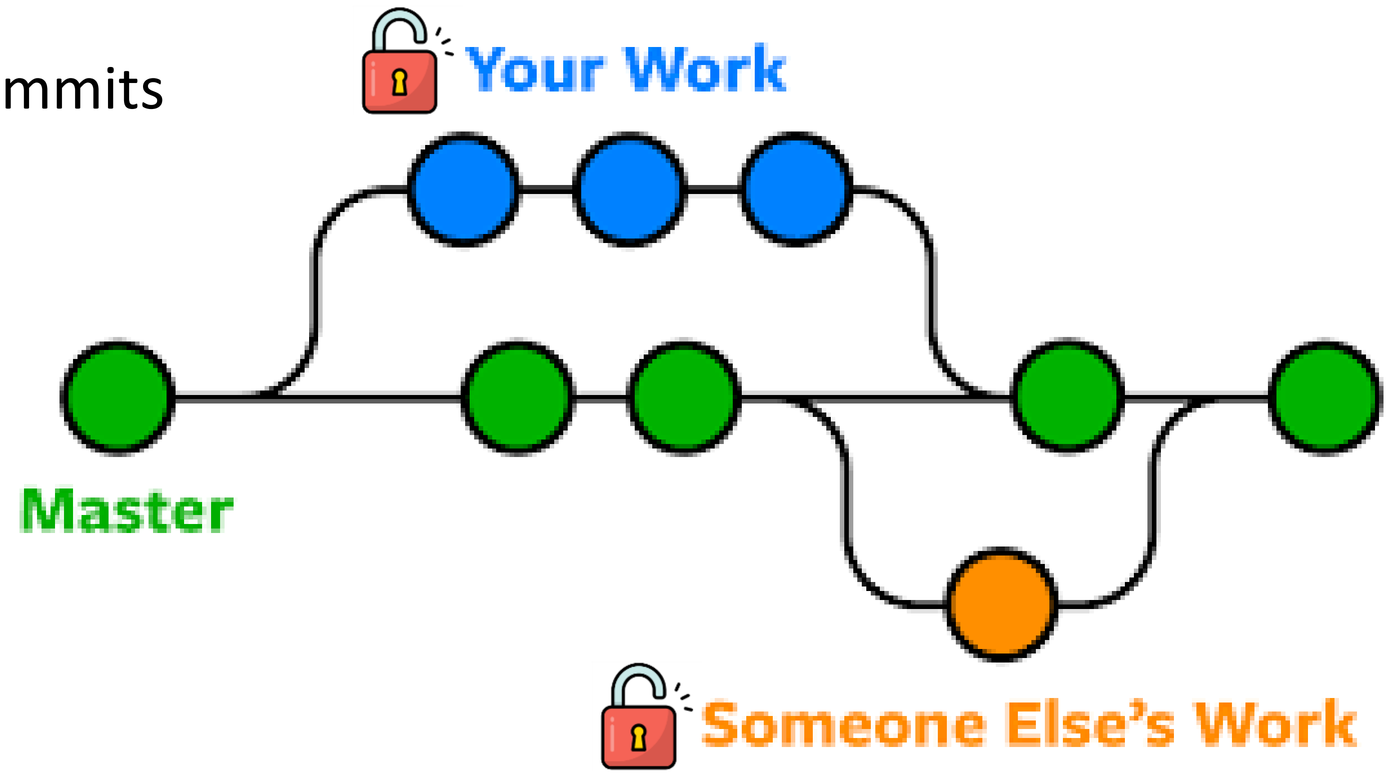
Git Fundamentals
Day 4:
Branches

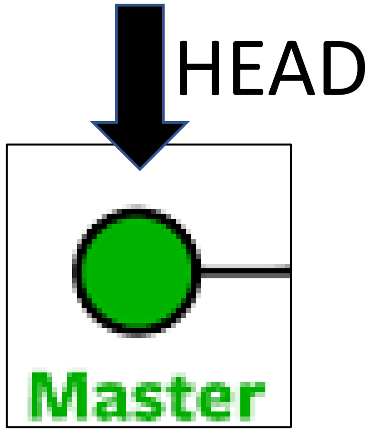
Today:

- ✓ Day 3 Recap + amend/reset commits
- ✓ Branches
- ✓ Stashing changes
- ✓ Introduction to merging



Recap: HEAD

- A reference to a commit
- Will follow us as we create new commits
- To what we compare versions



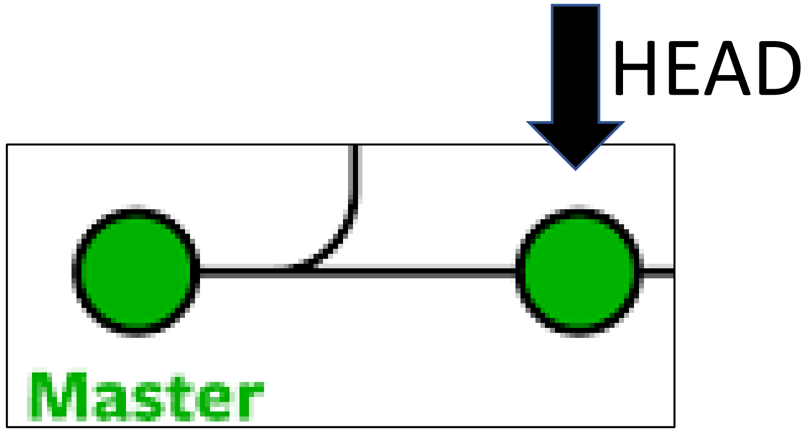
Recap: git checkout

We can deliberately move the HEAD reference

We can navigate by:

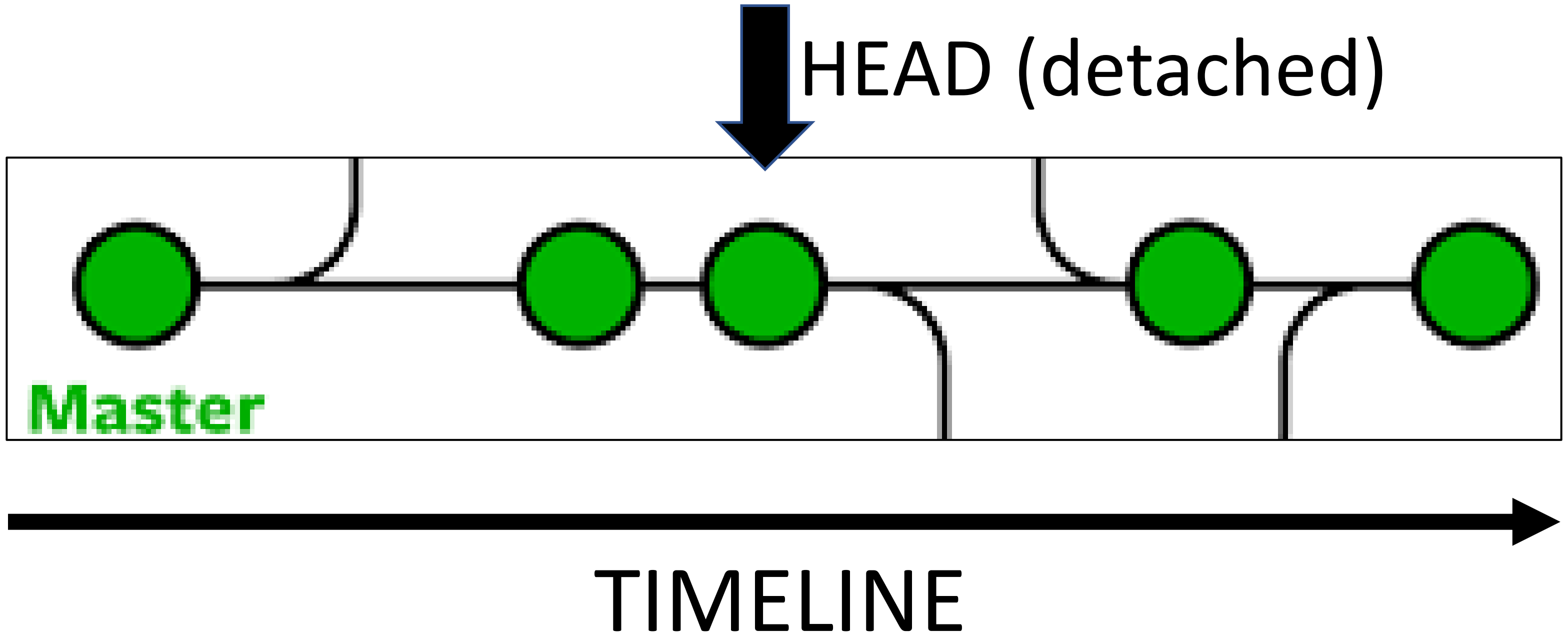
- Commit id (We can use the shortened hash)
- Relative position (HEAD~N)

Note: N is the number of commits before current HEAD position



Recap: git checkout

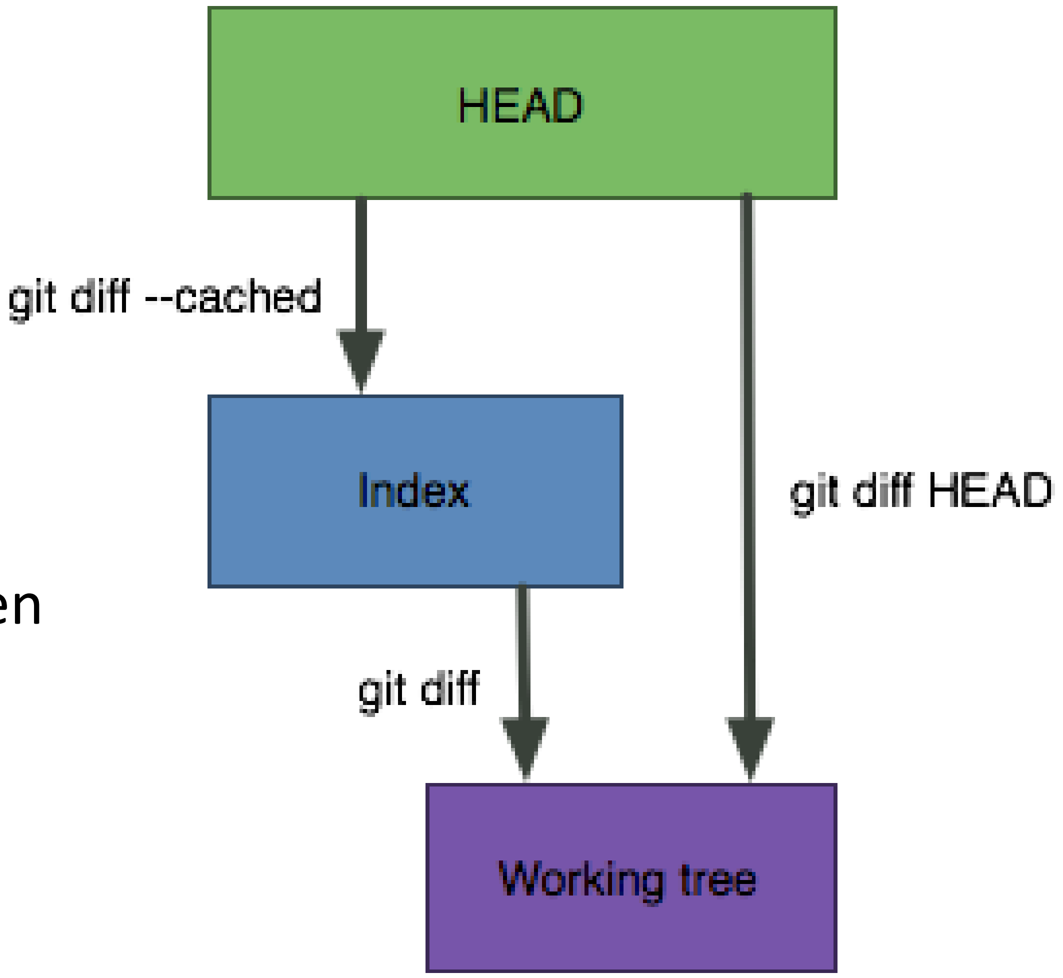
- When not pointing to the last commit is called detached HEAD



Recap: git diff

It is posible to view changes between commits

- By default git diff (without arguments) compares work tree with stage area
- Other use cases are viewing changes between commits, timelines, …

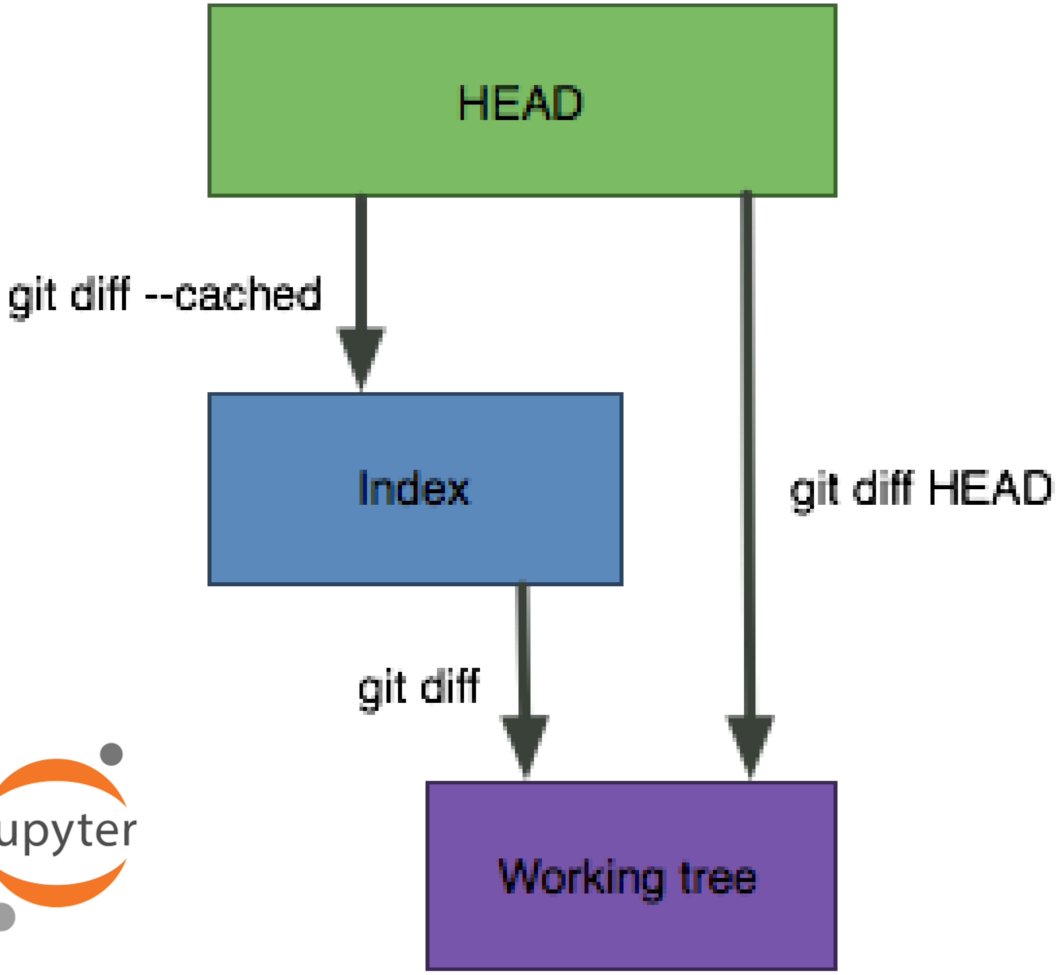


Recap: git diff

However … there are some nice GUI options:

- GitKraken
- Git Lens
- VS Code

Note: Jupyter notebooks not easy to visualize

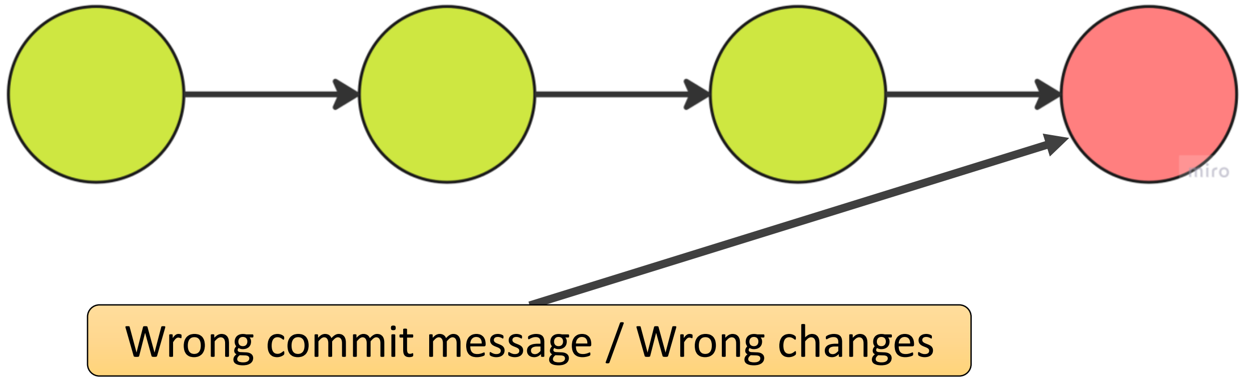


Commits: Amending

We can replace last commit with a modified one

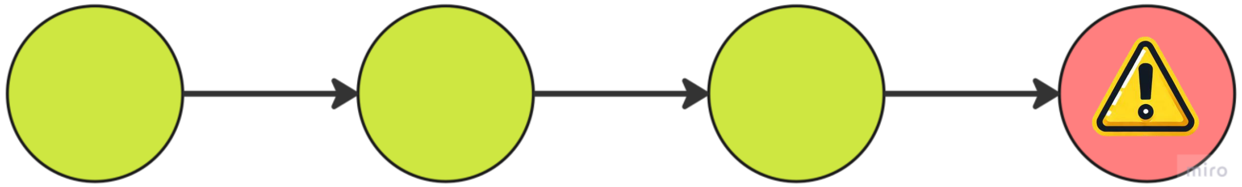
Command `git commit --amend` allows:

- Modifying commit message
- Modifying selected changes

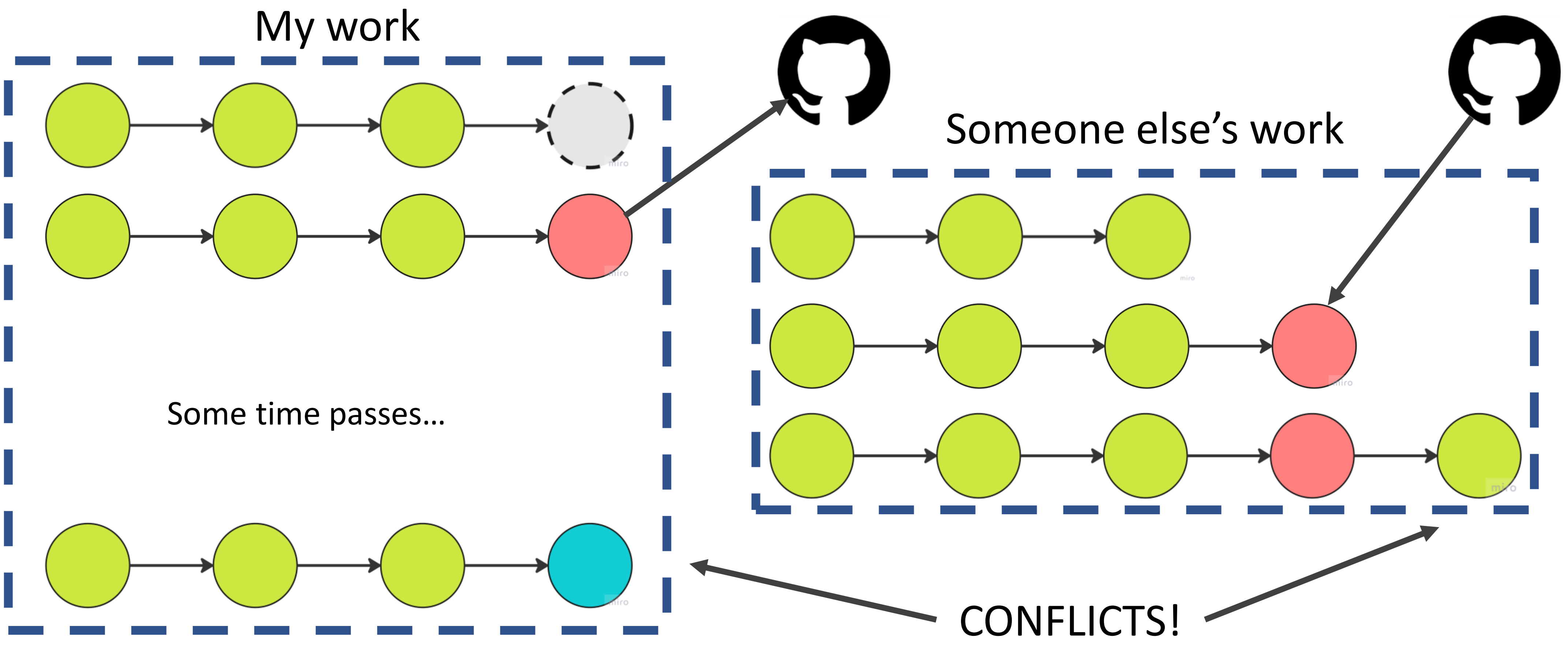


Beware: ⚠️ Amending

- Modifying the commit history is not always safe when working with others
- Amending commits is useful before publishing commits
- In general we only want to append to the commit history without further modifications

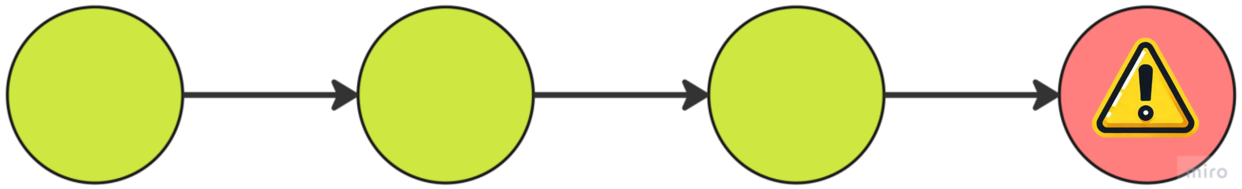


Beware: ⚠️ Amending Example



Solving conflicts

- Amending commits is useful before publishing commits
- In general we only want to append to the commit history without further modifications

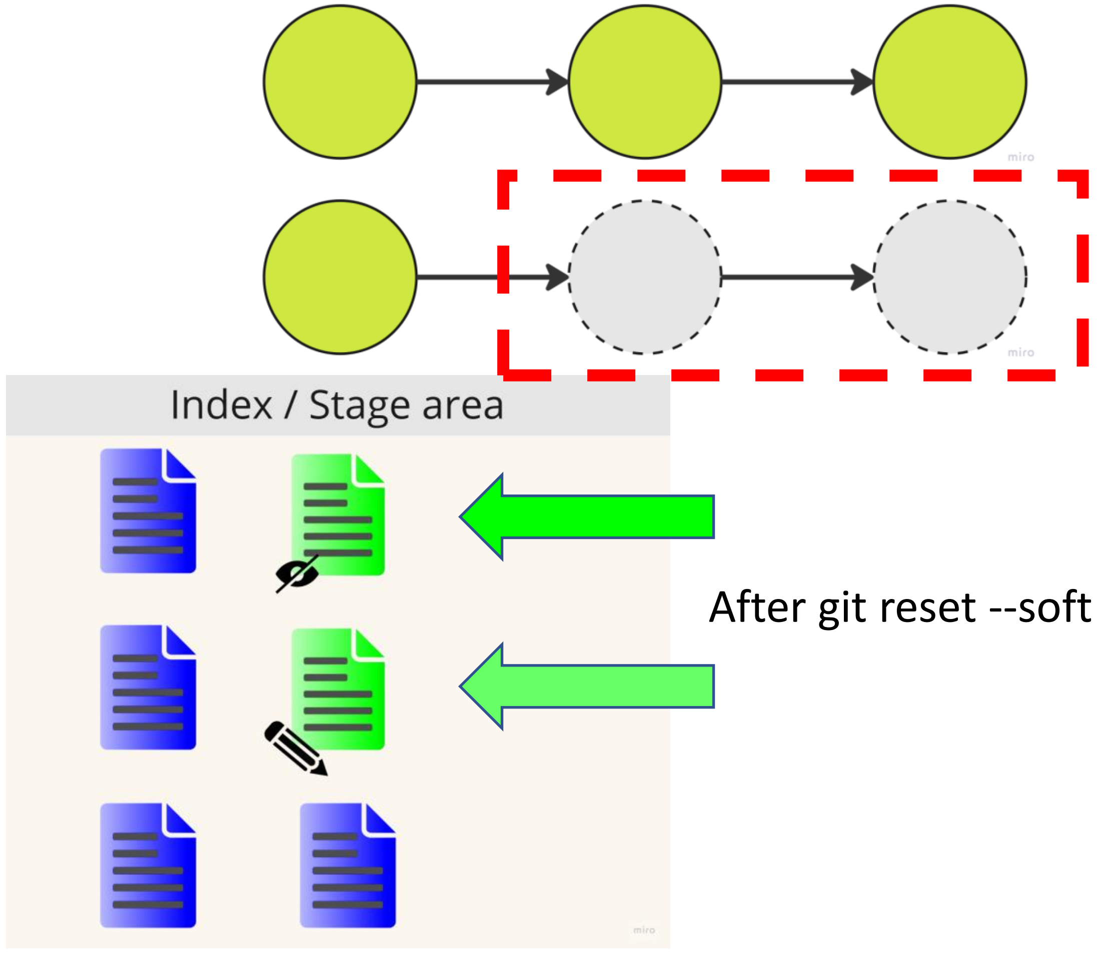


Commits: Reseting

Another option to modify the history and that is “non destructive” is git reset --soft

- Move HEAD pointer backwards and move the reseted commits to the Stage area

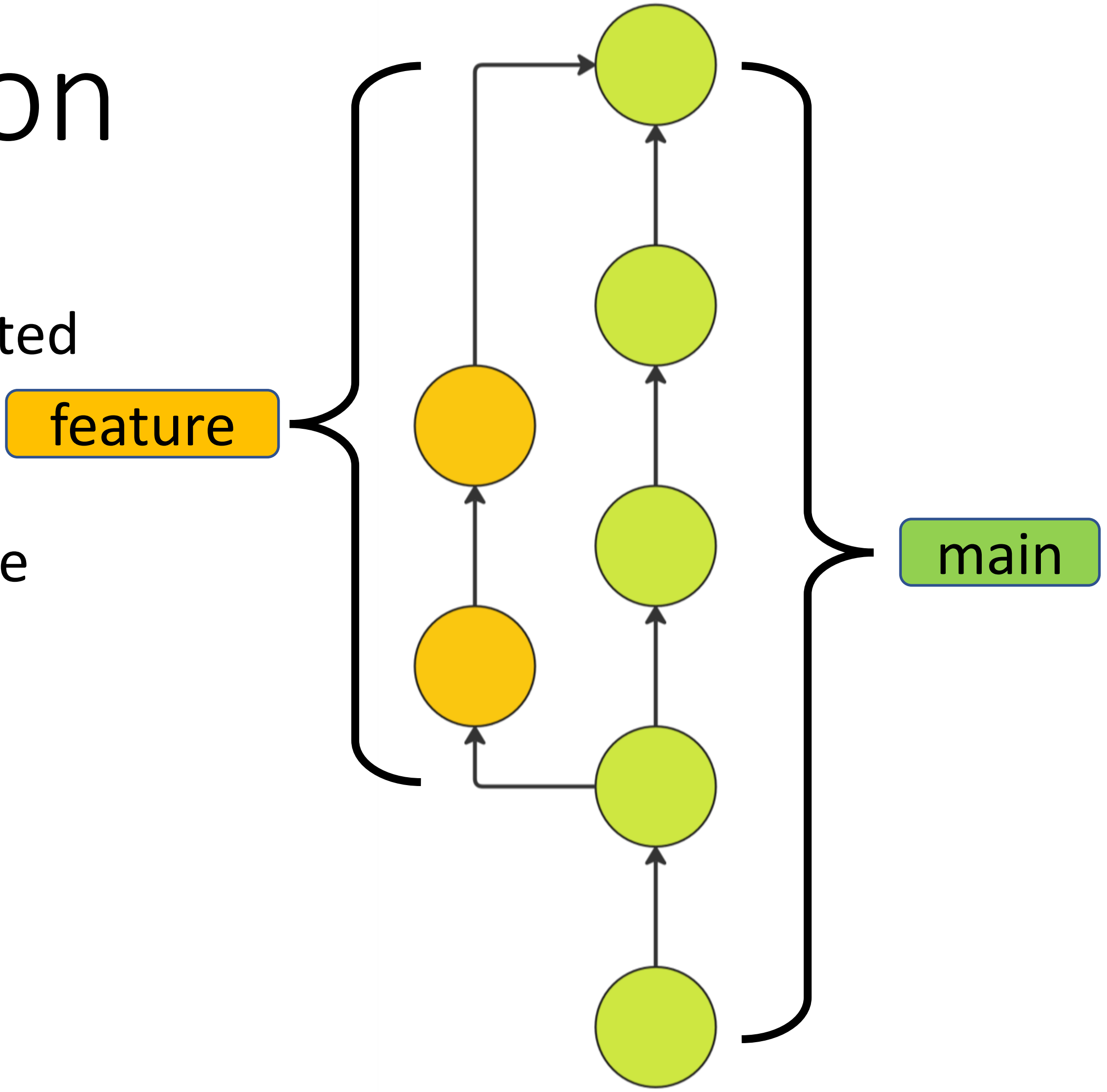
- Leaves working directory untouched

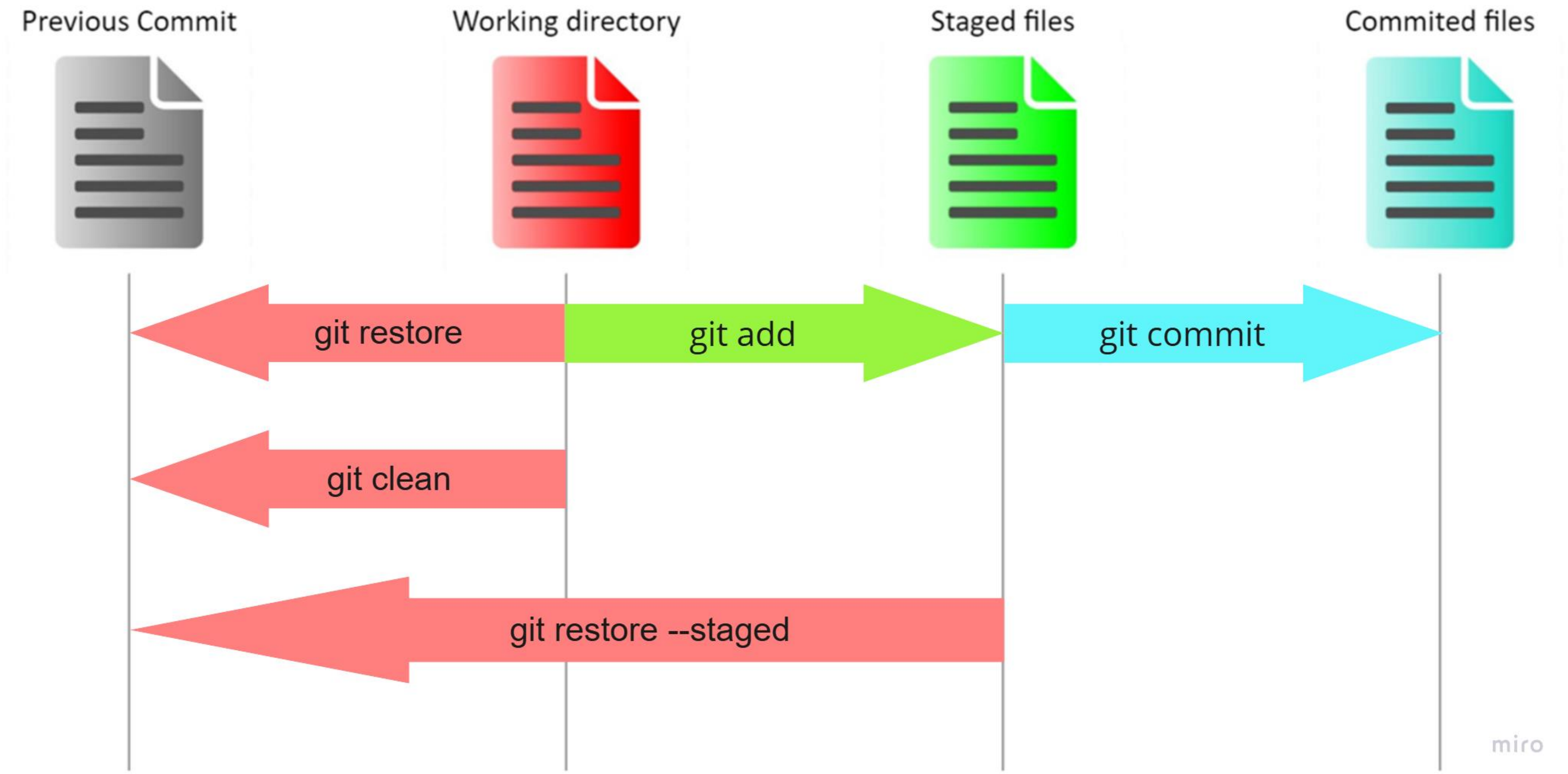


Branches: Introduction

When working in a project, we may be interested on working in parallel for 2 reasons:

- Do changes in different places of a file at the "same time"
- Modify already existent code to obtain behaviours that diverge

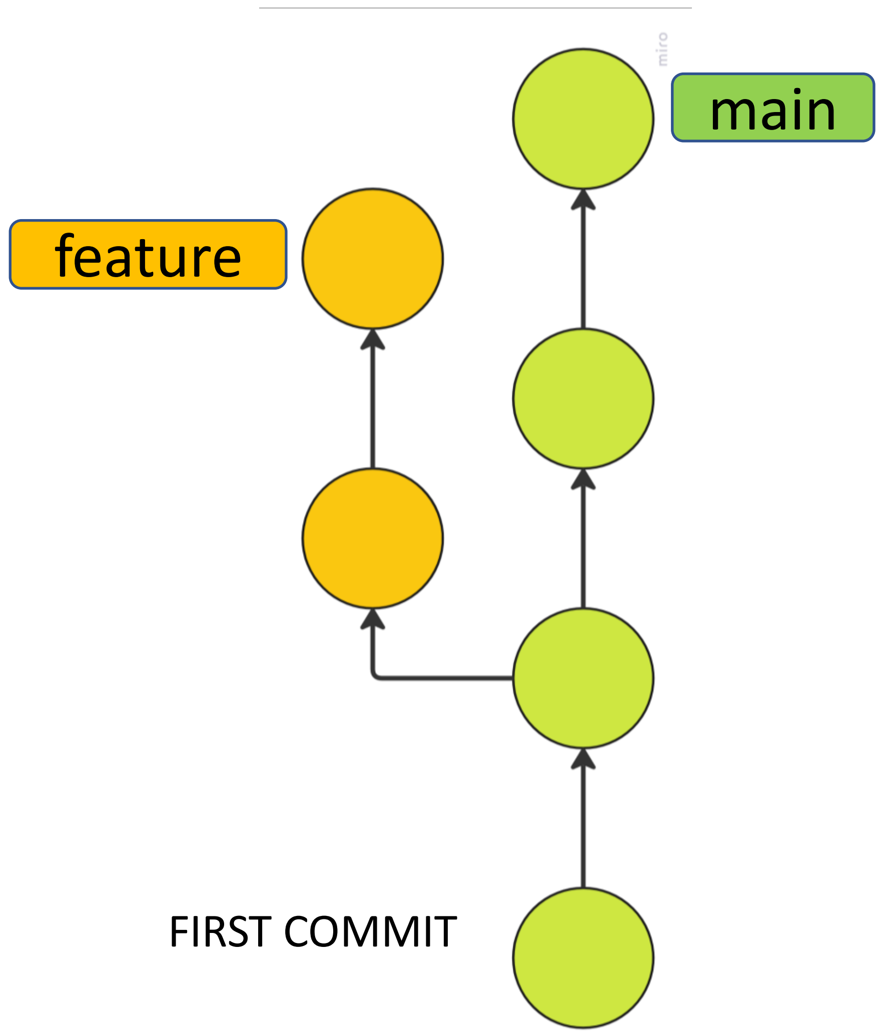



Previous Commit
Working directory
Staged files
Commited files
git restore
git add
git commit
git clean
git restore --staged
miro

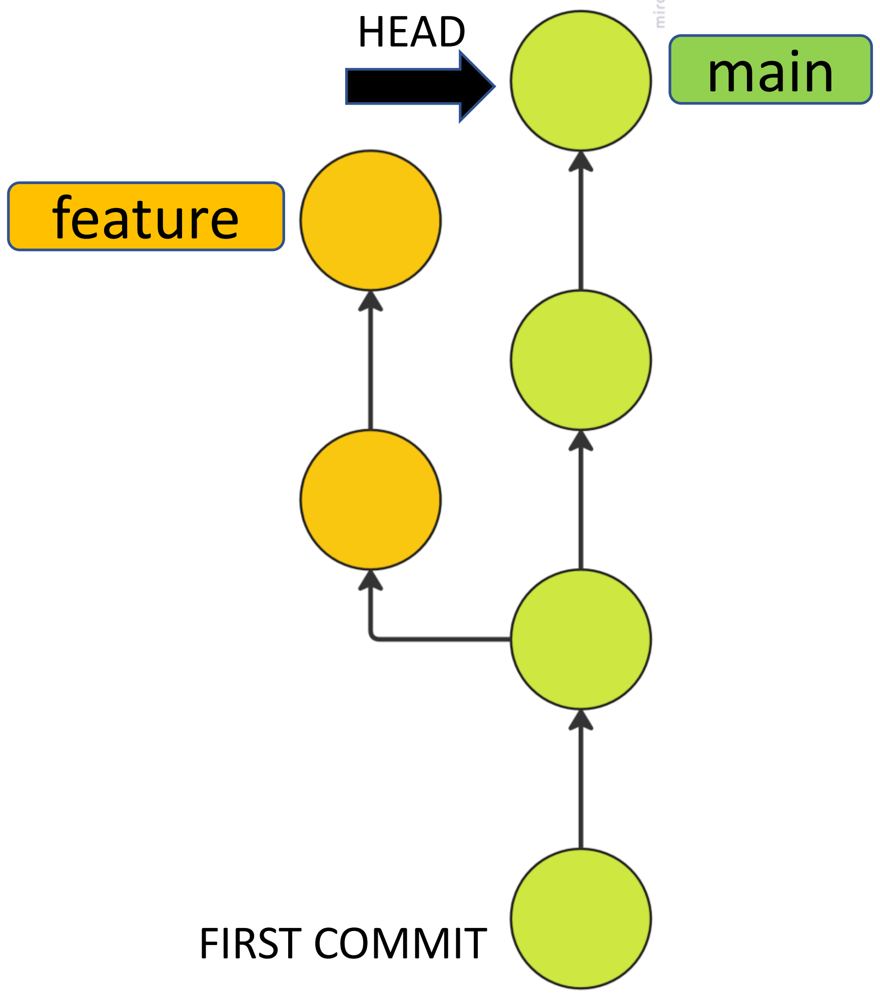
Branches: Switch

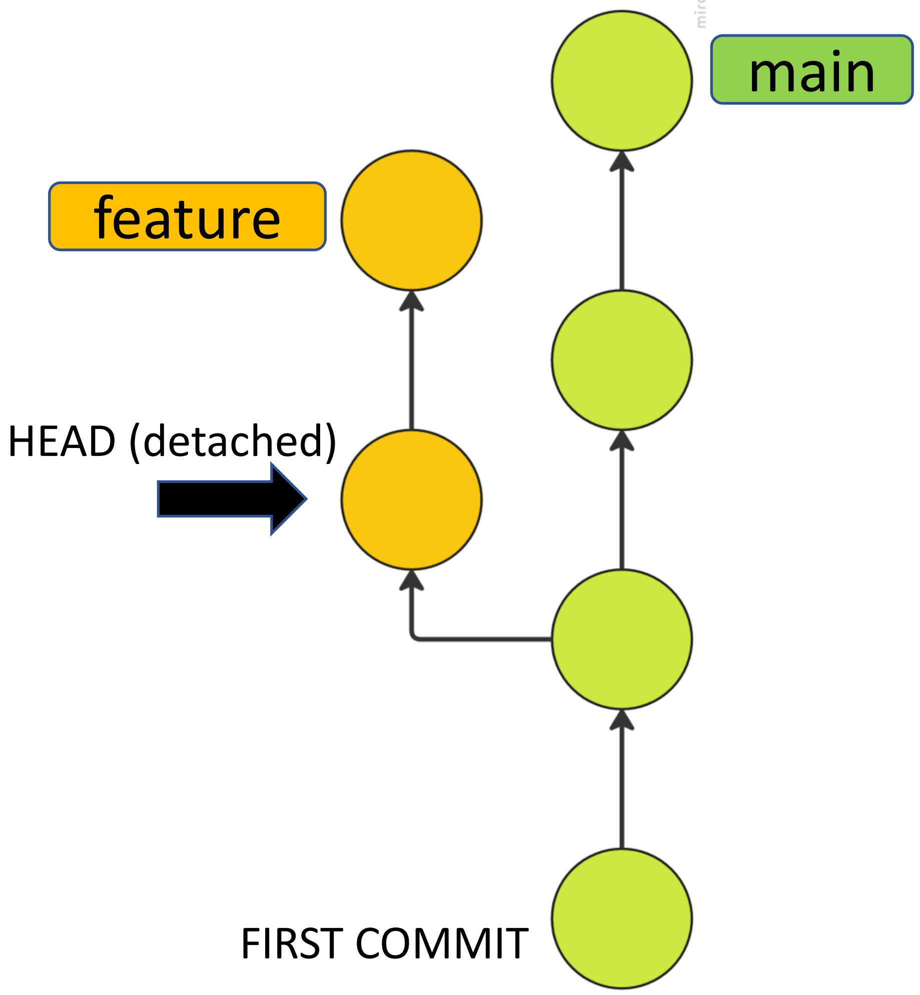
A branch is simply a pointer to a commit, that is at the end of a line of development

- We can create branch from a commit as its starting point with `git switch -c branch-name`
- Switching between branches can be achieved with `git switch branch-name`



Branches: HEAD vs branch

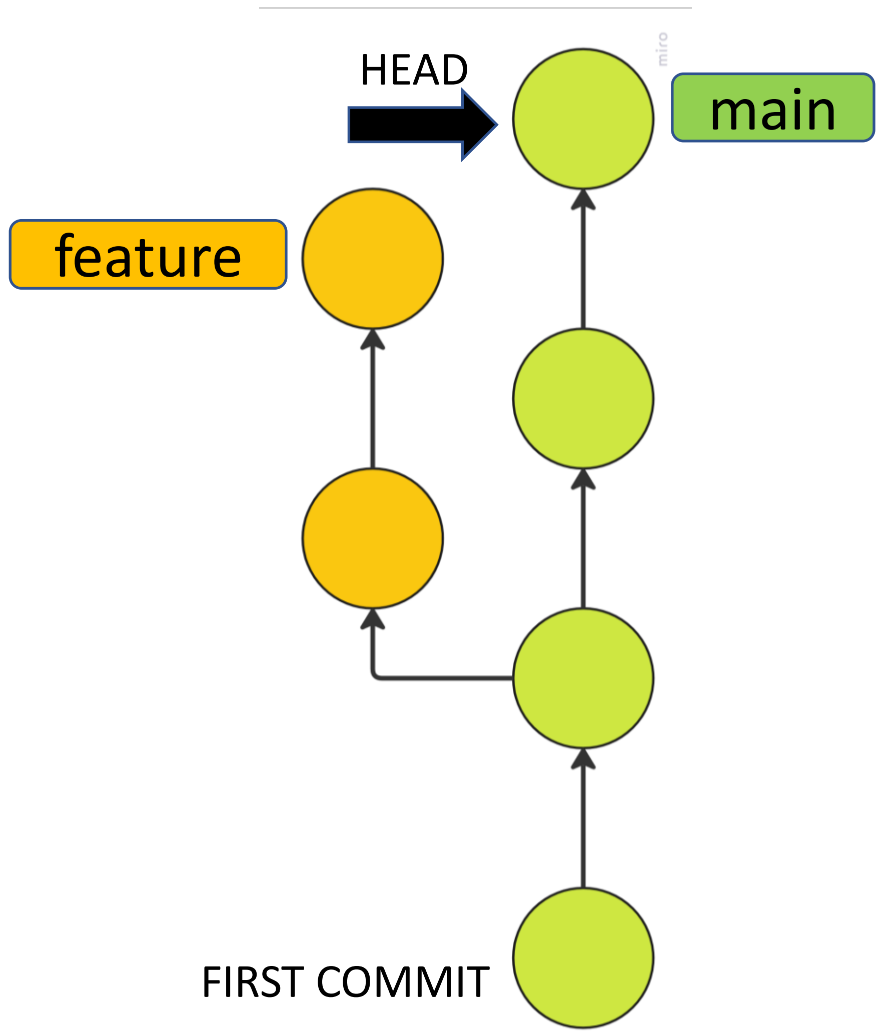


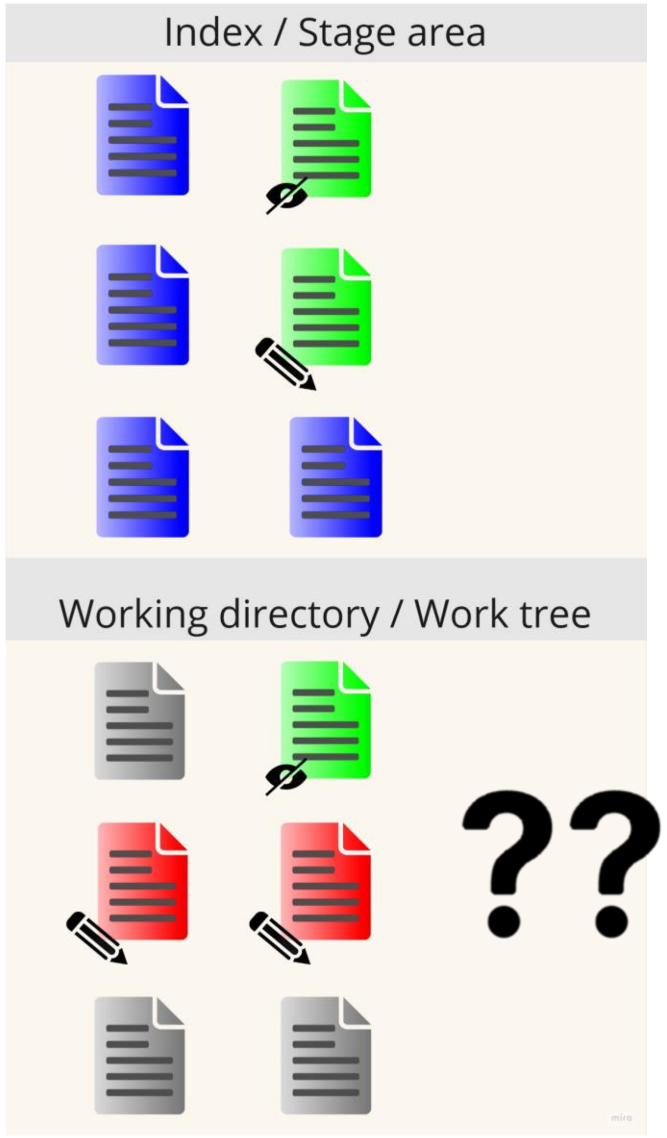


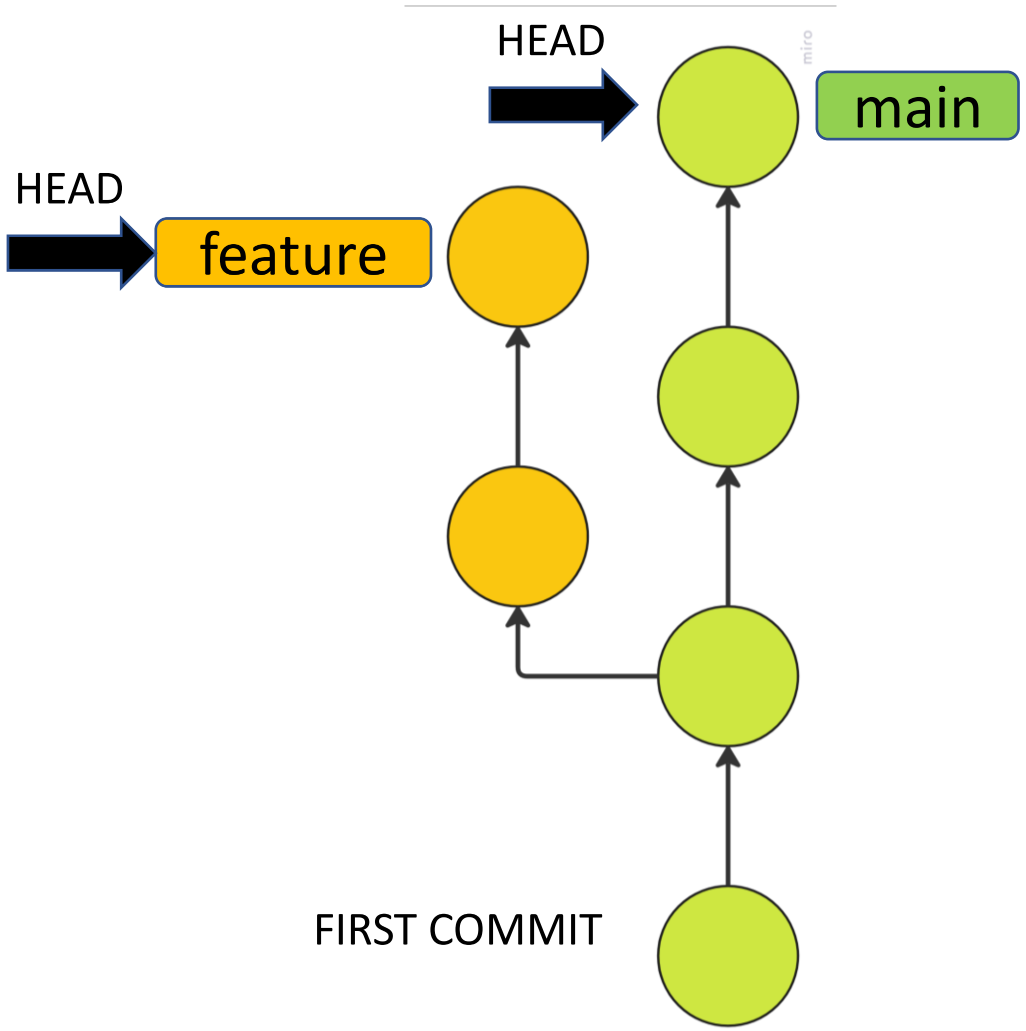
Stashing changes

We can't always move the HEAD reference to a commit if there are changes in our working or staging area, for both:

- Commits (git checkout)
- Branches (git switch)




Index / Stage area
Working directory / Work tree
??

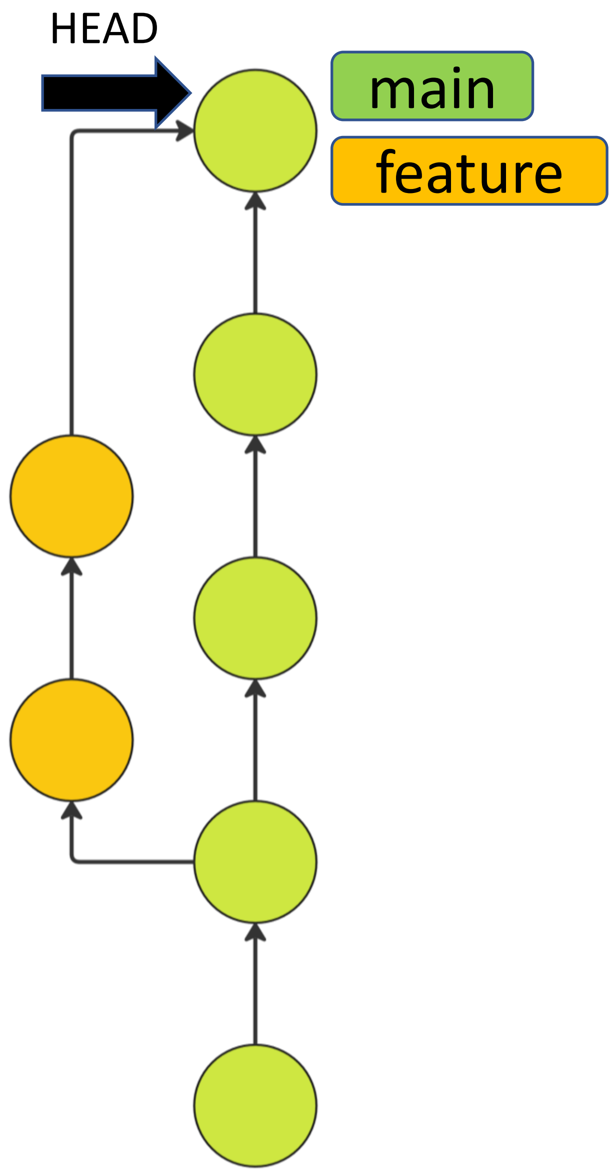

HEAD
main
HEAD
feature
FIRST COMMIT

Stashing changes

git stash allows us to save in a changes in a stack for later use

- Jump faster to a different line of development (otherwise discard or commit changes)

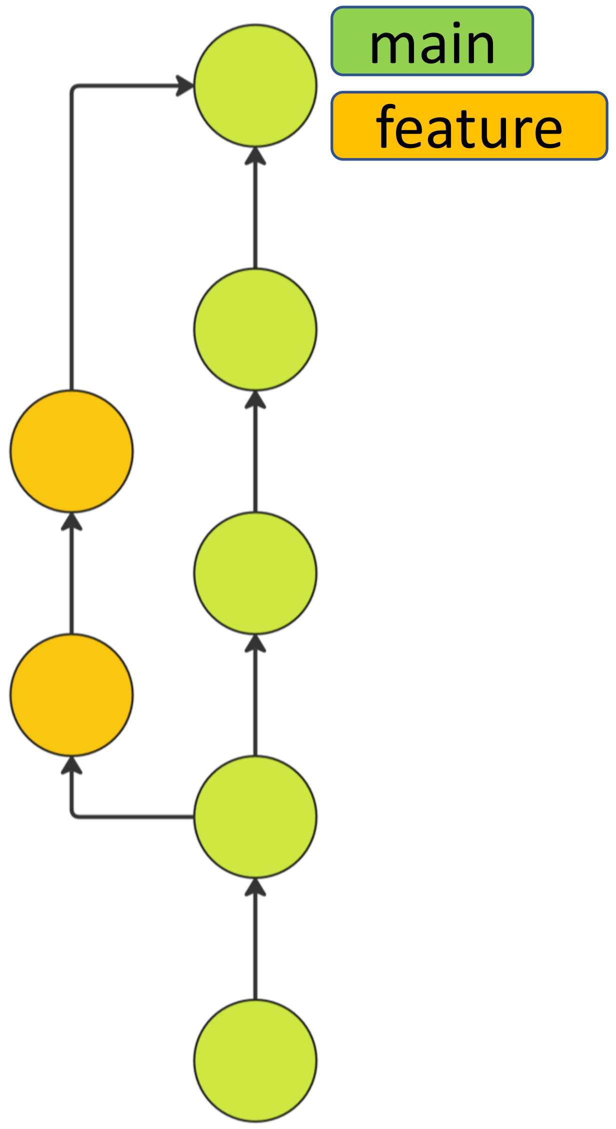
- We can pop stashed changes with git stash pop



Branches: Mixing changes

Once done working in a parallel line of development we want to integrate with other lines of development. There are 3 strategies:

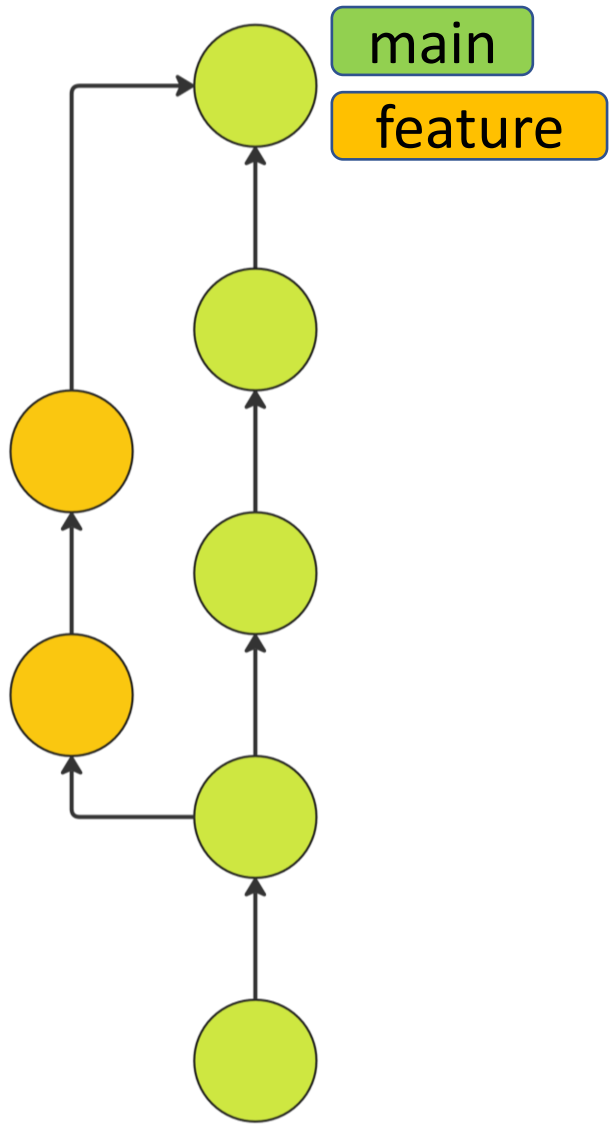
- Merge
- Rebase
- Squash and merge



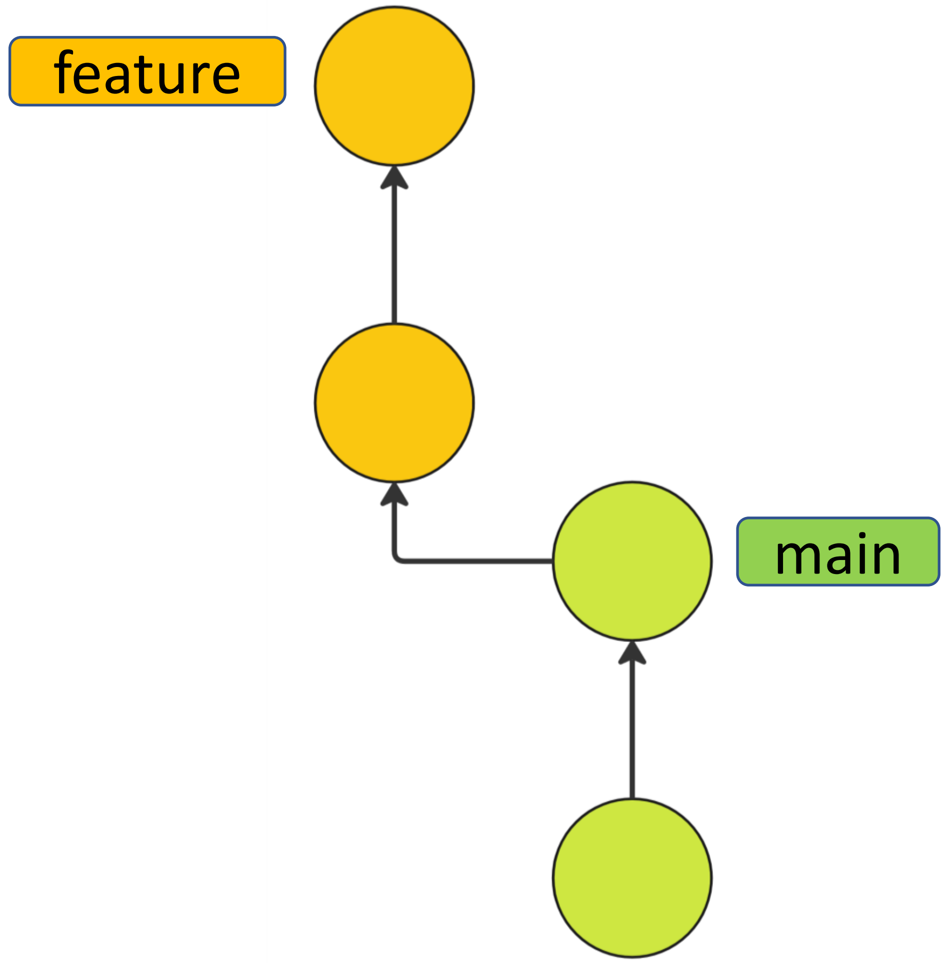
Branches: Merging

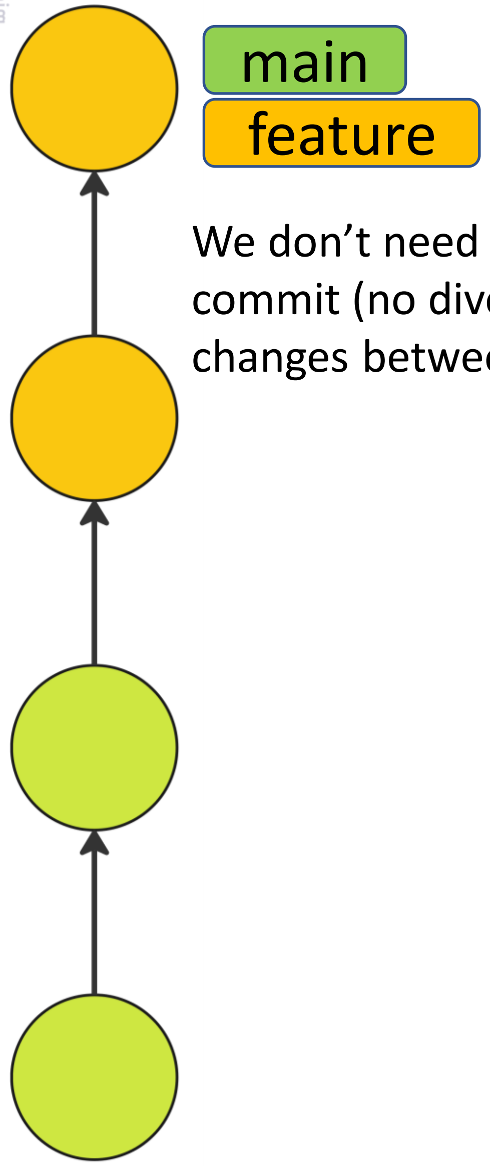
We will see 2 types of merge:

- Fast-forward (argument usually is --ff)
- 3 way merge



Merge: Fast forward



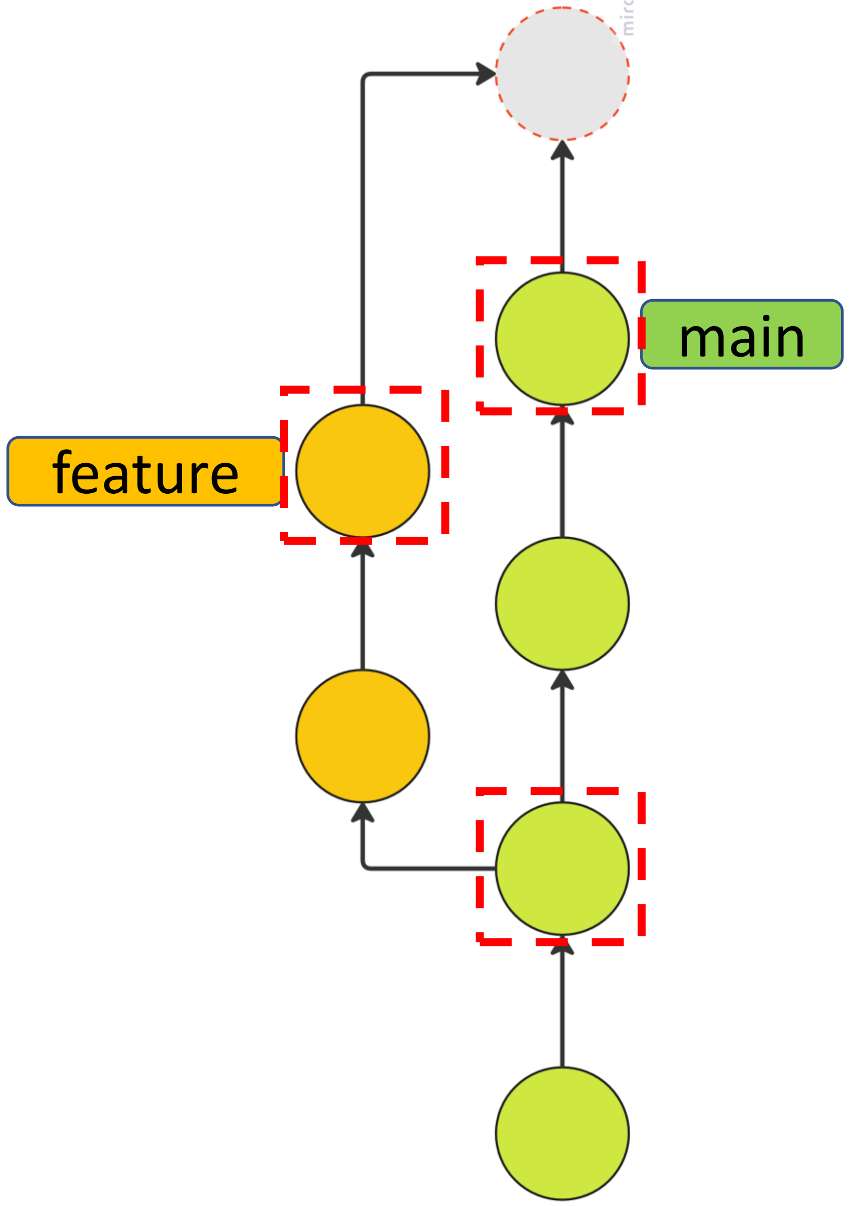


We don’t need an extra commit (no divergent changes between branches)

Merging conflicts

We can't use fast forward because there are divergent changes between branches:

- Modifications in files that are incompatible
- With git merge we enter a merge state in our working directory (git modifies conflictive files)
- We can stop this state with git merge --abort



TODO: Practice

- ✓ Amend/Reset commits
- ✓ Create branches
- ✓ Stash changes
- ✓ Merge fast forward and 3 way