GIT-3 of 4

Simanta Mitra

Main references:

https://betterexplained.com/articles/aha-moments-when-learning-git/

Learning Objectives

Now that you know how to do some basic operations on your local repository i.e. add, commit, checkout, reset, and diff, let us delve into branching.

Operation	Developer wants to	Command
CREATE	create a new branch	git branch other_branch
READ	switch to a branch	git checkout other_branch
UPDATE	make changes inside a branch	git commit etc
DELETE	delete a branch	git branch –d other_branch
LIST	list all branches	git branch -av
MERGE	merge changes from one branch to current	git merge other_branch

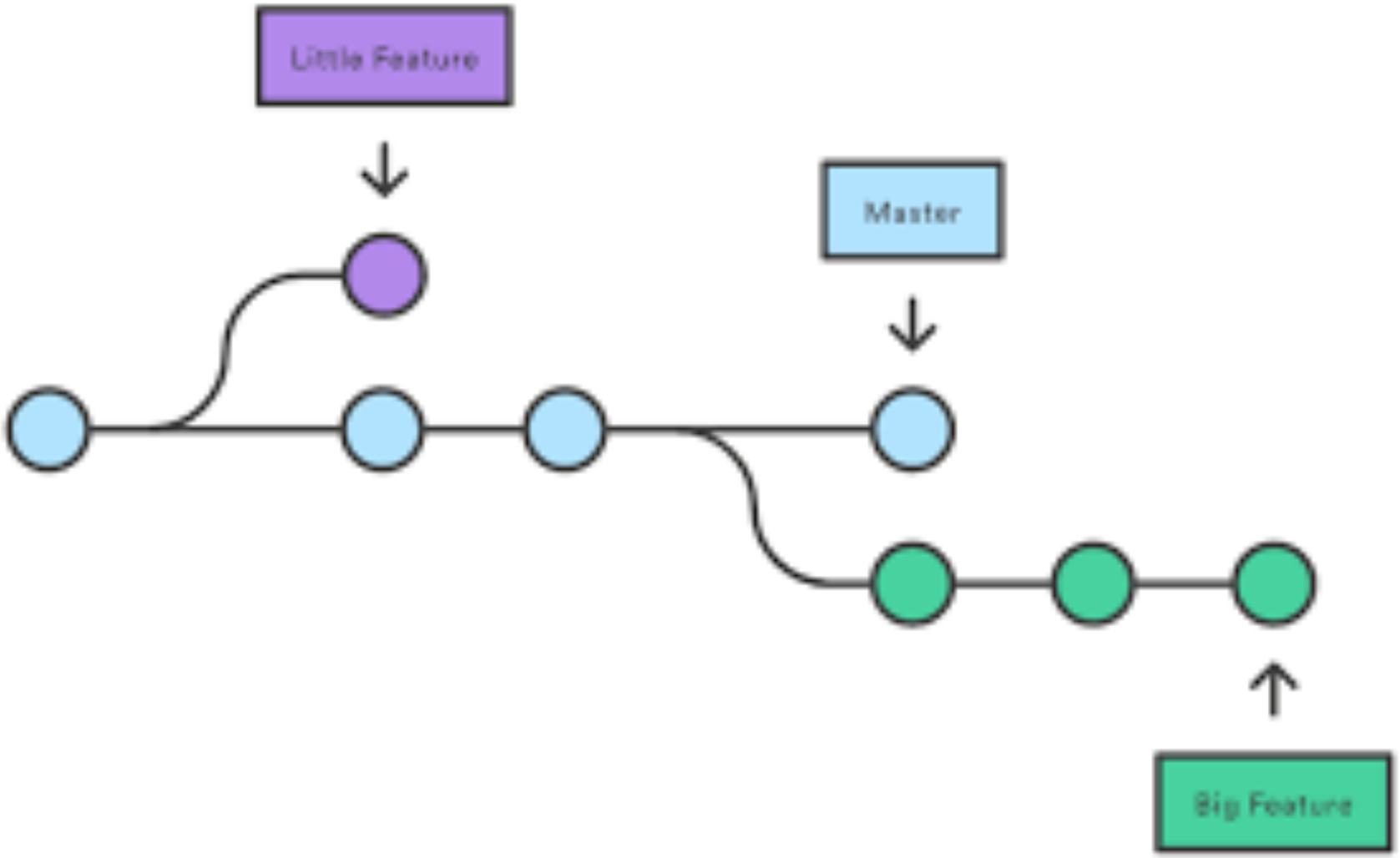
BRANCH

What's a branch?

- commits are snapshots in time.
- as you add commits, the chain of commits grows.

- so far, we have looked at a SINGLE chain of commits (default known as master branch).

- however, we can start a new chain of commits from any of the commits. These are called branches.


Little Feature
Master
Big Feature


v0.1
v0.2
v1.0
Master
Develop
Feature
Feature

BRANCH CRUDL

CRUDL

- Create a branch

  git branch abc (a link named abc to current commit is created)
- Read or switch to a branch

  git checkout abc (head points to abc. stage and WD changed)
- Update a branch

  git commit (when on branch)
- Delete a branch

  git branch –d abc
- List branches

  git branch –av (all + verbose)

MERGE

Git Merge

1. What is it? Why do it? How?
2. What are the three cases?
3. Fast Forward
4. Three Way Merge
5. Conflicts
6. Conflict resolution

1. What/Why merge? How?

- You do a merge when you want to **incorporate changes made on a branch to the current branch**.

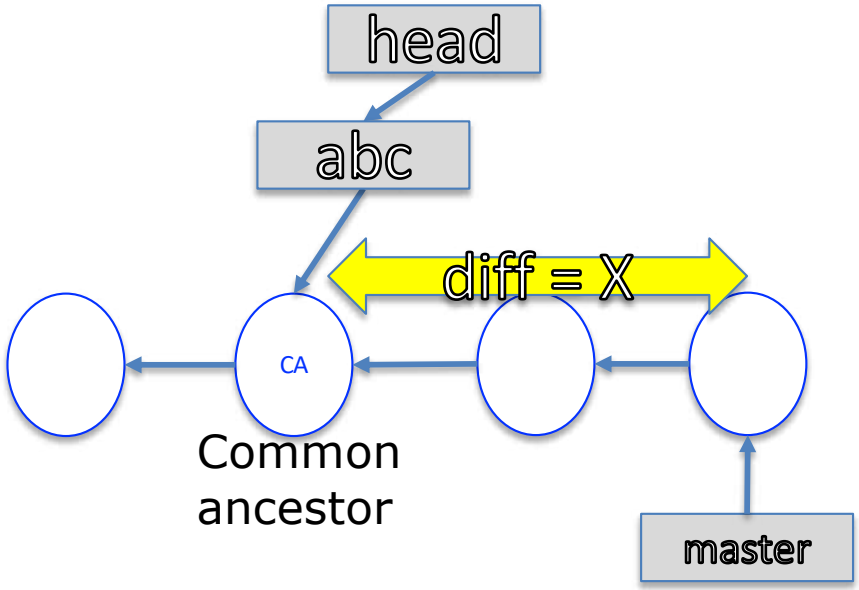
- For example, you developed and tested a new feature in a branch abc.
  - Now, you would like to incorporate this new feature into the master branch.
  - Assuming you are on master branch, you would type "git merge abc"

2. Three cases of Merge

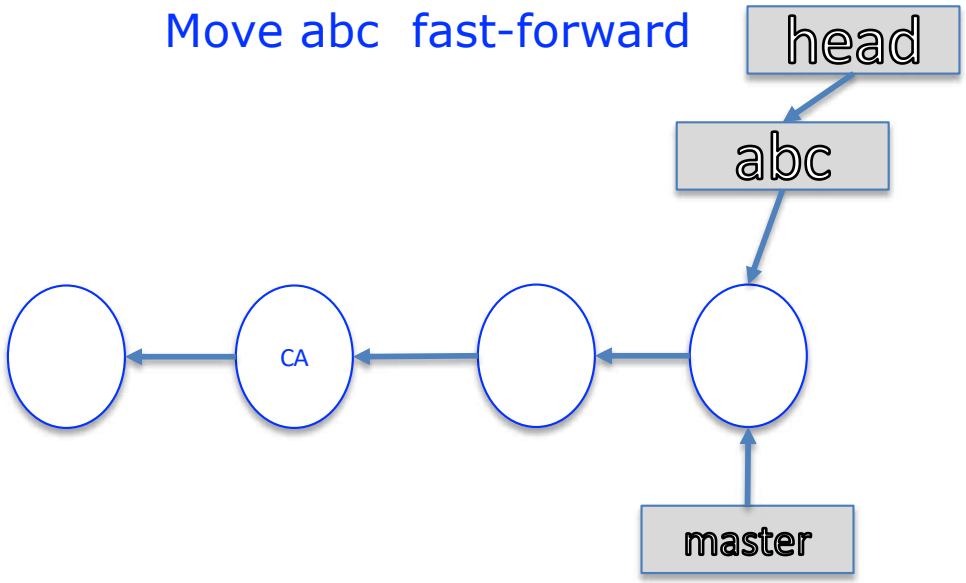
- Find a common ancestor commit object.
- Then (3 cases)
  - if commit-to-be incorporated is an ancestor of current commit object --- then do nothing? (why?)
  - else if current commit object is ancestor of commit-to-be-incorporated, then move reference AND checkout (fast-forward merge).
  - else three way merge.

3. fast forward merge

BEFORE MERGE



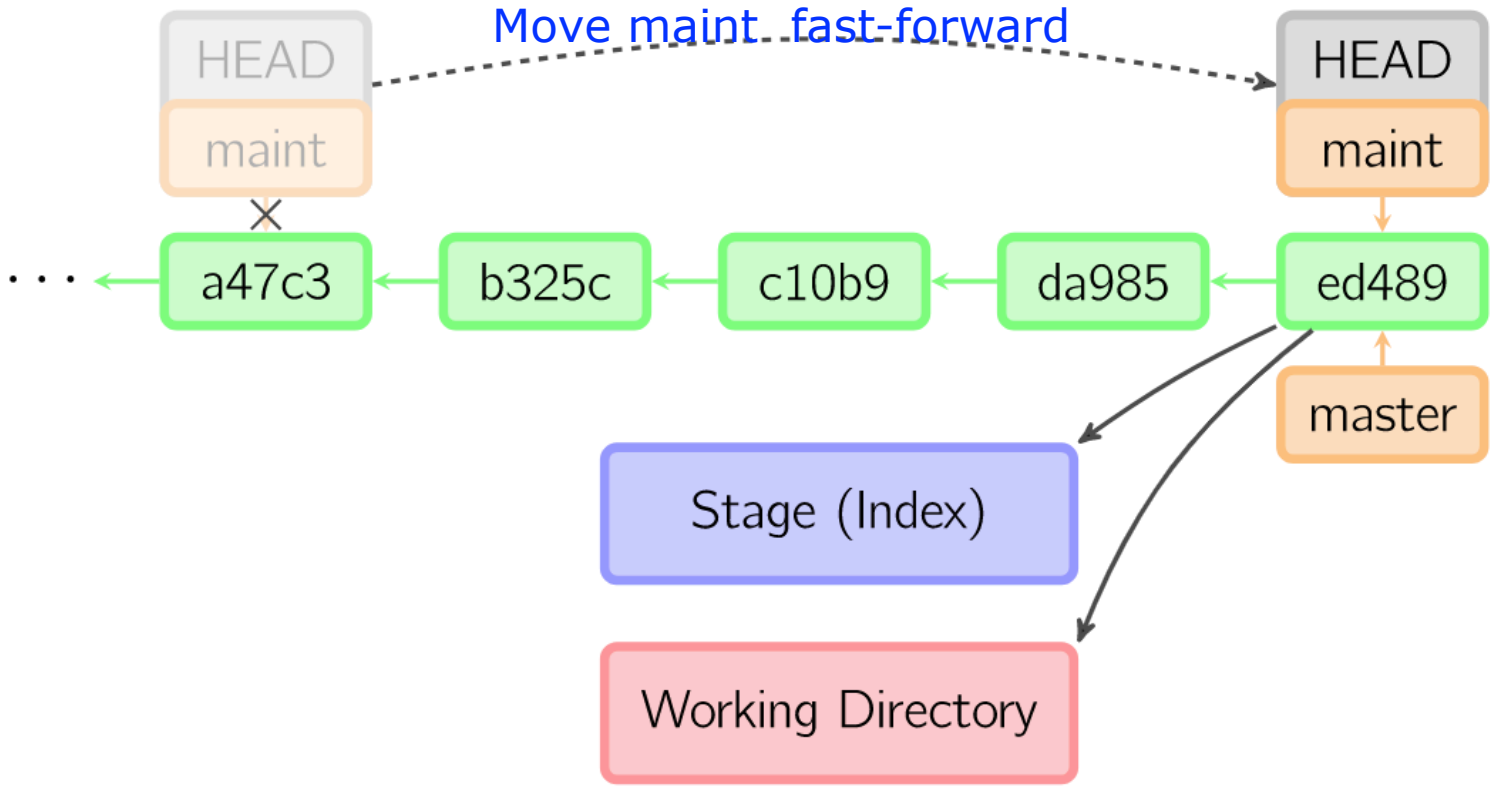
AFTER MERGE



(1) We are initially on branch abc (see HEAD).
(2) We want to apply changes in master to abc branch.
(3) Making those changes to abc will mean we will get to same commit as the master. So branch abc will move forward.
(4) (what's in stage? WD?)

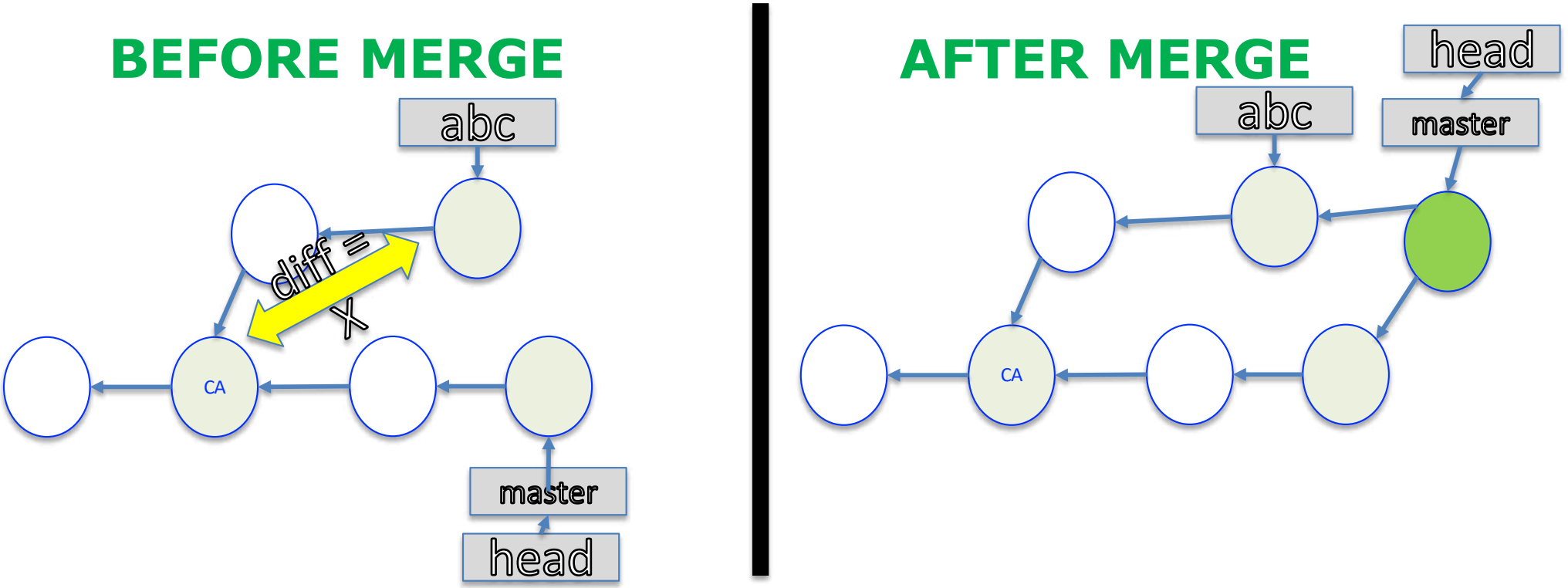
3. fast-forward merge

(another example)



First, branch maint was checked out
Next, "git merge master" was typed

4. Three way merge



git merge abc means that take diff X and apply to the current commit

(1)Create NEW commit object (green) with the changes.

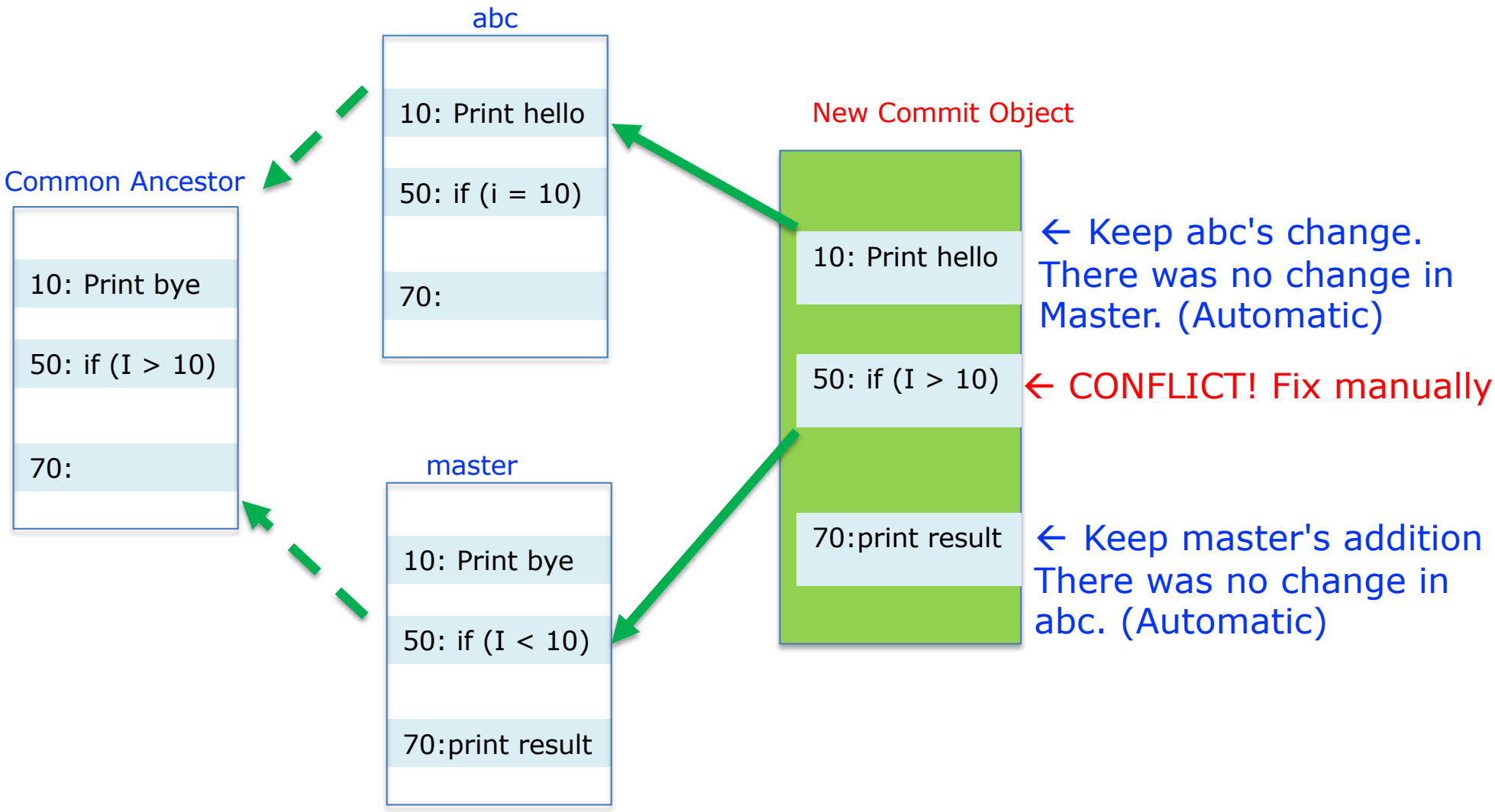
(2)Make it point to both abc and master

(3)Make this new commit object the master/head

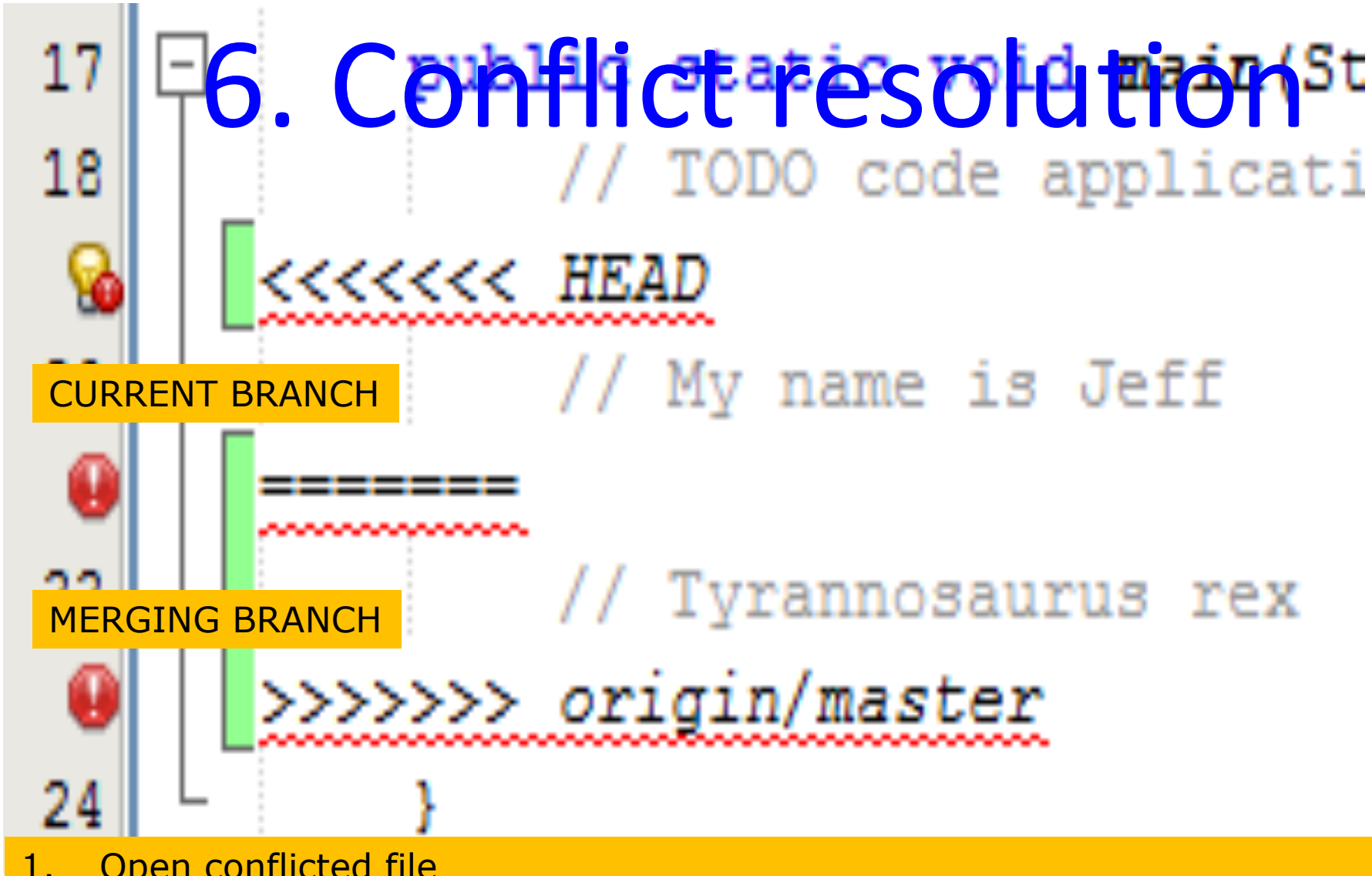
(4)Whats in stage? WD?

5. Conflicts (git merge abc)

This example shows a single file. Many files would be in similar states



6. Conflict resolution



1. Open conflicted file
2. decide how to resolve the differences between the marked "current branch" and "merging branch"
3. Edit the file whith your changes (also remove the <<<< ==== and >>>)
4. Add the modified file.

DO this for all the conflicted files.
THEN commit all the changes. That will RESOLVE the conflict problem.

THE END!