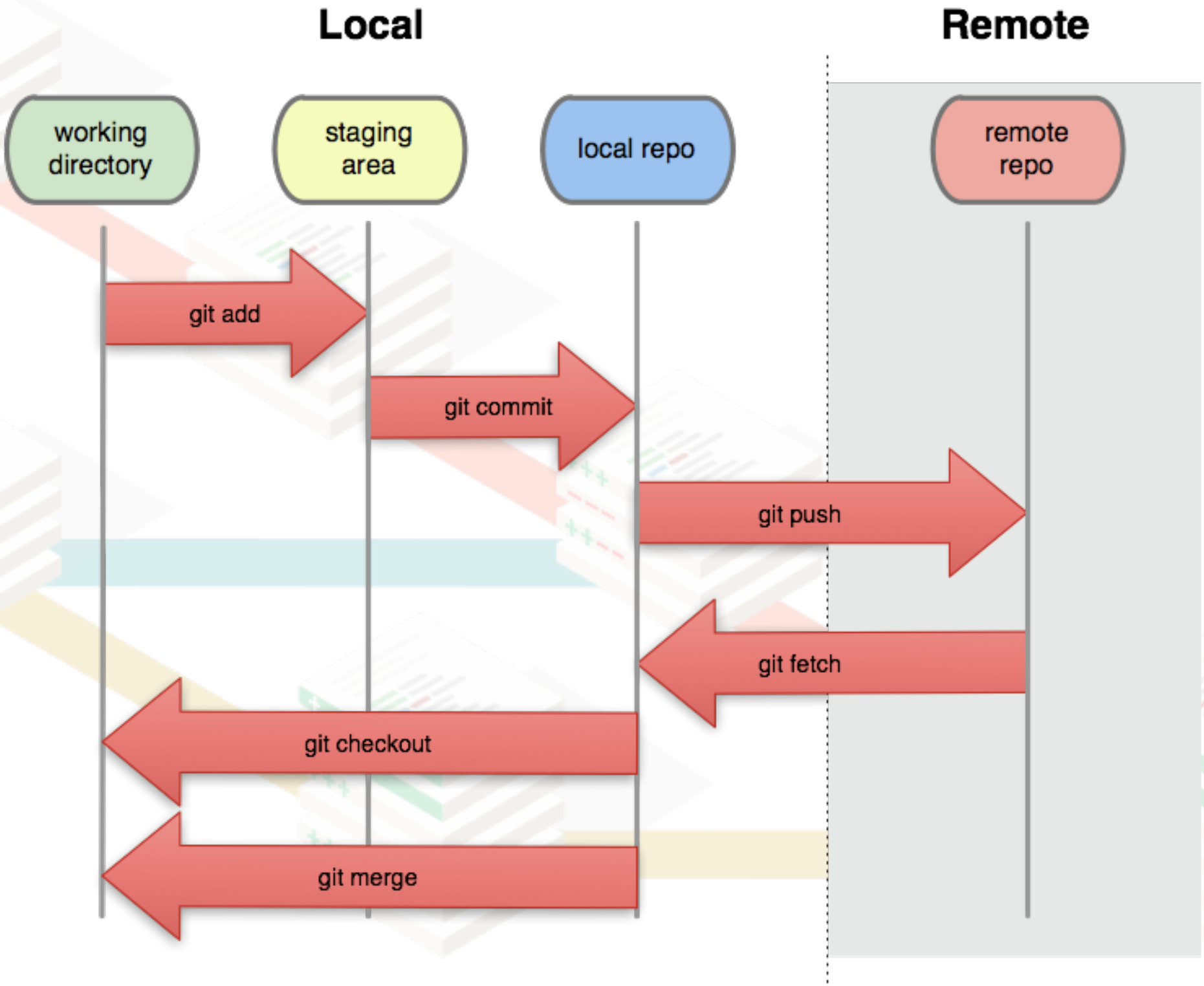
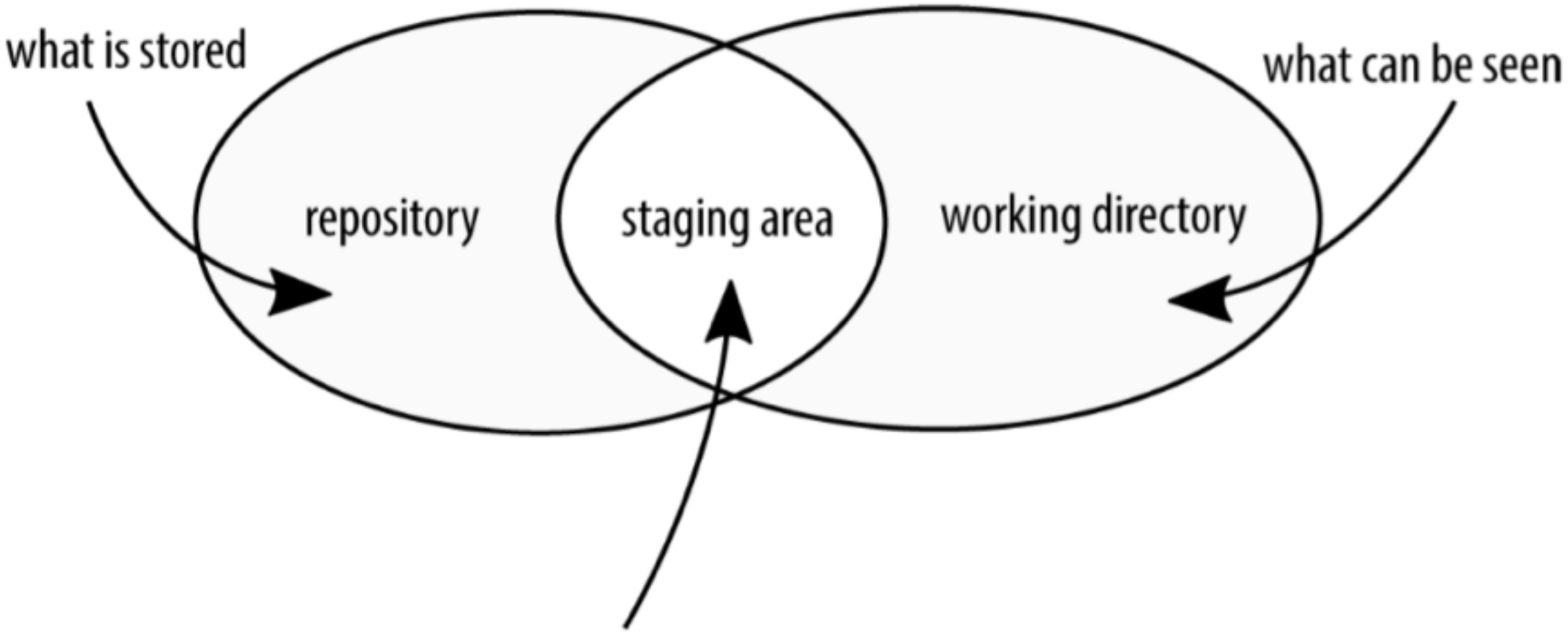
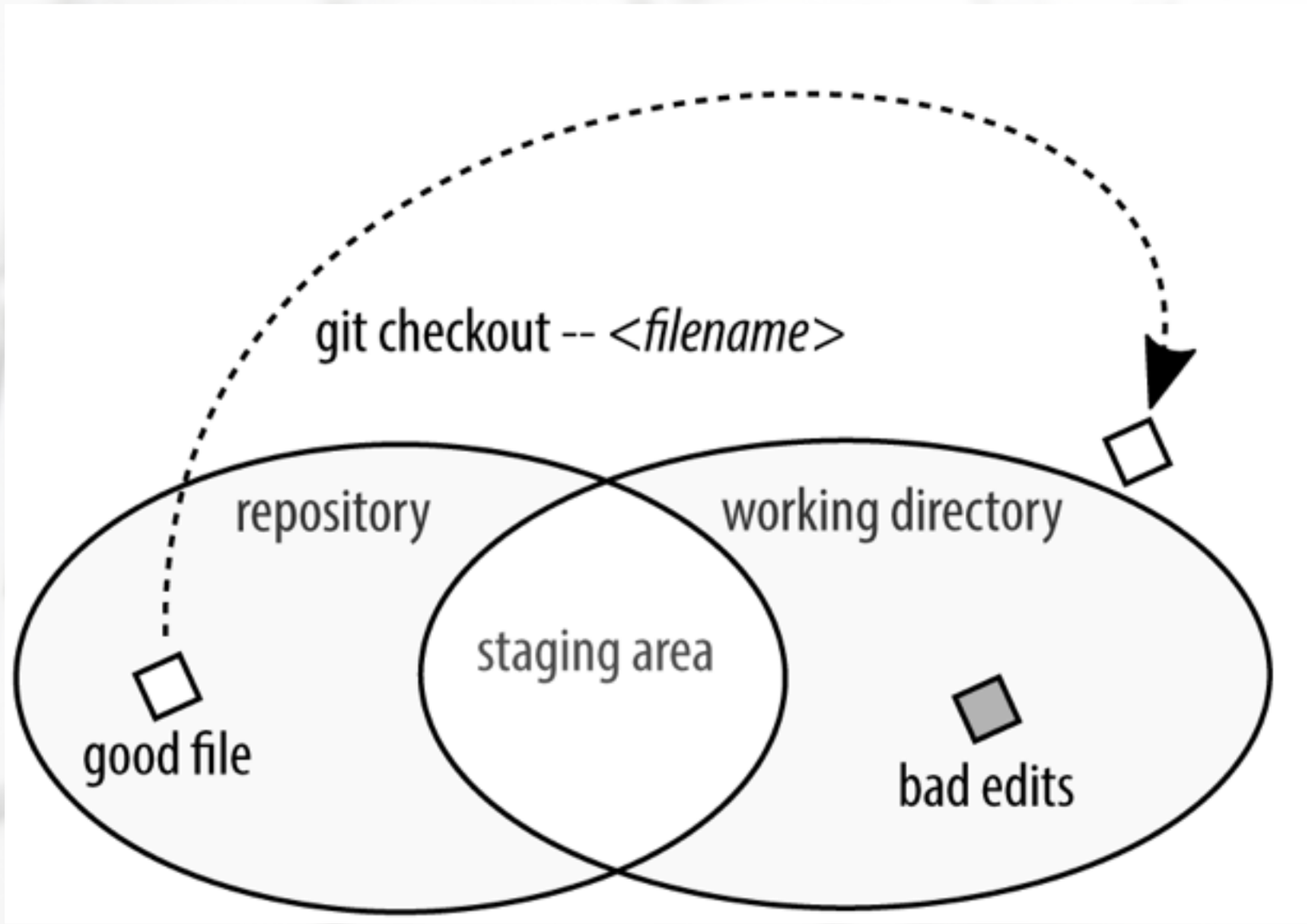
Rollback Revert Reset


Local
Remote
working directory
staging area
local repo
remote repo
git add
git commit
git push
git fetch
git checkout
git merge


what is stored
what can be seen
repository
staging area
working directory
the difference between what is stored and what is seen

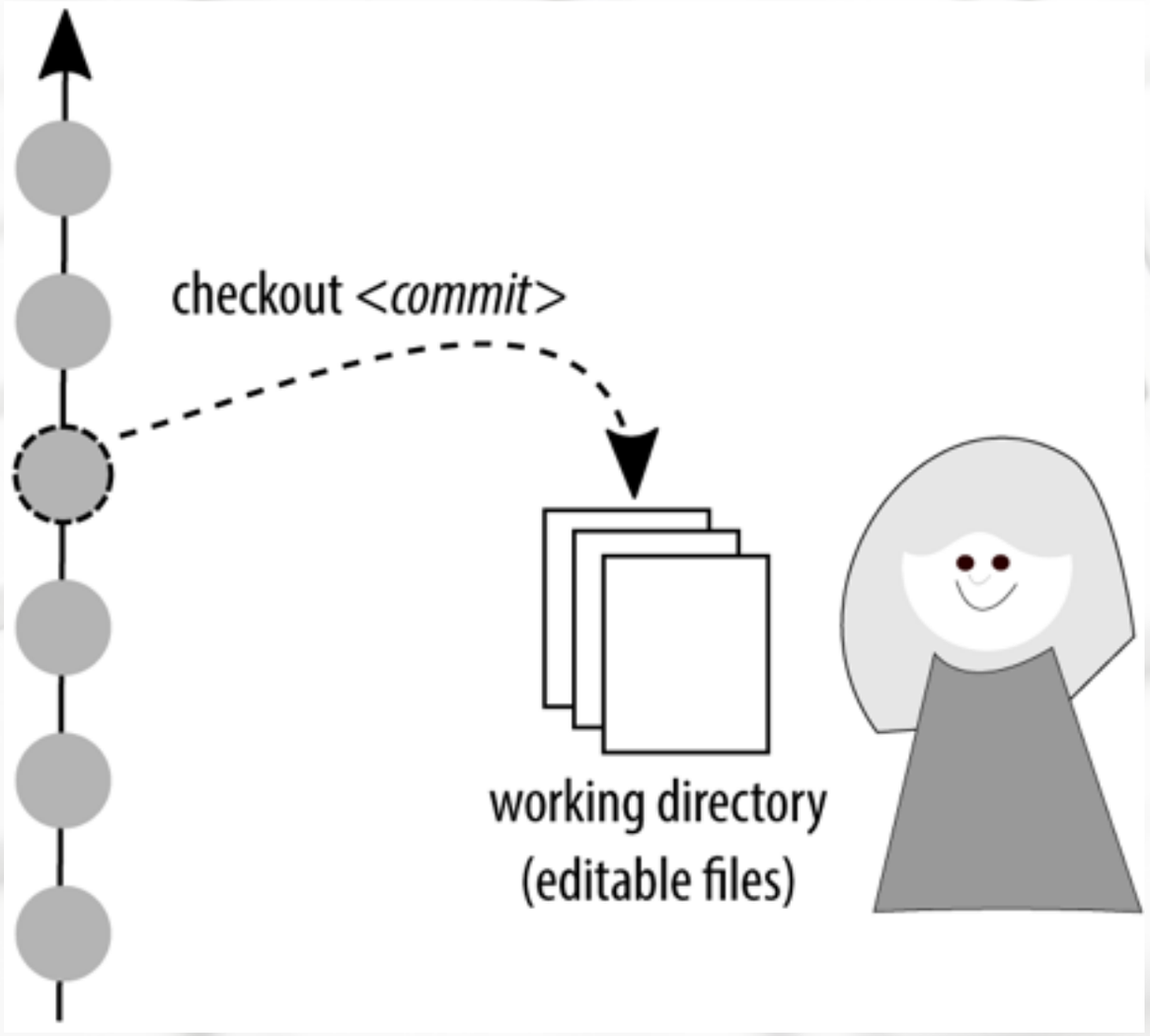
Discard change in working directory

$git **checkout -- <filename>**



View in history

$git **checkout <commit> <filename>**



Remove all files (not committed)

```
$git clean  -fd
```

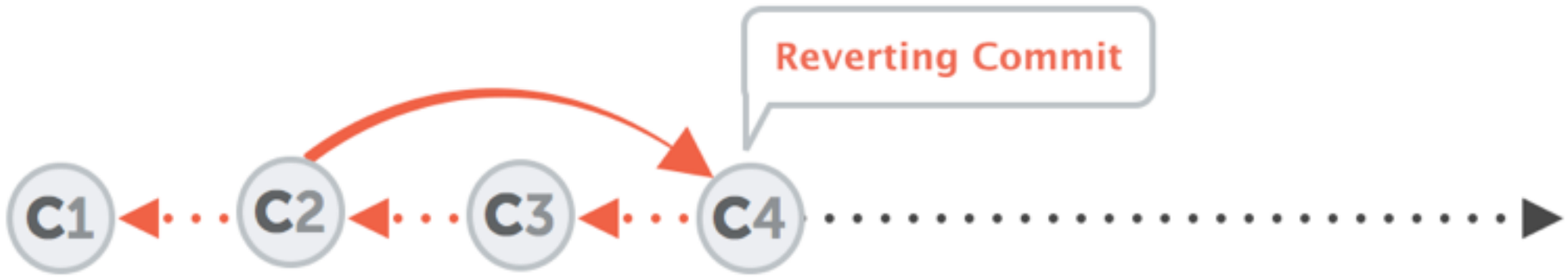
Reset vs Revert

$git **reset <commit>**

$git **revert <commit>**

Revert

$git **revert <commit>**

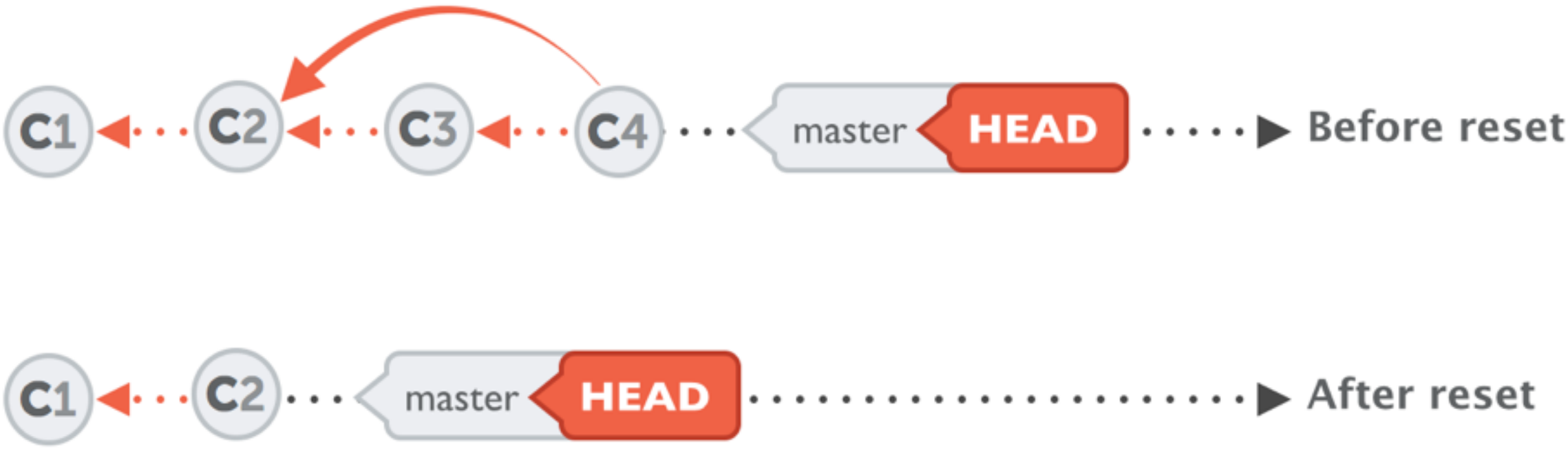


Modified “index.html”:
(old) About This Project
(new) About

Modifies “index.html” in the “opposite” way:
(old) About
(new) About This Project

Reset

$git **reset <commit>**



Restore files

```
$touch for_delete
```

```
$git add .
```

```
$git commit
```

```
$rm for_delete
```

Restore files

$git status

```
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        deleted:    for_delete

no changes added to commit (use "git add" and/or "git commit -a")
```

try to restore file for_delete !!

Restore files again !!

```
$rm for_delete
```

```
$git add for_delete
```

Restore files again !!

$git status

```
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        deleted:    for_delete
```

try to restore file for_delete !!

Restore files again !!

$git **reset HEAD** for_delete

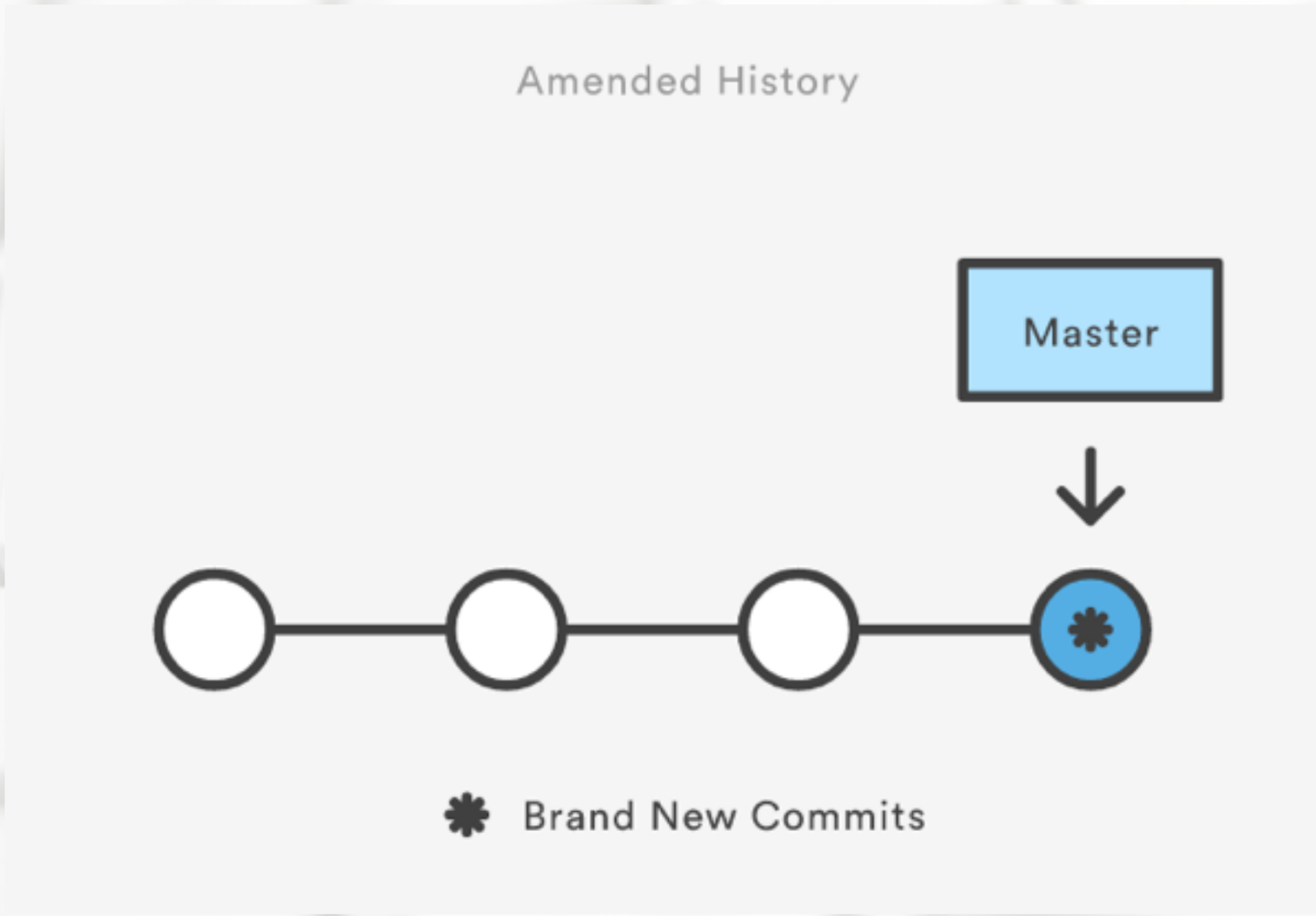
$git **checkout --** for_delete

Restore files in one line

```
$git reset --hard HEAD
```

Working with commit

$git **commit --amend**

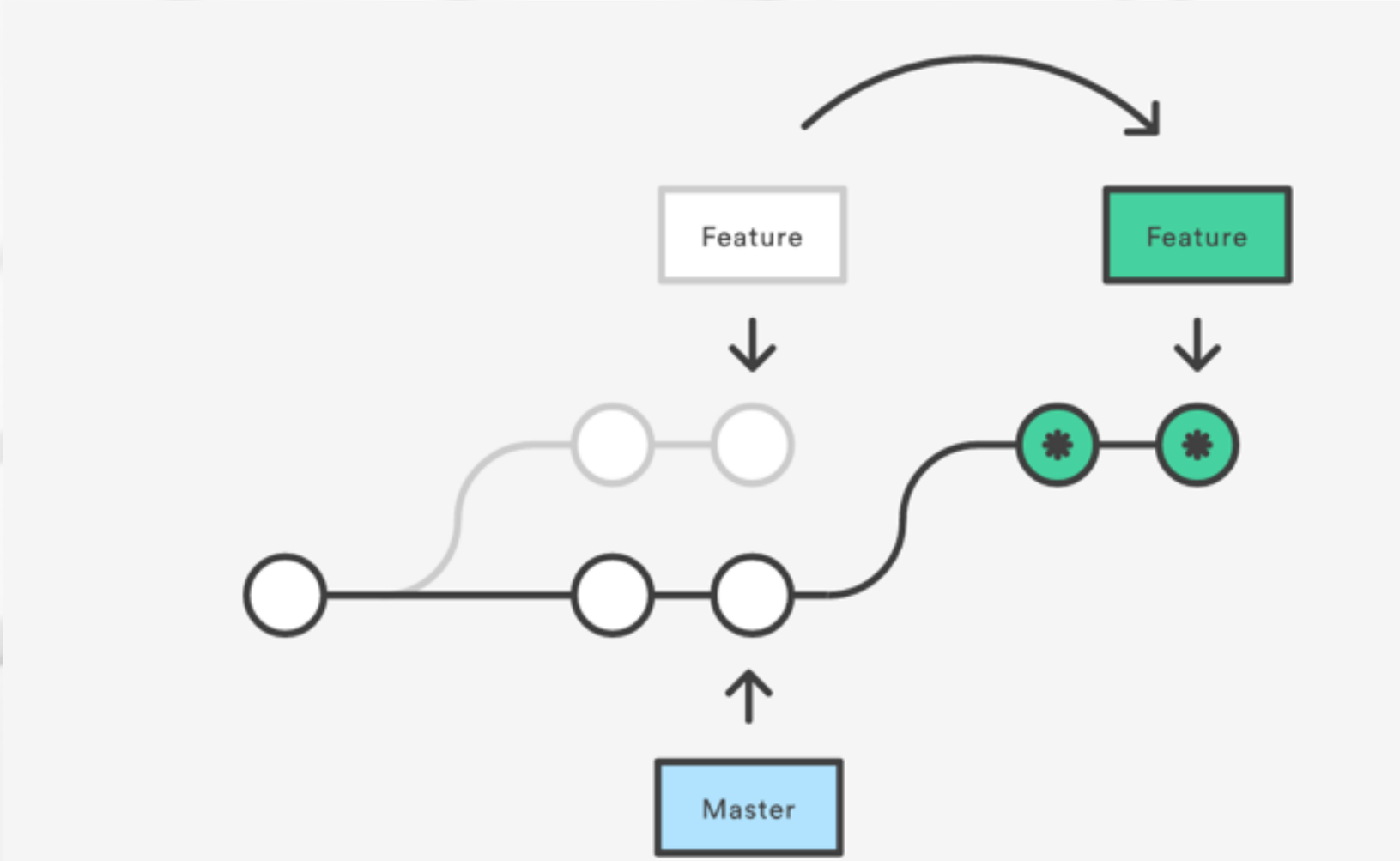


Without changing commit message

```
$git commit --amend --no-edit
```

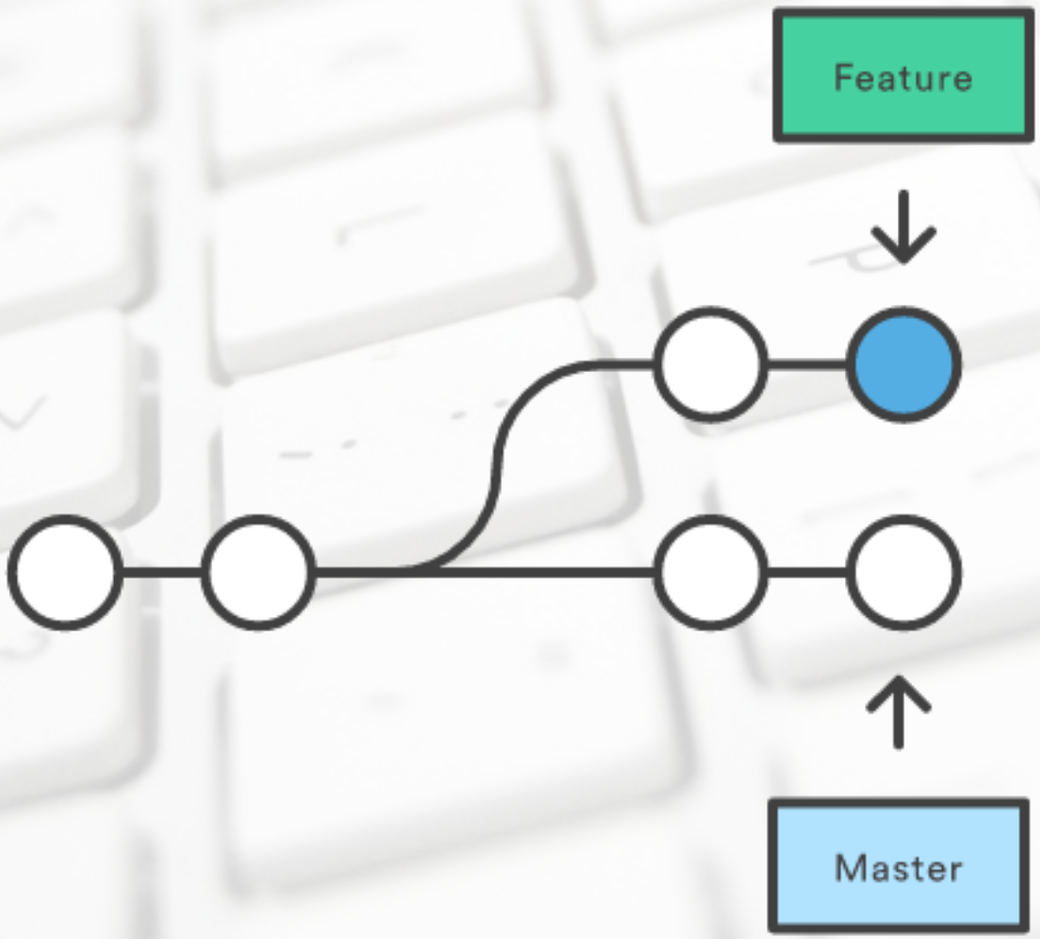
Rebase

$git **rebase <base>**

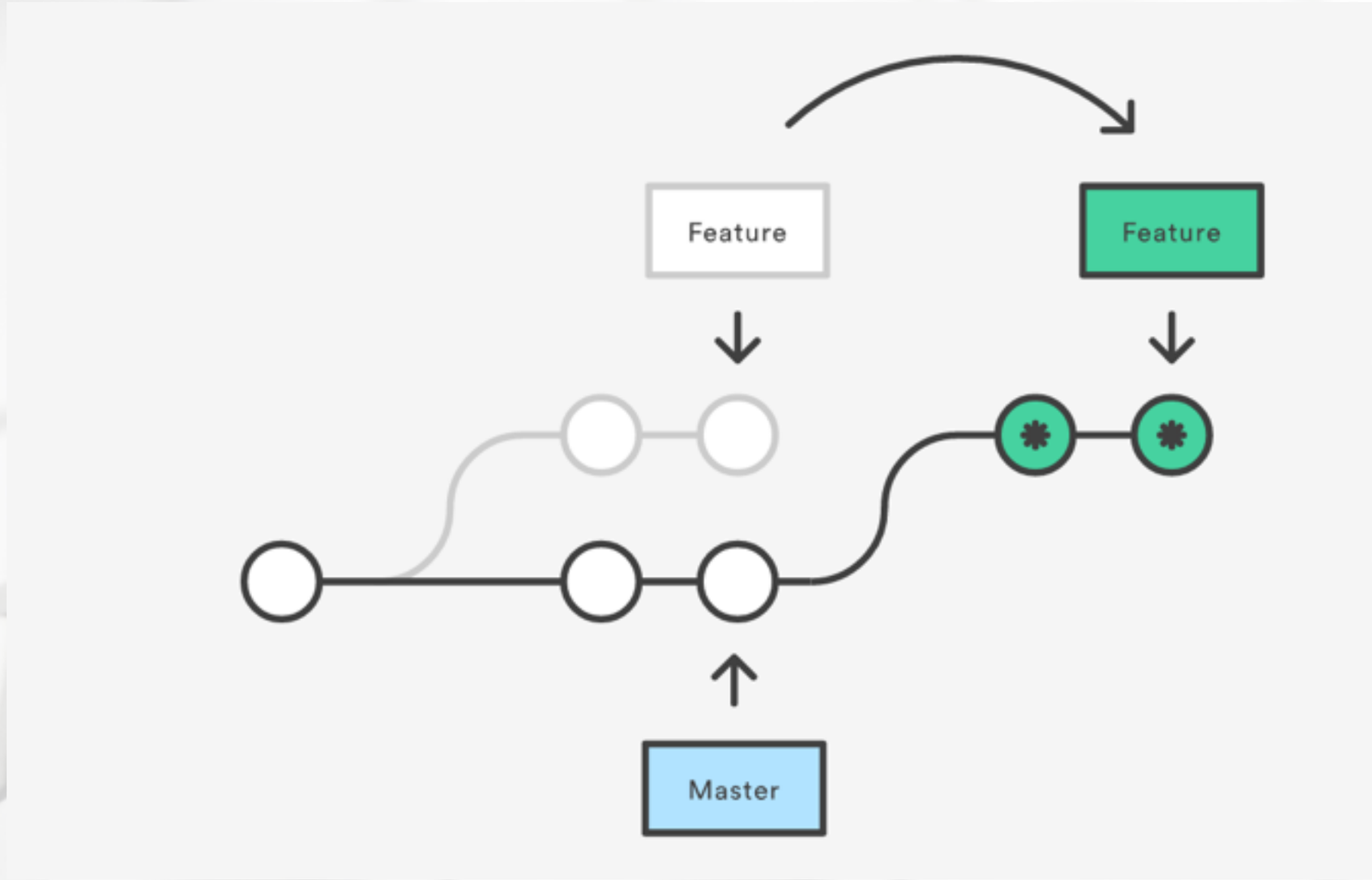


Workshop Rebase

try by yourself



1. Rebase feature to master



Workshop Rebase

$git **checkout feature**

$git **rebase master**

2. Merge to master

Workshop Rebase

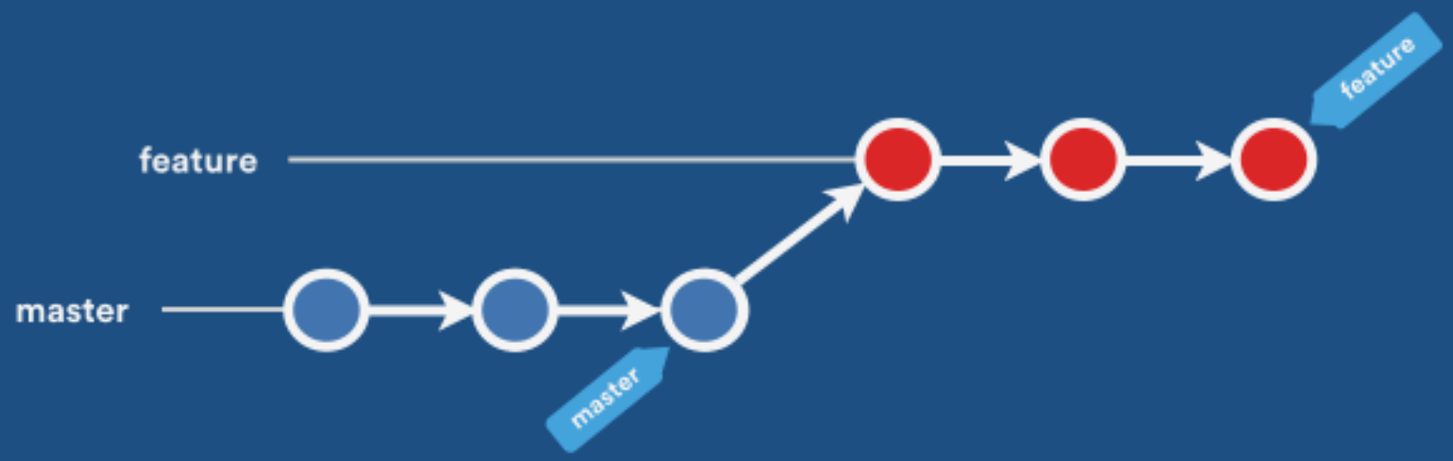
$git **checkout master**

$git **merge feature**

Squash

What is squash on merge?

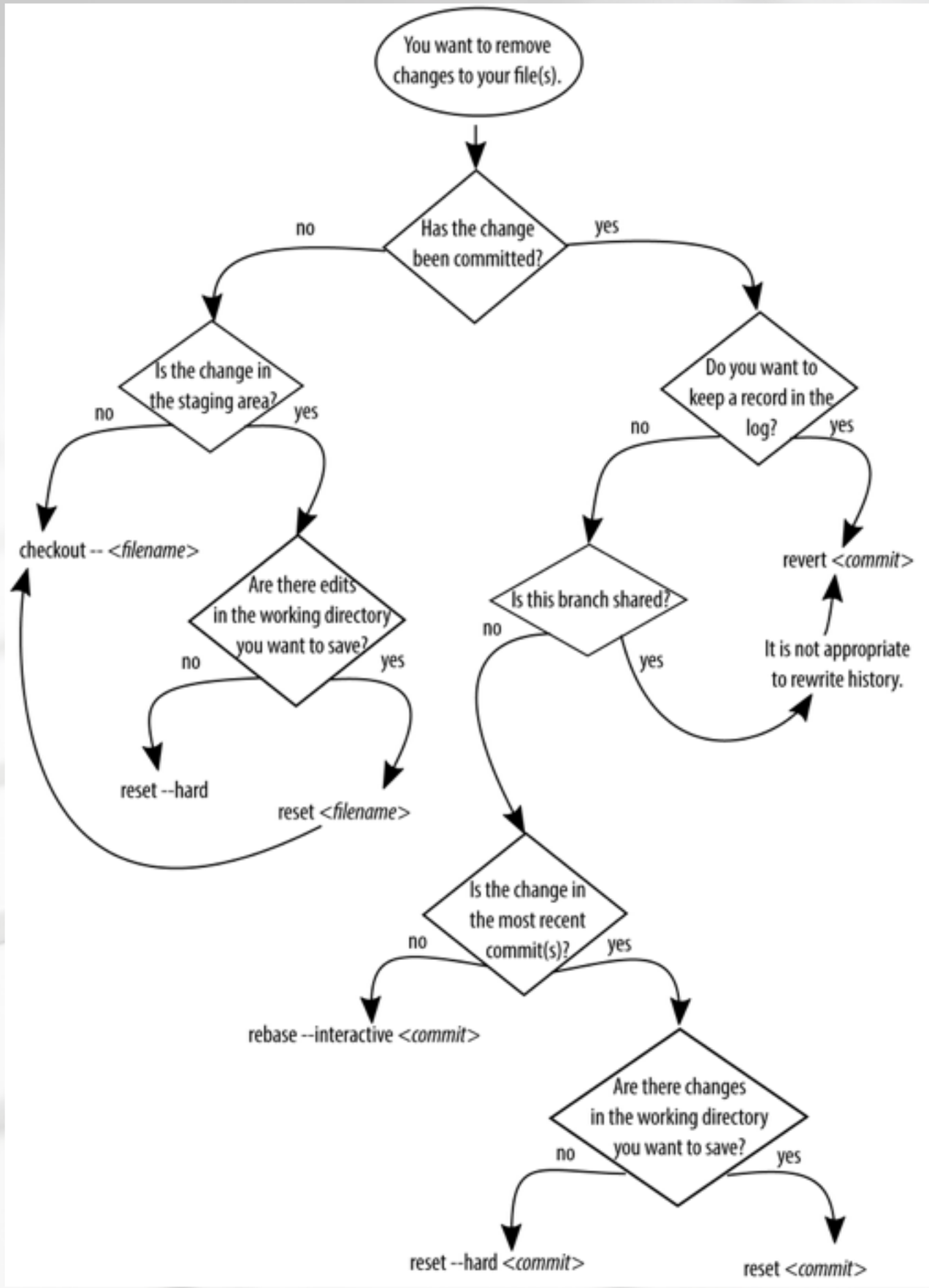
It will compact feature commits
into one before merging

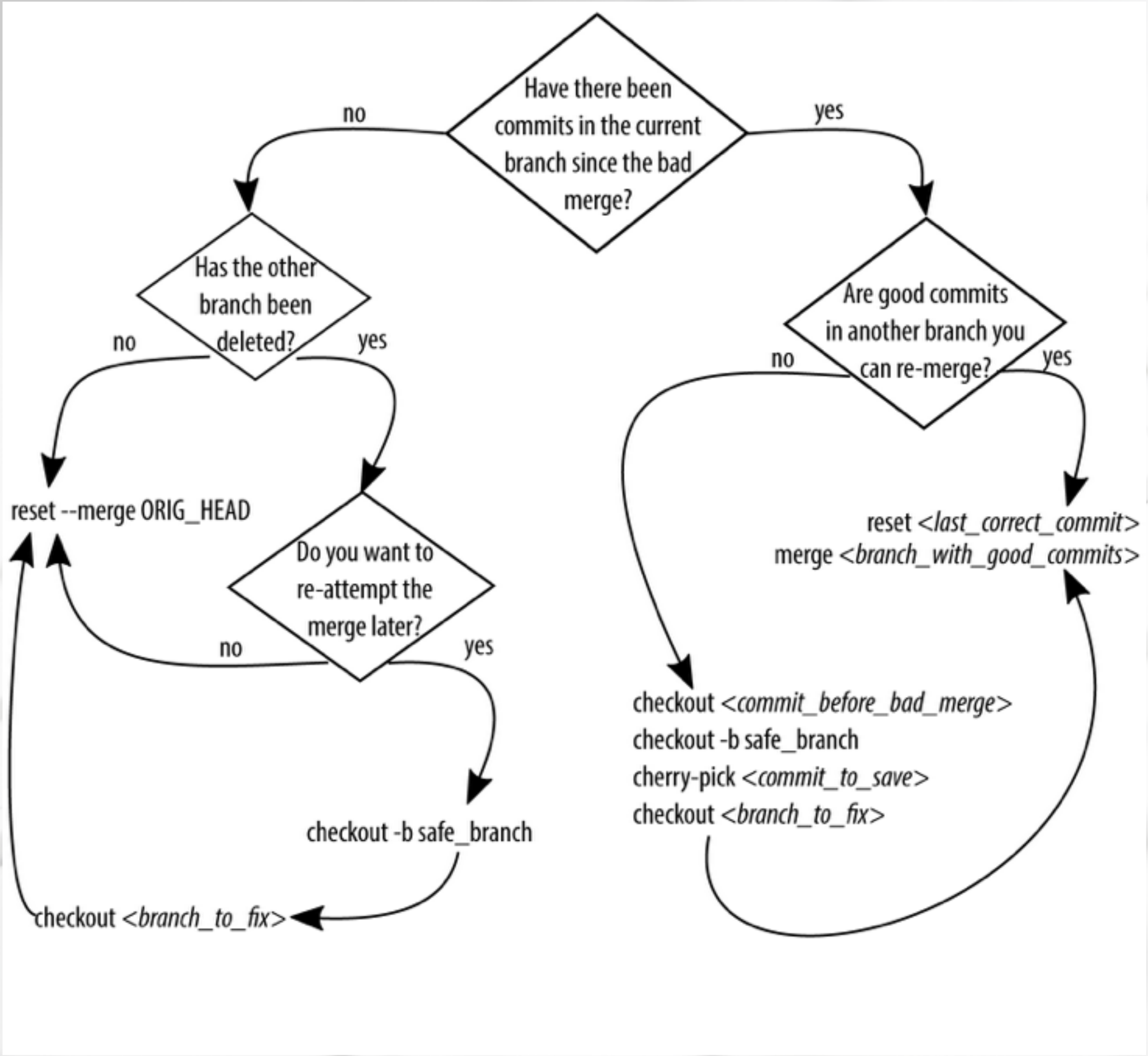


Squash

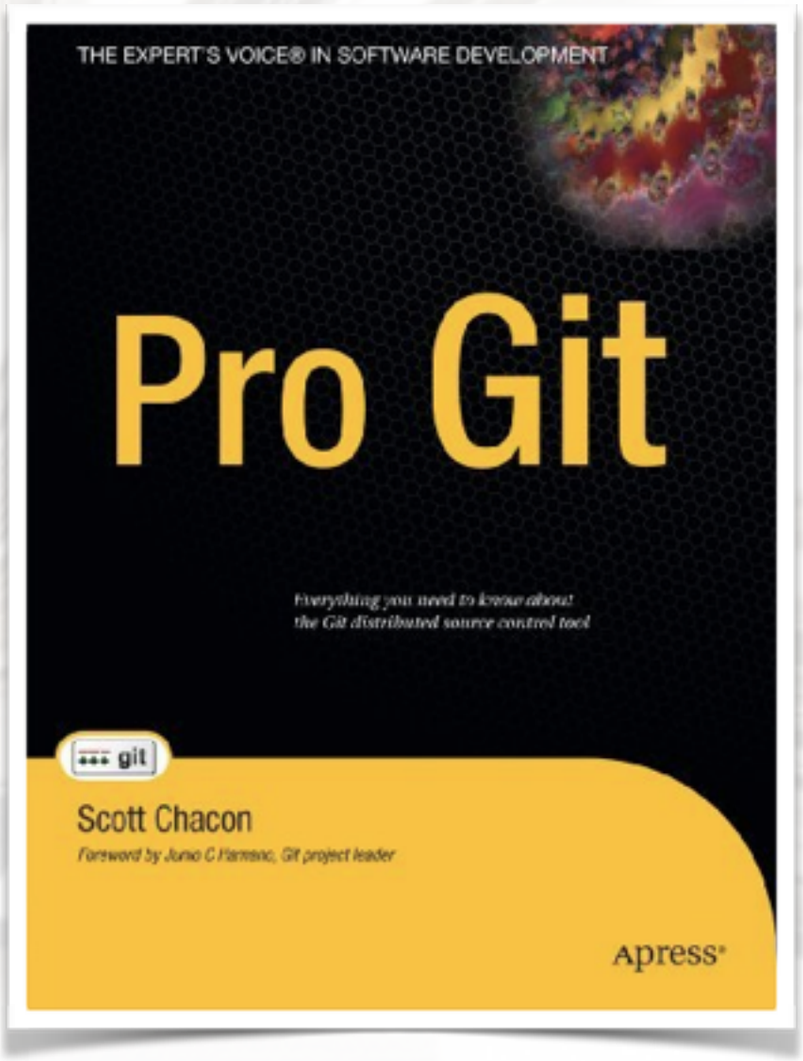
What is squash on merge?

It will compact feature commits
into one before merging

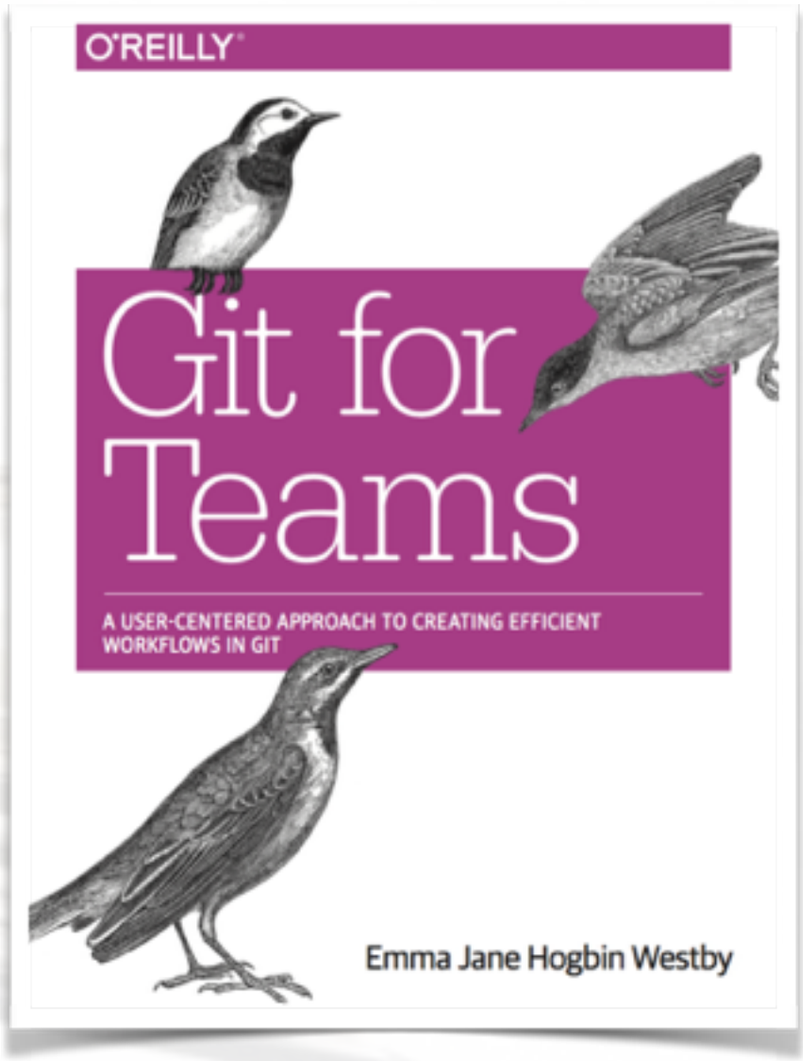

You want to remove changes to your file(s).
Has the change been committed?
no
yes
Is the change in the staging area?
no
yes
checkout -- <filename>
Are there edits in the working directory you want to save?
no
yes
reset --hard
reset <filename>
Do you want to keep a record in the log?
no
yes
revert <commit>
Is this branch shared?
no
yes
It is not appropriate to rewrite history.
Is the change in the most recent commit(s)?
no
yes
rebase --interactive <commit>
Are there changes in the working directory you want to save?
no
yes
reset --hard <commit>
reset <commit>


Have there been commits in the current branch since the bad merge?
no
yes
Has the other branch been deleted?
no
yes
Are good commits in another branch you can re-merge?
no
yes
reset --merge ORIG_HEAD
Do you want to re-attempt the merge later?
no
yes
checkout -b safe_branch
checkout <branch_to_fix>
reset <last_correct_commit>
merge <branch_with_good_commits>
checkout <commit_before_bad_merge>
checkout -b safe_branch
cherry-pick <commit_to_save>
checkout <branch_to_fix>

Books to Read and Practice



http://www.git-scm.com/book/en/v2



http://shop.oreilly.com/product/0636920034520.do