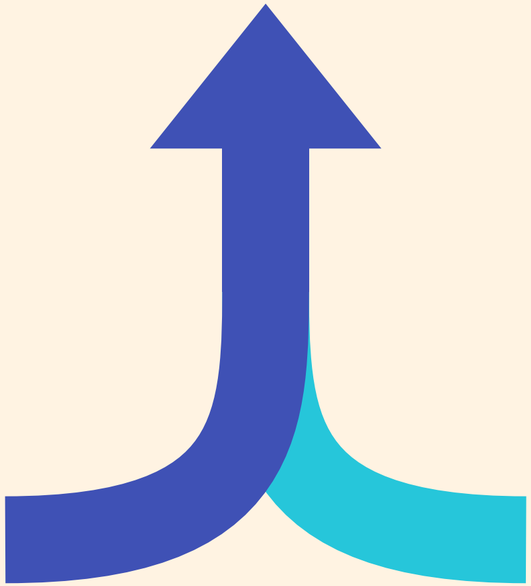
Merging Branches

Merging

Branching makes it super easy to work within self-contained contexts, but often we want to incorporate changes from one branch into another!

We can do this using the **git merge** command



Merging

The merge command can sometimes confuse students early on. Remember these two merging concepts:

- **We merge branches, not specific commits**
- **We always merge to the current HEAD branch**

Merging Made Easy

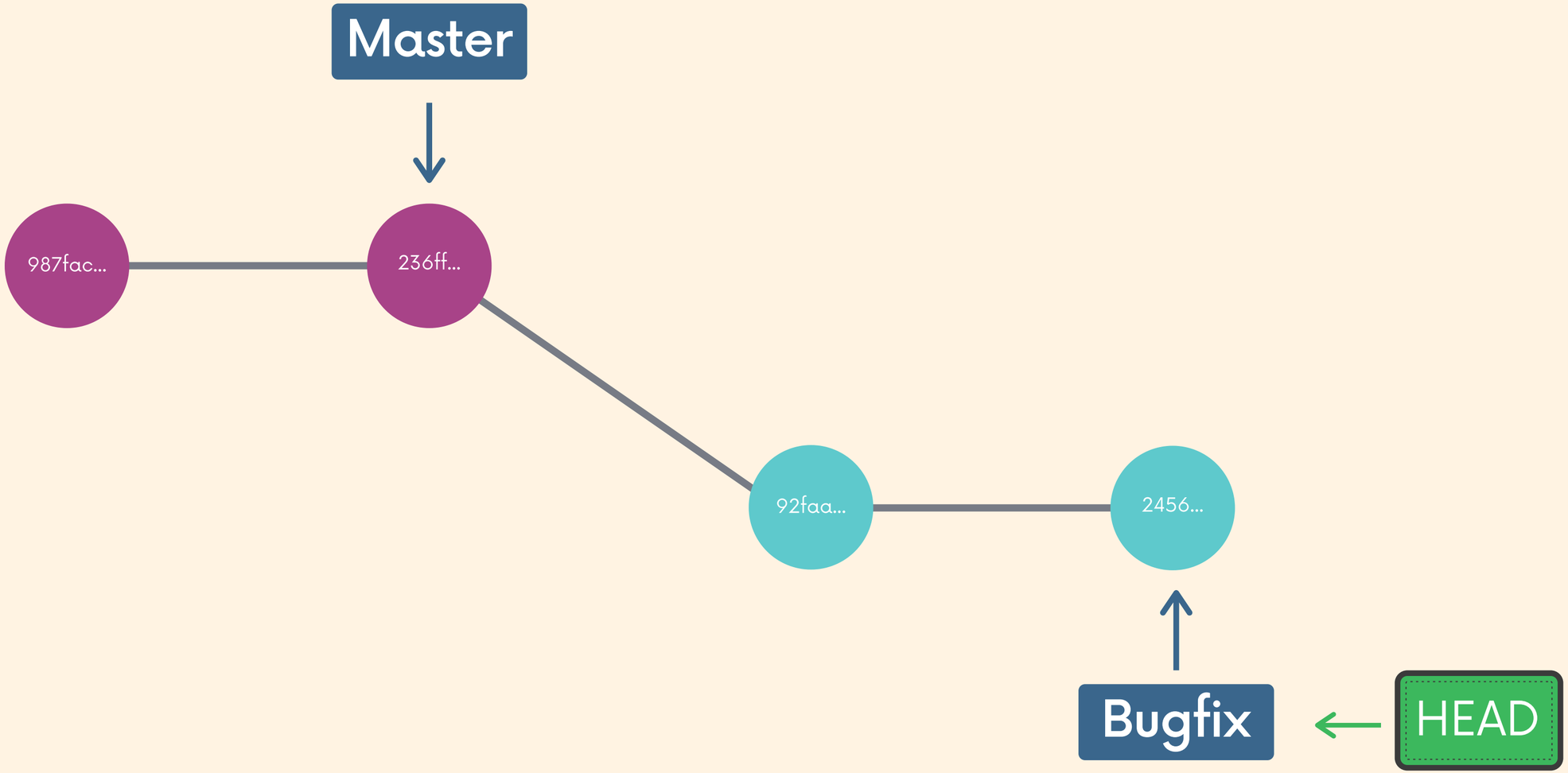
To merge, follow these basic steps:

1. Switch to or checkout the branch you want to merge the changes into (the receiving branch)
2. Use the **git merge** command to merge changes from a specific branch into the current branch.

To merge the bugfix branch into master...

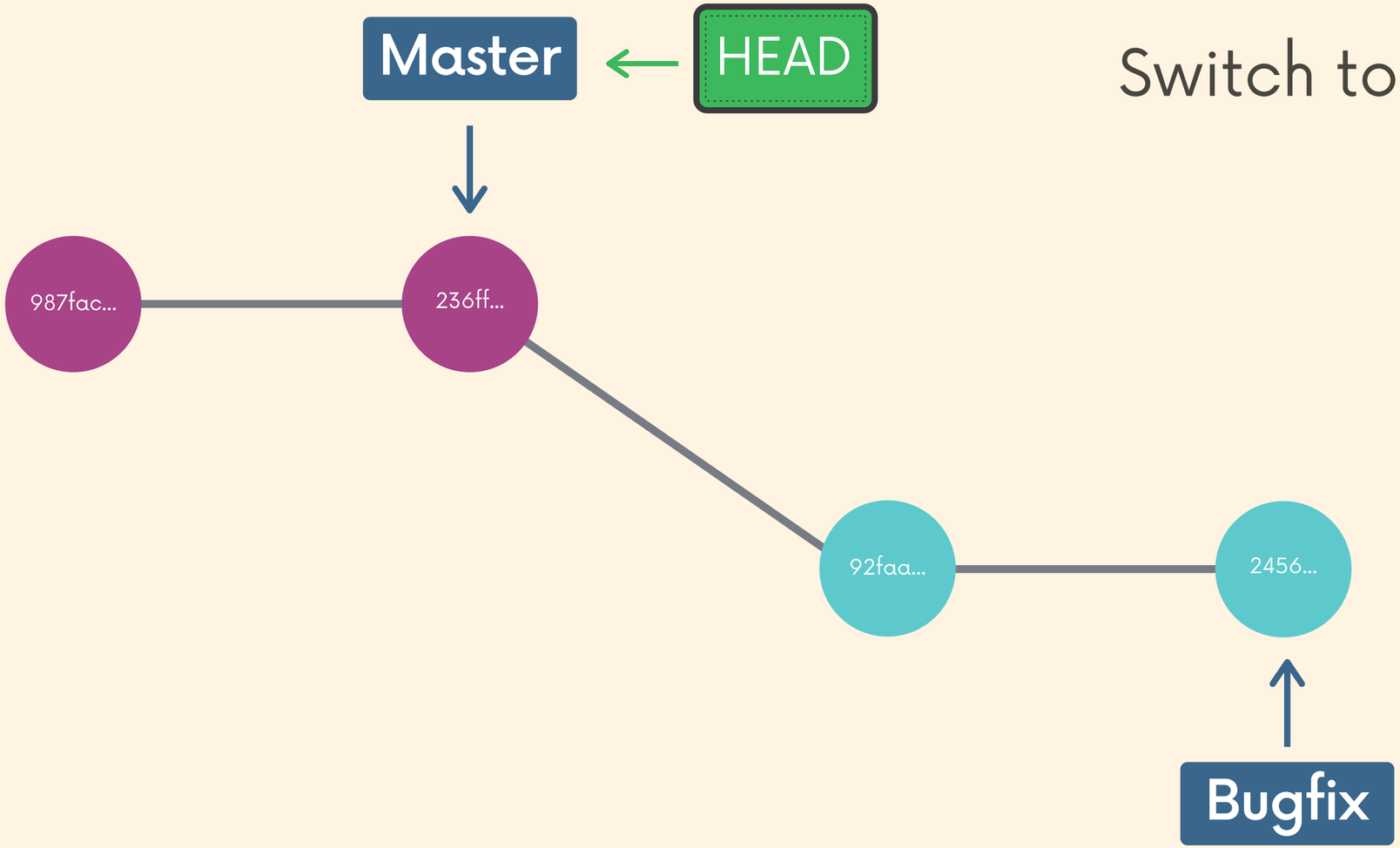
```
❯git switch master
❯git merge bugfix
```

To merge the bugfix branch into the master branch...



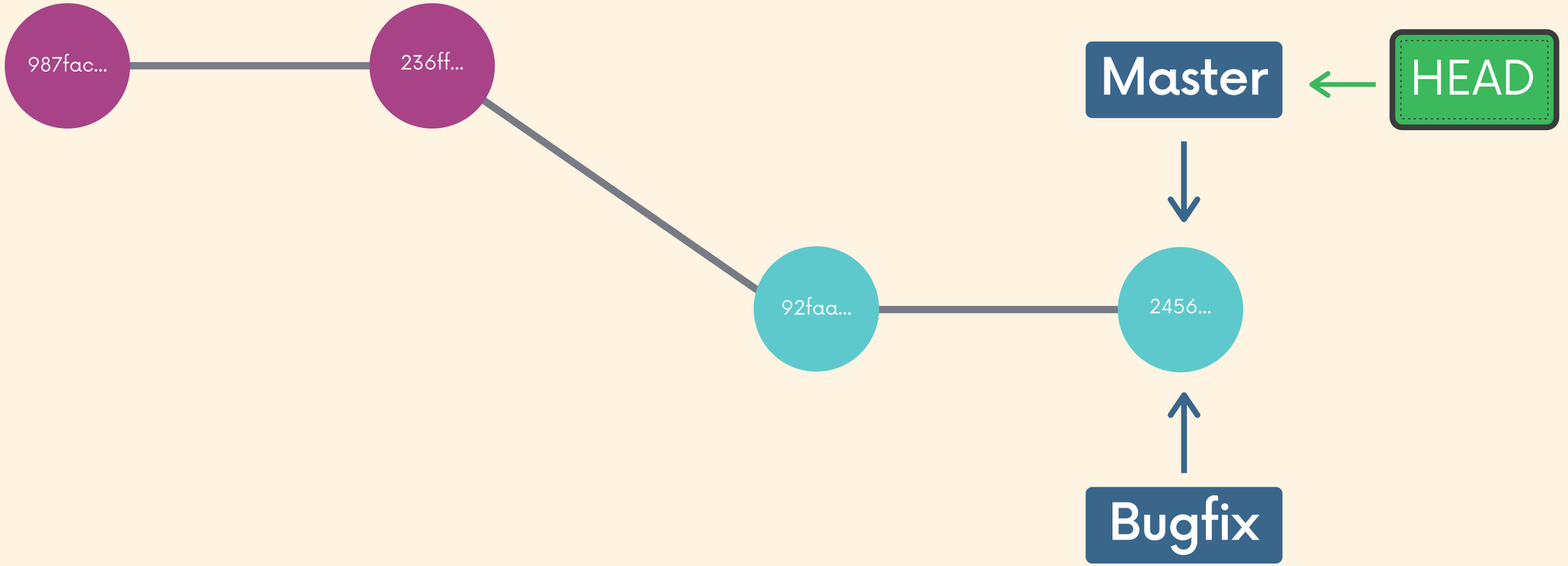
```
❯ git switch master
```

Switch to the master branch



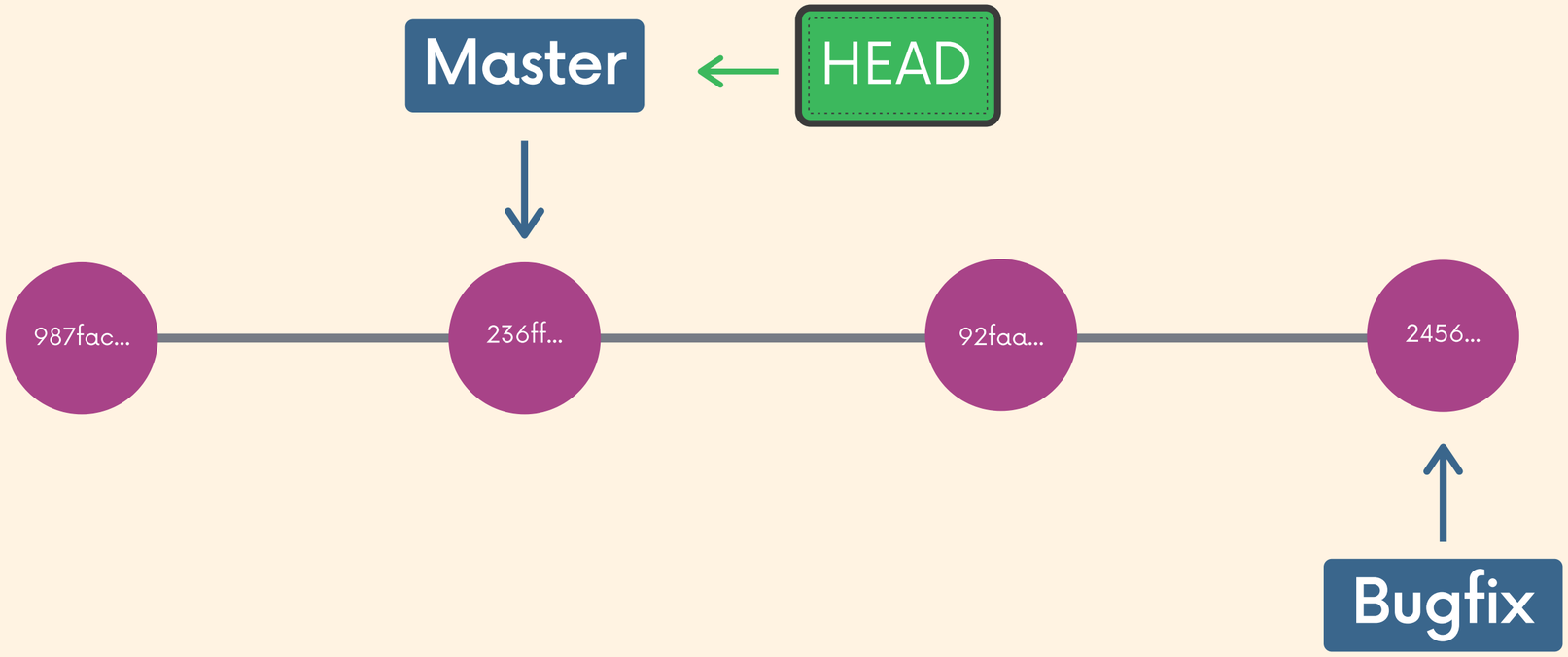
```
❯ git merge bugfix
```

Merge bugfix into master



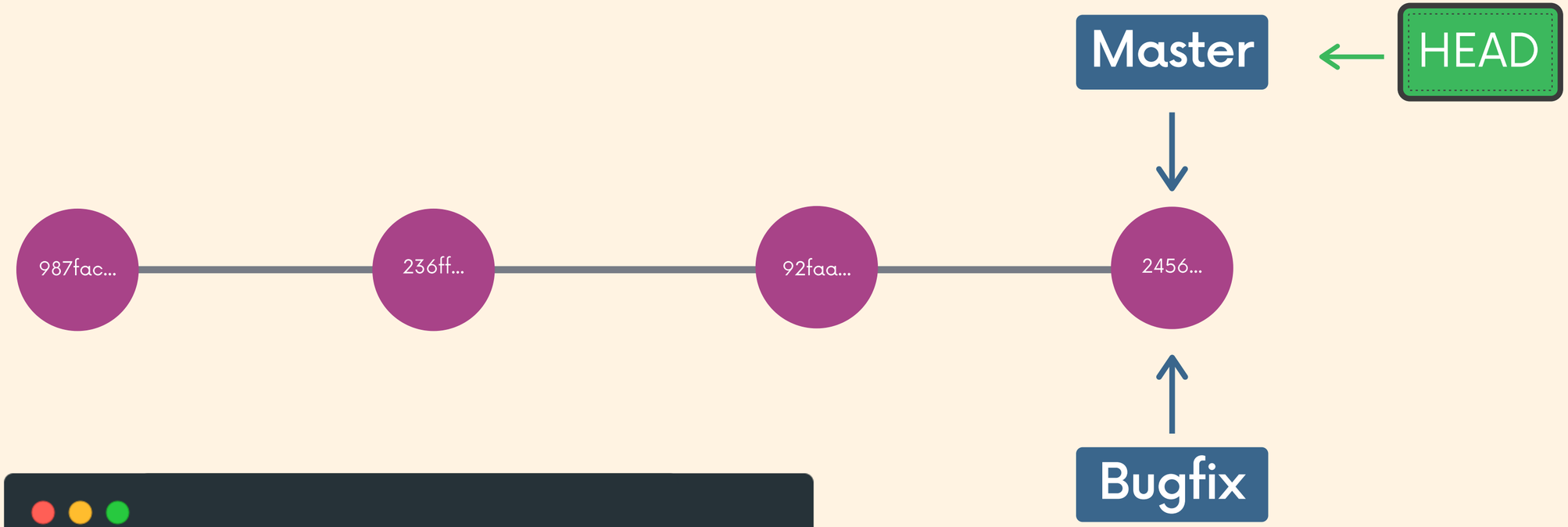
This is what it really looks like

Remember, branches are just defined by a branch pointer



This is what it really looks like

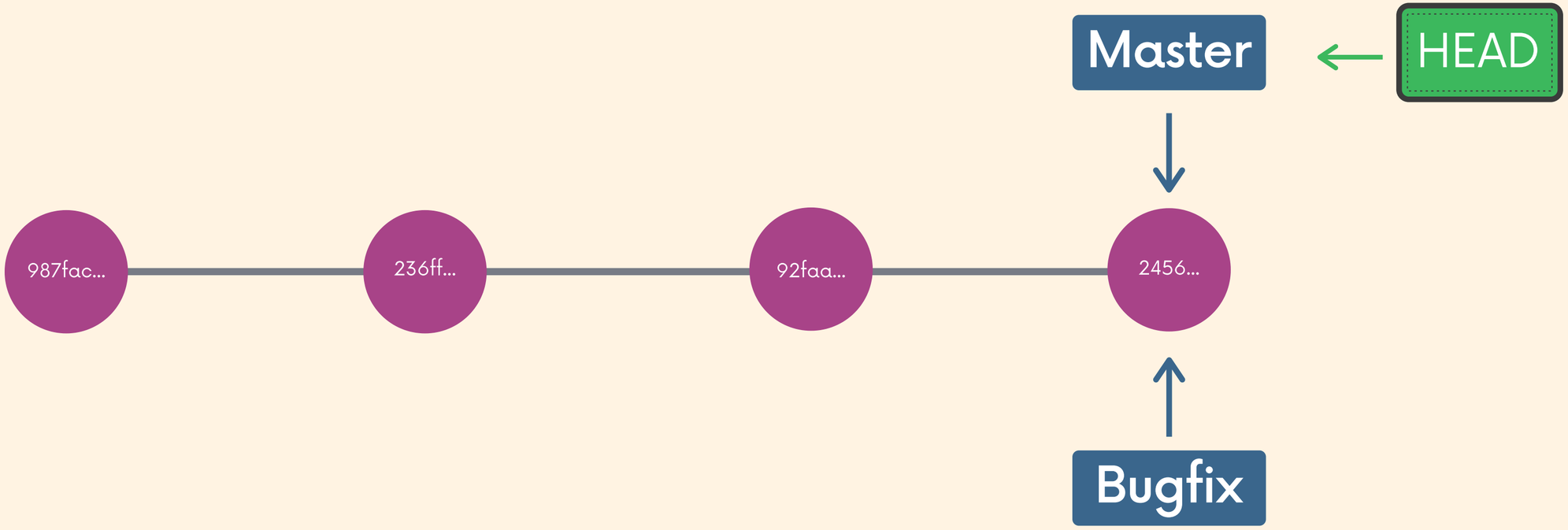
Master & Bugfix now point to the same commit, until they diverge again




```
❯ git merge bugfix
```

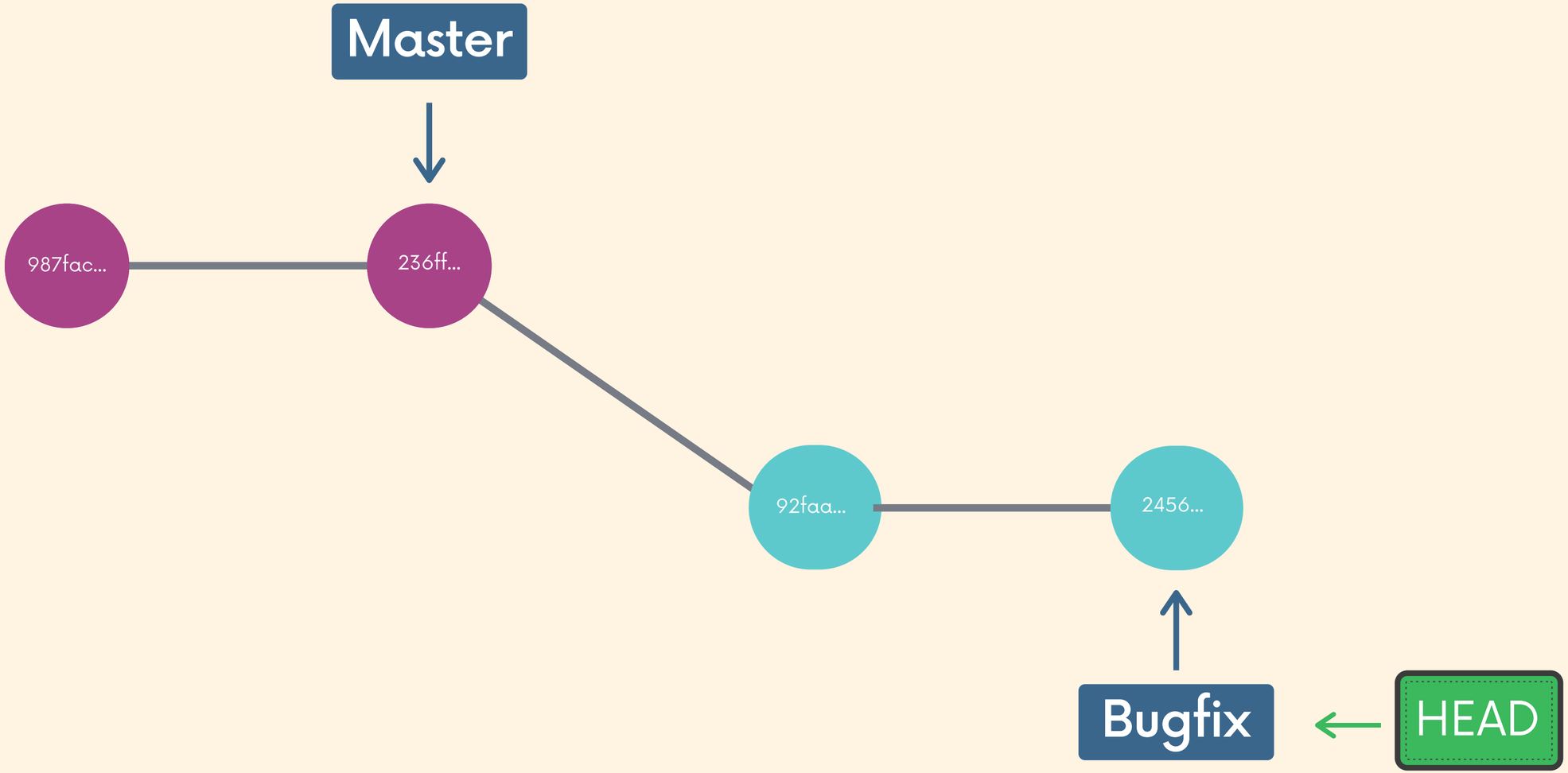
This Is Called A Fast-Forward

Master simply caught up on the commits from Bugfix



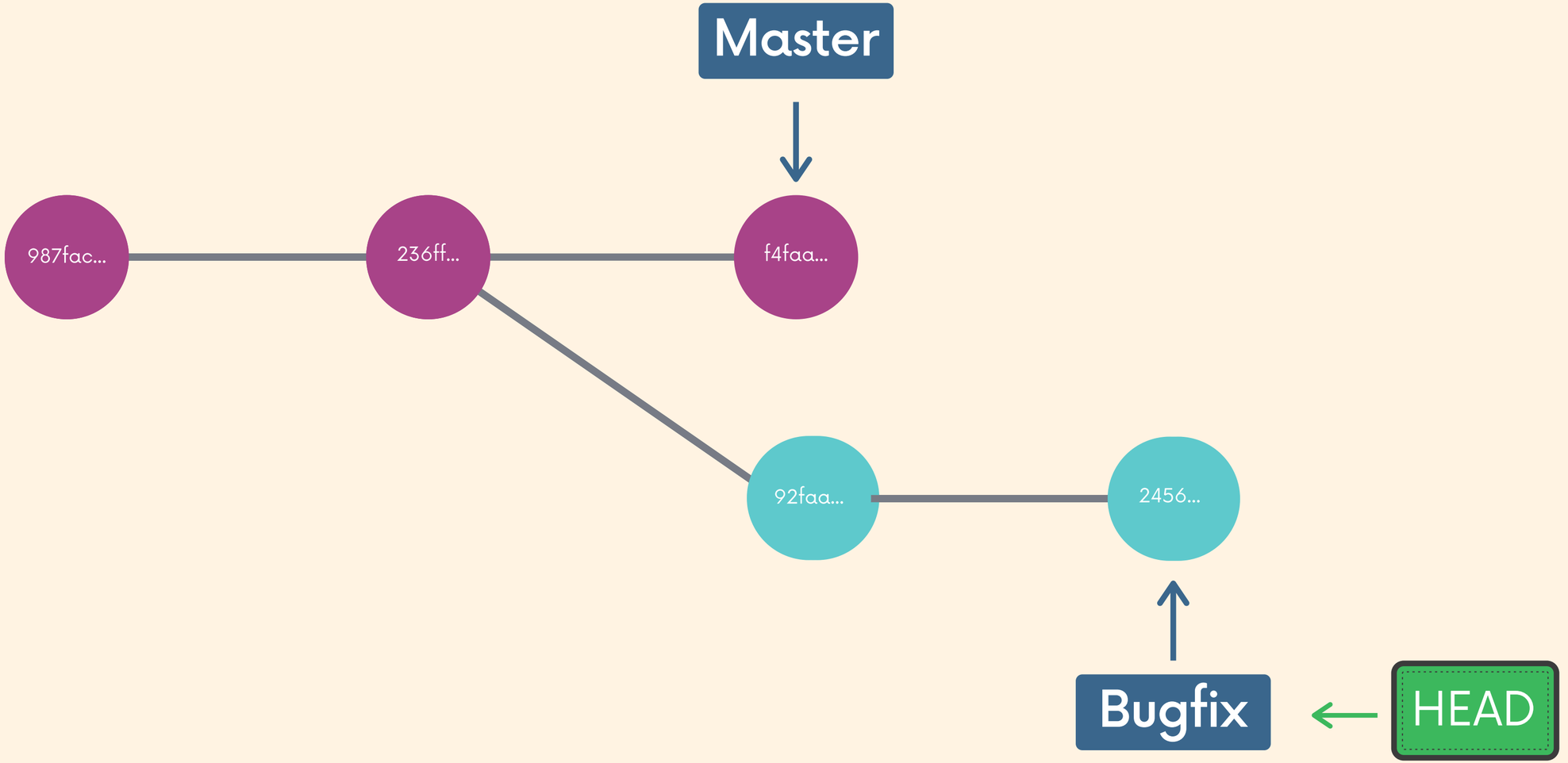
Not All Merges Are Fast Forwards!

We've branched off of master, just like before...



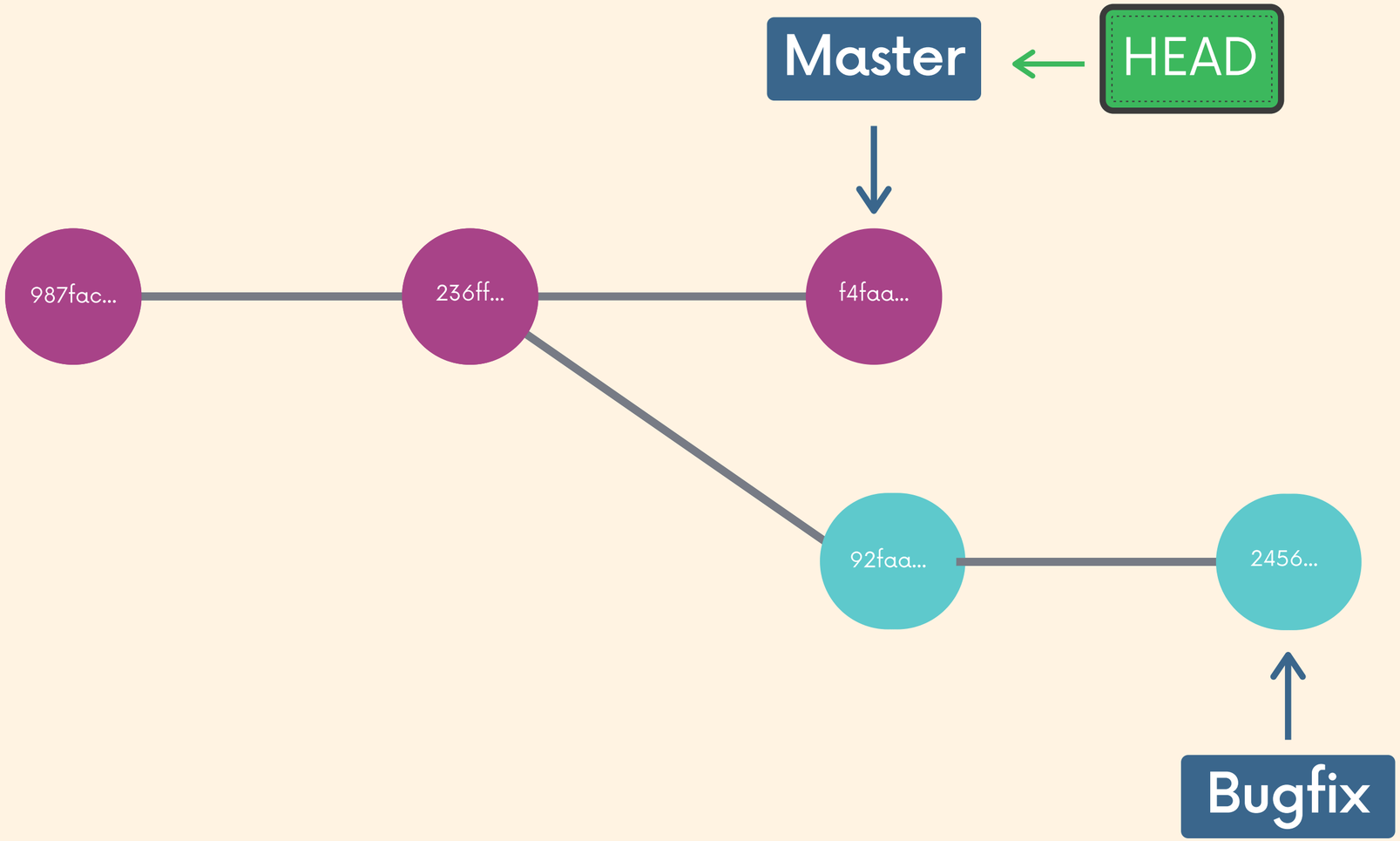
What if we add a commit on master?

This happens all the time! Imagine one of your teammates merged in a new feature or change to master while you were working on a branch



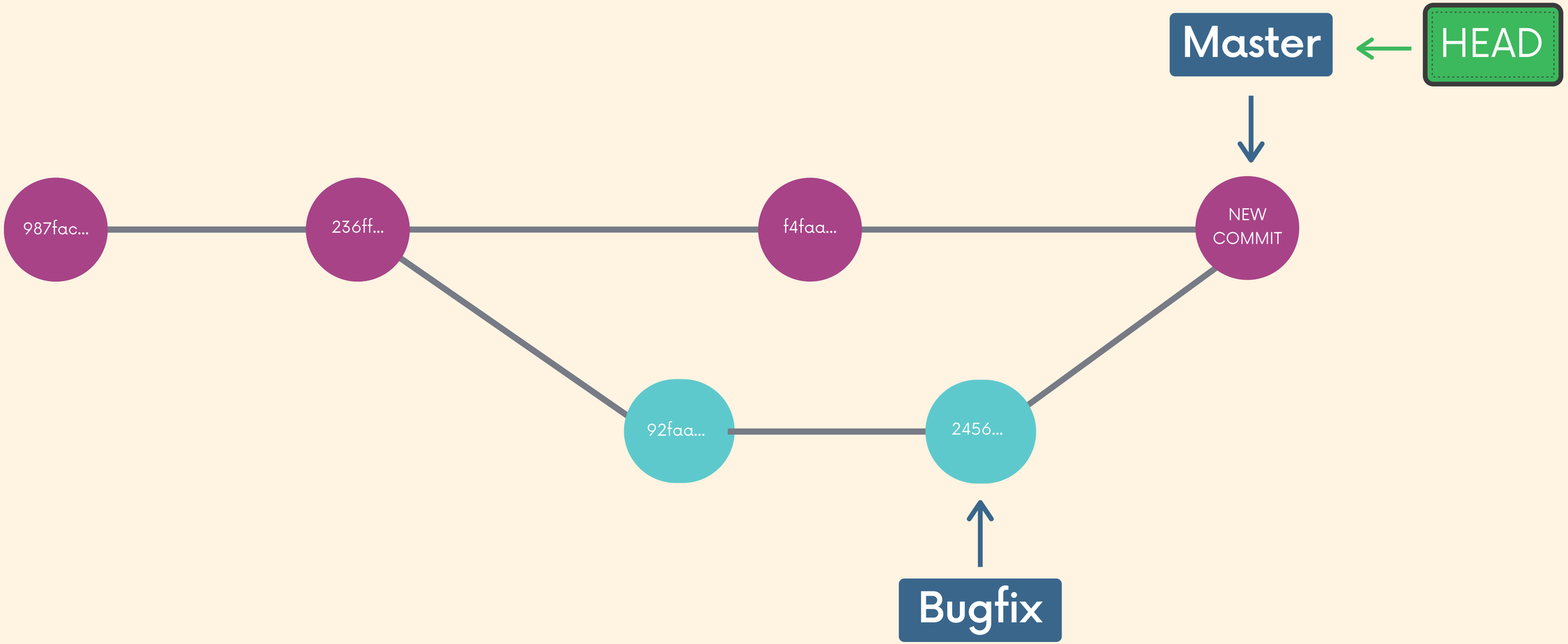
What happens when I try to merge?

Rather than performing a simple fast forward, git performs a "merge commit"
We end up with a new commit on the master branch.
Git will prompt you for a message.



What happens when I try to merge?

Rather than performing a simple fast forward, git performs a "merge commit"
We end up with a new commit on the master branch.
Git will prompt you for a message.



Heads Up!

Depending on the specific changes your are trying to merge, Git may not be able to automatically merge. This results in **merge conflicts**, which you need to manually resolve.

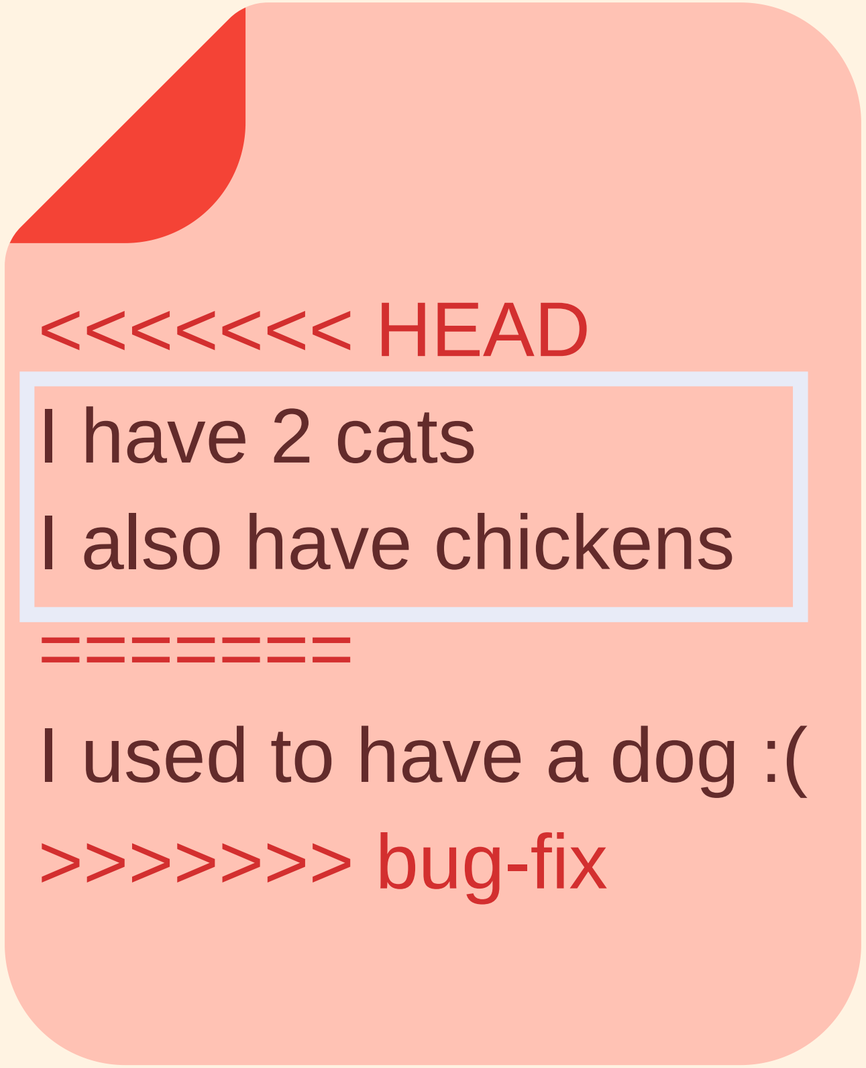

❯ CONFLICT (content): Merge conflict in blah.txt
Automatic merge failed; fix conflicts and then commit the result.

```
<<<<<<< HEAD
I have 2 cats
I also have chickens
=======
I used to have a dog :(
>>>>>>> bug-fix
```

WHAT THE...

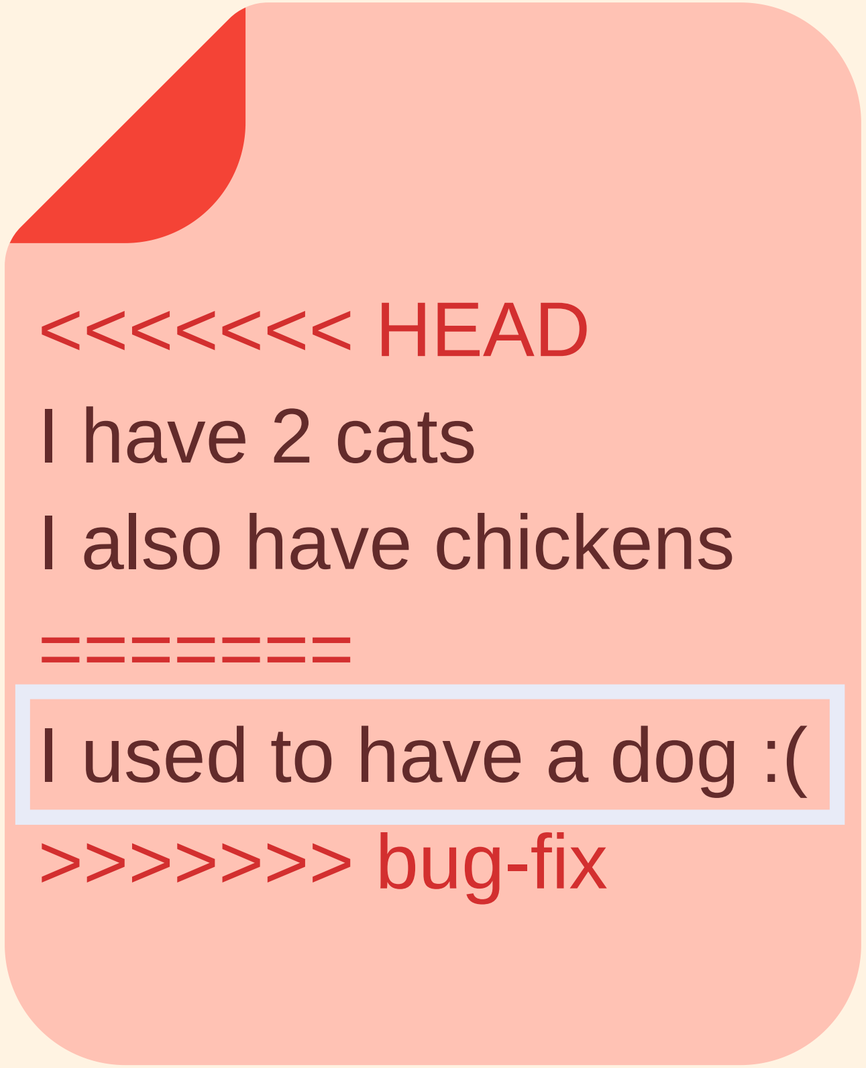
When you encounter a merge conflict, Git warns you in the console that it could not automatically merge.

It also changes the contents of your files to indicate the conflicts that it wants you to resolve.



Conflict Markers

The content from your current **HEAD** (the branch you are trying to merge content into) is displayed **between the** `<<<<<<< HEAD` **and** `=======`



Conflict Markers

The content from the branch you are trying to merge from is displayed **between the ======= and >>>>>>> symbols.**

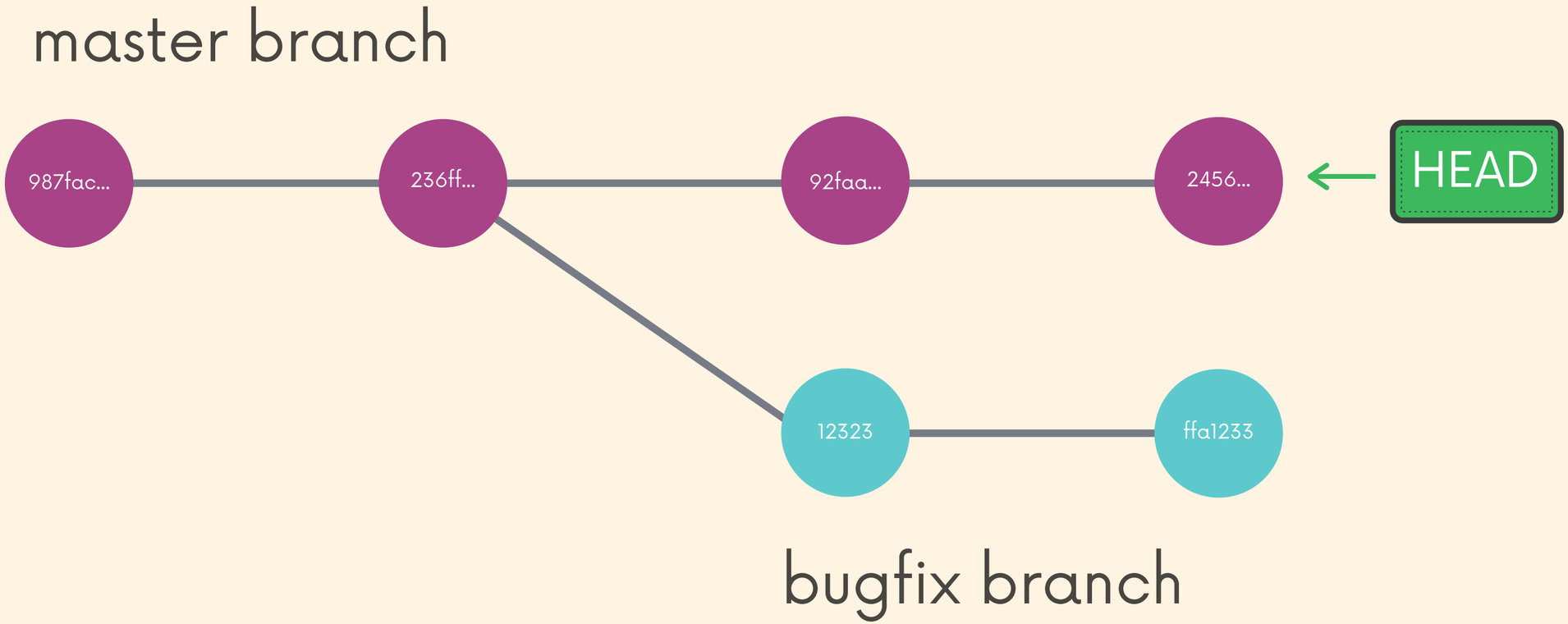
Resolving Conflicts

Whenever you encounter merge conflicts, follow these steps to resolve them:

1. Open up the file(s) with merge conflicts
2. Edit the file(s) to remove the conflicts. Decide which branch's content you want to keep in each conflict. Or keep the content from both.
3. Remove the conflict "markers" in the document
4. Add your changes and then make a commit!

Deleting A Branch

```
❯ git branch -d bugfix
```



Deleting A Branch