Git Branching

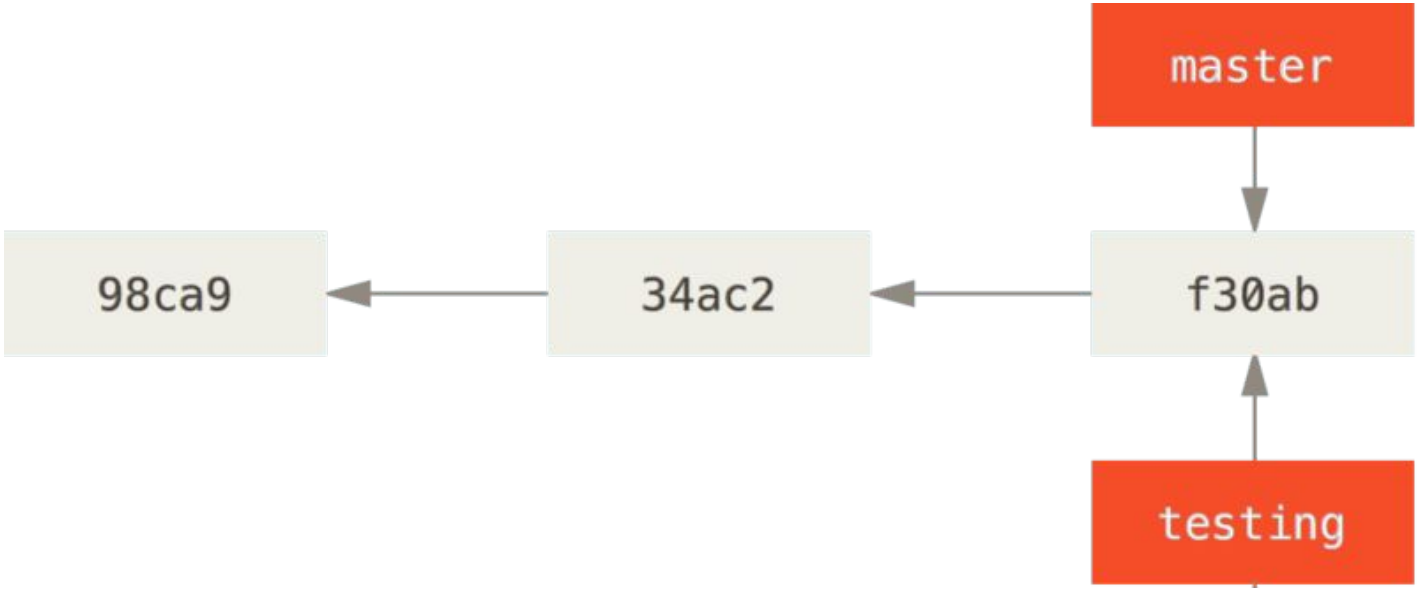
Outline

1. Branches in a Nutshell
2. Basic Branching and Merging
3. Branch Management
4. Branching Workflows
5. Remote Branches
6. Rebasing

1. Branches in a Nutshell

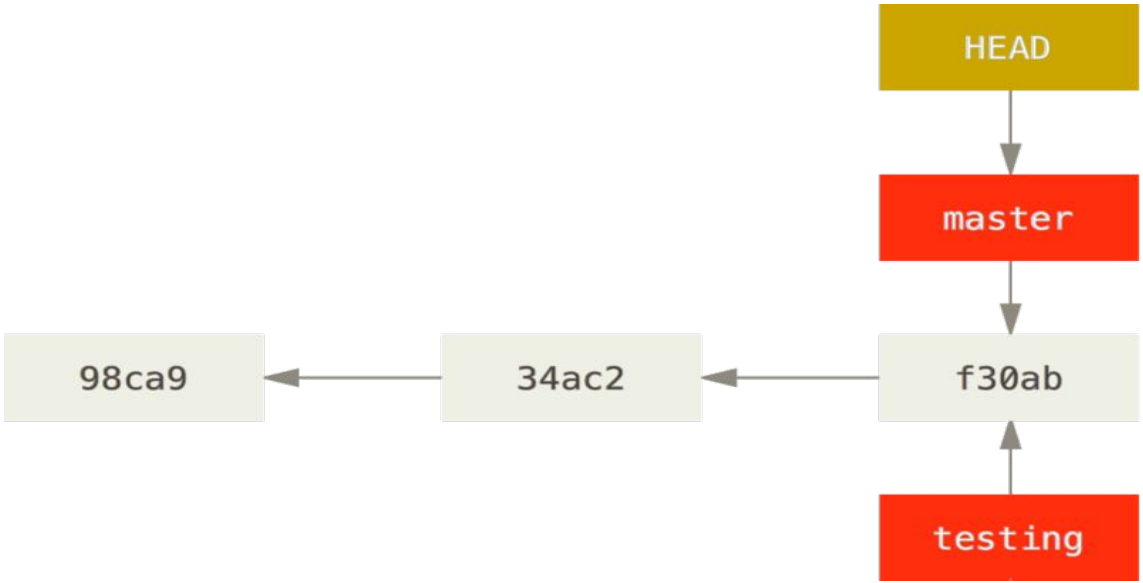
1.1 Creating a New branch

```
$ git branch testing
```



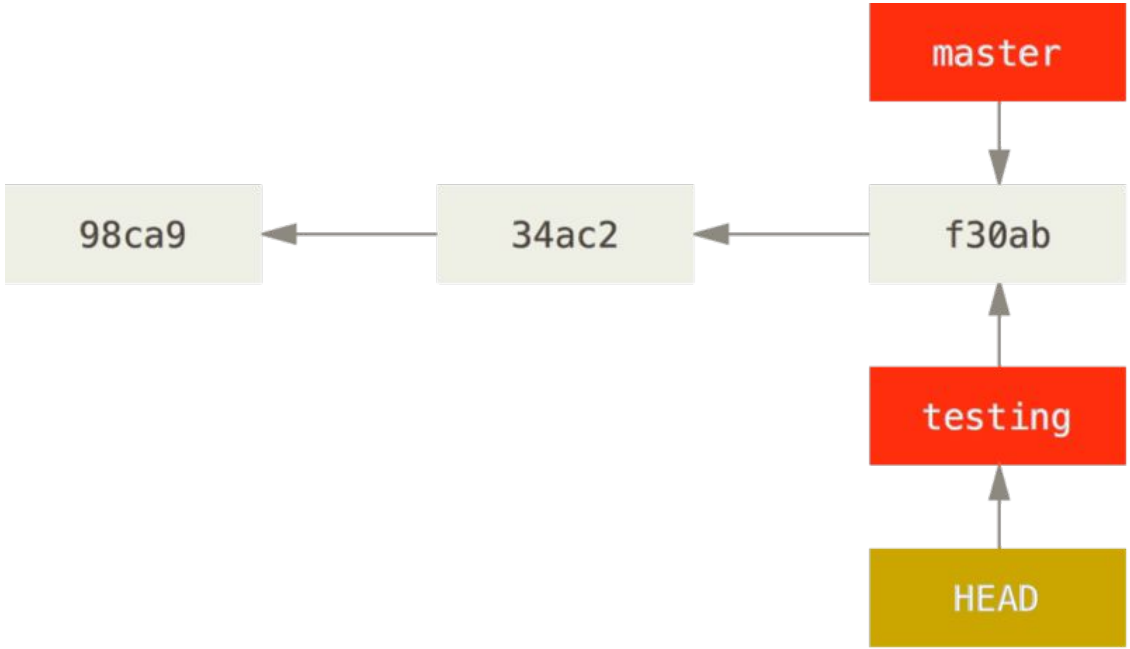
1.1 Creating a New branch

How does Git know what branch you're currently on?



1.2 Switching Branches

```
$ git checkout testing
```

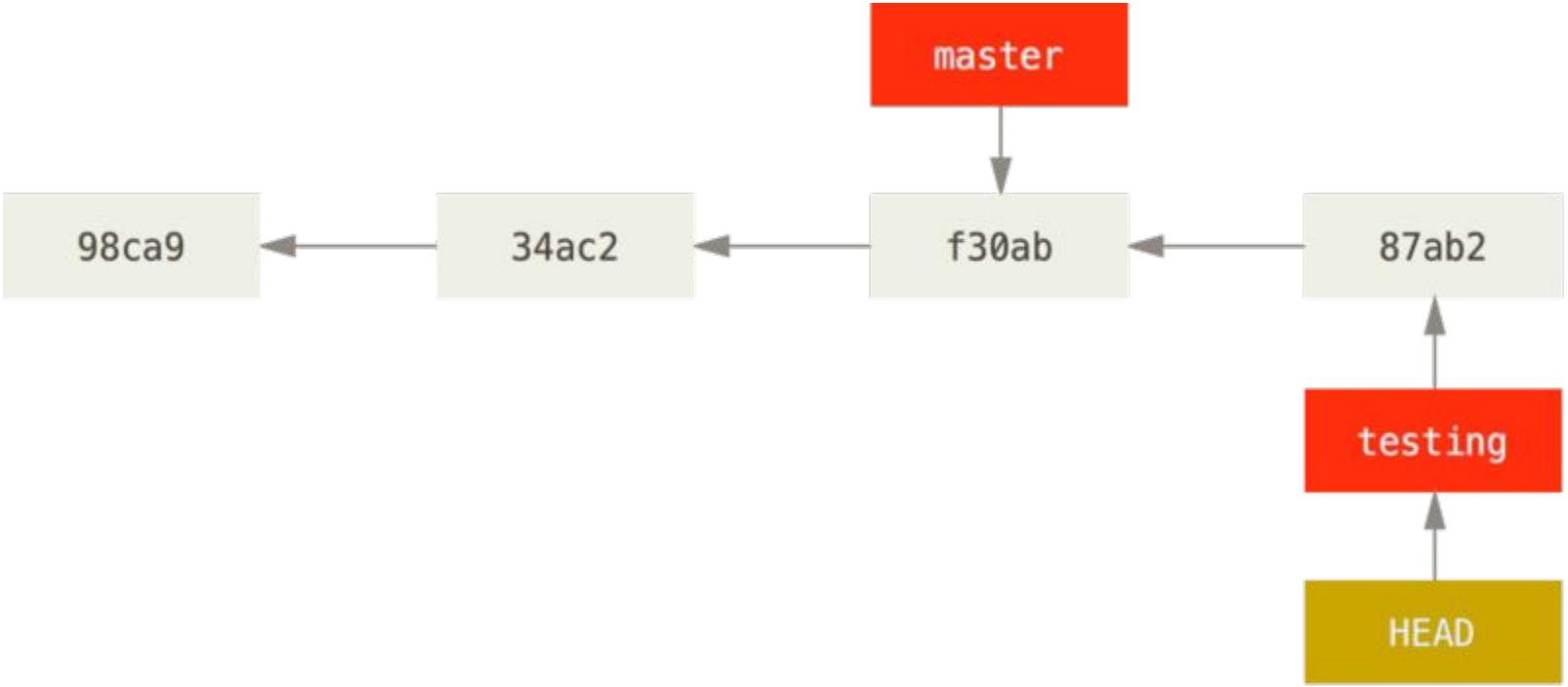


1.2 Switching Branches

What is the significance of that? Well, let's do another commit:

```
$ git commit -a -m 'made a change'
```

1.2 Switching Branches

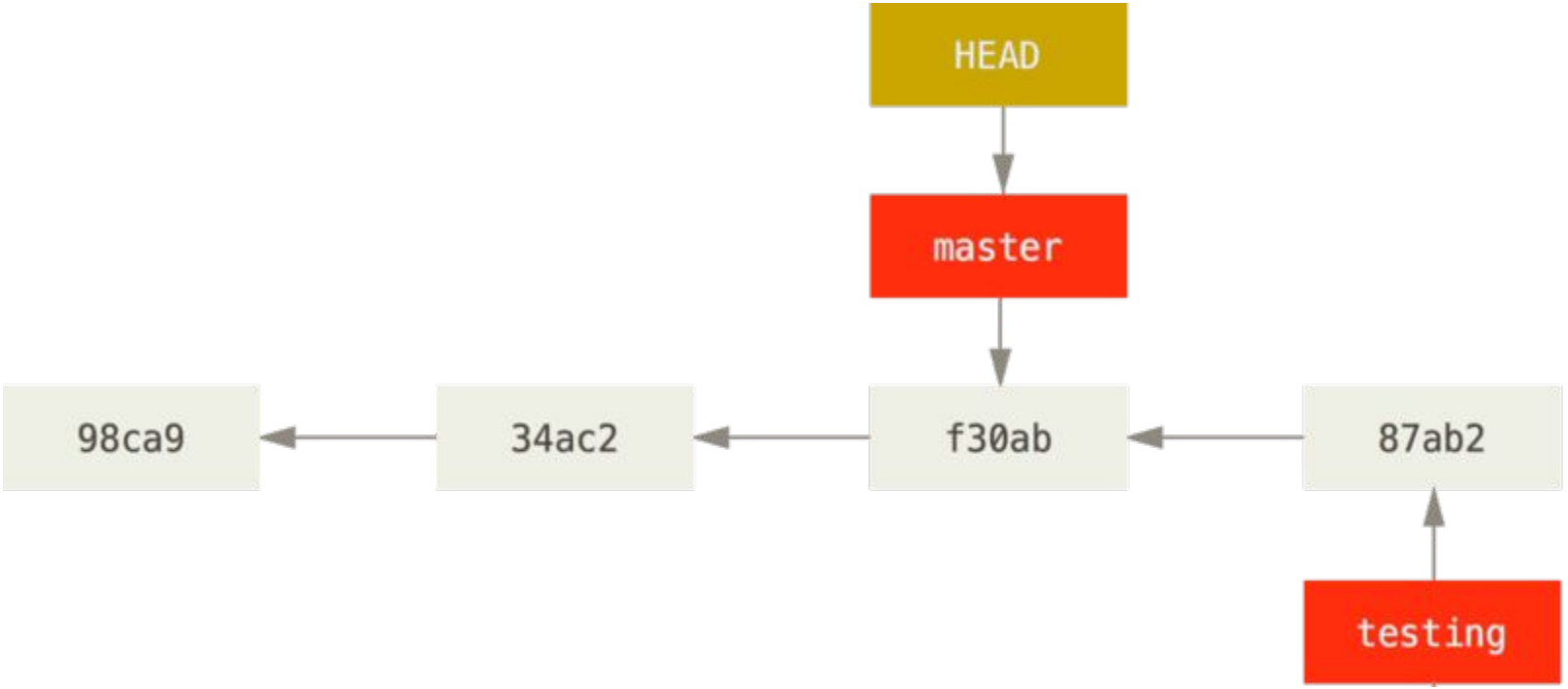


1.2 Switching Branches

Let’s switch back to the master branch:

```
$ git checkout master
```

1.2 Switching Branches

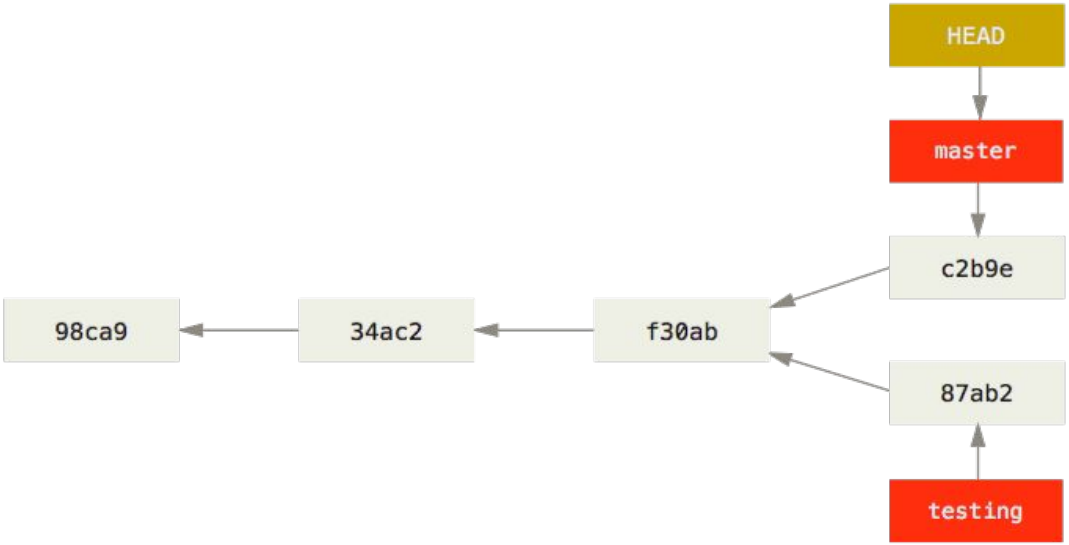


1.2 Switching Branches

Let’s make a few changes and commit again:

```
$ git commit -a -m 'made other changes'
```

1.2 Switching Branches



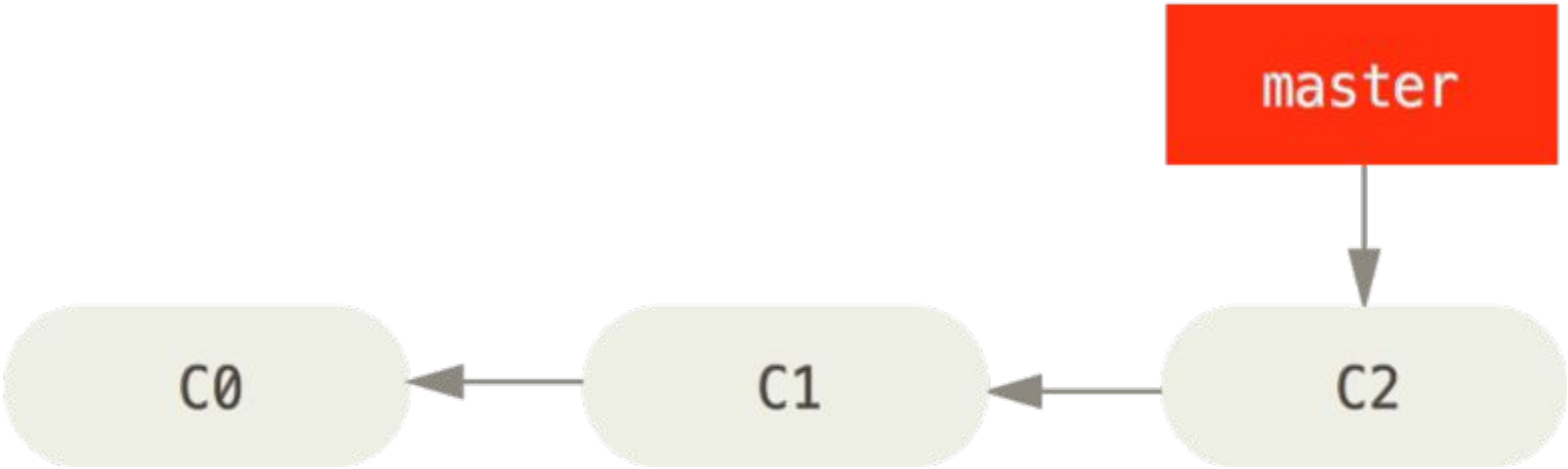
1.3 Summary

- A simple file that contains the 40 character SHA-1 checksum of the commit it points to.
- Branches are cheap to create and destroy.
- Creating a new branch is as quick and simple as writing 41 bytes to a file

2. Basic Branching and Merging

- Basic Branching
- Basic Merging

2.1 Basic Branching



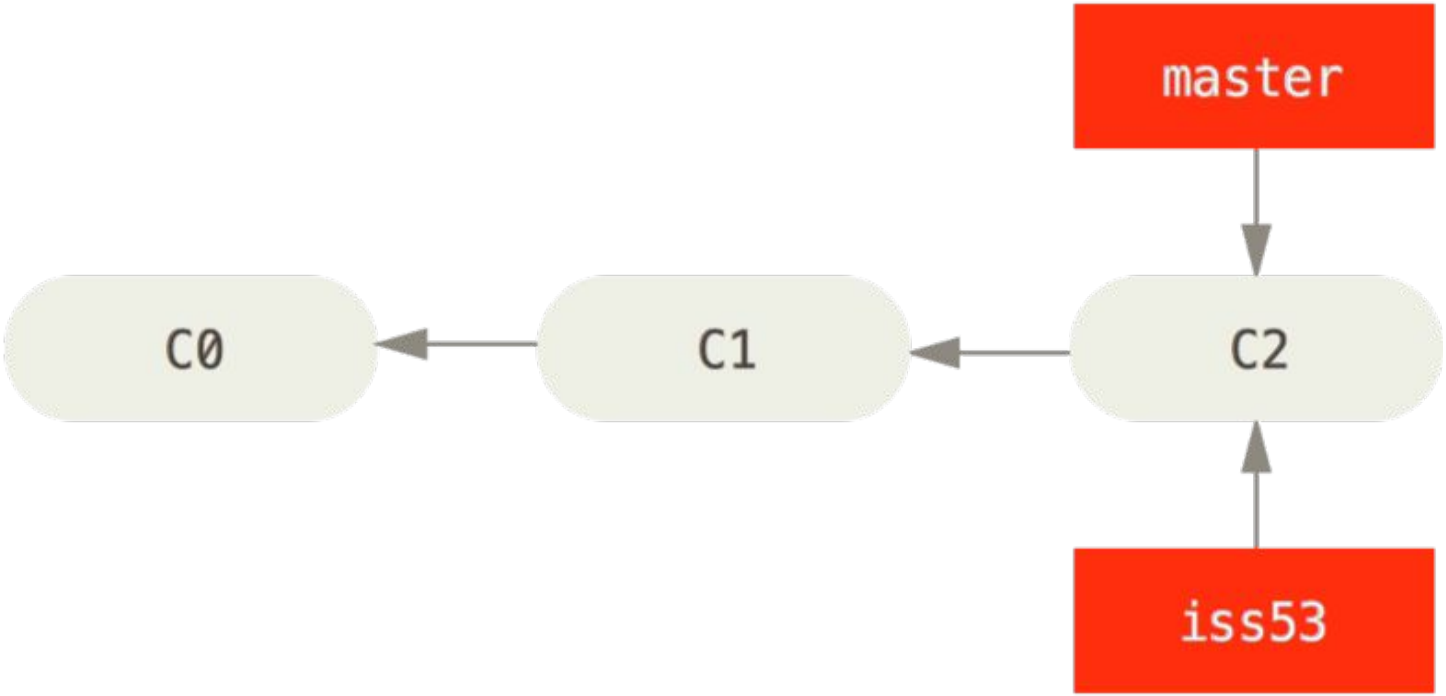
2.1 Basic Branching

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

Or

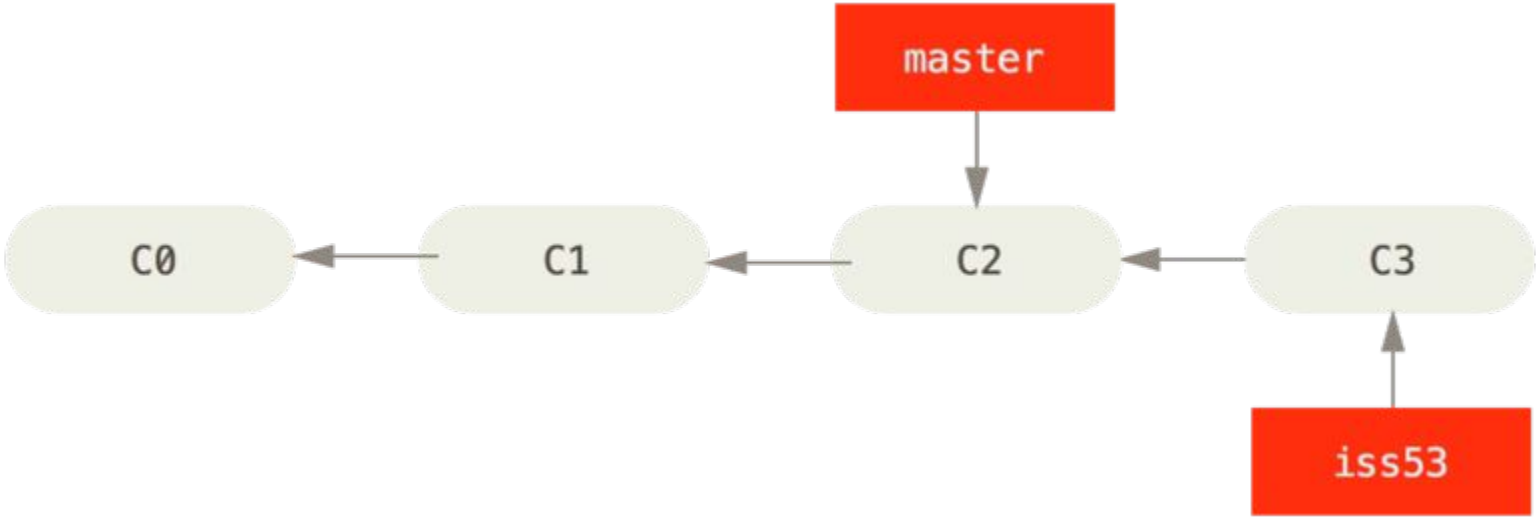
```
$ git branch iss53
$ git checkout iss53
```

2.1 Basic Branching



2.1 Basic Branching

```
$ git commit -a -m 'added a new footer [issue 53]'
```



2.1 Basic Branching

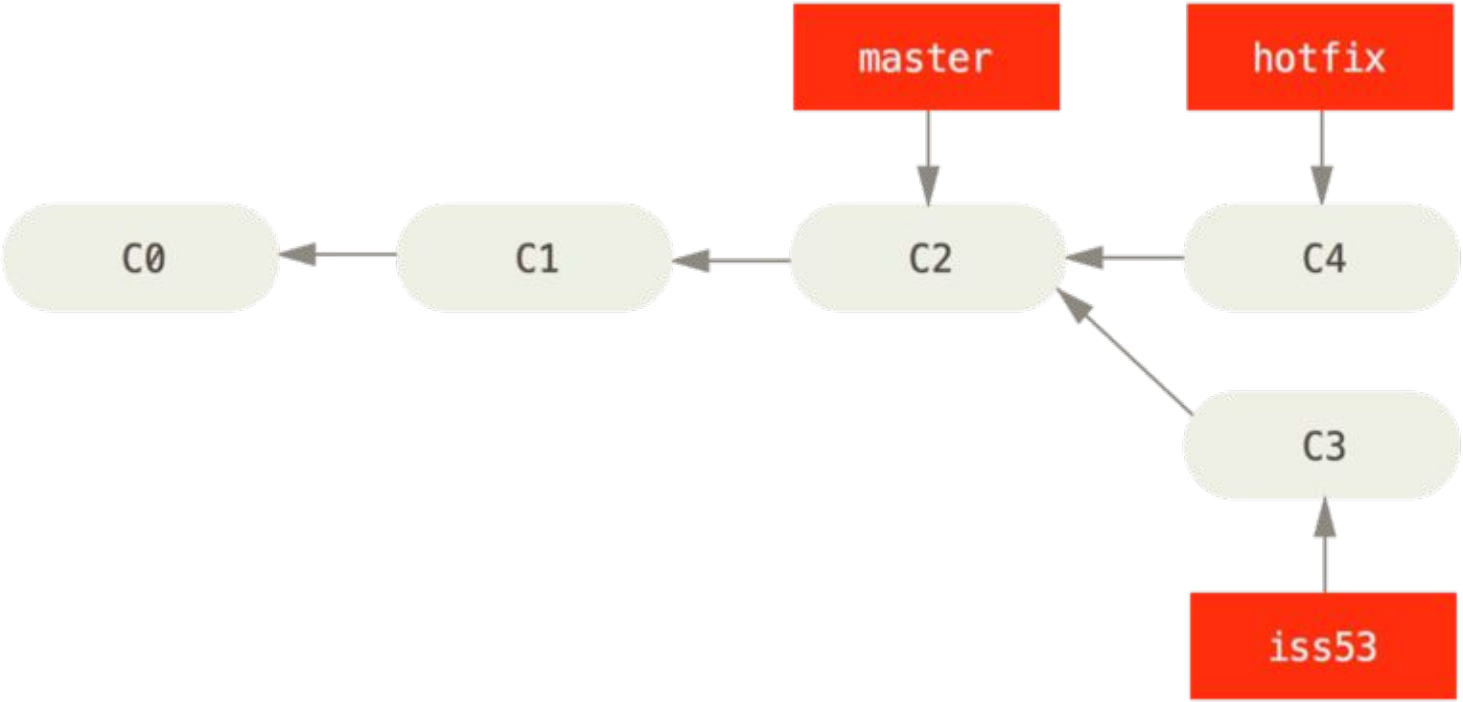
```
$ git checkout master
Switched to branch 'master'
```

2.1 Basic Branching

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

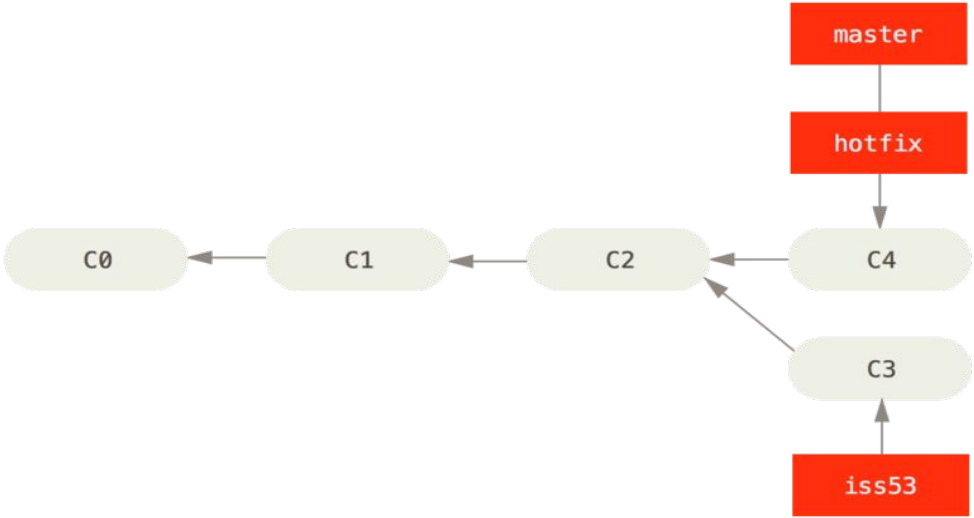
```
$ git commit -a -m 'fixed the broken email address'
```

2.1 Basic Branching



2.1 Basic Branching

```
$ git checkout master
$ git merge hotfix
```

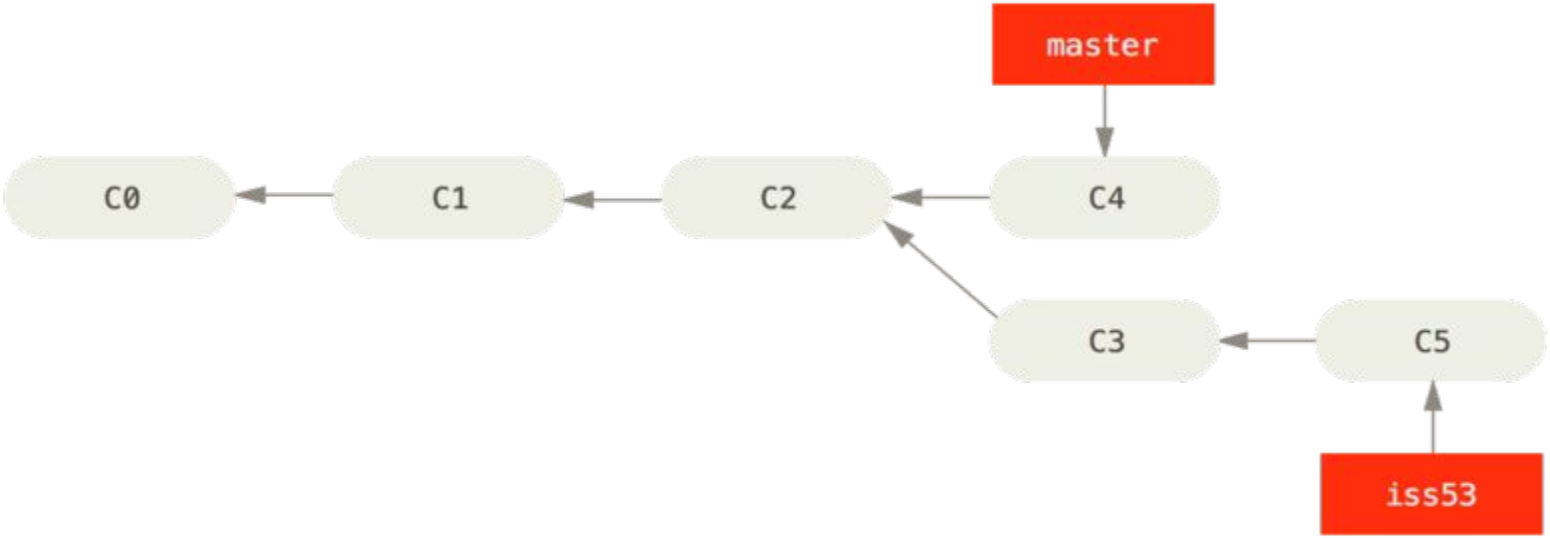


2.1 Basic Branching

```
$ git branch -d hotfix
```

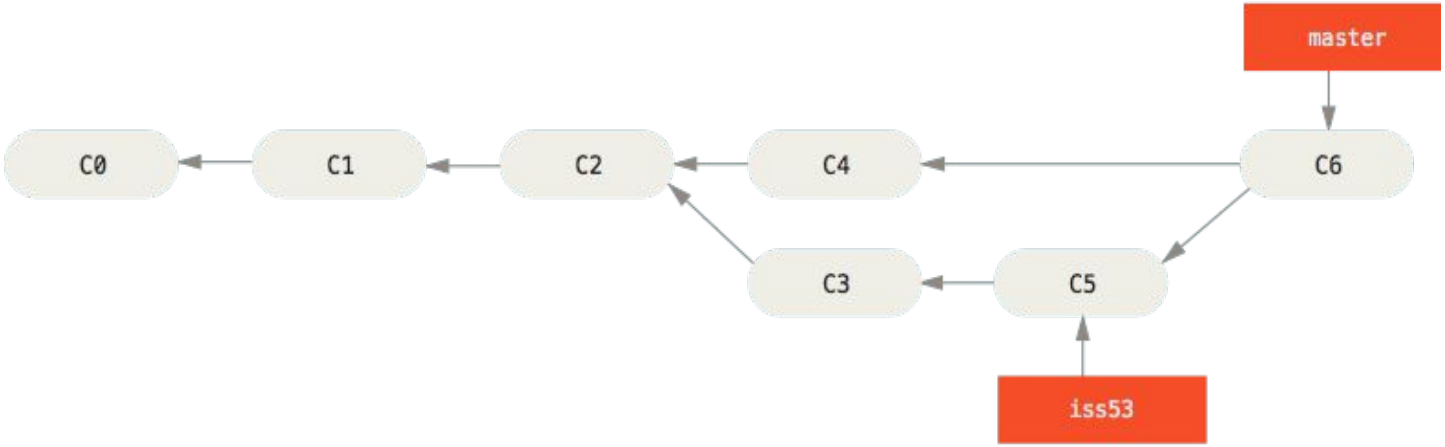
```
$ git branch -d hotfix
$ git checkout iss53
$ git commit -a -m 'finished the new footer [issue 53]'
```

2.1 Basic Branching



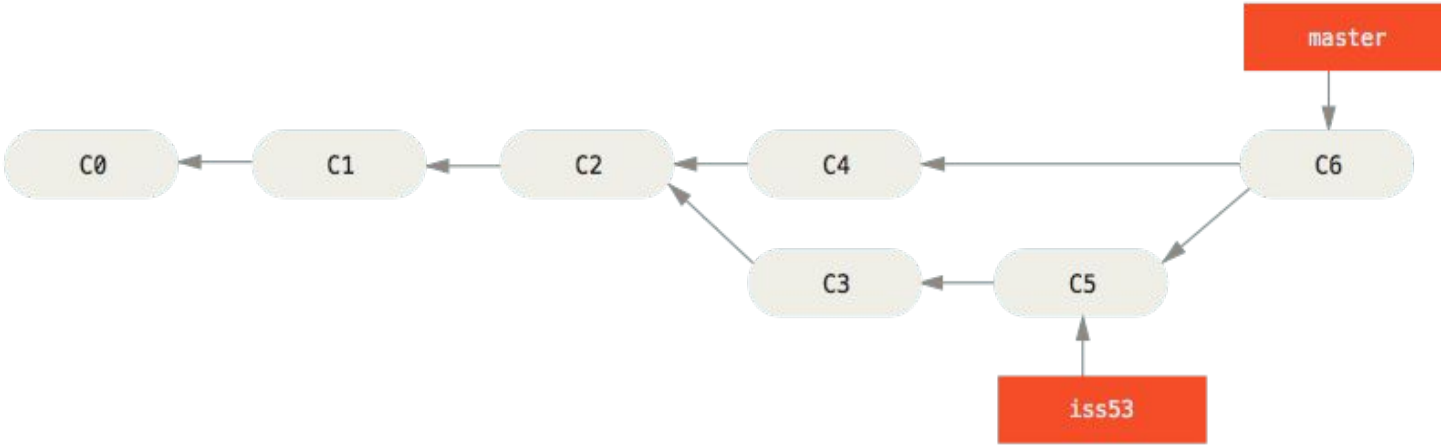
2.2 Basic Merging

```
$ git checkout master
$ git merge iss53
```



2.2 Basic Merging

```
$ git checkout master
$ git merge iss53
```



3.Branch Management

```
$ git branch
$ git branch -v
$ git branch --merged
$ git branch --no-merged
$ git branch -d [name branch]
```

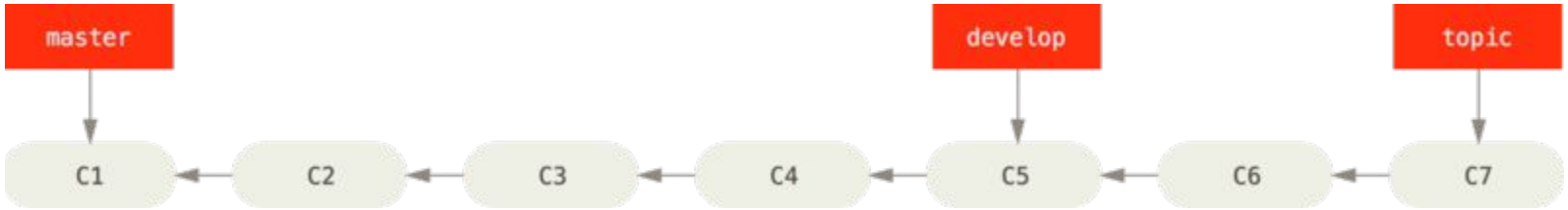
4.Branching Workflows

- Long-Running Branches
- Topic Branches

4.1 Long-Running Branches

```
$ git branch develop
$ git checkout -b topic
```

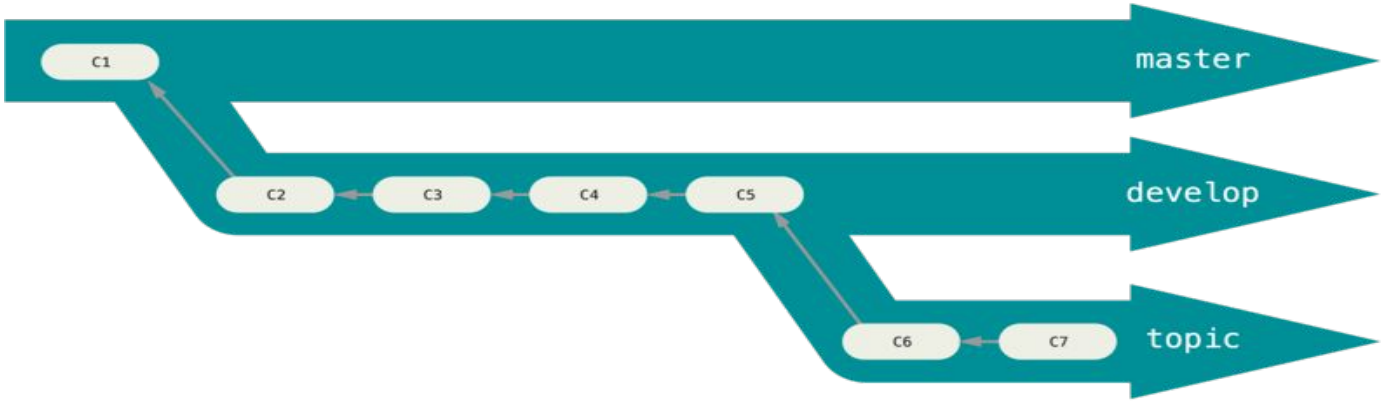
A linear view of progressive-stability branching



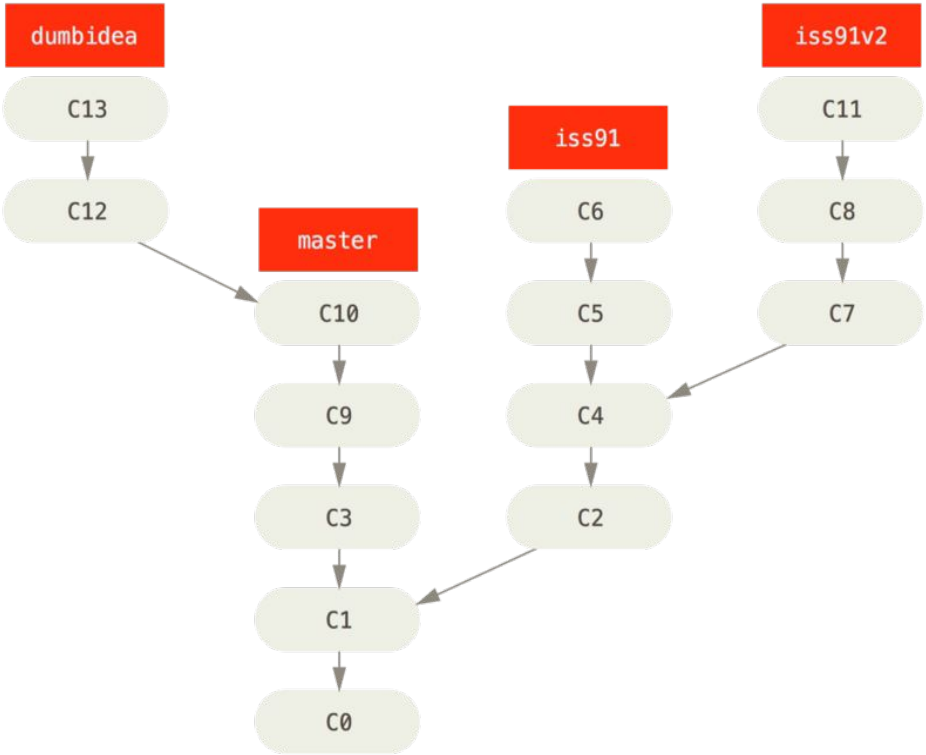
4.1 Long-Running Branches

```
$ git branch develop
$ git checkout -b topic
```

A “silo” view of progressive-stability branching

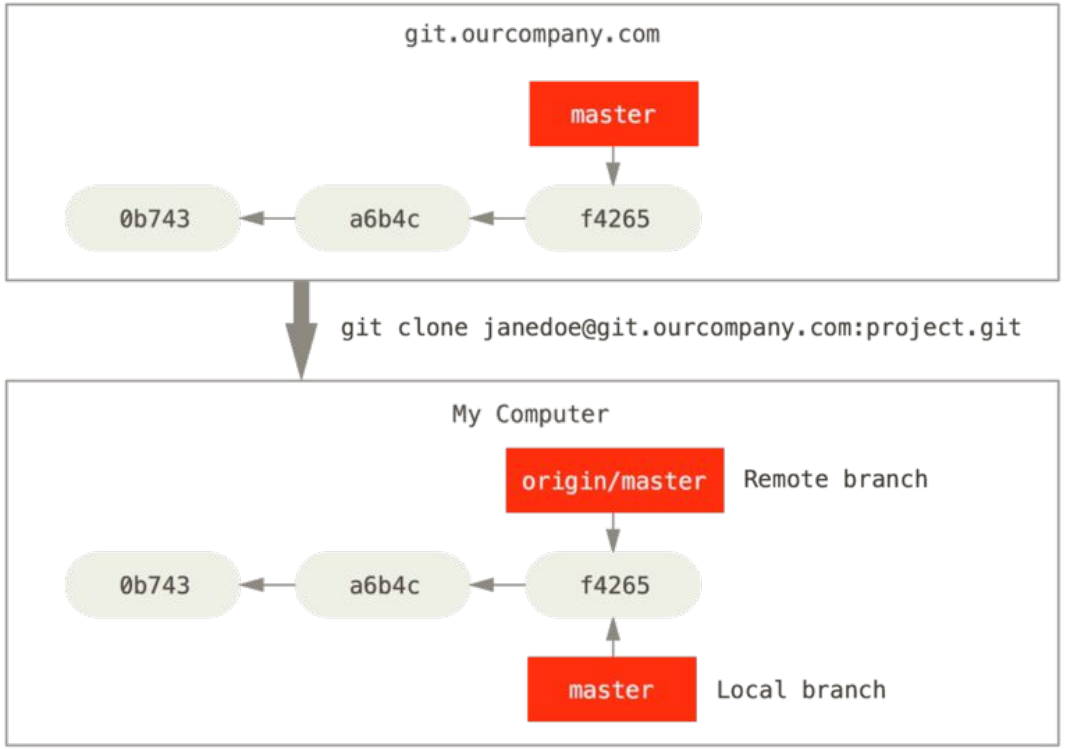


4.2 Topic Branches



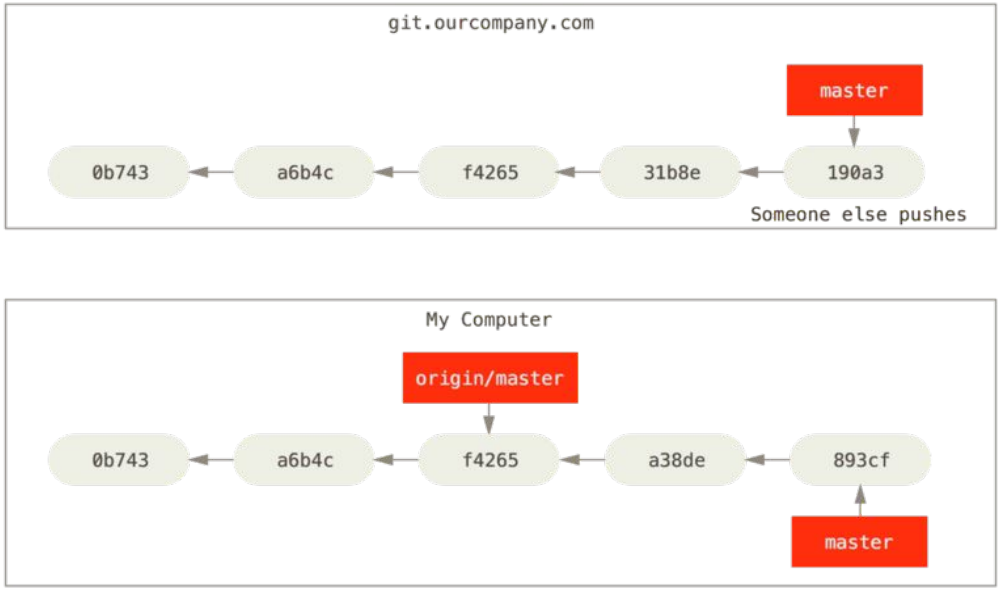
5.Remote Branches

Git clone



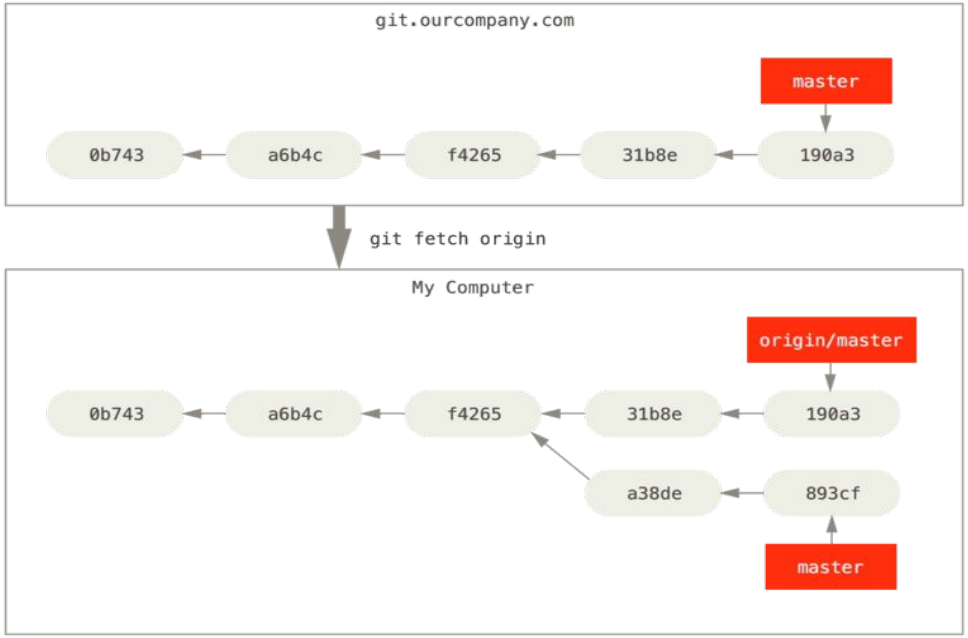
5.Remote Branches

Local and remote work can diverge

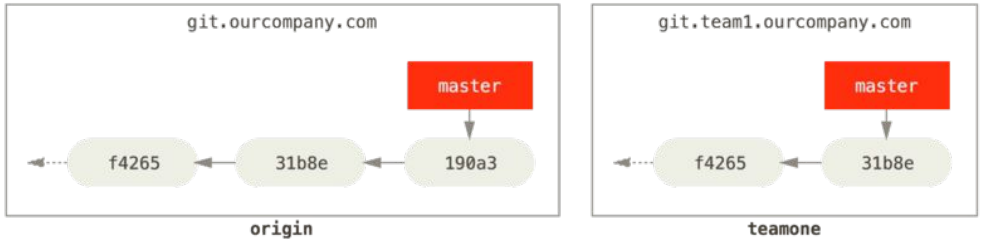


5.Remote Branches

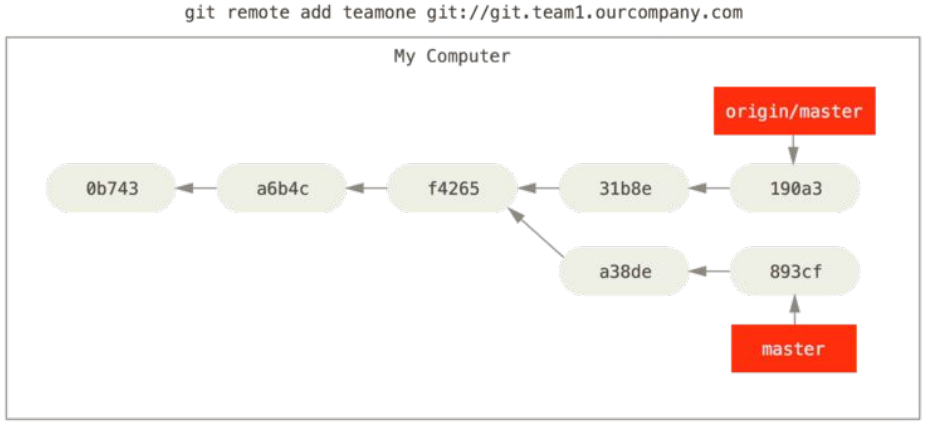
git fetch updates your remote references



5.Remote Branches

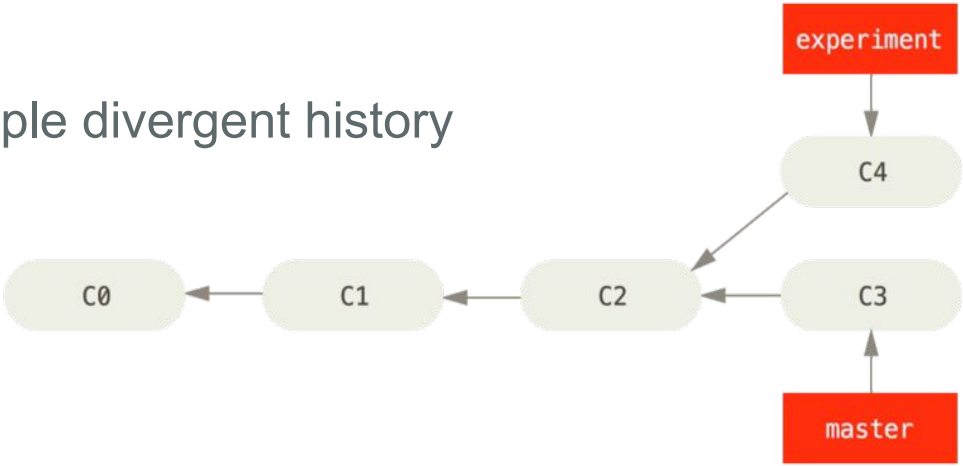


Add remote



6.Rebasing

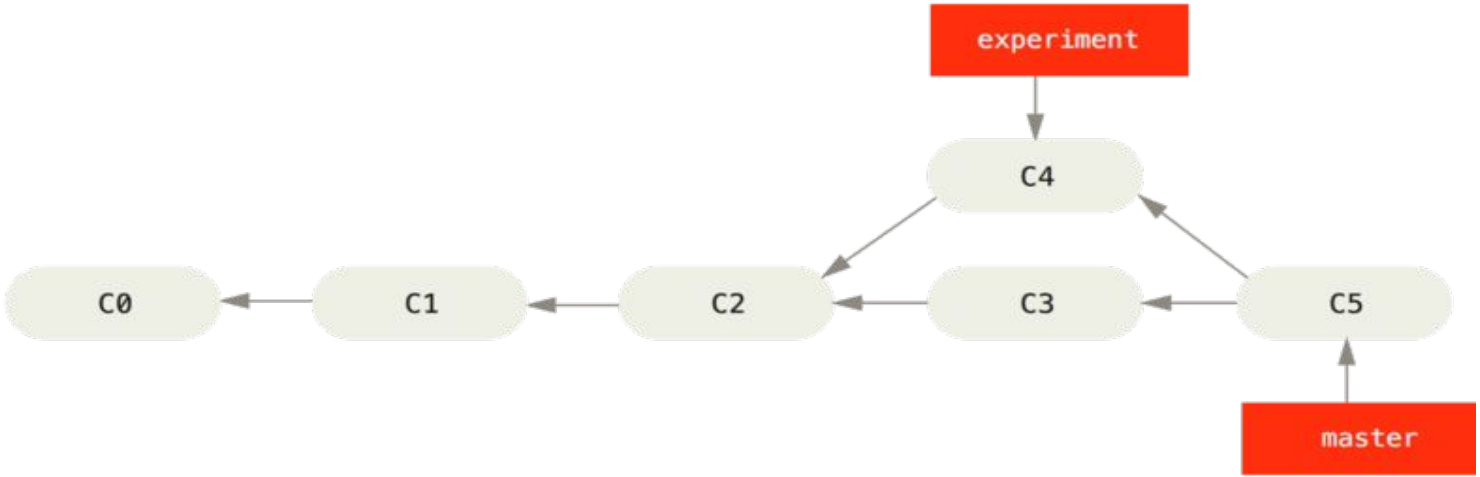
Simple divergent history



6.Rebasing

Merging to integrate diverged work history

```
$ git checkout master
$ git merge experiment
```



6.Rebasing

Rebasing the change introduced in C4 onto C3

```
$ git checkout experiment
$ git rebase master
```