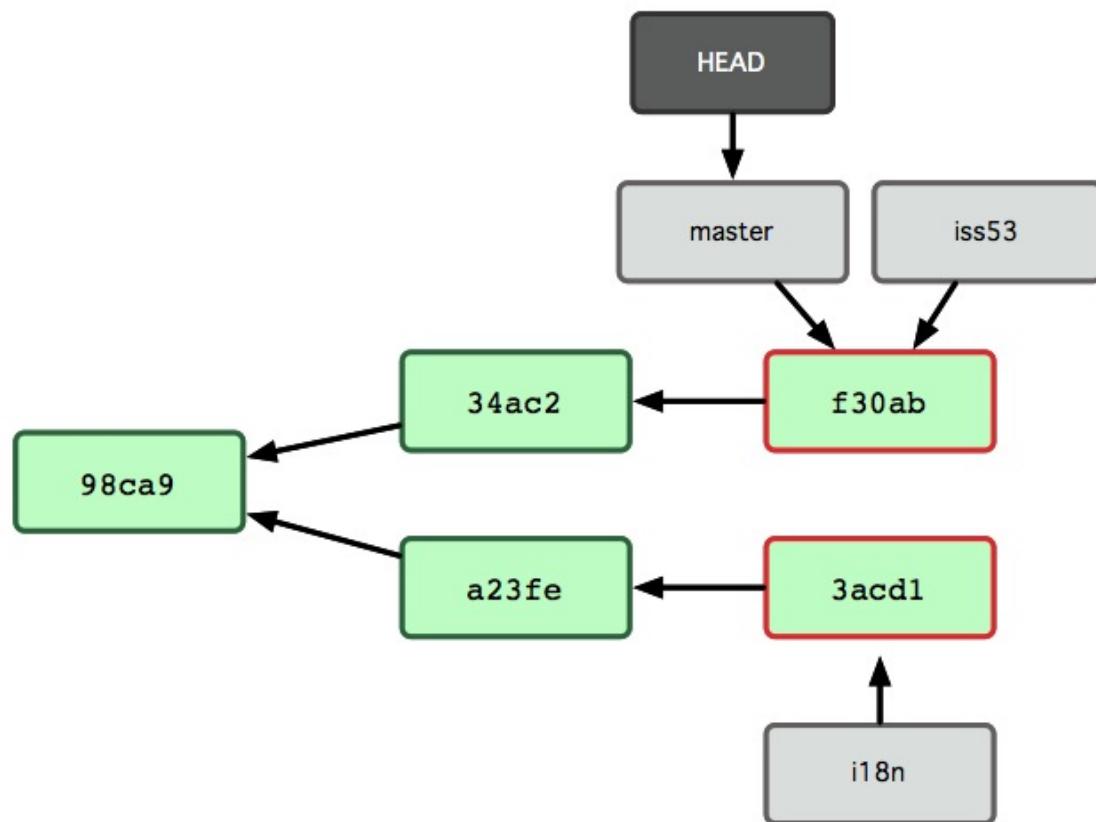


git merge i18n



non fast-forward merge

git merge i18n



--no-ff

```
$ git merge hotfix
Updating f42c576..3a0874c
Fast forward
 README |    1 -
 1 files changed, 0 insertions(+), 1 deletions(-)
```

```
$ git merge --no-ff hotfix
Merge made by recursive.
 README |    1 -
 1 files changed, 0 insertions(+), 1 deletions(-)
```

multi-head merge

```
git merge branchA branchB branchC branchD
```

```
$ git merge --no-ff topic1 topic2
Trying simple merge with topic1
Trying simple merge with topic2
Merge made by octopus.
```

```
$ git log --decorate --graph --oneline
*- . 4cfec1f (HEAD, master) Merge branches 'topic1' and 'topic2'
  \ \
  | | * 86cadbc (topic1) first topic fix
  | | * d91f9a8 first topic
  | |
  | /
  |/
* | a0fedb2 (origin/master) last commit on master
| * f659f02 (topic2) second topic work
| /
* aed1f32 base commit
```

--no-commit

```
$ git merge --no-commit topic1
Automatic merge went well; stopped before committing as requested.

$ cat .git/MERGE_HEAD
86cadbc69c20884d1fcb2494453bc44133b16d74

$ cat .git/MERGE_MSG
Merge branch 'topic1'
```

--squash

```
$ git merge --squash topic1
Squash commit -- not updating HEAD
Automatic merge went well; stopped before committing as requested
```

```
$ git commit
$ git log -1
commit 7657ee547b30f0f97519d14ba6577a999db48fd7
Author: Antonio Chacon <achacon@gmail.com>
Date:   Wed Jan 12 06:37:04 2011 -0800
```

Squashed commit of the following:

```
commit f659f023856092916a4681f7fa6d5f2a4ea246d0
Author: Scott Chacon <schacon@gmail.com>
Date:   Tue Jan 11 01:00:47 2011 -0800
```

Made everything better

Tutorial

merge the 'conflict' branch into master

view the conflict diff with `git diff`

view the conflict commits with `git log --merge`

resolve the conflict using `--thiers` or `--ours`

re-conflict the file with `checkout -m` or `--conflict`

resolve the conflict and commit

Tutorial

merge the sweet_potatoes and cookies branches in one at a time

revert the merge

merge both branches at the same time, non-fast-forward

Diff and Merge Tools

custom diff tool

```
git config --global diff.tool p4merge
```

```
$ git difftool
```

custom merge tool

ex: perform visual merge tool

```
git config --global merge.tool p4merge
```

kdiff3

tkdiff

meld

emerge

vimdiff

opendiff

```
$ git mergetool
```

non-supported merge tool

```
$ cat ~/.gitconfig
[merge]
    tool = extMerge
[mergetool "extMerge"]
    cmd = extMerge "$BASE" "$LOCAL" "$REMOTE" "$MERGED"
    trustExitCode = false
```

Submodules

Starting with Submodules

```
$ git submodule add git://github.com/rack/rack.git rack
Initialized empty Git repository in /opt/subtest/rack/.git/
remote: Counting objects: 3181, done.
remote: Compressing objects: 100% (1534/1534), done.
remote: Total 3181 (delta 1951), reused 2623 (delta 1603)
Receiving objects: 100% (3181/3181), 675.42 KiB | 422 KiB/s,
Resolving deltas: 100% (1951/1951), done.

$ git status
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       new file:   .gitmodules
#       new file:   rack
#
```

```
$ cat .gitmodules
[submodule "rack"]
    path = rack
    url = git://github.com/rack/rack.git

$ git diff --cached rack
diff --git a/rack b/rack
new file mode 160000
index 0000000..08d709f
--- /dev/null
+++ b/rack
@@ -0,0 +1 @@
+Subproject commit 08d709f78b8c5b0fbebe7821e37fa53e69afcf433
```

```
$ git commit -m 'first commit with submodule rack'  
[master 0550271] first commit with submodule rack  
2 files changed, 4 insertions(+), 0 deletions(-)  
create mode 100644 .gitmodules  
create mode 160000 rack
```

```
$ git log -1
```

```
commit 0550271328a0038865aad6331e620cd7238601bb
```

```
Author: Scott Chacon <schacon@gmail.com>
```

```
Date: Thu Apr 9 09:03:56 2009 -0700
```

```
first commit with submodule rack
```

```
$ cd rack/
```

```
$ git log -1
```

```
commit 08d709f78b8c5b0fbef7821e37fa53e69afcf433
```

```
Author: Christian Neukirchen <chneukirchen@gmail.com>
```

```
Date: Wed Mar 25 14:49:04 2009 +0100
```

```
Document version change
```

Cloning a Project with Submodules

```
$ git clone git://github.com/schacon/myproject.git
Initialized empty Git repository in /opt/myproject/.git/
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 6 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (6/6), done.
```

```
$ cd myproject
$ ls -l
total 8
-rw-r--r-- 1 schacon admin 3 Apr 9 09:11 README
drwxr-xr-x 2 schacon admin 68 Apr 9 09:11 rack
```

```
$ ls rack/
$
```

```
$ git submodule init
Submodule 'rack' (git://github.com/rack/rack.git) registered f

$ git submodule update
Initialized empty Git repository in /opt/myproject/rack/.git/
remote: Counting objects: 3181, done.
remote: Compressing objects: 100% (1534/1534), done.
remote: Total 3181 (delta 1951), reused 2623 (delta 1603)
Receiving objects: 100% (3181/3181), 675.42 KiB | 173 KiB/s,
Resolving deltas: 100% (1951/1951), done.
Submodule path 'rack': checked out '08d709f78b8c5b0fbef7821e3'
```

```
$ git fetch
# fetches commits with an update to the submodule

$ git merge origin/master
Updating 0550271..85a3eee
Fast forward
rack |    2 +- 
1 files changed, 1 insertions(+), 1 deletions(-)
[master*]$ git status
# On branch master
# Changed but not updated:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in w...
#
#       modified:   rack
#
```

```
$ git diff
diff --git a/rack b/rack
index 6c5e70b..08d709f 160000
--- a/rack
+++ b/rack
@@ -1 +1 @@
-Subproject commit 6c5e70b984a60b3cecd395edd5b48a7575bf58e0
+Subproject commit 08d709f78b8c5b0fbebe7821e37fa53e69afcf433

$ git submodule update
remote: Counting objects: 5, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 2 (delta 0)
Unpacking objects: 100% (3/3), done.
From git@github.com:schacon/rack
  08d709f..6c5e70b  master      -> origin/master
Submodule path 'rack': checked out '6c5e70b984a60b3cecd395edd5
```

Non-public Submodules

```
$ git submodule update
fatal: reference isn't a tree: 6c5e70b984a60b3cecd395edd5b48a]
Unable to checkout '6c5e70b984a60b3cecd395edd5ba7575bf58e0' in

$ git log -1 rack
commit 85a3eee996800fcfa91e2119372dd4172bf76678
Author: Scott Chacon <schacon@gmail.com>
Date:   Thu Apr 9 09:19:14 2009 -0700

    added a submodule reference I will never make public. haha
```

Issues with Submodules

submodules in branches

```
$ git checkout -b rack
Switched to a new branch "rack"

$ git submodule add git@github.com:schacon/rack.git rack
Initialized empty Git repository in /opt/myproj/rack/.git/
...
Receiving objects: 100% (3184/3184), 677.42 KiB | 34 KiB/s, do
Resolving deltas: 100% (1952/1952), done.

$ git commit -am 'added rack submodule'
[rack cc49a69] added rack submodule
 2 files changed, 4 insertions(+), 0 deletions(-)
 create mode 100644 .gitmodules
 create mode 160000 rack

$ git checkout master
Switched to branch "master"

$ git status
# On branch master
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       rack/
```

changing from a directory to a
submodule

```
$ rm -Rf rack/
$ git submodule add git@github.com:schacon/rack.git rack
'rack' already exists in the index

$ git rm -r rack

$ git submodule add git@github.com:schacon/rack.git rack
Initialized empty Git repository in /opt/testsub/rack/.git/
remote: Counting objects: 3184, done.
remote: Compressing objects: 100% (1465/1465), done.
remote: Total 3184 (delta 1952), reused 2770 (delta 1675)
Receiving objects: 100% (3184/3184), 677.42 KiB | 88 KiB/s, do
Resolving deltas: 100% (1952/1952), done.

$ git checkout master
error: Untracked working tree file 'rack/AUTHORS' would be ove

$ mv rack /tmp/
$ git checkout master
Switched to branch "master"

$ ls
README  rack
```

Subtree Merging

```
$ git remote add rack_remote git@github.com:schacon/rack.git  
  
$ git fetch rack_remote  
warning: no common commits  
remote: Counting objects: 3184, done.  
remote: Compressing objects: 100% (1465/1465), done.  
remote: Total 3184 (delta 1952), reused 2770 (delta 1675)  
Receiving objects: 100% (3184/3184), 677.42 KiB | 4 KiB/s, done.  
Resolving deltas: 100% (1952/1952), done.  
From git@github.com:schacon/rack  
 * [new branch]      build      -> rack_remote/build  
 * [new branch]      master     -> rack_remote/master  
 * [new branch]      rack-0.4   -> rack_remote/rack-0.4  
 * [new branch]      rack-0.9   -> rack_remote/rack-0.9  
  
$ git checkout -b rack_branch rack_remote/master  
Branch rack_branch set up to track remote branch refs/remotes/  
Switched to a new branch "rack_branch"
```

```
$ ls
AUTHORS           KNOWN-ISSUES      Rakefile        contrib      lik
COPYING          README            bin             example     tes
```

```
$ git checkout master
Switched to branch "master"
```

```
$ ls
README
```

```
$ git read-tree --prefix=rack/ -u rack_branch  
$ git checkout rack_branch  
$ git pull  
$ git checkout master  
$ git merge --squash -s subtree --no-commit rack_branch  
Squash commit -- not updating HEAD  
Automatic merge went well; stopped before committing as requeste
```

Tracking Branches

```
$ git fetch
remote: Counting objects: 535, done.
remote: Compressing objects: 100% (154/154),
remote: Total 426 (delta 317), reused 369 (de
Receiving objects: 100% (426/426), 52.49 KiB,
Resolving deltas: 100% (317/317), completed w
From github.com:github/github
 * [new branch]      frotz    -> origin/frotz
 9893ff7..1549652  master   -> origin/master
```

```
git checkout origin/frotz          -> detaches HEAD  
git checkout -b frotz origin/frotz -> tracking  
git checkout -t origin/frotz      -> tracking  
git checkout frotz                -> tracking  
git checkout -t frotz             -> fatal: Missing branch name
```

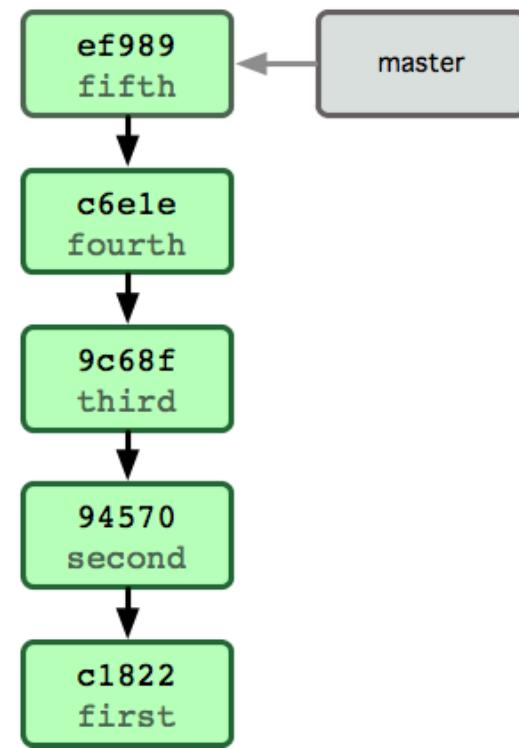
```
$ git push -u origin frotz  
# pushes the "frotz" branch to "origin"  
remote and sets up tracking
```

push.default

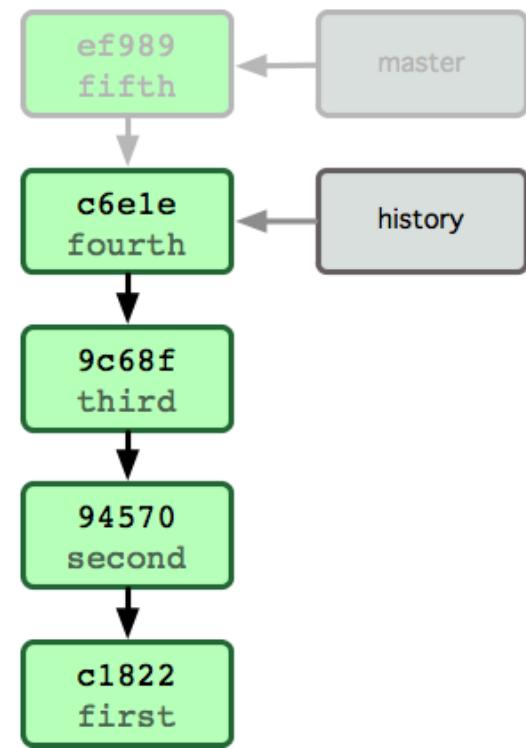
```
$ git config --global push.default matching  
$ git config --global push.default nothing  
$ git config --global push.default tracking  
$ git config --global push.default current
```

Replacements and Grafting

```
$ git log --oneline  
ef989d8 fifth commit  
c6e1e95 fourth commit  
9c68fdc third commit  
945704c second commit  
c1822cf first commit
```

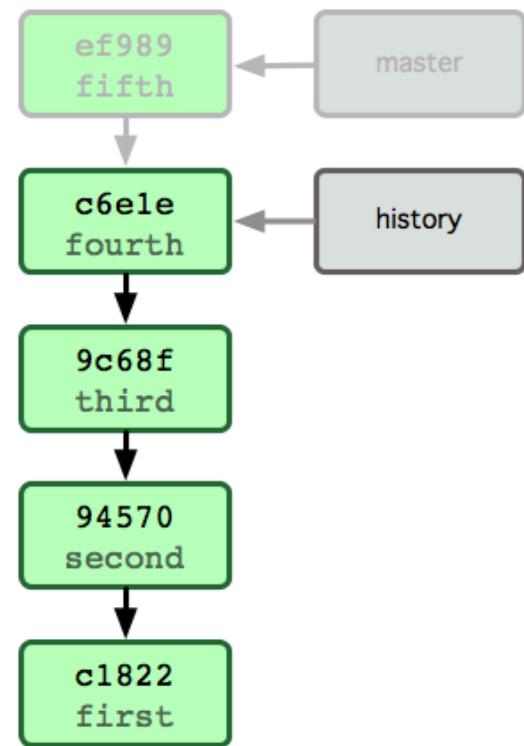


```
$ git branch history master~1
$ git log --oneline --decorate
ef989d8 (HEAD, master) fifth commit
c6e1e95 (history) fourth commit
9c68fdc third commit
945704c second commit
c1822cf first commit
```

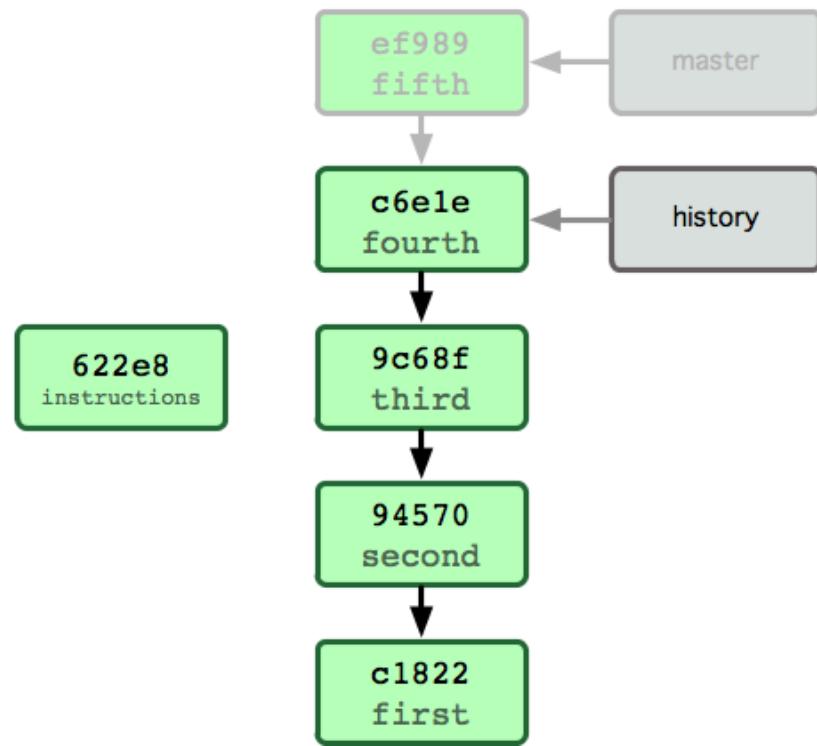


```
$ git remote add history git@github.com:schacon/project-history
$ git push history history:master
Counting objects: 12, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (12/12), 907 bytes, done.
Total 12 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (12/12), done.
To git@github.com:schacon/project-history.git
 * [new branch]      history -> master

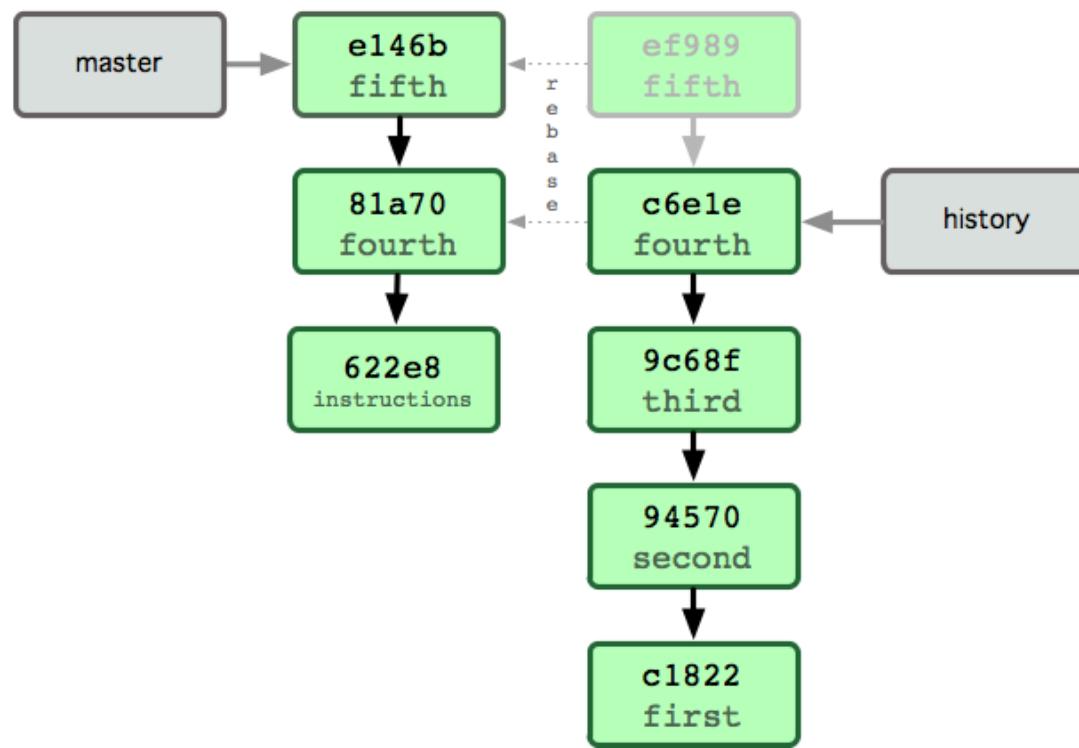
$ git log --oneline --decorate
ef989d8 (HEAD, master) fifth commit
c6e1e95 (history) fourth commit
9c68fdc third commit
945704c second commit
c1822cf first commit
```



```
$ echo 'get history from blah blah blah' > msg
$ cat msg | git commit-tree history~1^{tree}
622e88e9cbfbacfb75b5279245b9fb38dfea10cf
```



```
$ git rebase --onto 622e88 9c68fdc
First, rewinding head to replay your work on top of it...
Applying: fourth commit
Applying: fifth commit
```



```
$ git clone git://github.com/schacon/project.git
$ cd project

$ git log --oneline master
e146b5f fifth commit
81a708d fourth commit
622e88e get history from blah blah blah

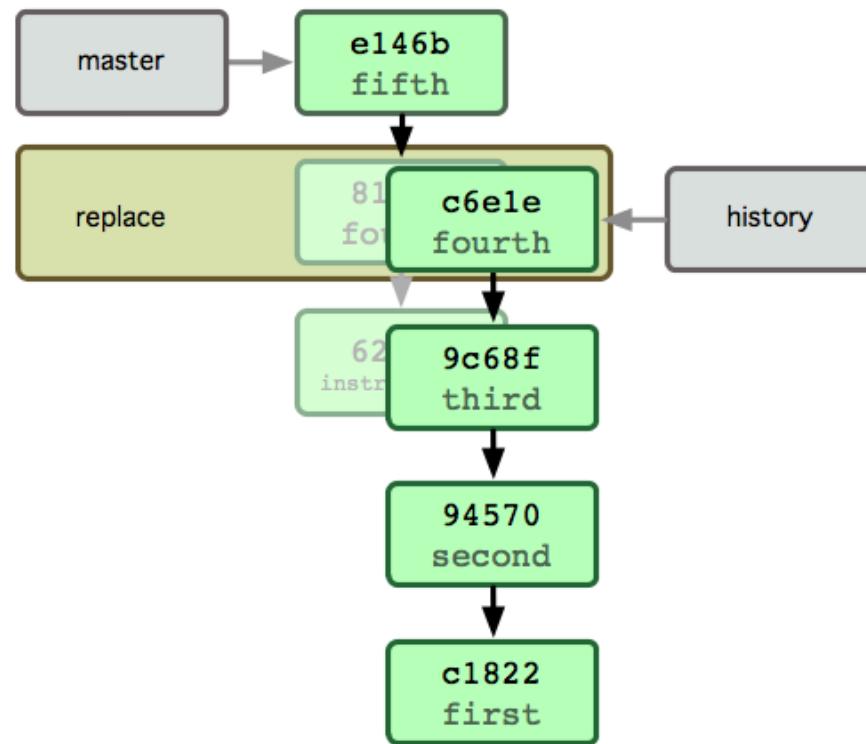
$ git remote add history git://github.com/schacon/project-history
$ git fetch history
From git://github.com/schacon/project-history.git
 * [new branch]          master      -> history/master

$ git log --oneline history/master
c6e1e95 fourth commit
9c68fdc third commit
945704c second commit
c1822cf first commit
```

```
$ git log --oneline  
e146b5f fifth commit  
81a708d fourth commit  
622e88e get history from blah blah blah
```

```
$ git replace 81a708d c6e1e95
```

```
$ git log --oneline  
e146b5f fifth commit  
81a708d fourth commit  
9c68fdc third commit  
945704c second commit  
c1822cf first commit
```



```
$ git cat-file -p 81a708d
tree 7bc544cf438903b65ca9104a1e30345eee6c083d
parent 9c68fdceee073230f19ebb8b5e7fc71b479c0252
author Scott Chacon <schacon@gmail.com> 1268712581 -0700
committer Scott Chacon <schacon@gmail.com> 1268712581 -0700
```

fourth commit

```
$ git --no-replace-references cat-file -p 81a708d
tree 7bc544cf438903b65ca9104a1e30345eee6c083d
parent 622e88e9cbfbacfb75b5279245b9fb38dfa10cf
author Scott Chacon <schacon@gmail.com> 1268712581 -0700
committer Scott Chacon <schacon@gmail.com> 1268712581 -0700
```

fourth commit

```
$ git for-each-ref  
e146b5f14e79d49351 commit refs/heads/master  
c6e1e95051d41771a6 commit refs/remotes/history/master  
e146b5f14e79d49351 commit refs/remotes/origin/HEAD  
e146b5f14e79d49351 commit refs/remotes/origin/master  
c6e1e95051d41771a6 commit refs/replace/81a708dd0e167a3f69154
```

Grafts

.git/info/grafts

```
$ echo "84adef33143d 9c6fdd21e4e65" > .git/info/grafts
```

git diff grafts replace

easier to share (refs)

easier to ignore

any object

Erlang

<http://github.com/erlang/otp/wiki/Extending-the-history-of-Erlang-OTP>

Tutorial

clone `git://github.com/ghtraining/re-small.git` and
follow the instructions to graft the history back
on

re-graft the history a few commits higher

Reuse Recorded Resolution

rerere

```
git config --global rerere.enabled 1
```

Original File

```
#! /usr/bin/env ruby  
def hello  
  puts 'hello world'  
end
```

in master branch

```
#! /usr/bin/env ruby
```

```
def hello  
  puts 'hello mundo'  
end
```

in i18n-world branch

```
#! /usr/bin/env ruby
```

```
def hello  
  puts 'hola world'  
end
```

```
$ git merge i18n-world
Auto-merging hello.rb
CONFLICT (content): Merge conflict in hello.rb
Recorded preimage for 'hello.rb'
Automatic merge failed; fix conflicts and then commit the resu
```

```
$ git rerere status
hello.rb
```

```
$ # edit hello.rb to be 'hola mundo'  
$ git add hello.rb
```

```
$ git rerere diff
--- a/hello.rb
+++ b/hello.rb
@@ -1,11 +1,7 @@
#! /usr/bin/env ruby

def hello
-
-  puts 'hello mundo'
=====
-  puts 'hola world'
->>>>>
+  puts 'hola mundo'
end
```

```
$ git commit  
Recorded resolution for 'hello.rb'.  
[master 68e16e5] Merge branch 'i18n'
```

```
$ git reset --hard HEAD^
HEAD is now at ad63f15 i18n the hello

$ git checkout i18n-world
Switched to branch 'i18n-world'

$ git rebase master
First, rewinding head to replay your work on top of it...
Applying: i18n one word
Using index info to reconstruct a base tree...
Falling back to patching base and 3-way merge...
Auto-merging hello.rb
CONFLICT (content): Merge conflict in hello.rb
Resolved 'hello.rb' using previous resolution.
Failed to merge in the changes.
Patch failed at 0001 i18n one word
```

```
$ cat hello.rb
#!/usr/bin/env ruby

def hello
    puts 'hola mundo'
end

$ git diff
diff --cc hello.rb
index a440db6,54336ba..0000000
--- a/hello.rb
+++ b/hello.rb
@@@ -1,7 -1,7 +1,7 @@@
  #! /usr/bin/env ruby

      def hello
-     puts 'hola world'
-     puts 'hello mundo'
++     puts 'hola mundo'
    end
```

```
$ git checkout --conflict=merge hello.rb
$ cat hello.rb
#!/usr/bin/env ruby

def hello
<<<<< ours
  puts 'hola mundo'
=====
  puts 'hello mundo'
>>>>> theirs
end

$ git rerere
Resolved 'hello.rb' using previous resolution.
$ cat hello.rb
#!/usr/bin/env ruby

def hello
  puts 'hola mundo'
end

$ git add hello.rb
$ git rebase --continue
Applying: i18n one word
```

```
git merge --rerere-autoupdate
```

```
git config --global rerere.autoupdate 1
```

Debugging

git blame

ie: "what dumbass did this? oh yeah, it was me..."

```
$ git blame daemon.c
979e32fa (Randal L. Schwartz) 2005-10-25 16:29:09 -0700 1) #include "
85023577 (Junio C Hamano) 2006-12-19 14:34:12 -0800 2) #include "
77cb17e9 (Michal Ostrowski) 2006-01-10 21:12:17 -0500 3) #include "
49ba83fb (Jon Loeliger) 2006-09-19 20:31:51 -0500 4) #include "
f8ff0c06 (Petr Baudis) 2005-09-22 11:25:28 +0200 5)
85023577 (Junio C Hamano) 2006-12-19 14:34:12 -0800 6) #include <
85023577 (Junio C Hamano) 2006-12-19 14:34:12 -0800 7)
695dffe2 (Johannes Schindelin) 2006-09-28 12:00:35 +0200 8) #ifndef HC
695dffe2 (Johannes Schindelin) 2006-09-28 12:00:35 +0200 9) #define HC
695dffe2 (Johannes Schindelin) 2006-09-28 12:00:35 +0200 10) #endif
695dffe2 (Johannes Schindelin) 2006-09-28 12:00:35 +0200 11)
415e7b87 (Patrick Welche) 2007-10-18 18:17:39 +0100 12) #ifndef NI
415e7b87 (Patrick Welche) 2007-10-18 18:17:39 +0100 13) #define NI
415e7b87 (Patrick Welche) 2007-10-18 18:17:39 +0100 14) #endif
415e7b87 (Patrick Welche) 2007-10-18 18:17:39 +0100 15)
9048fe1c (Petr Baudis) 2005-09-24 16:13:01 +0200 16) static int
f8ff0c06 (Petr Baudis) 2005-09-22 11:25:28 +0200 17) static int
1955fabf (Mark Wooding) 2006-02-03 20:27:04 +0000 18) static int
f8ff0c06 (Petr Baudis) 2005-09-22 11:25:28 +0200 19)
960deccb (H. Peter Anvin) 2005-10-19 14:27:01 -0700 20) static con
1b1dd23f (Stephan Beyer) 2008-07-13 15:36:15 +0200 21) "git daem
3bd62c21 (Stephen R. van den Berg) 2008-08-14 20:02:20 +0200 22) "
3bd62c21 (Stephen R. van den Berg) 2008-08-14 20:02:20 +0200 23) "
73a7a656 (Jens Axboe) 2007-07-27 14:00:29 -0700 24) "
49ba83fb (Jon Loeliger) 2006-09-19 20:31:51 -0500 25) "
678dac6b (Tilman Sauerbeck) 2006-08-22 19:37:41 +0200 26) "
d9edcbd6 (Junio C Hamano) 2006-09-07 01:40:04 -0700 27) "
dd467629 (Jon Loeliger) 2006-09-26 09:47:43 -0500 28) "
```

```
$ git blame -C GITPackUpload.m
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 12)
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 13)
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 14)
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 15)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 16)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 17)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 18)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 19)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 20)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 21)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 22)
a2cbabf5 Network/GITPackUpload.m (Scott Chacon 2009-03-25 22:29:39 23)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 24)
a2cbabf5 Network/GITPackUpload.m (Scott Chacon 2009-03-25 22:29:39 25)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 26)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 27)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 28)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 29)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 30)
ad11ac80 Network/GITPackUpload.m (Scott Chacon 2009-03-24 18:32:50 31)
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 32)
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 33)
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 34)
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 35)
f344f58d Network/GITServerHandler.m (Scott Chacon 2009-01-04 18:59:04 36)
56ef2caf Network/GITServerHandler.m (Scott Chacon 2009-01-05 21:44:26 37)
```

```
$ git blame -L23,29 Source/Network/GITPackUpload.m a2cbabf5
a2cbabf5 (Scott Chacon 2009-03-25 22:29:39 +0100 23) - (id) initWithGit:(GIT
ad11ac80 (Scott Chacon 2009-03-24 18:32:50 +0100 24) {
a2cbabf5 (Scott Chacon 2009-03-25 22:29:39 +0100 25)     gitRepo = gRepo;
ad11ac80 (Scott Chacon 2009-03-24 18:32:50 +0100 26)     needRefs = nRefs;
ad11ac80 (Scott Chacon 2009-03-24 18:32:50 +0100 27)     gitSocket = gSocket;
ad11ac80 (Scott Chacon 2009-03-24 18:32:50 +0100 28)     return self;
ad11ac80 (Scott Chacon 2009-03-24 18:32:50 +0100 29) }
```

Bisecting

**binary search for where a bug
was introduced**

```
$ git bisect start
$ git bisect bad
$ git bisect good v1.0
Bisection: 9 revisions left to test after this (roughly 3 steps)
[1fe13deb3187e193b9ec9ba6d70ac58934c6aa32] Adding routes to he
$ git bisect bad
Bisection: 4 revisions left to test after this (roughly 2 steps)
[93f150ab7e6bda806148205dab5a50cca16ef7a3] Move bounces to fix
$ git bisect good
Bisection: 2 revisions left to test after this (roughly 1 step)
[ac7c7346526f7b1be68775d2f7620f726813e229] Fix 404 pages. Clos
$ git bisect bad
Bisection: 0 revisions left to test after this (roughly 0 steps)
[c4131e24d1f97b8ff105d184d5cf4a954c1c70ec] fix org tests
$ git bisect bad
c4131e24d1f97b8ff105d184d5cf4a954c1c70ec is the first bad commit
commit c4131e24d1f97b8ff105d184d5cf4a954c1c70ec
Author: rick <technoweenie@gmail.com>
Date:   Thu Oct 14 13:09:10 2010 -0700

    fix org tests

:040000 040000 05e813cf1a32269ed023982dd9bfe433934a34b6 187593
```

Tutorial

find the user who last edited line 5 of
mexican_food_ingredients.txt

find the user who edited that line before him

find the file that line originally came from

find the first commit that broke the raisins.sh
script (it works in 5a1aae2)

Scripting

Environment Variables

moving around your git pieces

git directory

index file

working directory

GIT_DIR

```
$ mv .git /opt/repo.git  
$ git --git-dir=/opt/repo.git log  
$ export GIT_DIR=/opt/repo.git  
$ git log
```

GIT_INDEX_FILE

```
$ git status -s
M README
M kidgloves.rb

$ git add kidgloves.rb

$ git status -s
M README
S kidgloves.rb
```

```
$ export GIT_INDEX_FILE=/tmp/index
$ git read-tree HEAD
$ git add README

$ git status -s
S  README
M  kidgloves.rb
```

```
$ unset GIT_INDEX_FILE  
  
$ git status -s  
 M README  
S  kidgloves.rb  
  
$ export GIT_INDEX_FILE=/tmp/index  
  
$ git status -s  
S  README  
M  kidgloves.rb
```

GIT_WORK_TREE

```
$ git status -s
M README
S kidgloves.rb

$ export GIT_DIR=$(pwd) /.git
$ export GIT_WORK_TREE=$(pwd)

$ cd /tmp
$ git status -s
M README
S kidgloves.rb
```

Overwriting your credentials

GIT_AUTHOR_NAME

GIT_AUTHOR_EMAIL

Debugging Git

GIT_MERGE_VERBOSITY

```
$ git merge 06ca03a
Auto-merging kidgloves.rb
Merge made by recursive.
  kidgloves.rb |    3 +--
  1 files changed, 1 insertions(+), 2 deletions(-)

$ git reset --hard 096880d
HEAD is now at 096880d use idiomatic rack :Host and :Port opti

$ GIT_MERGE_VERBOSITY=5 git merge 06ca03a
Merging:
  096880d use idiomatic rack :Host and :Port options
  virtual 06ca03a
  found 1 common ancestor(s):
  66d35ee wasnt setting the header vars properly
  Auto-merging kidgloves.rb
  Merge made by recursive.
  kidgloves.rb |    3 +--
  1 files changed, 1 insertions(+), 2 deletions(-)
```

GIT_TRACE

```
$ GIT_TRACE=1 git lol
trace: exec: 'git-lol'
trace: run_command: 'git-lol'
trace: alias expansion: lol => 'log' '--oneline' '--graph' '--'
trace: run_command: 'less'
trace: exec: 'less'
trace: built-in: git 'log' '--oneline' '--graph' '--decorate'
*   b896af7 (HEAD, test) Merge commit '06ca03a' into test
| \
| * 06ca03a (rtomayko/headernames) no need to delete these hea
| * 8cebd9f CONTENT_TYPE and CONTENT_LENGTH are not prefixed w
| * 8aff92c request_header env keys get HTTP_ prefix
* | 096880d (rtomayko/bindopts) use idiomatic rack :Host and :
* | 801e40a log to stderr instead of stdout
* | 7295993 take listen host/port
| /
```

GIT_CURL_VERBOSE

```
$ GIT_CURL_VERBOSE=1 git clone http://github.com/schacon/showoff
Cloning into showoff...
* Couldn't find host github.com in the .netrc file; using defa
* About to connect() to github.com port 80 (#0)
* Trying 207.97.227.239... * Connected to github.com (207.97.227.239) port 80 (#0)
> GET /schacon/showoff.git/info/refs?service=git-upload-pack HTTP/1.1
User-Agent: git/1.7.2.1
Host: github.com
Accept: */*
Pragma: no-cache

< HTTP/1.1 200 OK
< Server: nginx/0.7.67
< Date: Wed, 11 Aug 2010 20:21:35 GMT
< Content-Type: application/x-git-upload-pack-advertisement
< Connection: keep-alive
< Status: 200 OK
< Pragma: no-cache
< Content-Length: 1316
< Expires: Fri, 01 Jan 1980 00:00:00 GMT
< Cache-Control: no-cache, max-age=0, must-revalidate
<
* Expire cleared
* Connection #0 to host github.com left intact
* Couldn't find host github.com in the .netrc file; using defa
* About to connect() to github.com port 80 (#0)
* Trying 207.97.227.239... * connected
* Connected to github.com (207.97.227.239) port 80 (#0)
> POST /schacon/showoff.git/info/refs?service=git-upload-pack HTTP/1.1
```

Rabbit

Ruby + Bindings + Git == Ribbit

libgit2 - GSoC

libgit2.github.com

Reclist / Log

```
@ribbit = Ribbit::Repository.new(path)
@walker = Ribbit::Walker.new(@ribbit)

def rabbit_log(walker, sha)
  walker.push(sha)
  data = []
  while sha = walker.next do
    data << sha
  end
  data
end

rabbit_log(@walker, head)
```

Rev List - 16,000 Commits | x

Grit : 1.067s

Rabbit: 1.053s

Rev List - 210 Commits 200x

Grit : 3.370s

Rabbit: 0.081s

40x

Rev List - 15 Commits x 200

Grit : 2.7126s

Rabbit: 0.0067s

450x

Revision Walker API

```
walker = Rabbit::Walker.new(repo)

walker.push(hex_sha_interesting)
walker.hide(hex_sha_uninteresting)
hex_sha = walker.next # false if none left
walker.reset
```

Small Blob Read

```
@ribbit = Rabbit::Repository.new(path)
o = @ribbit.lookup(blob_sha)
puts o.sha
puts o.data
```

```
>> c = @ribbit.lookup(commit_sha)

>> puts c.message
"my commit message\n"

>> pp c.author
{ "name"=>"Scott Chacon", "time"=>1273360386,
  "email"=>"schacon@gmail.com"}
```

Small Blob Read

1k file x 10000

Grit : 0.96461s
Rabbit: 0.27974s

4x

Object Exists

```
@rabit = Rabbit::Repository.new(path)
@rabit.exists(sha)
```

Object Exists (50x)

Grit (loose) : 23.263354

Grit (packd) : 0.388390

Rabbit (loose) : 0.036332

Rabbit (packd) : 0.030283

10x to 1000x

Low-Level Object API

```
repo = Rabbit::Repository.new(path)

          bool = repo.exists(hex_sha)
data, length, type = repo.read(hex_sha) # or false
          hex_sha = repo.hash(content, type)
          hex_sha = repo.write("my content\n", "blob")
```

Other Languages

Geef - Erlang Git NIF

<http://github.com/schacon/geef>

```
geef:start().  
Raw = geef:hex_to_raw(HexSha).  
Exists = geef:object_exists(ObjectsPath, HexSha).
```

Customizing Git

Autocorrect

```
$ git com
git: 'com' is not a git-command. See 'git --help'.

Did you mean this?
  commit
```

help.autocorrect

```
git config --global help.autocorrect 1
```

```
$ git com
```

```
WARNING: You called a Git program named 'com', which does not  
Continuing under the assumption that you meant 'commit'
```

Colors

```
git config --global color.ui true
```

Git Attributes

.gitattributes

Diff Binary Files

```
$ git diff
diff --git a/image.png b/image.png
index 88839c4..4afcb7c 100644
Binary files a/image.png and b/image.png differ
```

tell Git how to diff a binary file

exiftool


```
$ echo '*.png diff=exif' >> .gitattributes
# take every file that ends in png
# and pre-process them with a strategy called 'exif'

$ git config diff.exif.textconv exiftool
# the 'exif' strategy is to run exiftool on the file
```

```
$ git diff
diff --git a/image.png b/image.png
index 88839c4..4afcb7c 100644
--- a/image.png
+++ b/image.png
@@ -1,12 +1,12 @@
    ExifTool Version Number          : 7.74
-File Size                         : 70 kB
-File Modification Date/Time       : 2009:04:21 07:02:45-07:
+File Size                         : 94 kB
+File Modification Date/Time       : 2009:04:21 07:02:43-07:
    File Type                        : PNG
    MIME Type                         : image/png
-Image Width                        : 1058
-Image Height                       : 889
+Image Width                        : 1056
+Image Height                       : 827
    Bit Depth                         : 8
    Color Type                        : RGB with Alpha
```

Filtering

Staging Area

fileA.txt

fileB.txt

fileC.rb

Staging Area

fileA.txt

fileB.txt

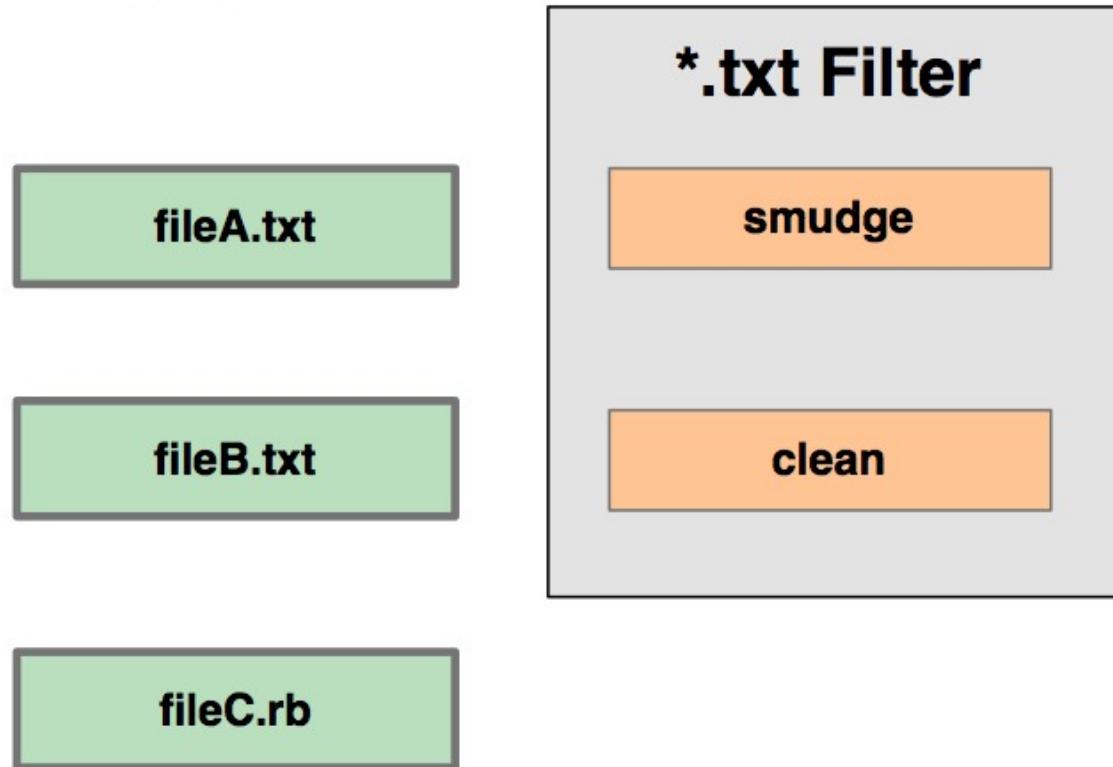
fileC.rb

***.txt Filter**

smudge

clean

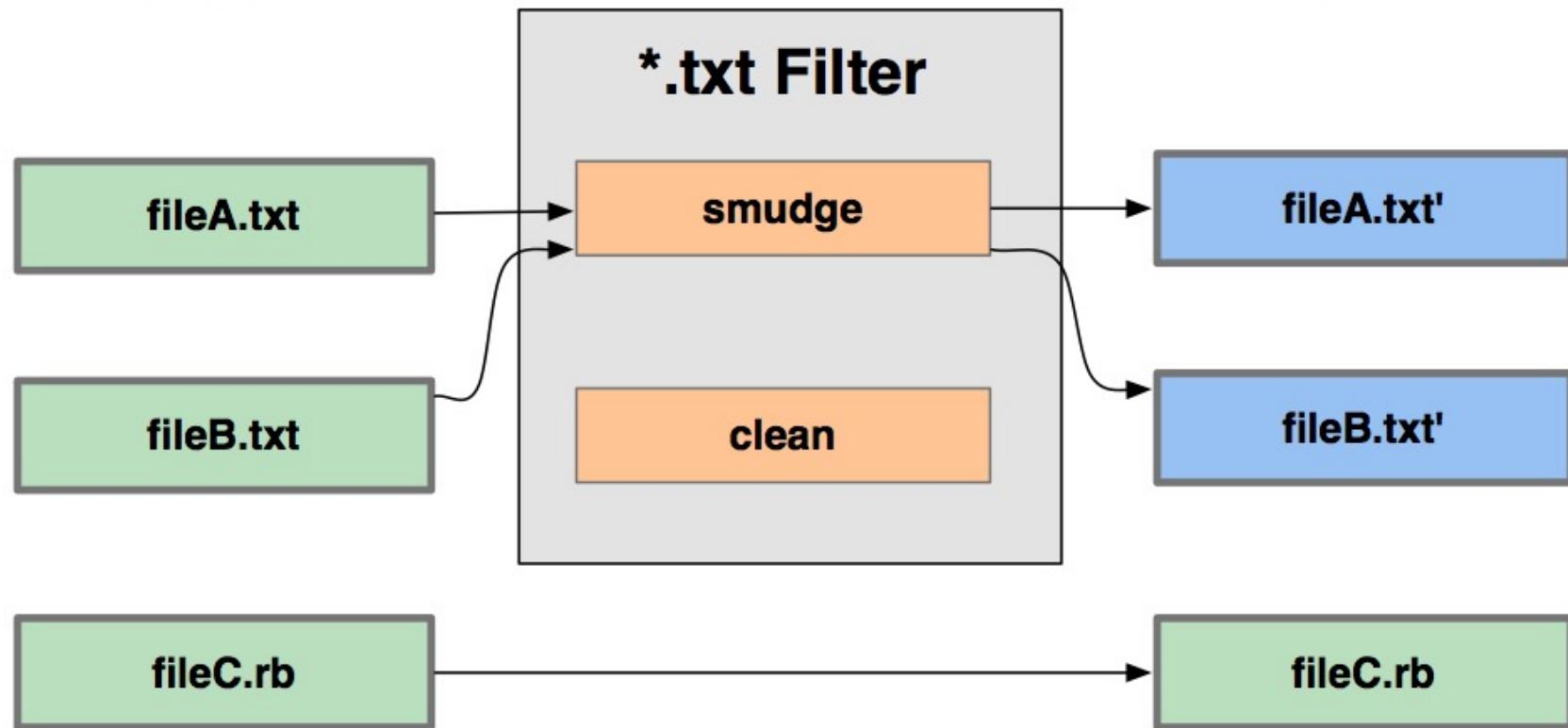
Staging Area



git checkout

Staging Area

Working Directory



git checkout

Staging Area

fileA.txt

fileB.txt

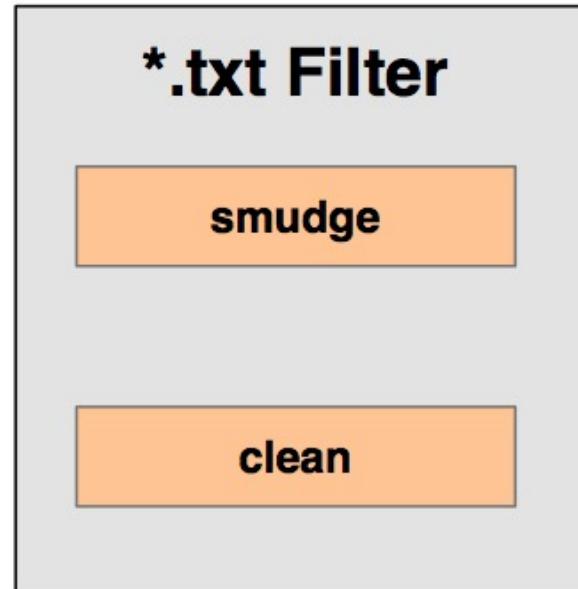
fileC.rb

Working Directory

fileA.txt'

fileB.txt'

fileC.rb



Staging Area

fileA.txt

fileB.txt

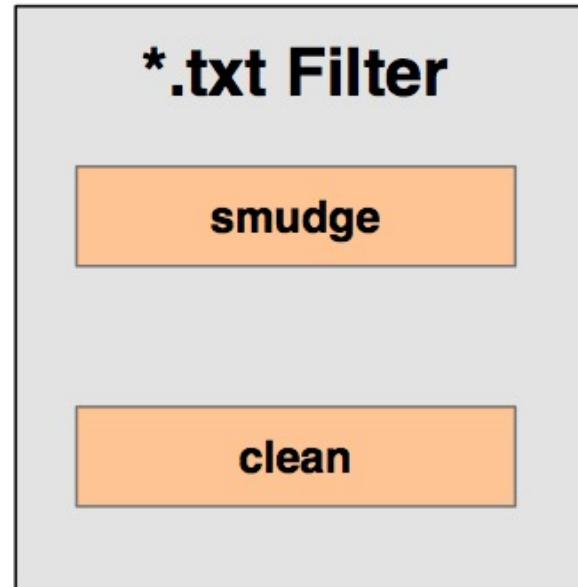
fileC.rb

Working Directory

fileA.txt'

fileB.txt'

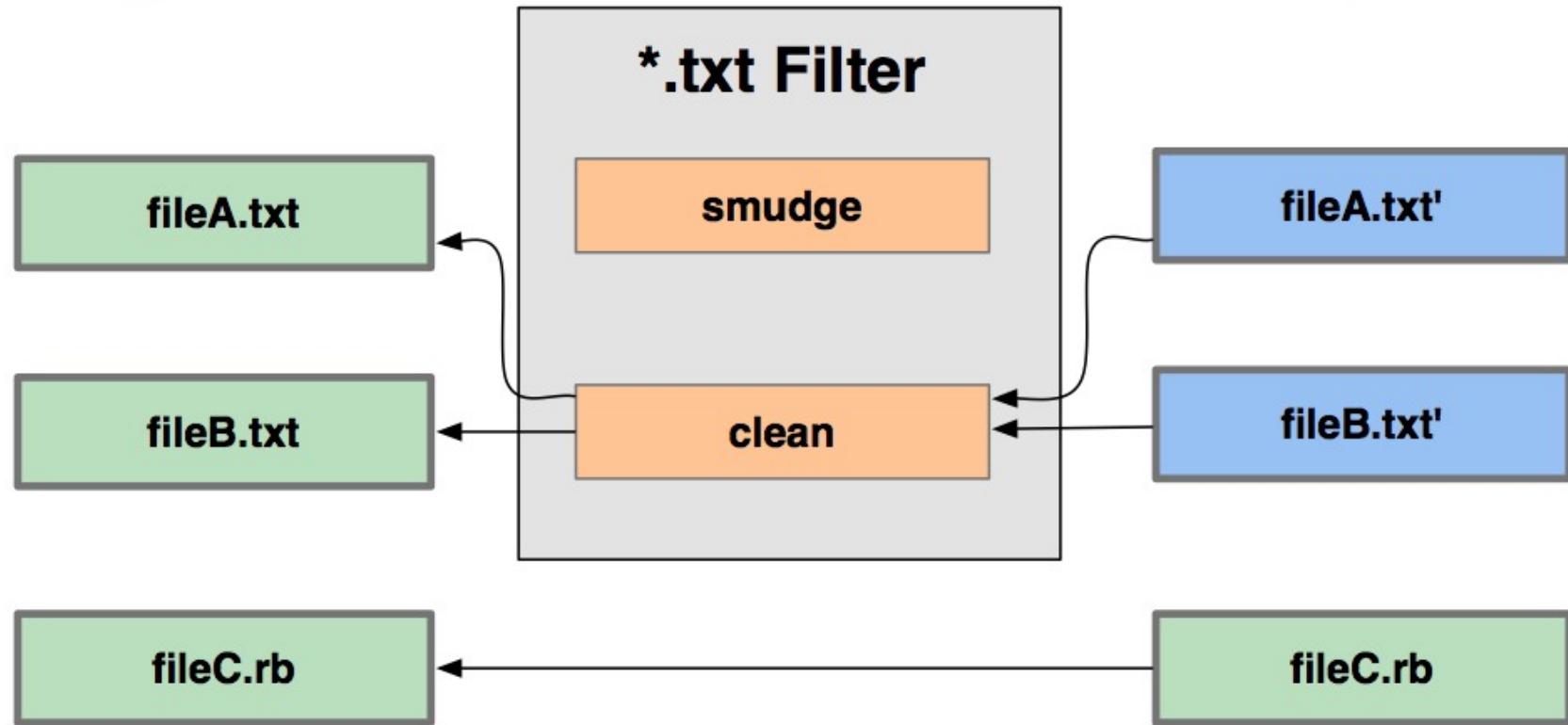
fileC.rb



git commit

Staging Area

Working Directory



git commit

expanding a \$Date\$

/usr/bin/expand_date

```
#! /usr/bin/env ruby
data = STDIN.read
date = `git log --pretty=format:"%ad" -1`
puts data.gsub('$Date$',
               '$Date: ' + date + '$')
```

```
$ git config filter.dater.smudge expand_date  
$ git config filter.dater.clean \  
'perl -pe "s/\\\\$Date[^\\\\$]*\\\$//\\\$Date\\\$/"'
```

test it

```
$ echo '# $Date$' > date_test.rb
$ echo 'date*.rb filter=dater' >> .gitattributes
$ git add date_test.rb .gitattributes
$ git commit -m "Testing date expansion in Git"
$ rm date_test.rb
$ git checkout date_test.rb

$ cat date_test.rb
# $Date: Tue Apr 21 07:26:52 2009 -0700$
```

Aliases

```
$ git config --global alias.lol \
    "log --oneline --decorate --graph"
$ git lol
* 6b490d2 (HEAD, locale) updated almost all of the explore are
*   2a062e7 (origin/locale) Merge remote branch 'origin/master'
| \
| * 29ffbab (origin/master, origin/HEAD) Merge branch 'master'
| |
| * e3f4eaf Merge branch 'master' of github.com:defunkt/gi
| |
| * | c12388e Kill site layout again
* | | 12e882a random repo link
| |
| /
| |
* | 154ab93 80c
* | ce2d3f6 use the master branch (not necessarily master) w
* | 81de6b0 1 per gist, less strict jetpack search
* | 8eb5cc1 it only worked in my mind
```

Local Hooks

```
$ tree .git/hooks/
.git/hooks/
├── applypatch-msg.sample
├── commit-msg.sample
├── post-commit.sample
├── post-receive.sample
├── post-update.sample
├── pre-applypatch.sample
├── pre-commit.sample
├── pre-rebase.sample
└── prepare-commit-msg.sample
    └── update.sample

$ mv .git/hooks/pre-commit.sample .git/hooks/pre-commit
$ chmod +x .git/hooks/pre-commit
```

Commit Process Hooks

pre-commit

```
#!/usr/bin/env ruby

# only allows you to modify certain subdirs in a project
def check_directory_perms
  access = ['doc', 'tests']

  mod = `git diff-index --cached --name-only HEAD`
  files_mod = mod.split("\n")
  files_mod.each do |path|
    next if path.size == 0
    has_file_access = false
    access.each do |access_path|
      if !access_path || (path.index(access_path) == 0)
        has_file_access = true
      end
    end
    if !has_file_access
      puts "[POLICY] You do not have access to push to #{path}"
      exit 1
    end
  end
end

check_directory_perms
```

prepare-commit-msg

commit-msg

```
#!/usr/bin/env ruby
message_file = ARGV[0]
message = File.read(message_file)

$regex = /\b[ref: (\d+)\b]/

if !$regex.match(message)
  puts "[POLICY] Your message is not formatted correctly"
  exit 1
end
```

```
$ git commit -am 'test'  
[POLICY] Your message is not formatted correctly  
  
$ git commit -am 'test [ref: 132]'  
[master e05c914] test [ref: 132]  
 1 files changed, 1 insertions(+), 0 deletions(-)
```

post-commit

Other Local Hooks

pre-rebase

```
#!/usr/bin/env ruby

base = ARGV[0]
if ARGV[1]
  topic = ARGV[1]
else
  topic = "HEAD"
end

target_shas = `git rev-list #{base}..#{topic}`.split("\n")
remote.refs = `git branch -r`.split("\n").map { |r| r.strip }

target_shas.each do |sha|
  remote.refs.each do |rem_ref|
    pushed = `git rev-list ^#{sha}^@ refs/remotes/#{rem_ref}`
    if pushed.split("\n").include?(sha)
      puts "[POLICY] Commit #{sha} already been pushed"
      exit 1
    end
  end
end
```

post-checkout

post-merge

Fun Bonus Git Internals

read-tree

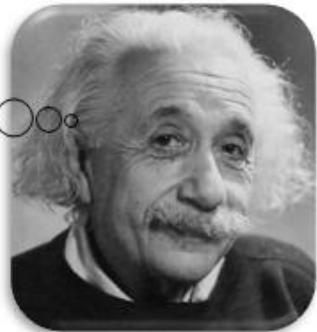
write-tree

commit-tree

that's it!



Your
feedback
is relative
to us!



In order for us to continue bring you quality learning programs, we need your input.

Survey website: go/eval or

<http://www.MetricsThatMatter.com/QCOM>

Password: **f33dback10**

Click on **NEXT**

Submit your feedback now for a course completion certificate!

