GIT BRANCHING

BASIC BRANCHING AND MERGING

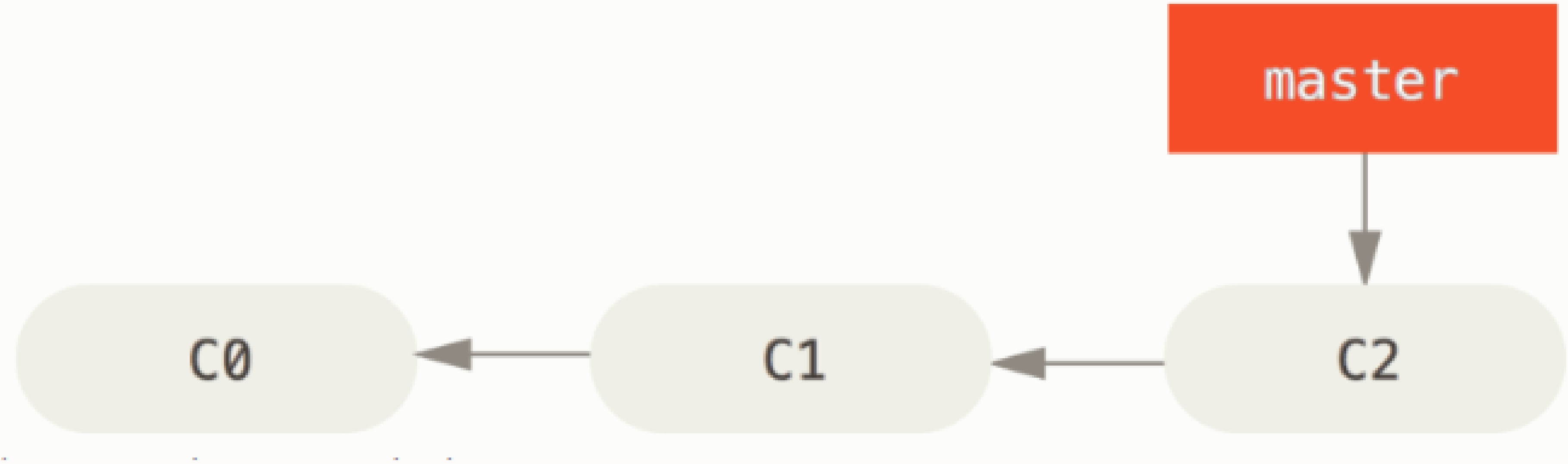
saadaslam.net/linkedin

@saadaslam.dev

Saad Aslam
saadaslams.com

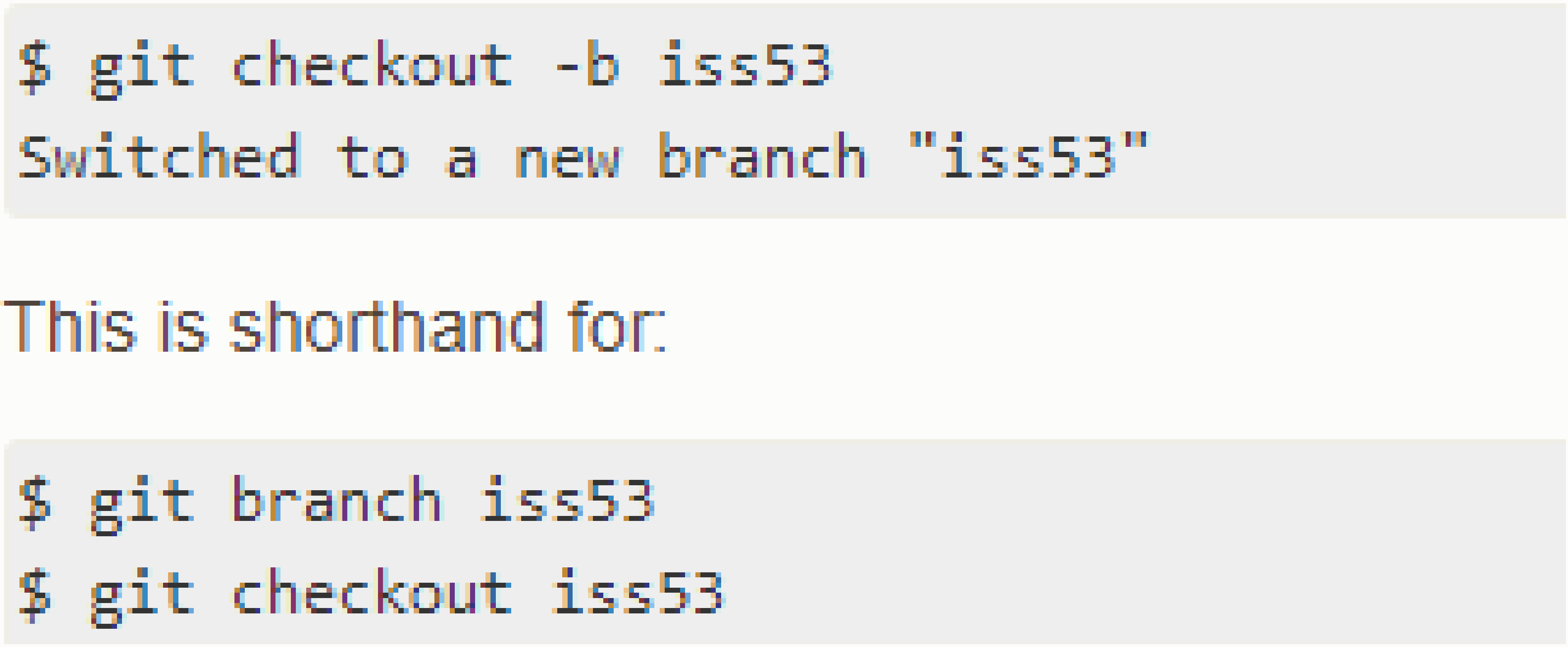
Basic Branching

First, let’s say you’re working on your project and have a couple of commits already on the master branch.



Basic Branching

You’ve decided that you’re going to work on issue #53. To create a new branch and switch to it at the same time, run the following



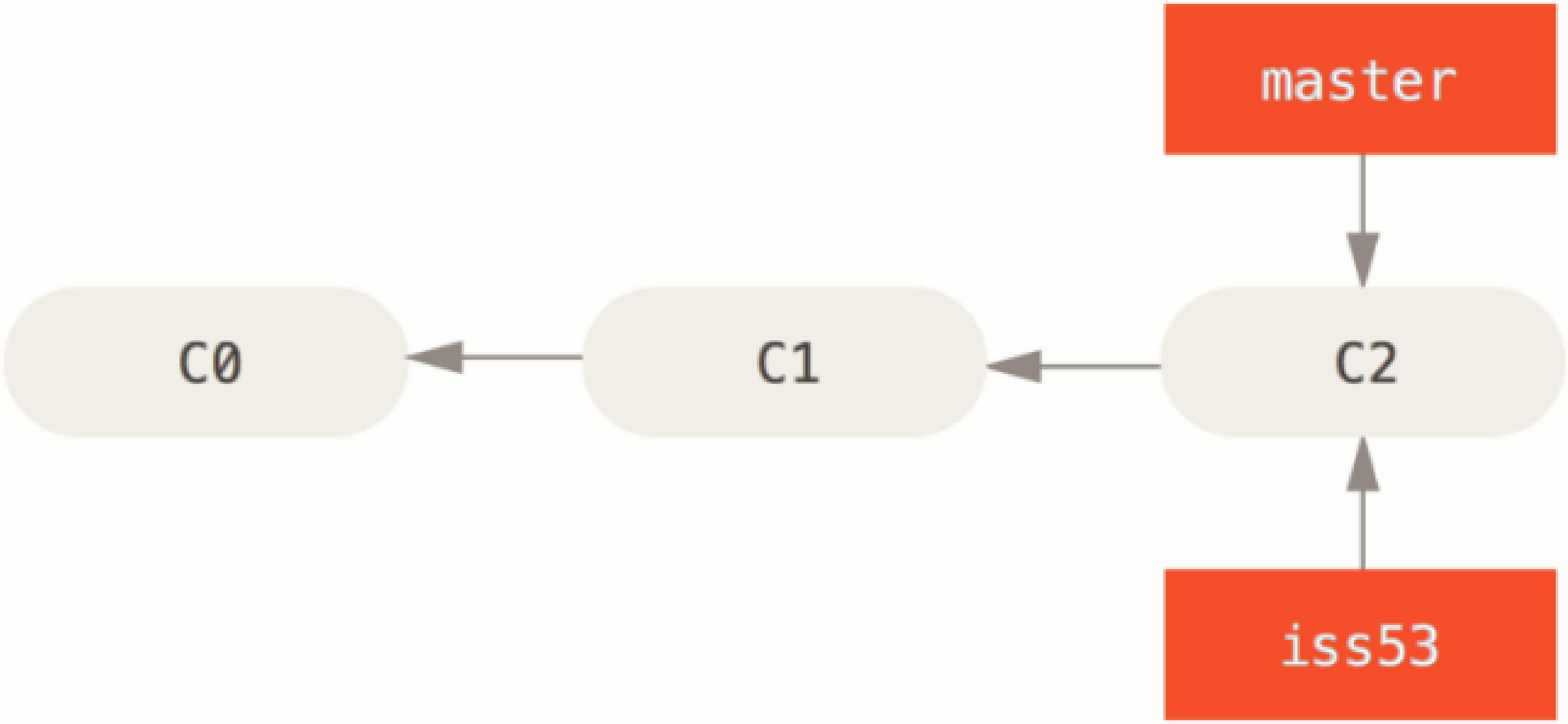
```
$ git checkout -b iss53
Switched to a new branch "iss53"
```

This is shorthand for:

```
$ git branch iss53
$ git checkout iss53
```

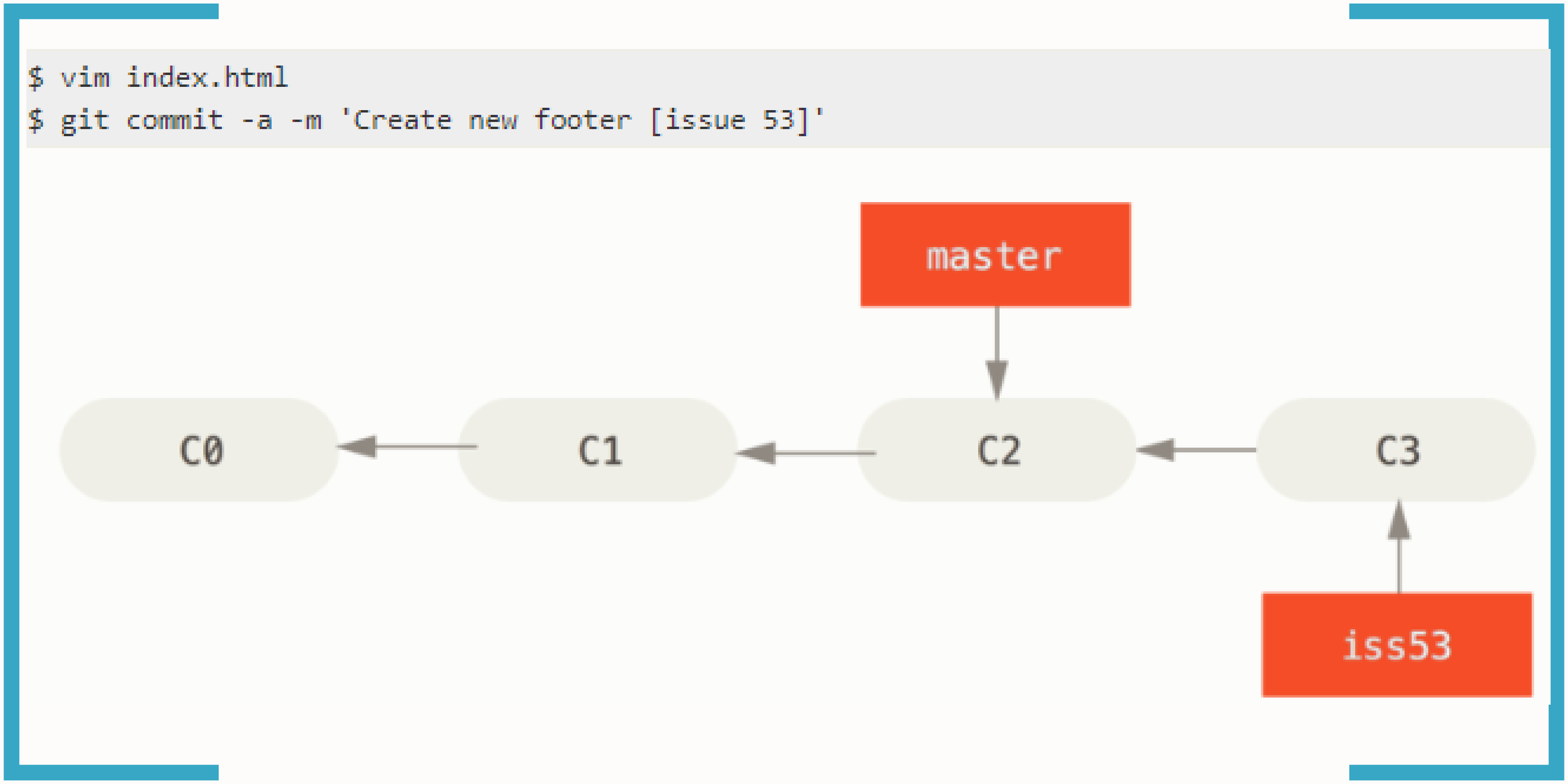
Creating a new branch pointer

After running the command **git checkout -b iss53** , you have switched to newly created branch



iss53 branch has moved forward

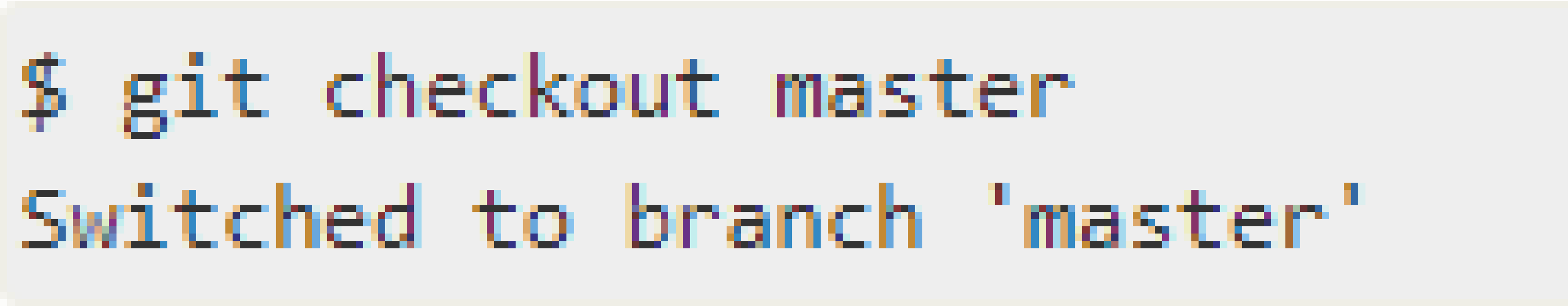
You work on your website and do some commits. Doing so moves the iss53 branch forward.



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

If there is any Hotfix to deal with

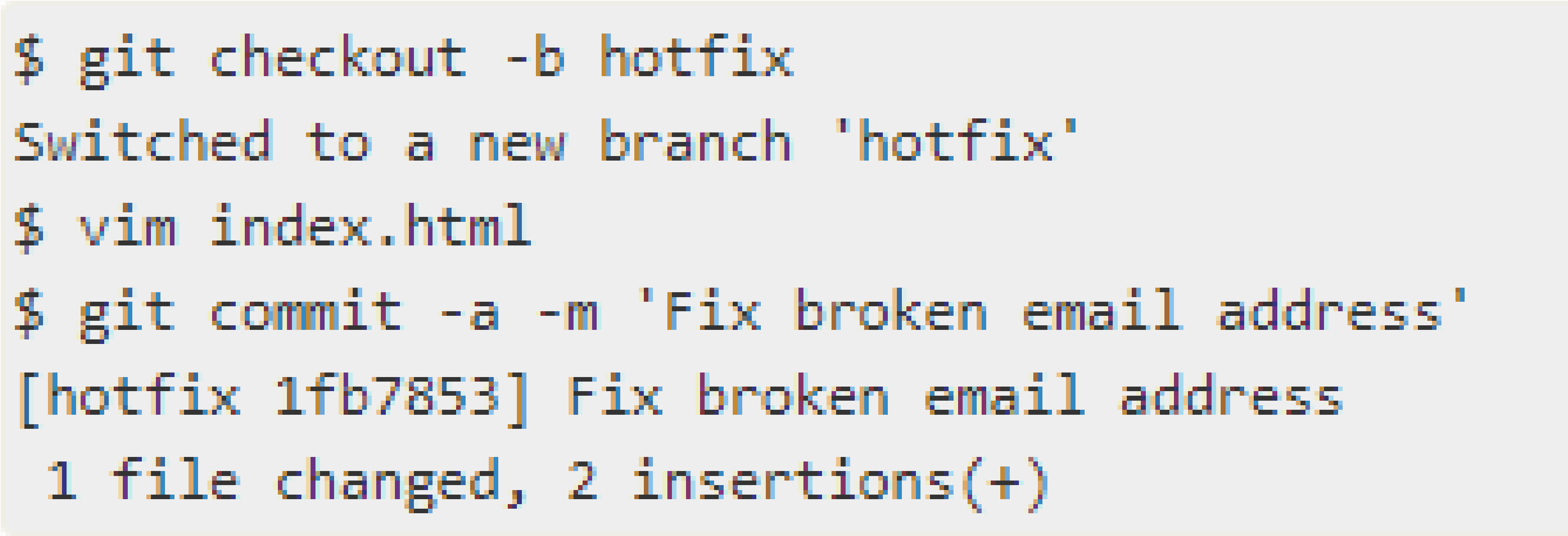
All you have to do is switch back to your master branch. Your project working directory is exactly the way it was before



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

Hotfix branch based on master

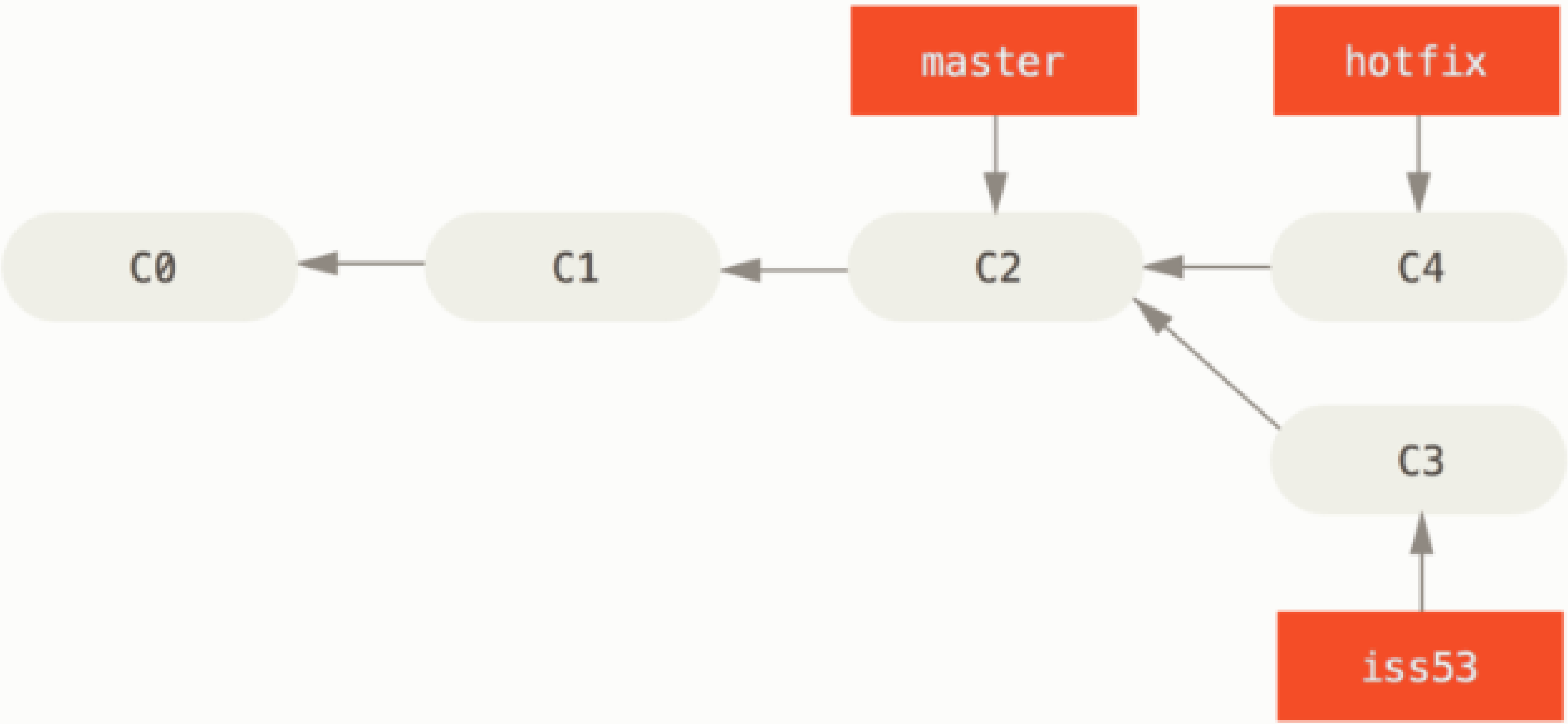
Let’s create a hotfix branch on which to work until it’s completed:



```
$ git checkout -b hotfix
Switched to a new branch 'hotfix'
$ vim index.html
$ git commit -a -m 'Fix broken email address'
[hotfix 1fb7853] Fix broken email address
 1 file changed, 2 insertions(+)
```

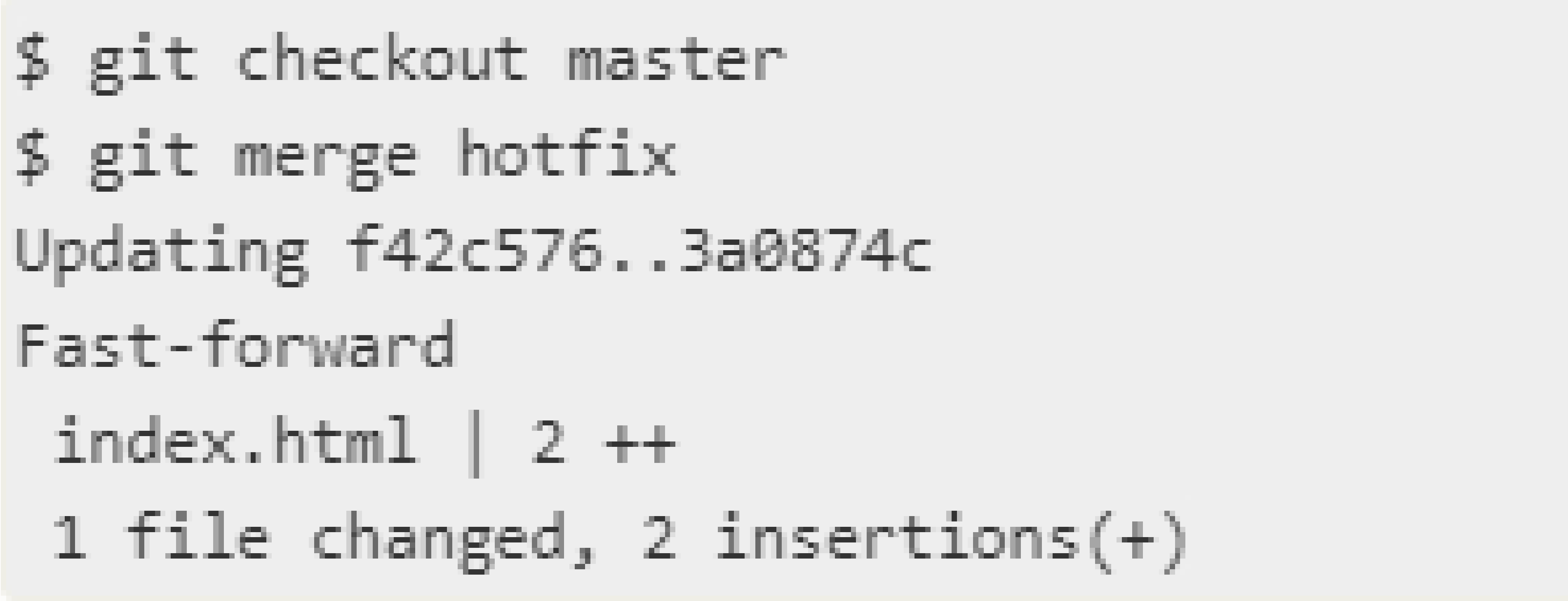
Hotfix branch based on master

You can run your tests, make sure the hotfix is what you want



Merge the Hotfix branch

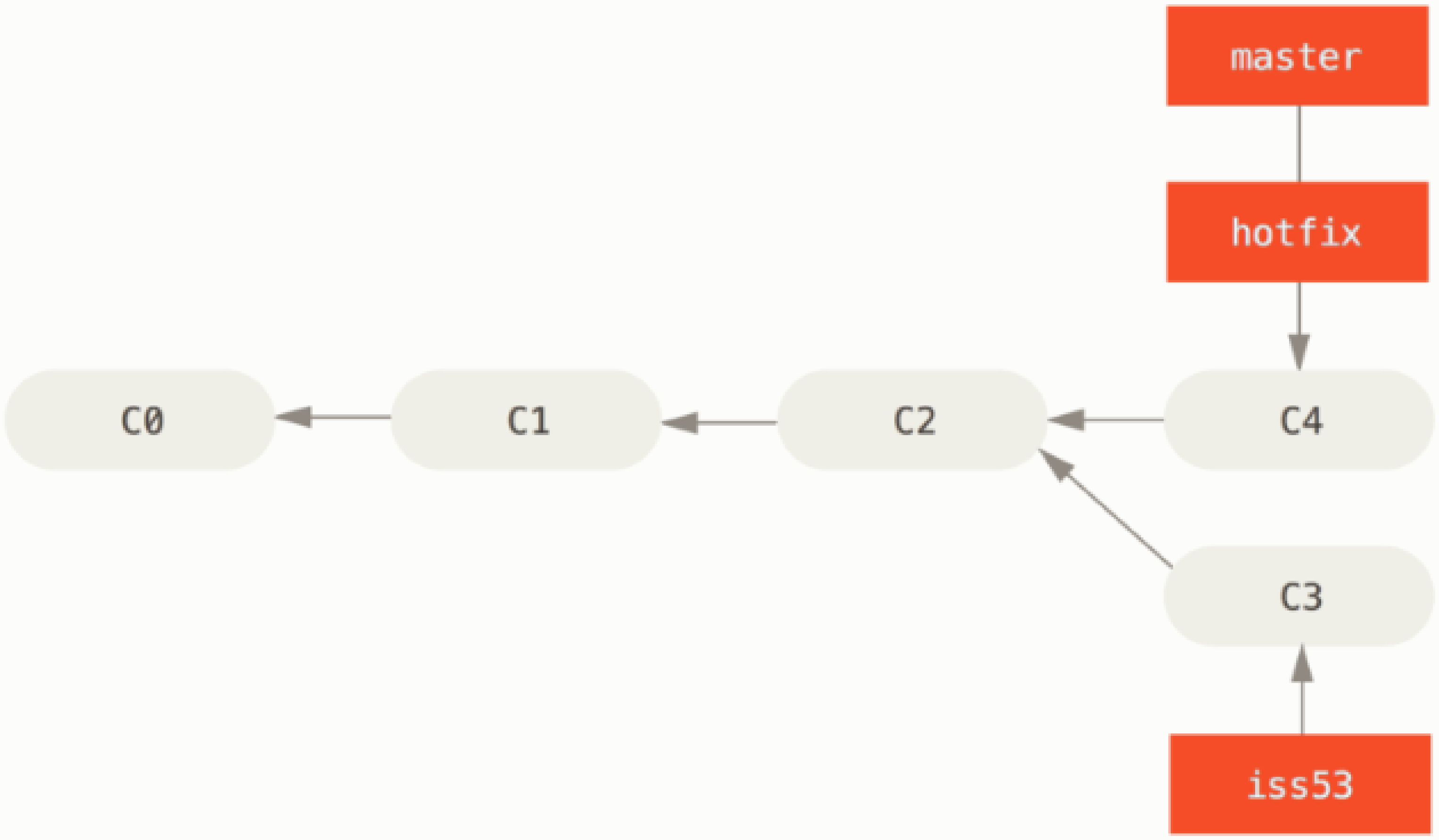
finally merge the hotfix branch back into your master branch to deploy to production. You do this with the git merge command



```
$ git checkout master
$ git merge hotfix
Updating f42c576..3a0874c
Fast-forward
 index.html | 2 ++
 1 file changed, 2 insertions(+)
```

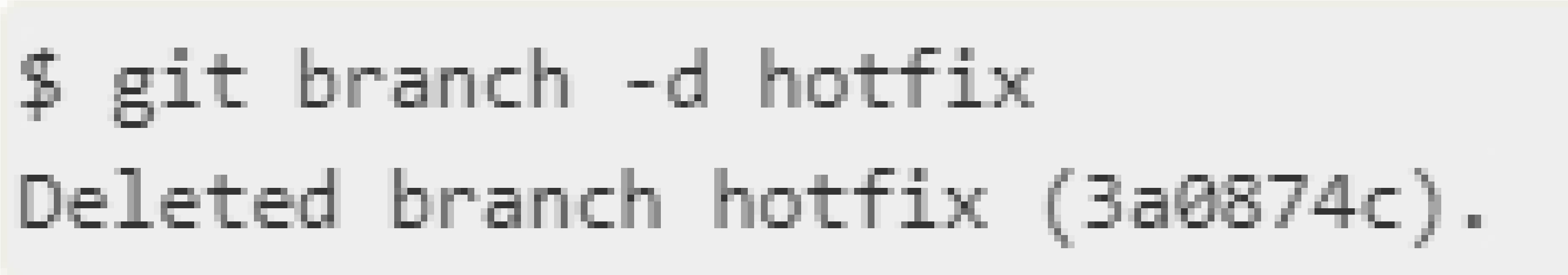
Ready for Deployment

Your change is now in the snapshot of the commit pointed to by the master branch, and you can deploy the fix.



Delete Hotfix branch

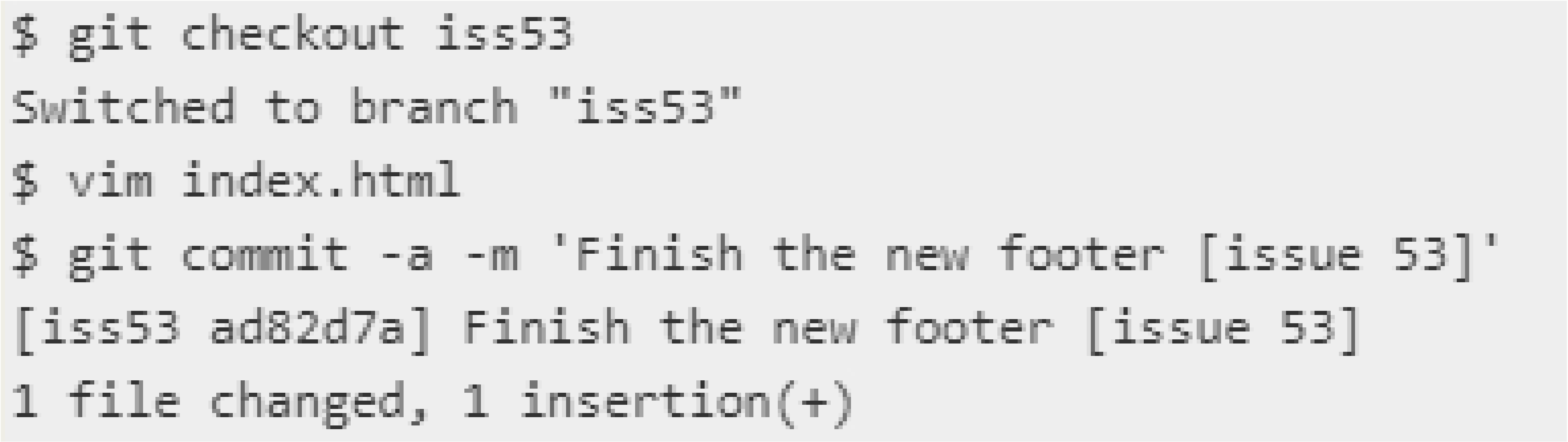
Delete the hotfix branch, because you no longer need it.



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

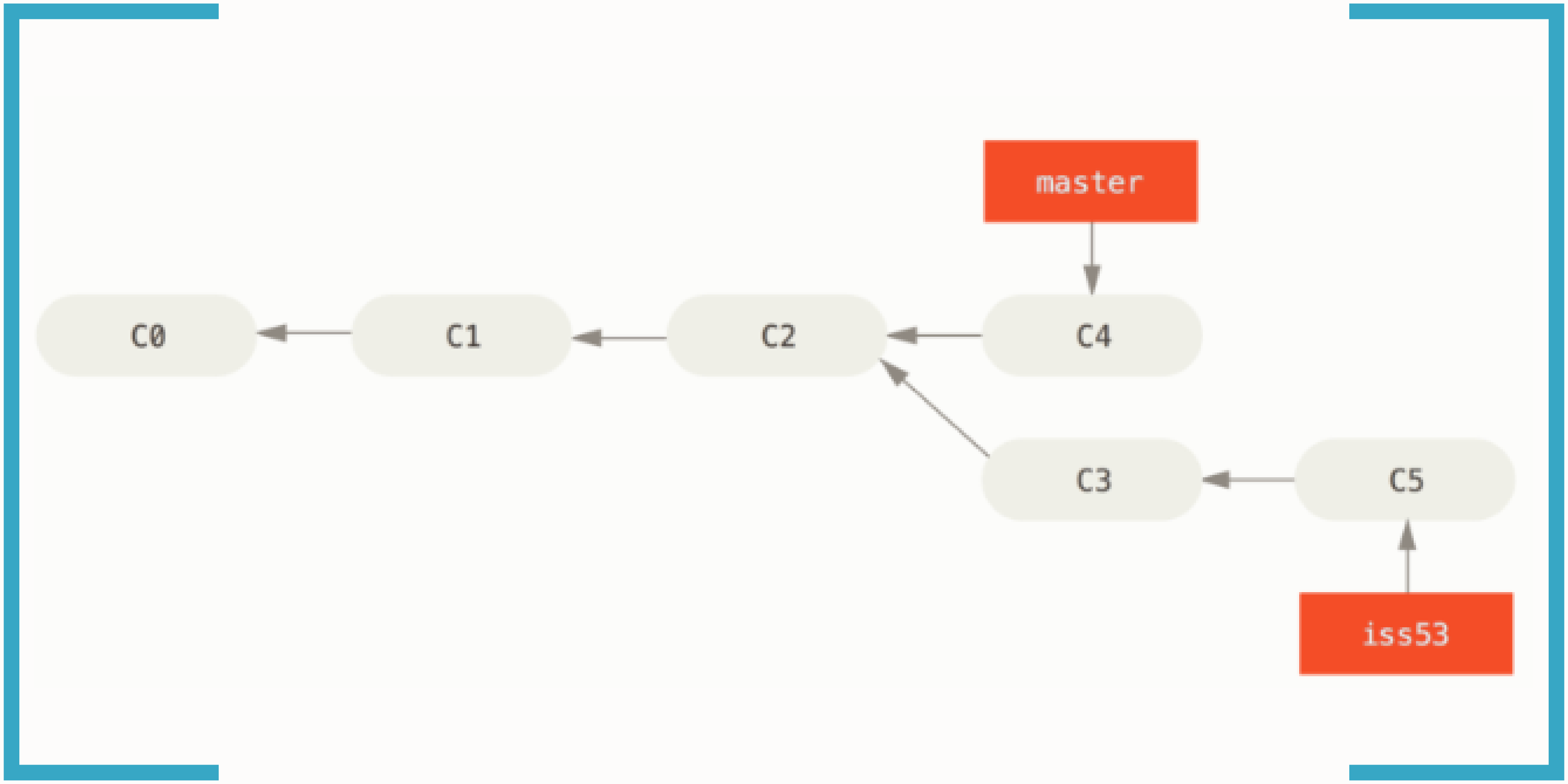
Switch to #53 branch

Now you can switch back to your work-in-progress branch on issue #53 and continue working on it.



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

Work continues on iss53



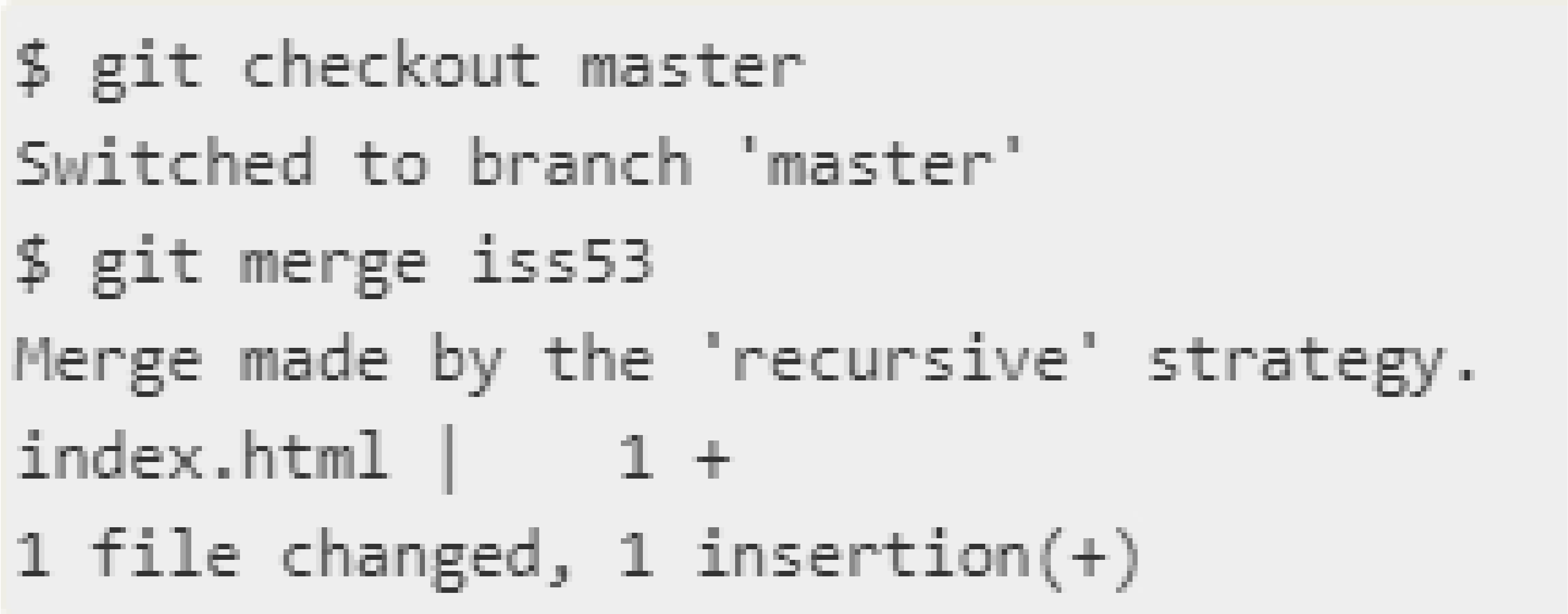
Work continues on iss53

It's worth noting here that the work you did in your hotfix branch is not contained in the files in your iss53 branch.

If you need to pull it in, you can merge your master branch into your iss53 branch by running git merge master, or you can wait to integrate those changes

Basic Merging

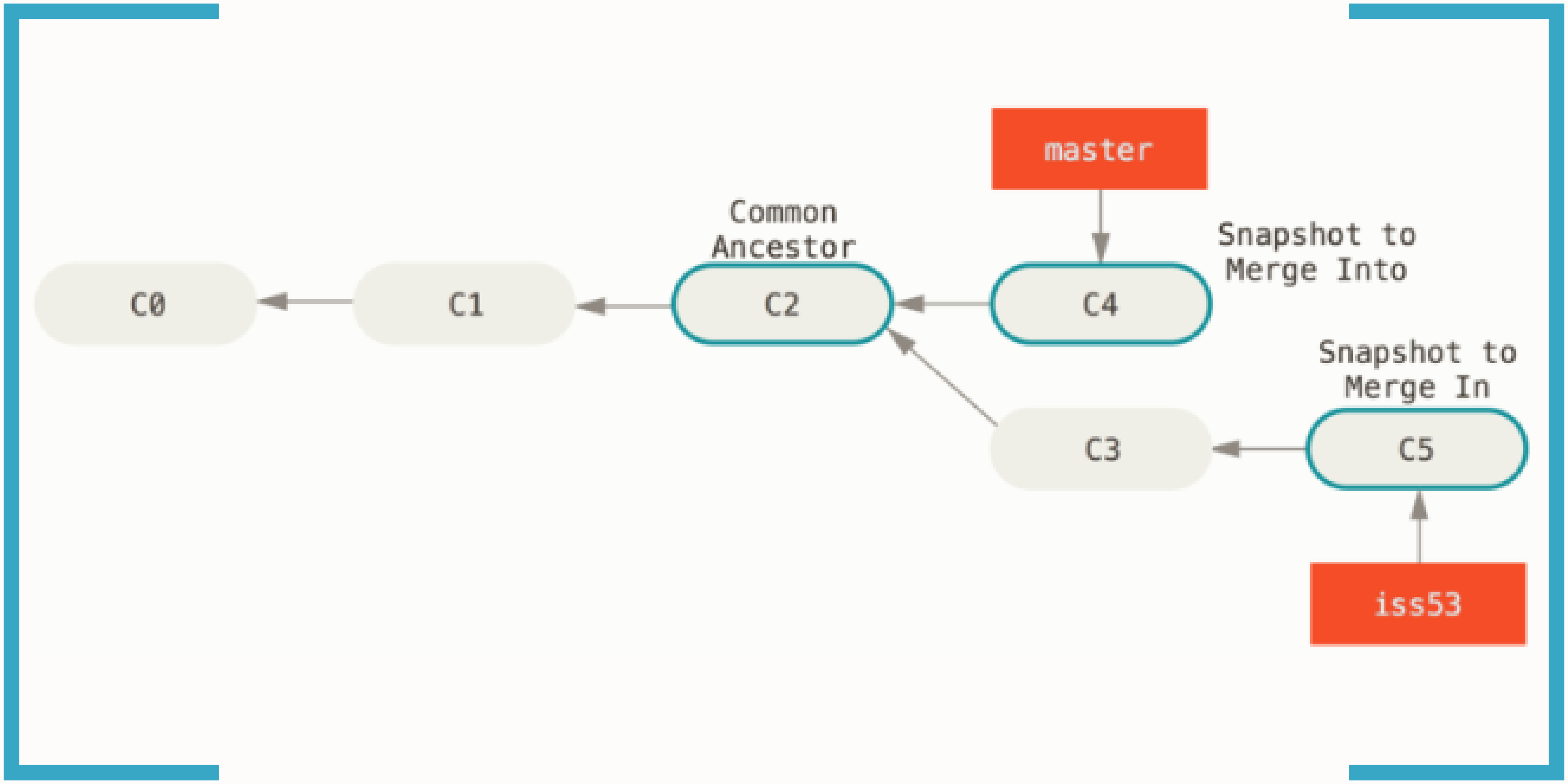
Suppose you've decided that your issue #53 work is complete and ready to be merged into your master branch



```
$ git checkout master
Switched to branch 'master'
$ git merge iss53
Merge made by the 'recursive' strategy.
index.html |    1 +
1 file changed, 1 insertion(+)
```

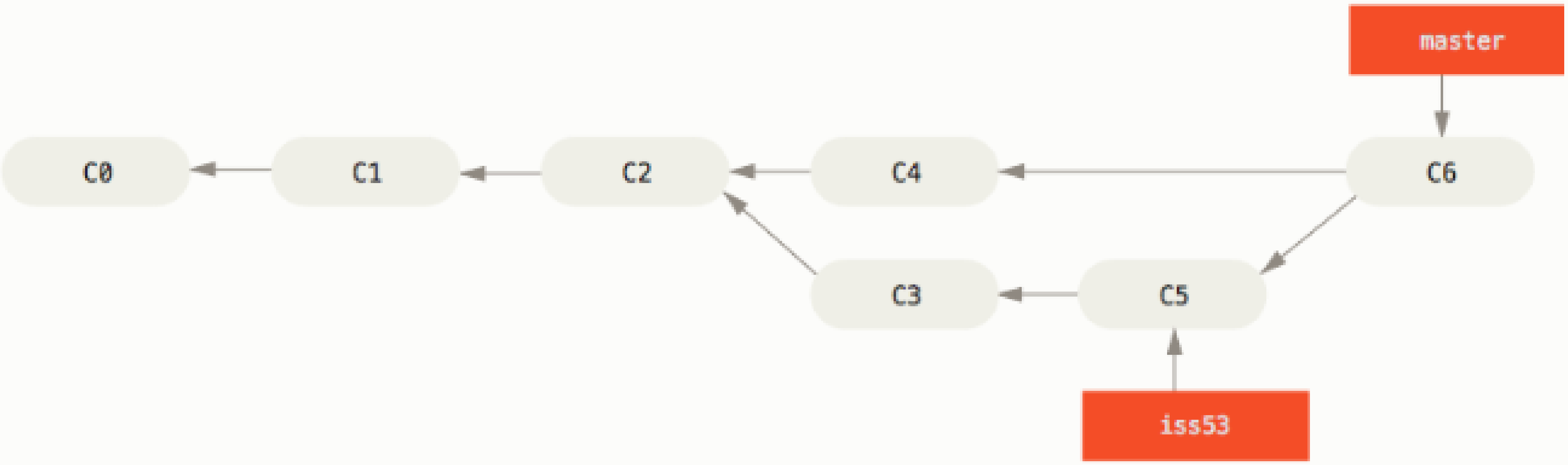
Three snapshots used in typical merge

Git does a simple three-way merge, using the two snapshots pointed to by the branch tips and the common ancestor of the two



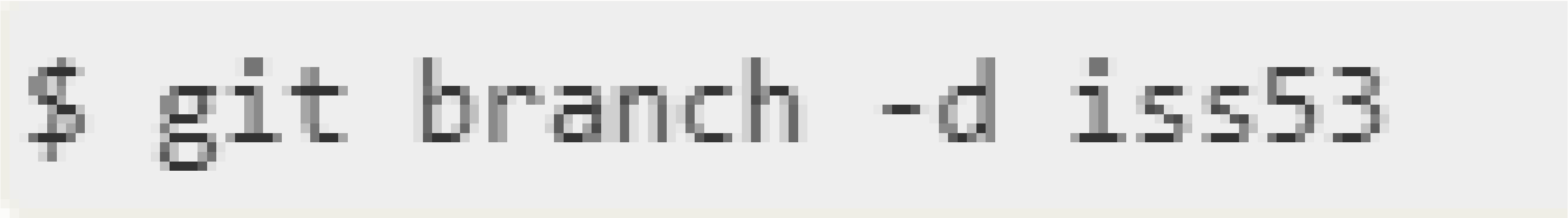
Merge Commit

Git creates a new snapshot that results from this three-way merge and automatically creates a new commit that points to it.



Delete iss53 branch

Now that your work is merged in, you have no further need for the iss53 branch. You can delete the branch:



```
$ git branch -d iss53
```