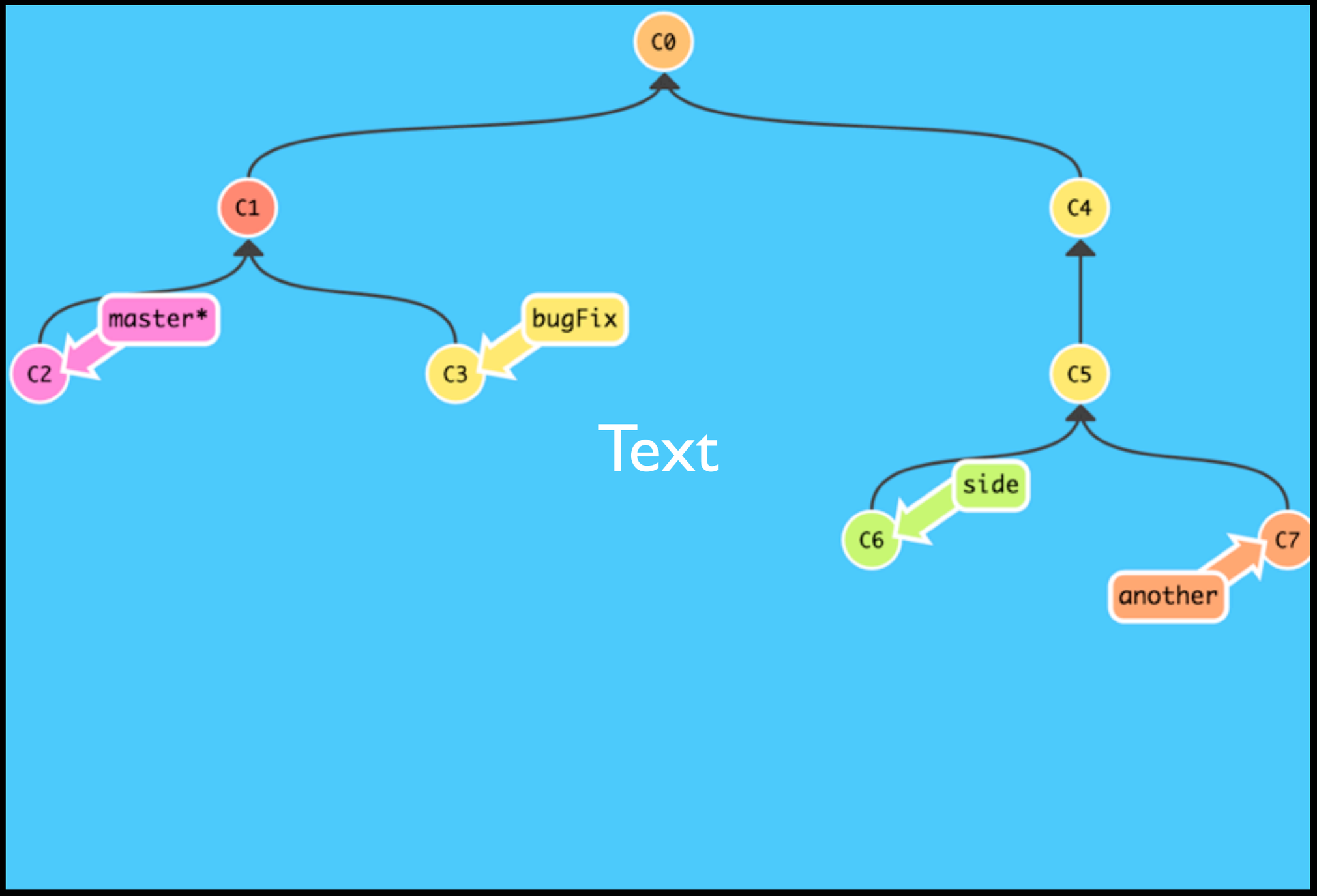
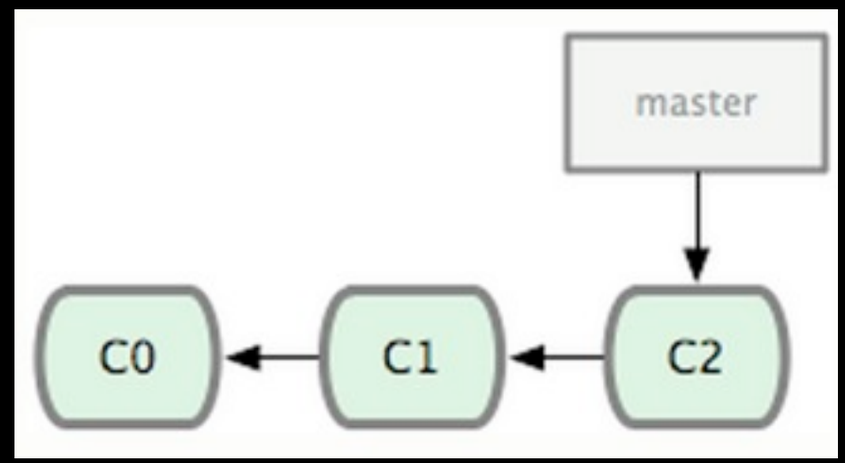
Git - Flow

Ran Tavory
Totango

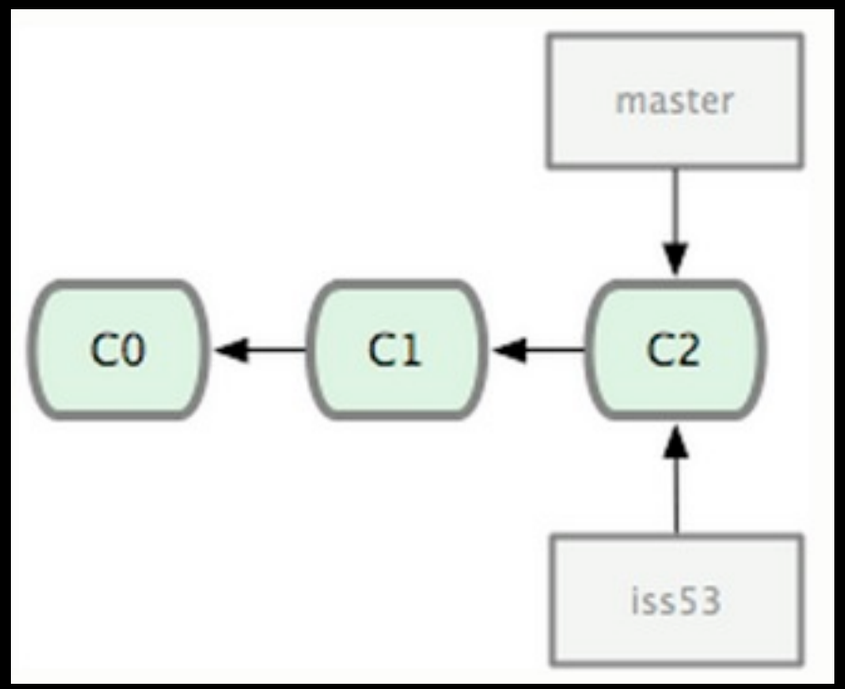


http://pcottle.github.io/learnGitBranching/?demo

Branching

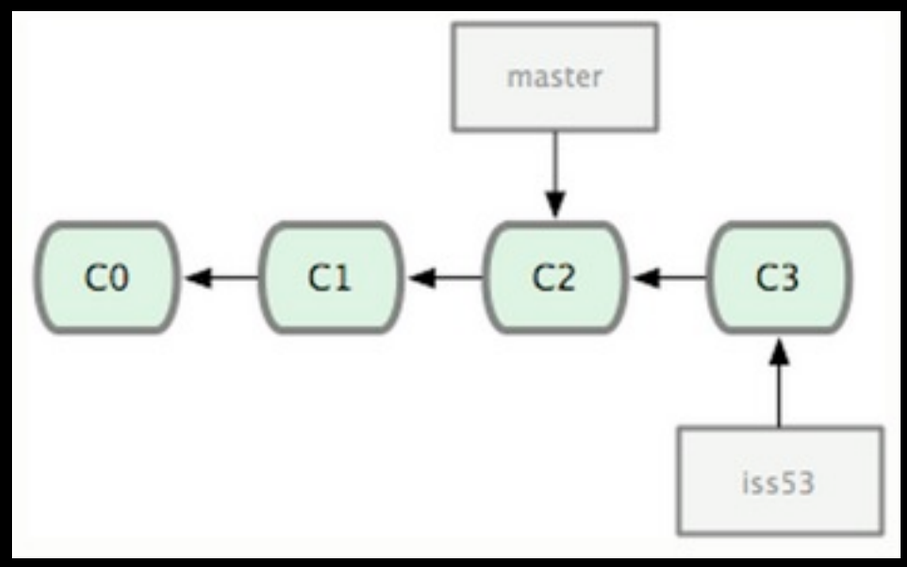



```
$ git checkout -b iss53
Switched to a new branch 'iss53'
```



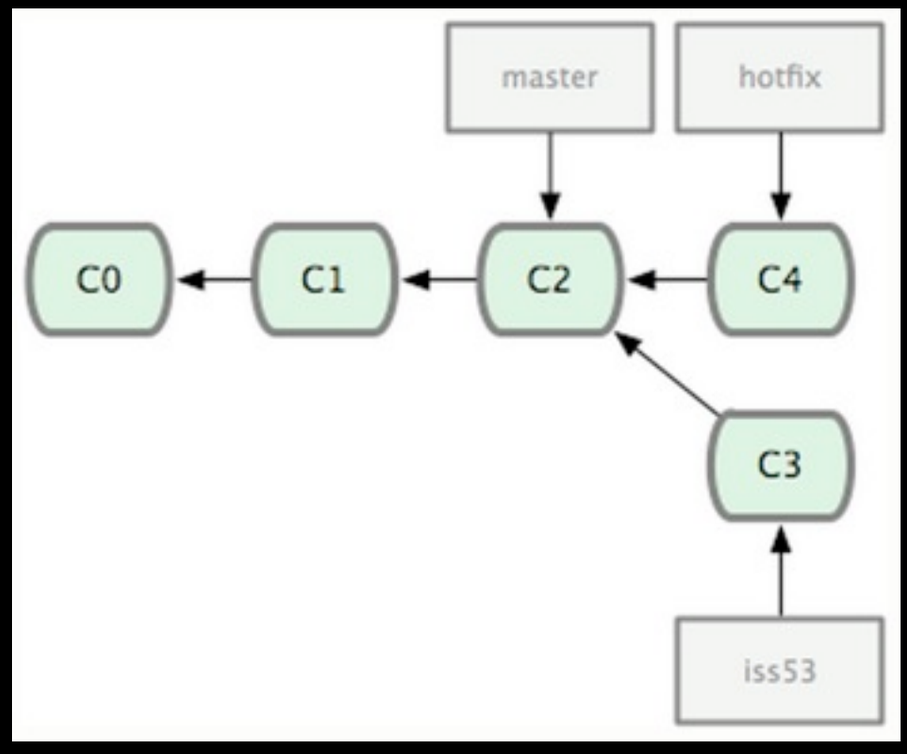
Branching

```
$ vim index.html
$ git commit -a -m 'added a new footer [issue 53]'
```




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

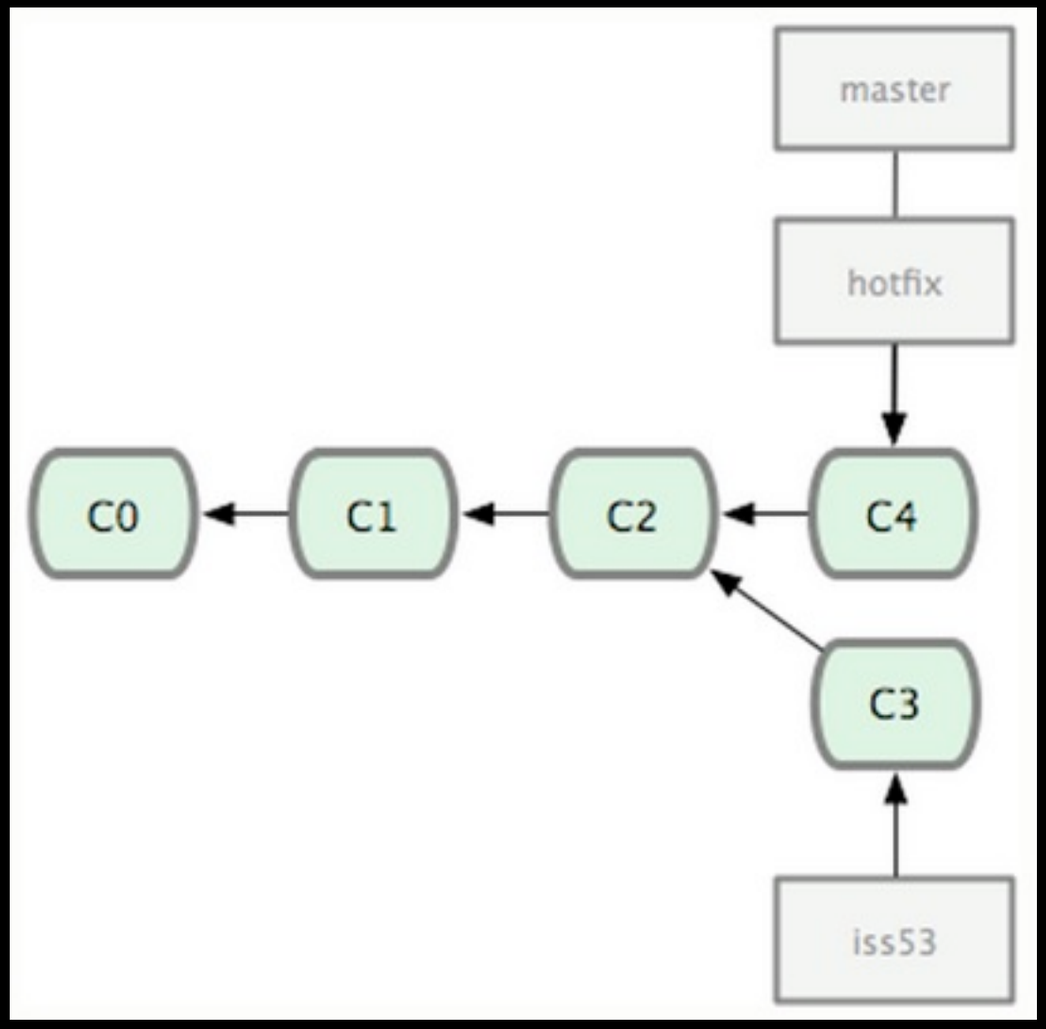
```
$ git checkout -b hotfix
Switched to a new branch 'hotfix'
$ vim index.html
$ git commit -a -m 'fixed the broken email address'
[hotfix 3a0874c] fixed the broken email address
 1 files changed, 1 deletion(-)
```



Branching - Merging

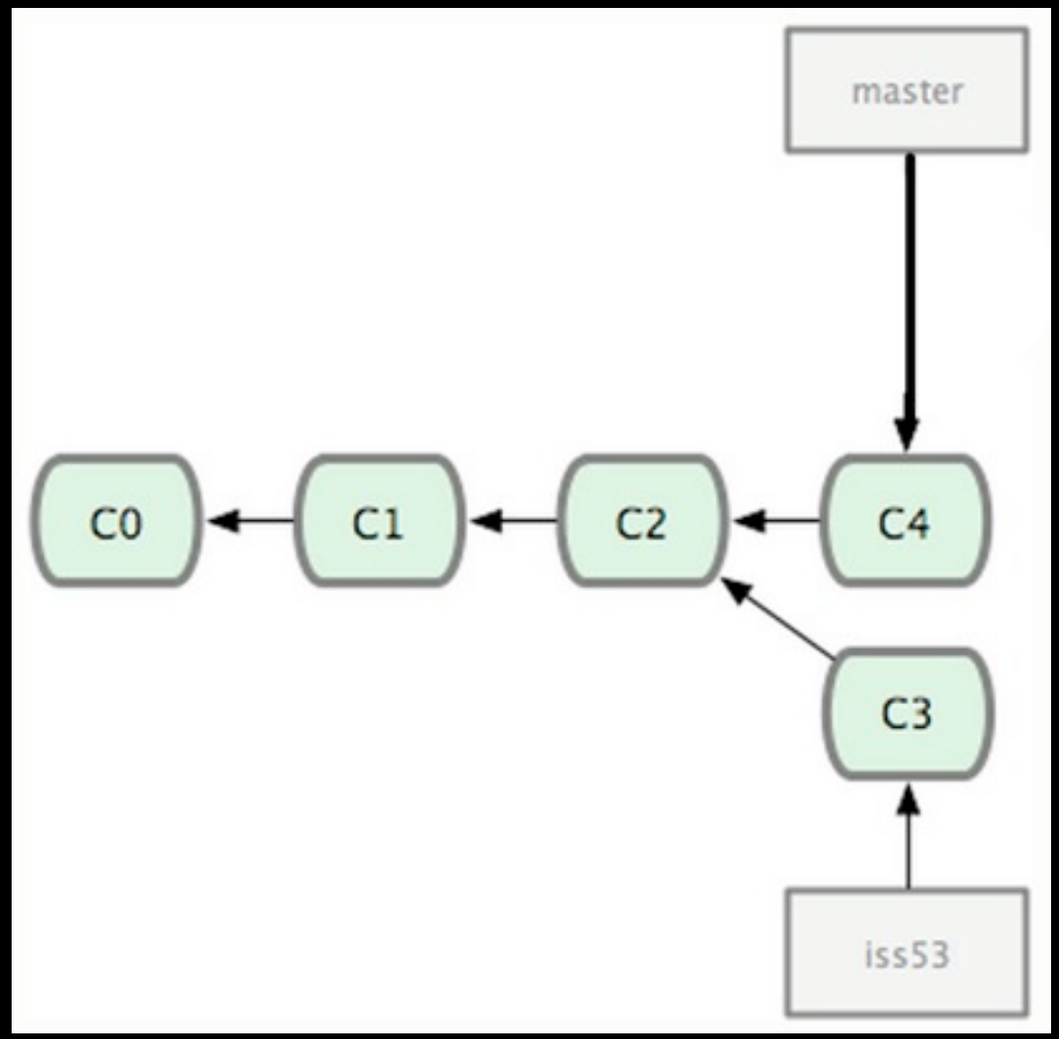
(fast forward)

```
$ git checkout master
$ git merge hotfix
Updating f42c576..3a0874c
Fast-forward
 README | 1 -
 1 file changed, 1 deletion(-)
```



Branching - Deleting

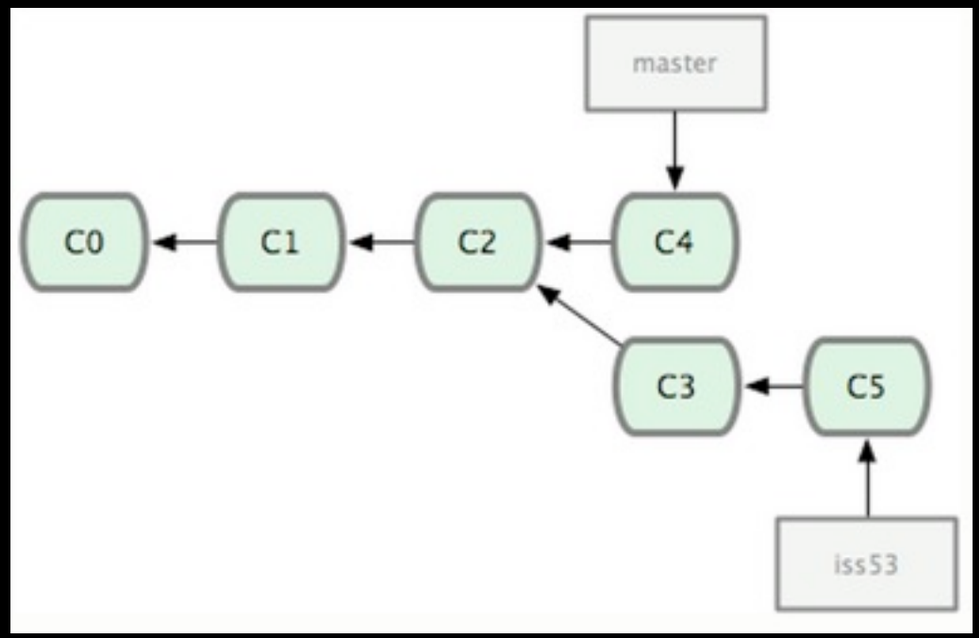
```
$ git branch -d hotfix
Deleted branch hotfix (was 3a0874c).
```



Branching

cont

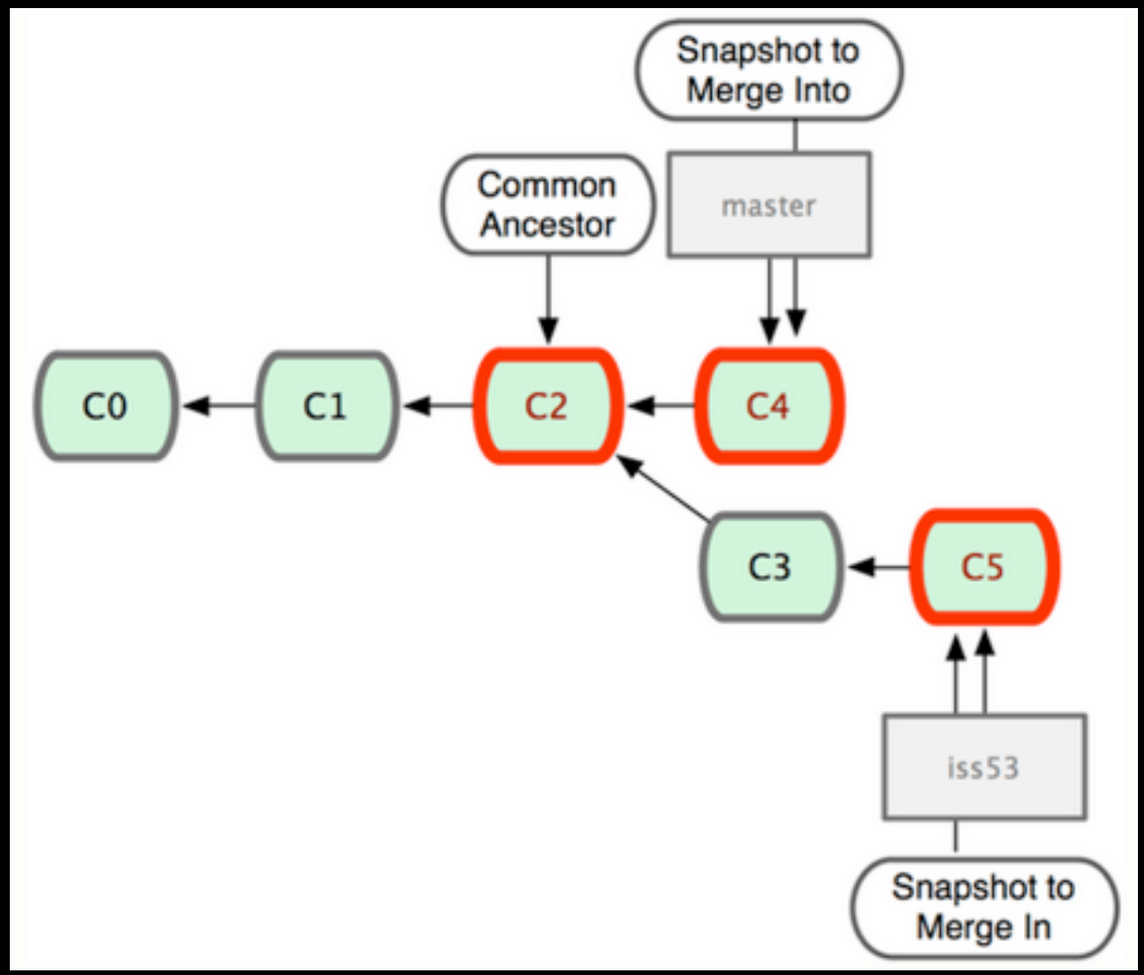
```
$ git checkout iss53
Switched to branch 'iss53'
$ vim index.html
$ git commit -a -m 'finished the new footer [issue 53]'
[iss53 ad82d7a] finished the new footer [issue 53]
 1 file changed, 1 insertion(+)
```

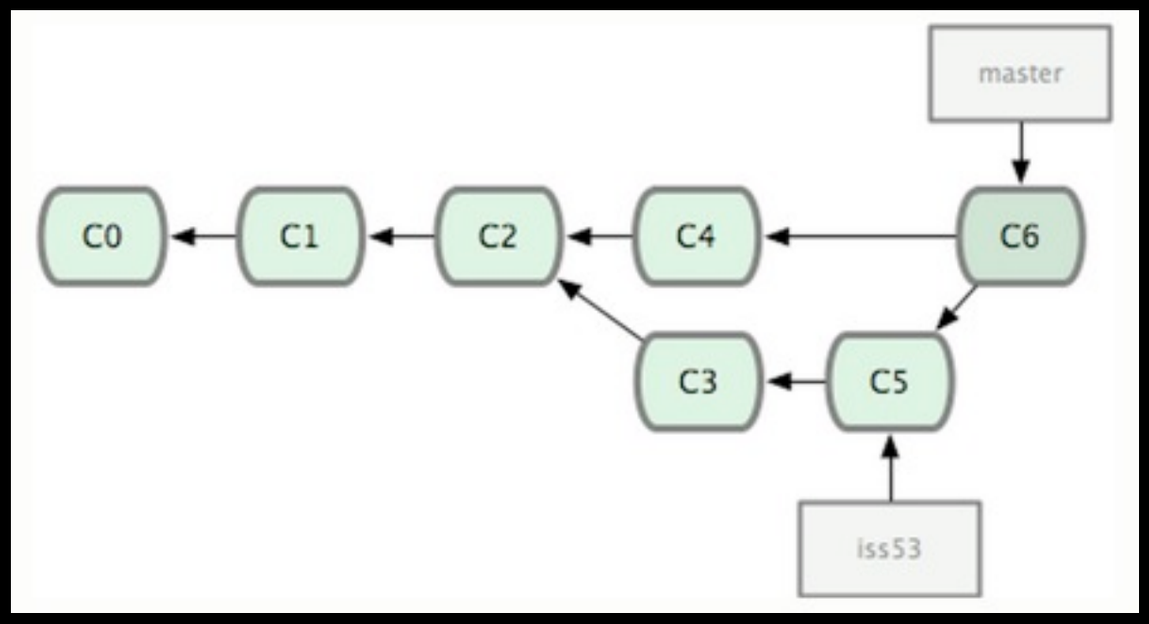


Branching - Merging

(3 way merge)

```
$ git checkout master
$ git merge iss53
Auto-merging README
Merge made by the 'recursive' strategy.
 README | 1 +
 1 file changed, 1 insertion(+)
```





Branching - Management

```
$ git branch
  iss53
* master
  testing
```

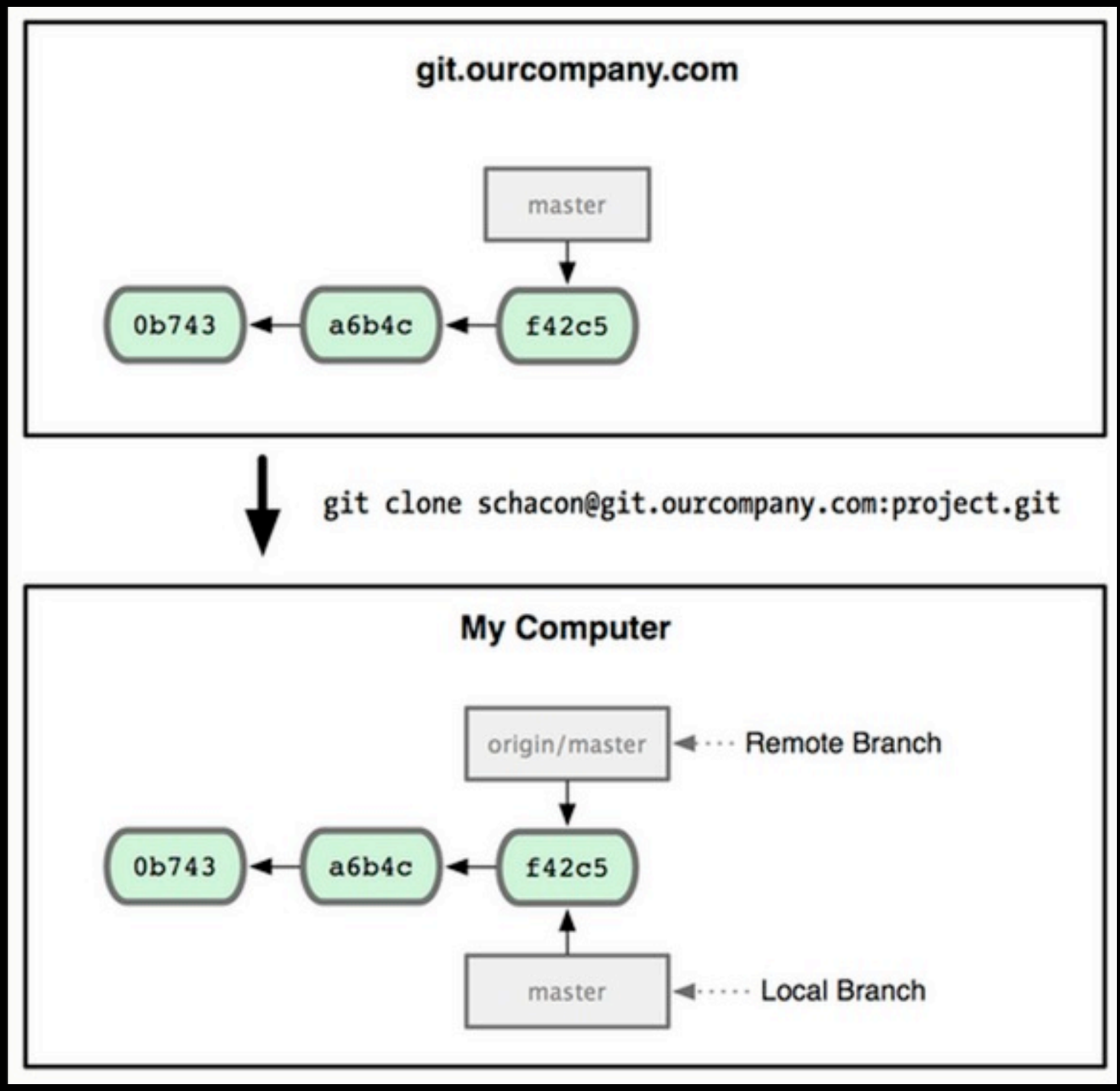
```
$ git branch -v
  iss53   93b412c fix javascript issue
* master  7a98805 Merge branch 'iss53'
  testing 782fd34 add scott to the author list in the readmes
```

```
$ git branch --merged
  iss53
* master
```

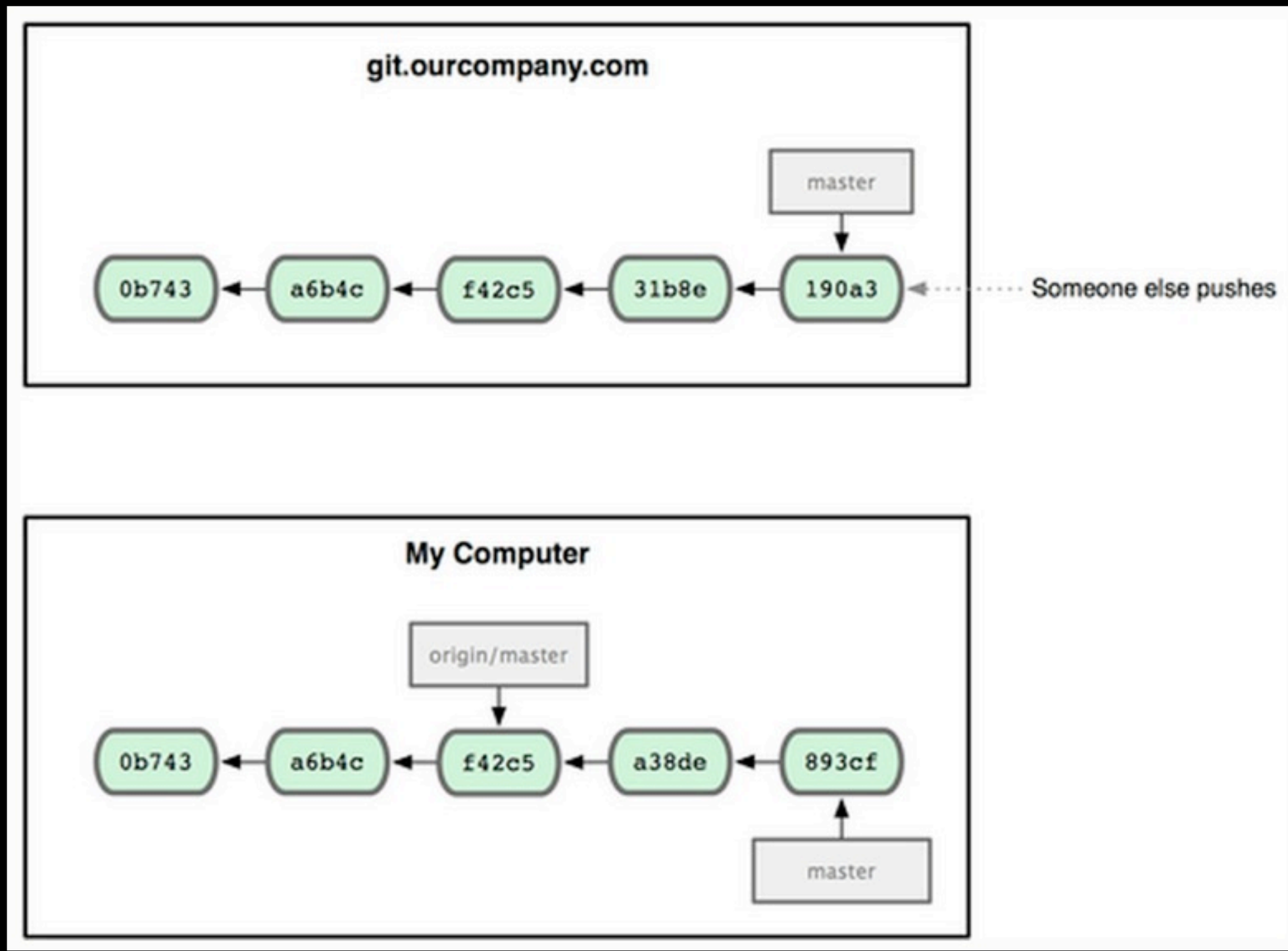
```
$ git branch --no-merged
  testing
```

```
$ git branch -d testing
error: The branch 'testing' is not fully merged.
If you are sure you want to delete it, run 'git branch -D testing'.
```

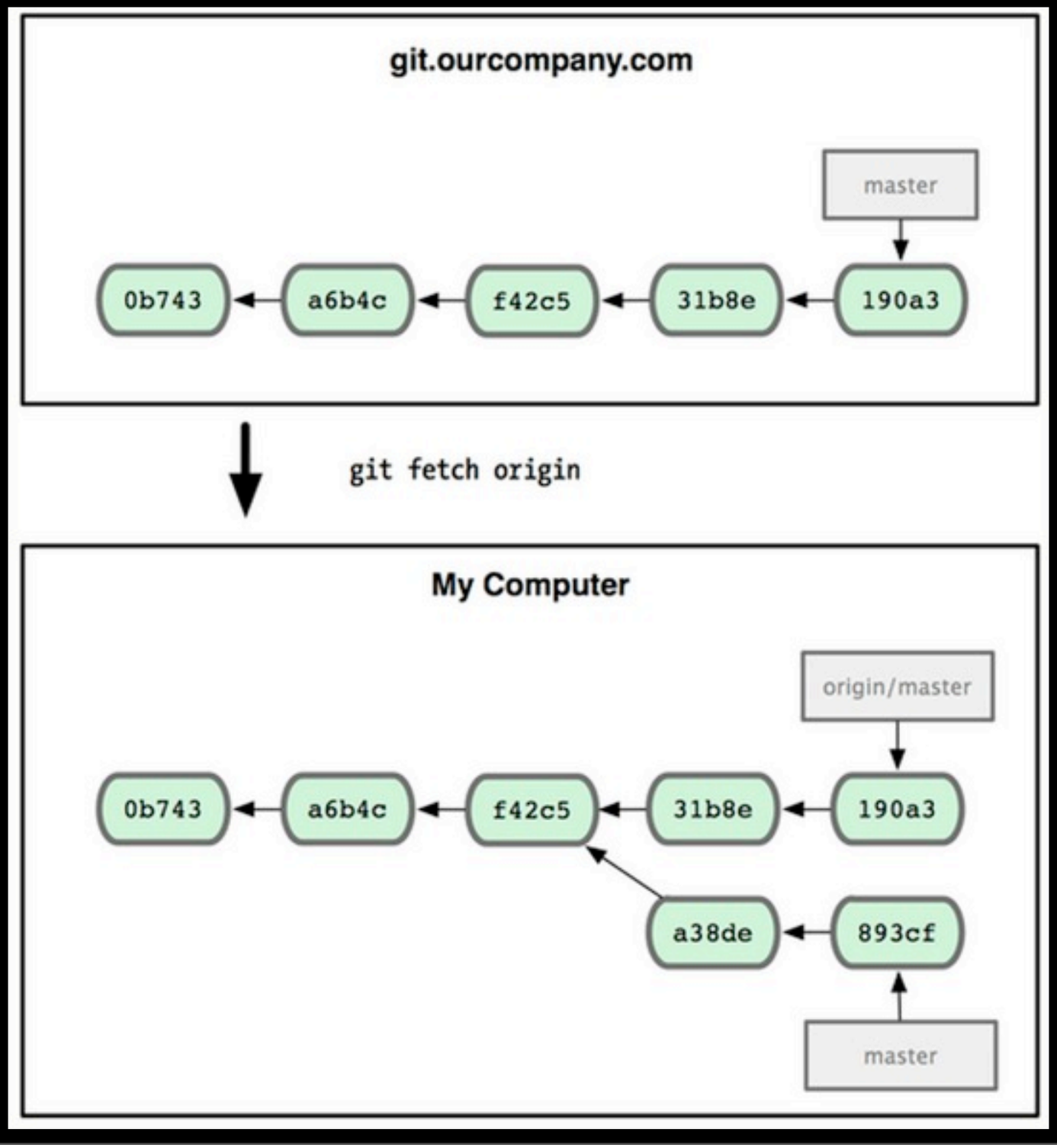
Branching - Remote



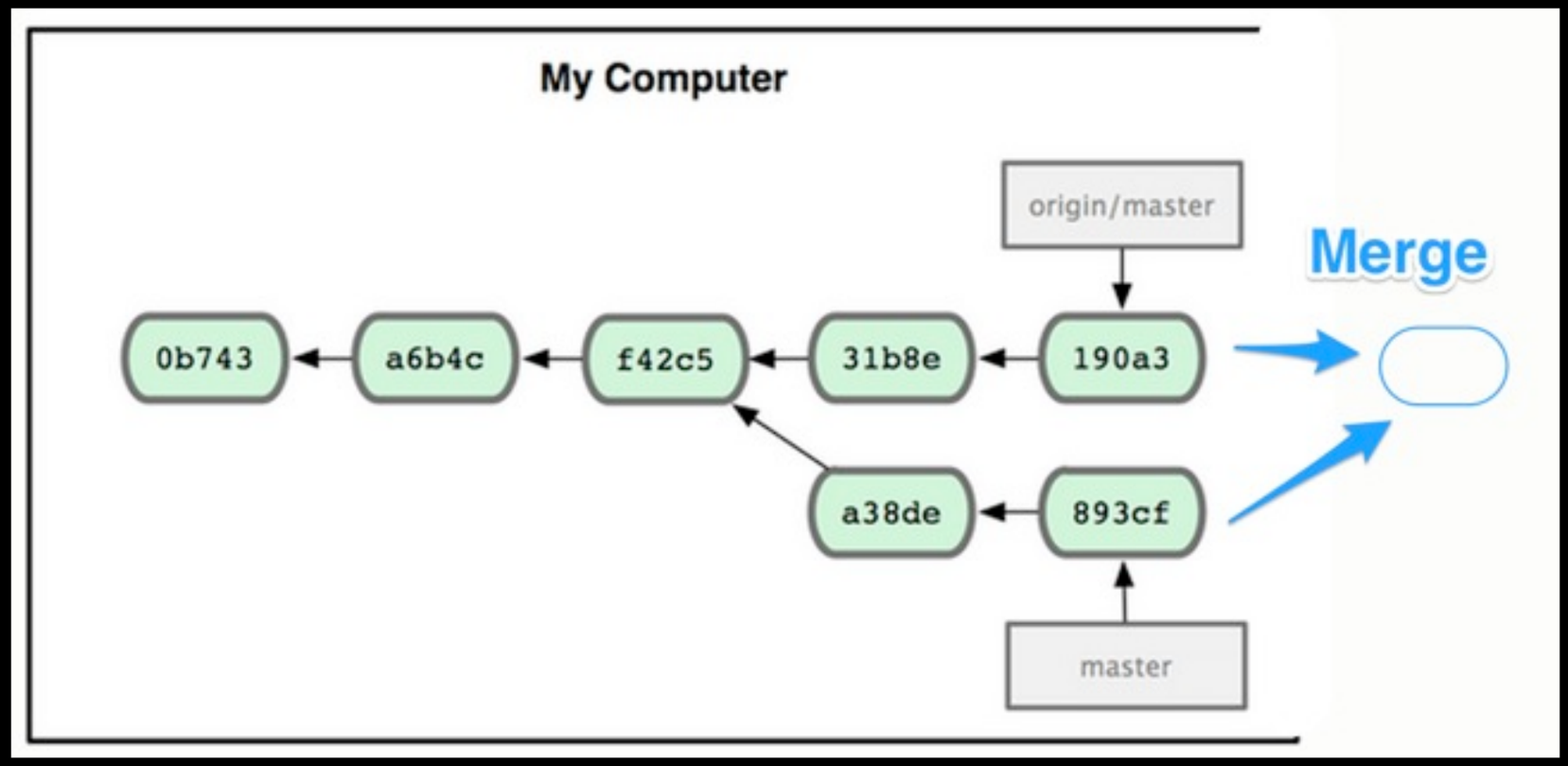
Branching - Remote



Branching - Remote



Branching - Remote




```
git pull == git fetch && git merge
```

Branching - Pushing

```
$ git push origin serverfix
Counting objects: 20, done.
Compressing objects: 100% (14/14), done.
Writing objects: 100% (15/15), 1.74 KiB, done.
Total 15 (delta 5), reused 0 (delta 0)
To git@github.com:schacon/simplegit.git
 * [new branch]      serverfix -> serverfix
```

`git push`

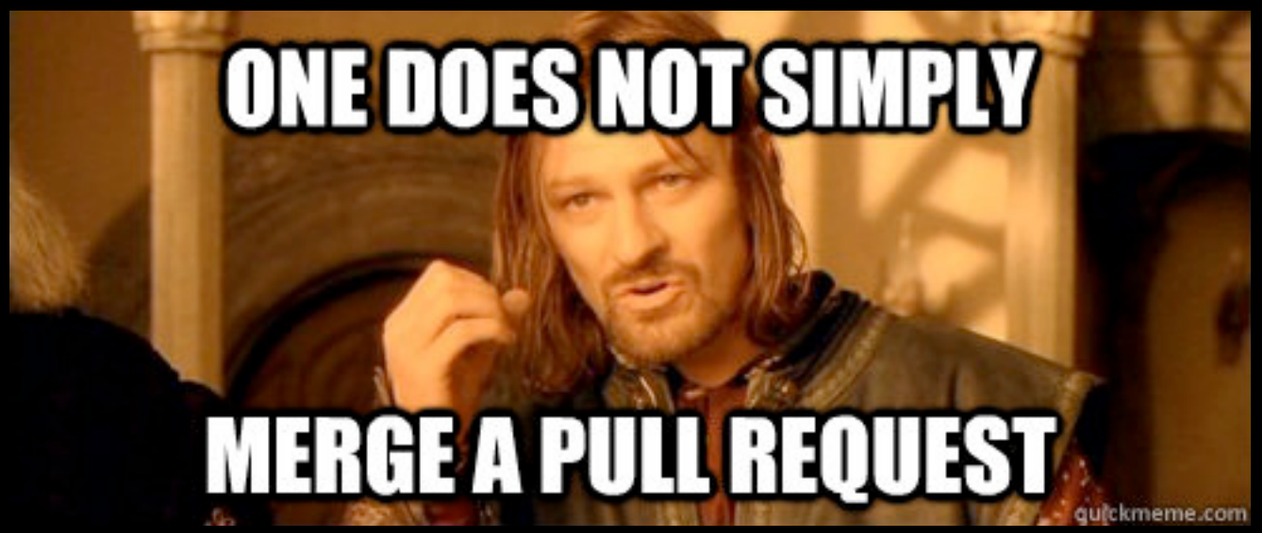
Requires tracking

```
git push origin serverfix:awesomebranch
```

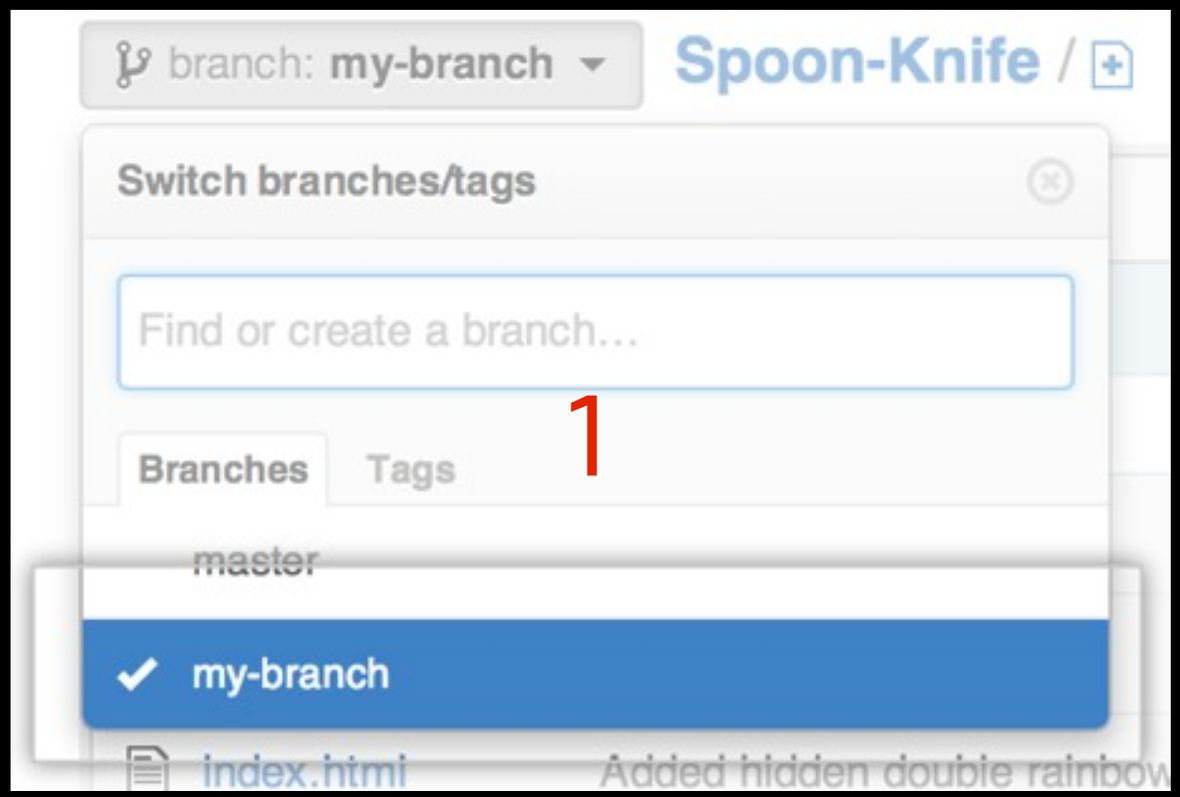
Deleting Remote Branches

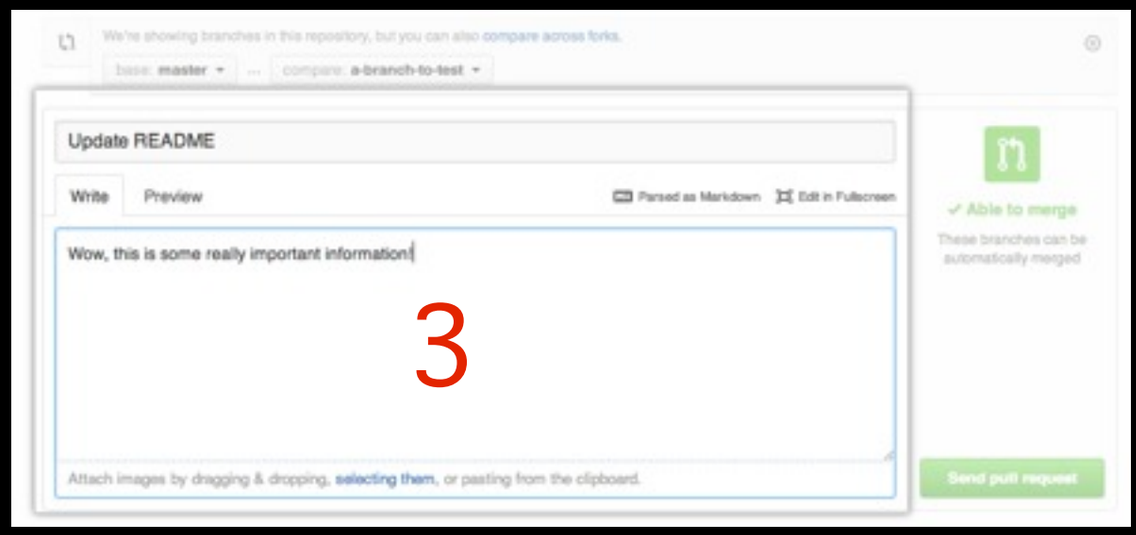
```
$ git push origin :serverfix
To git@github.com:schacon/simplegit.git
 - [deleted]         serverfix
```

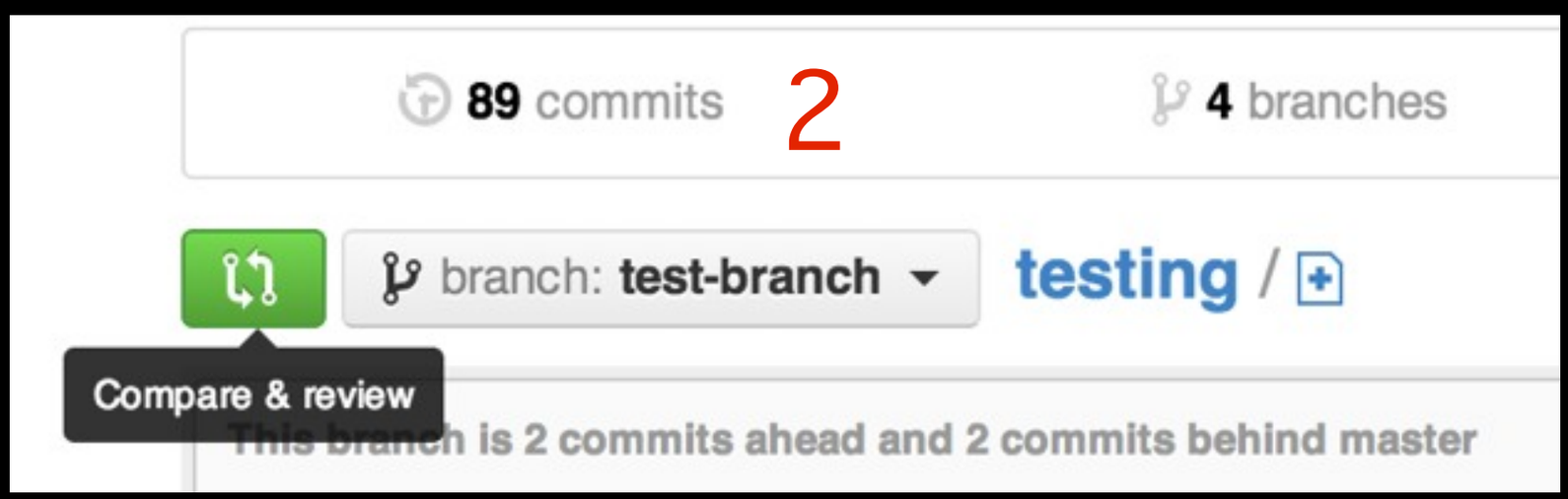
Pull Requests

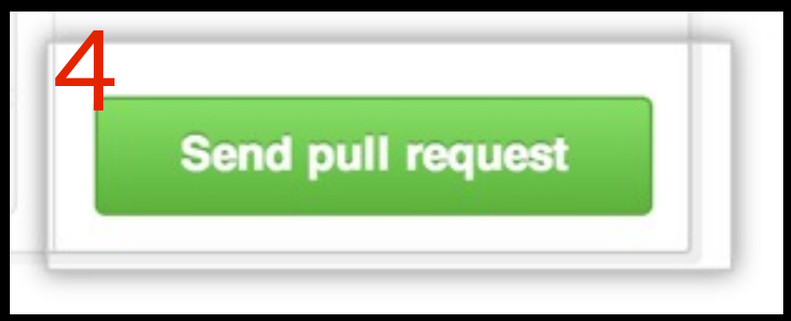


Creating a Pull Request

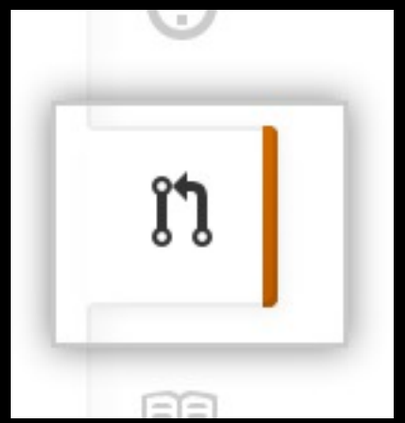


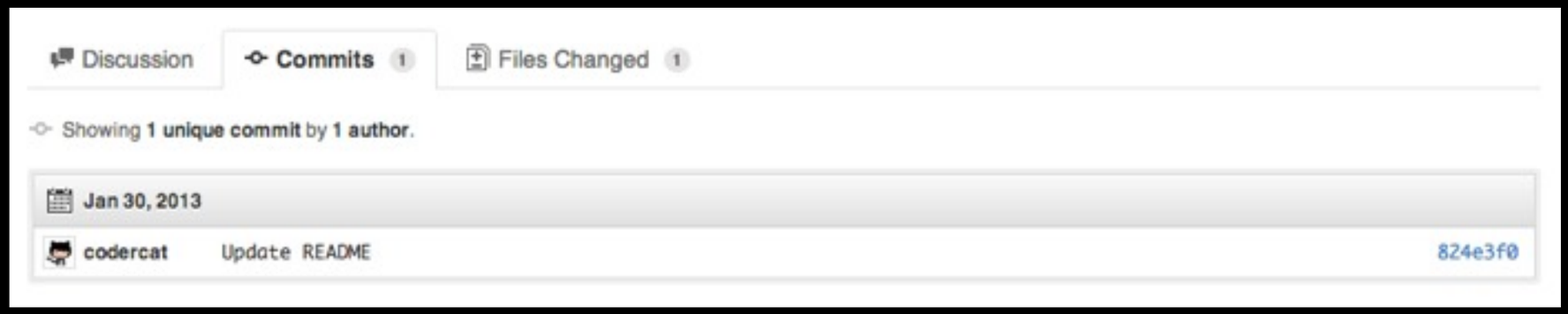


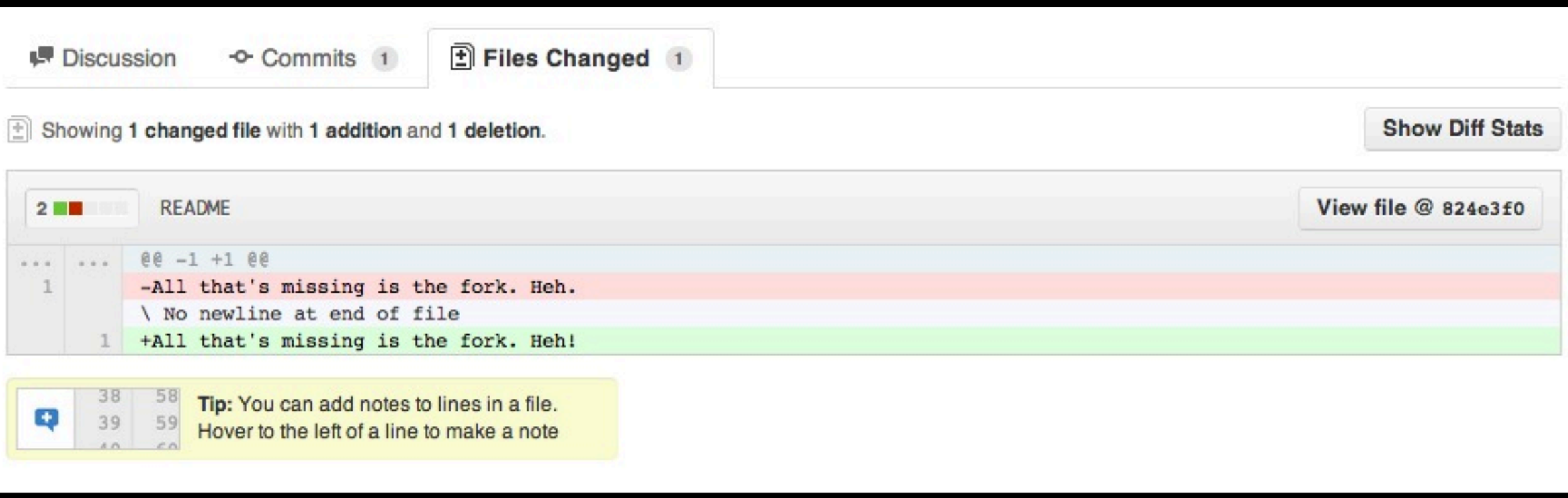




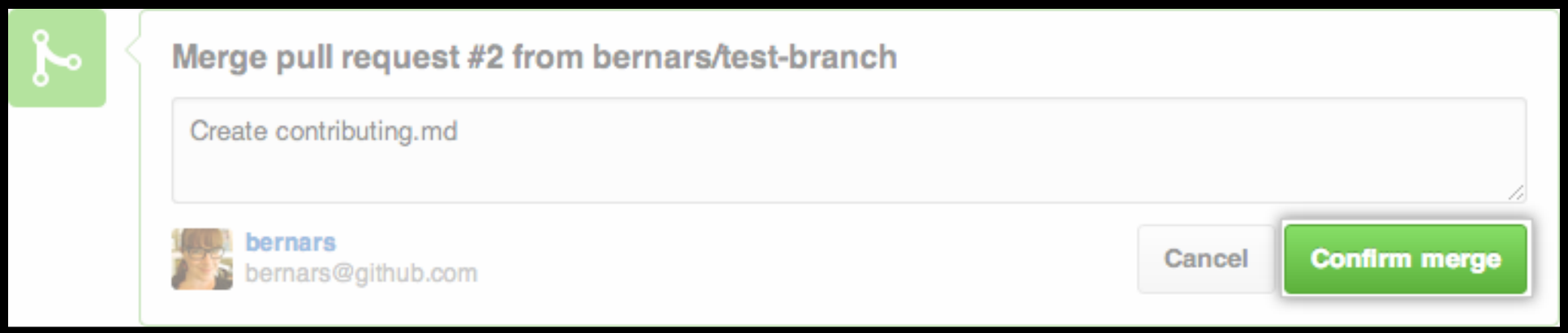
Reviewing a Pull Request







Merging Pull Request Online



Merging Pull Requests Offline

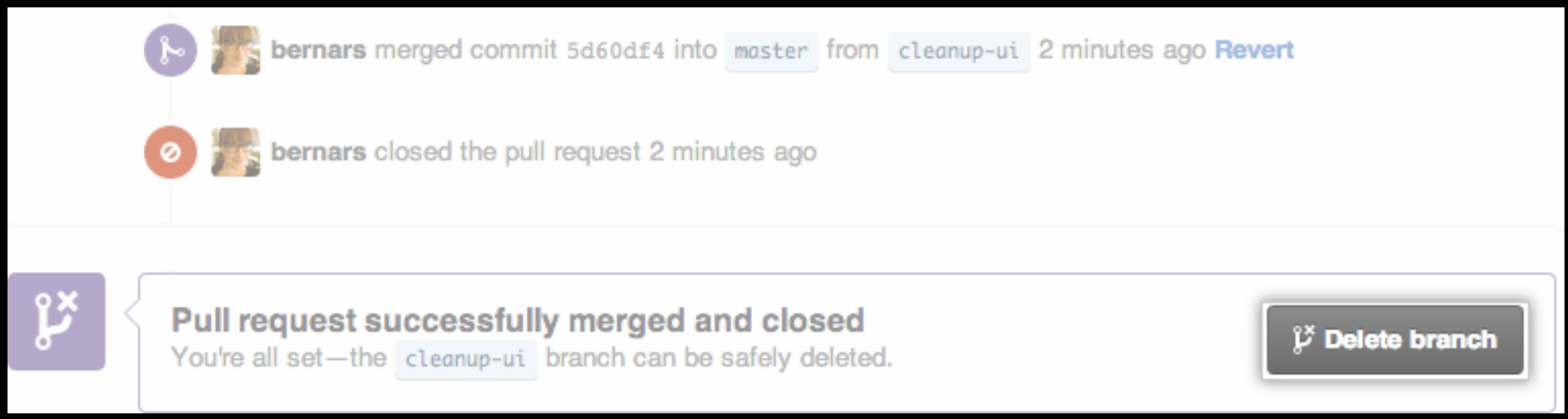
```
$ git checkout master
```

```
$ git pull https://github.com/octocat/repo.git branchname
```

- Resolve Conflicts
- Test
- Improve

```
$ git push origin master
```

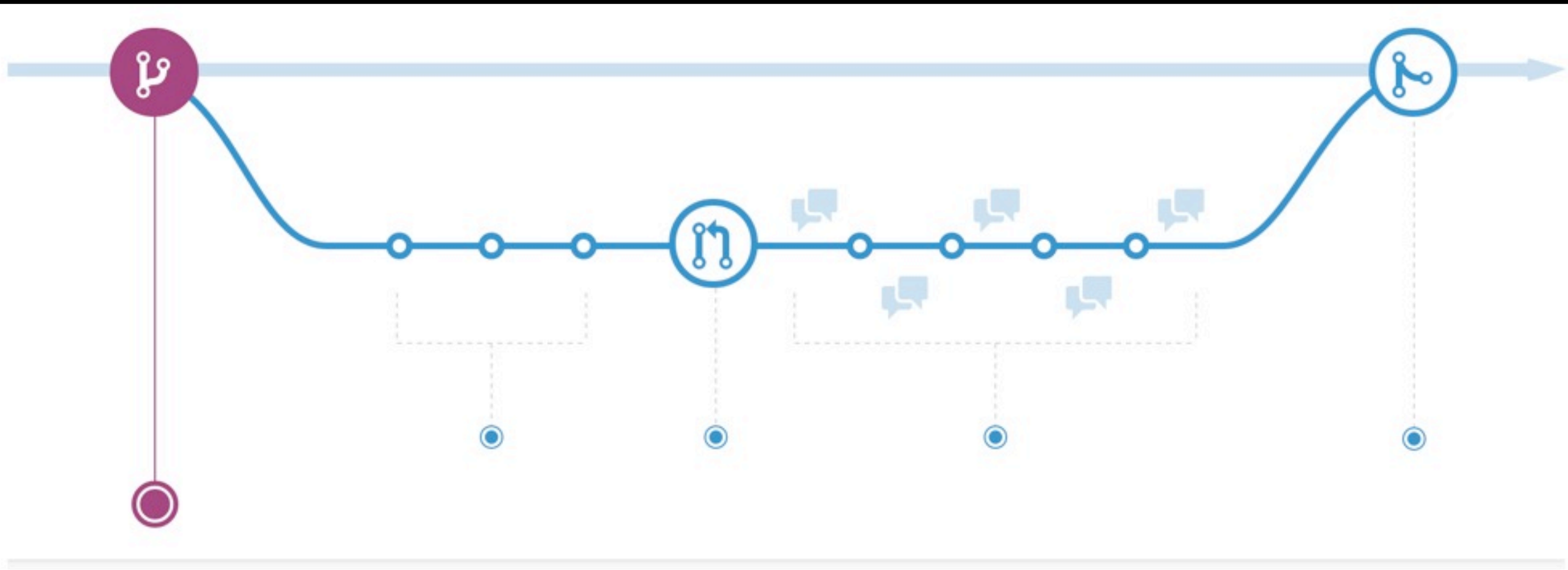
Deleting Branch After PR Merge



The Flow

KEEP
CALM
AND
GO WITH
THE FLOW

1. Create a Branch

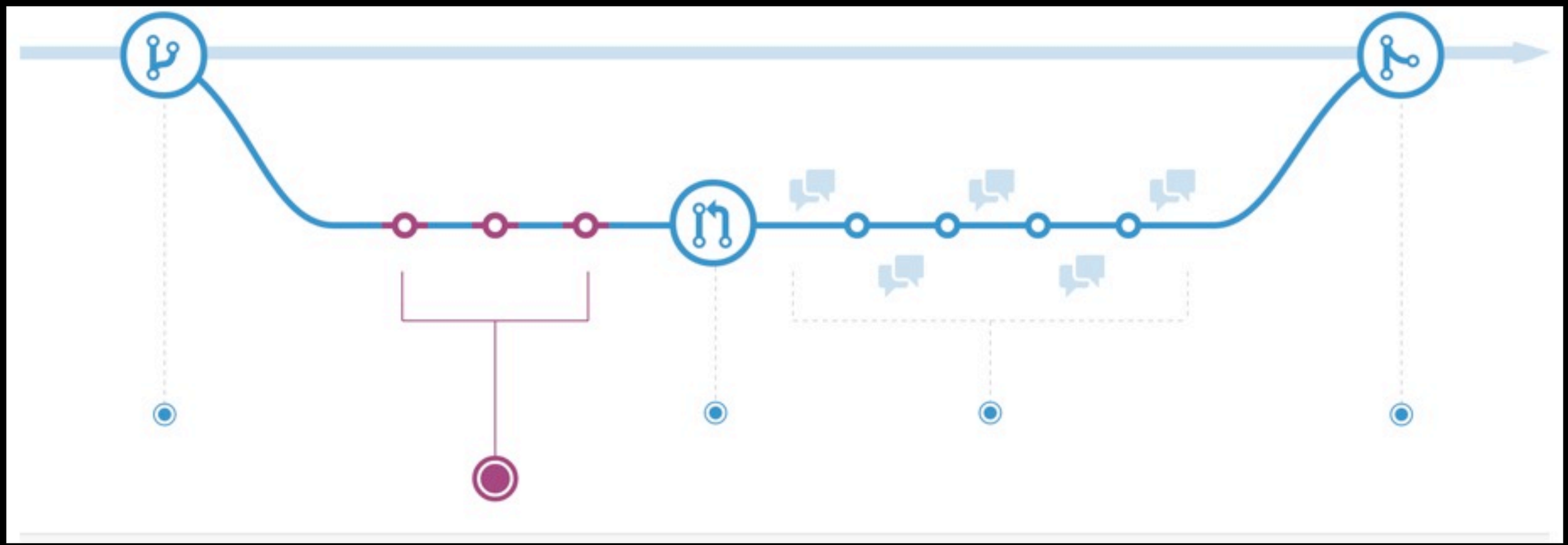


```
git co -b bug-157
```

Branch Naming:

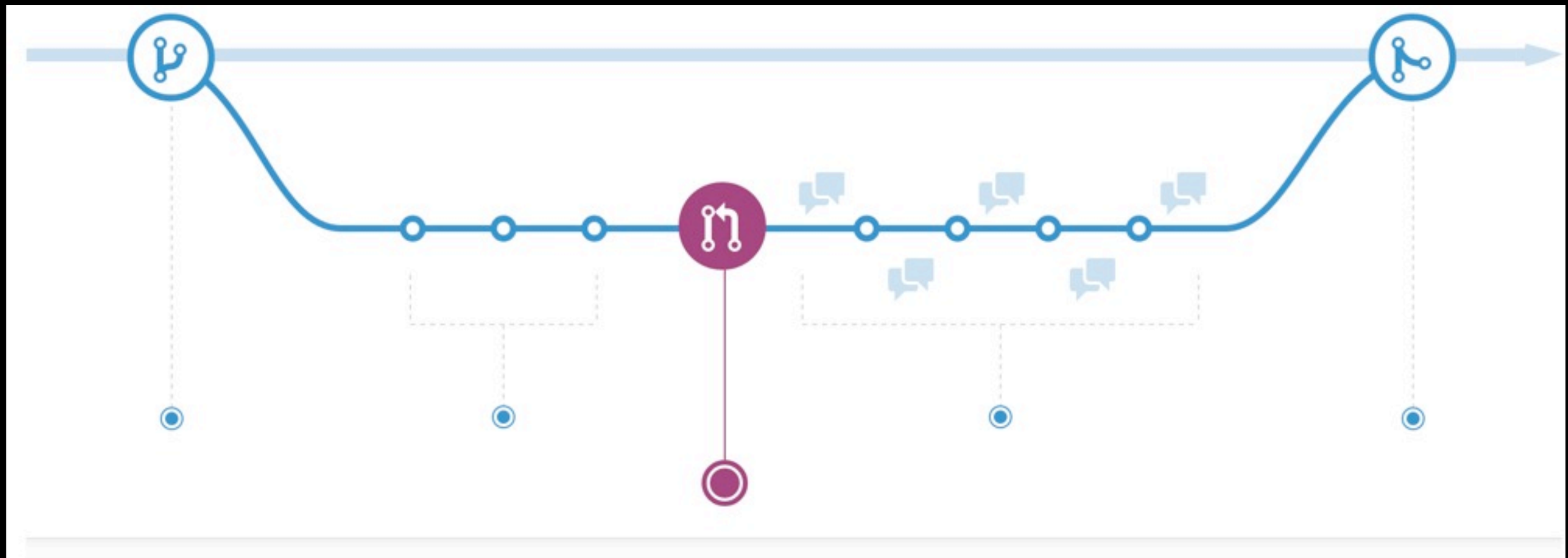
```
bug-xxx
feature-yyy
wip-zzz
```

2. Add Commits



Commit messages are super important!!!

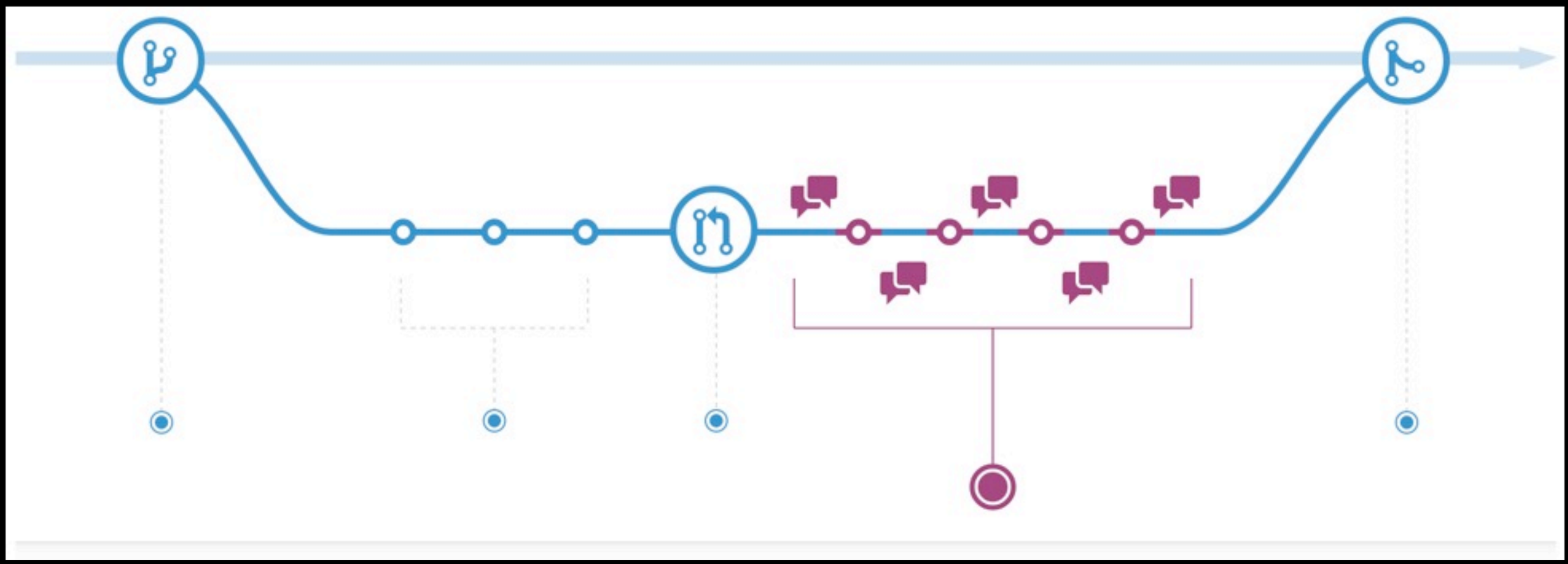
3. Open a Pull Request



Make sure you assign a reviewer!

4. Discuss

AKA Code Review



Continue to commit and push.
This will be reflected

5. Merge

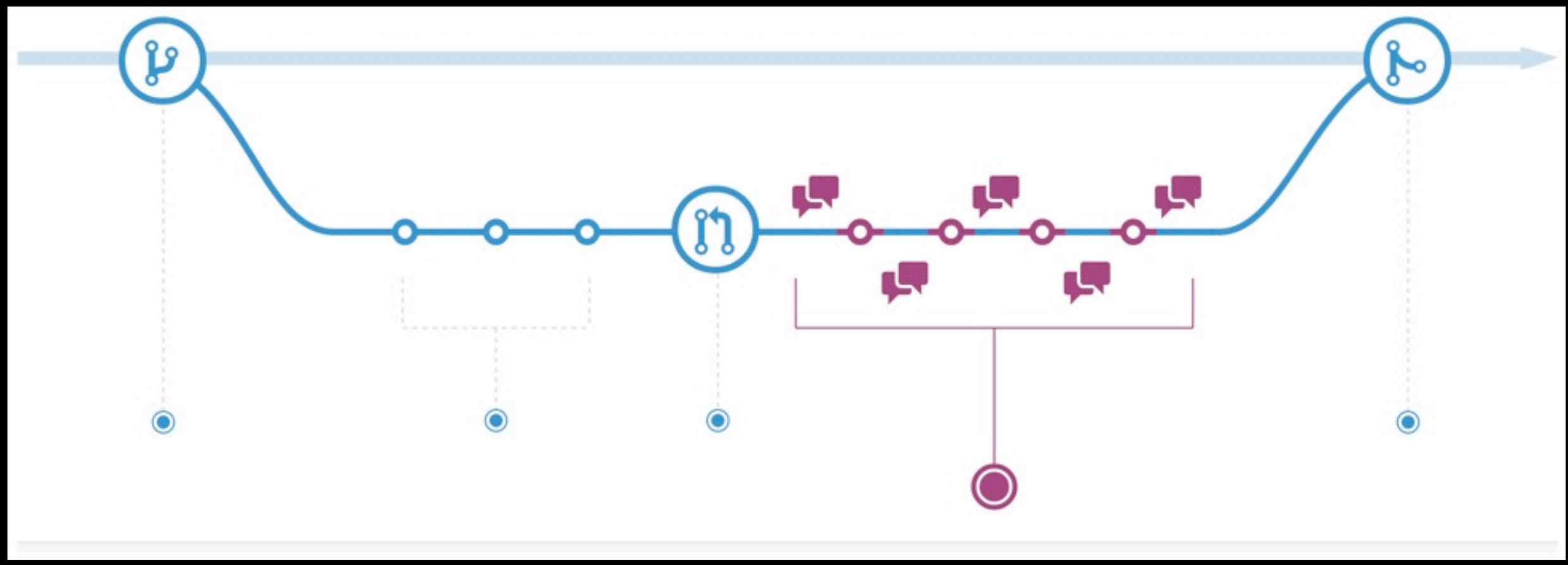
Merge means: It's good for production.

`master` is always deployable

Pull Request Demo

Code Review

Code Review



Code Review

Golden Rules

- Assign author
  - Determine ownership
  - Not managers. Peers.
- Review ASAP. Same day.
  - Highest priority, after prod
- Refer to Coding Guidelines (Java, Javascript, Python)
- 200 - 400 LOC Max. 400 LOC = 1 hour
- Verify that defects are actually fixed!

Code Review

What to look for

- Reusability
- Is the code in the right location in the codebase?
- Style guidelines
- חוק השיבר
- Coupling and Cohesion
- Are the loops bounded?
- Is the number of I/O operations minimized?
- Is the code metered?
- Is the code tested?
  - Each public method must have a few unit tests.
- ... and there's more...

Extra

Conflicts

Conflicts

Edit Collision

```
$ git status
# On branch branch-b
# You have unmerged paths.
#   (fix conflicts and run "git commit")
#
# Unmerged paths:
#   (use "git add ..." to mark resolution)
#
# both modified:      planets.md
#
no changes added to commit (use "git add" and/or "git commit -a")
```

```
the number of planets are
<<<<<<< HEAD
nine
=======
eight
>>>>>>> branch-a
```

Edit file and then

```
$ git add
$ git commit
```

Conflicts

Removed File

```
CONFLICT (modify/delete): README.md deleted in HEAD and modified in branch-b.
Version branch-b of README.md left in tree.
Automatic merge failed; fix conflicts and then commit the result.

$ git status
# On branch branch-c
# You have unmerged paths.
#   (fix conflicts and run "git commit")
#
# Unmerged paths:
#   (use "git add/rm ..." as appropriate to mark resolution)
#
# deleted by us:      README.md
#
no changes added to commit (use "git add" and/or "git commit -a")
```

Conflicts

Removed File - Keep

```
$ git add README.md

$ git commit
[branch-c 9bc3b01] Merge branch 'branch-b' into branch-c

$ git show | head
commit 9bc3b0130fe0178359d51243b5b882076a12f554
Merge: 4c80a63 7e8b679
Author: tekkub :bear:
Date:   Sat Jun 1 18:39:40 2013 -0700

    Merge branch 'branch-b' into branch-c
>
    Conflicts:
        README.md
```

Conflicts

Removed File - Remove

```
$ git rm README.md
README.md: needs merge
rm 'README.md'
```

Looks good, so commit it with the default message:

```
$ git commit
[branch-d 6f89e49] Merge branch 'branch-c' into branch-d

$ git show | head
commit 6f89e49189ba3a2b7440fc434f351cb041b3999e
Merge: 211261b fcc1093
Author: tekkub :bear:
Date:   Sat Jun 1 18:43:01 2013 -0700

    Merge branch 'branch-c' into branch-d
>
    Conflicts:
        README.md
```

Conflicts

Git Mergtool

- Not covered here...
- http://git-scm.com/docs/git-mergetool

Conflicts

While Rebasing

```
$ git rebase develop
First, rewinding head to replay your work on top of it...
Applying: feature/login
Using index info to reconstruct a base tree...
Falling back to patching base and 3-way merge...
Auto-merging config/environment.rb
CONFLICT (content): Merge conflict in config/environment.rb
Failed to merge in the changes.
Patch failed at 0001 feature/login

When you have resolved this problem run "git rebase --continue".
If you would prefer to skip this patch, instead run "git rebase --skip".
To restore the original branch and stop rebasing run "git rebase --abort".
```

Refs

https://help.github.com/articles/resolving-a-merge-conflict-from-the-command-line

http://pcottle.github.io/learnGitBranching/

http://jeffkreeftmeijer.com/2010/the-magical-and-not-harmful-rebase/

https://help.github.com/articles/using-pull-requests

https://help.github.com/articles/creating-a-pull-request

http://en.wikipedia.org/wiki/Coupling_(computer_programming)