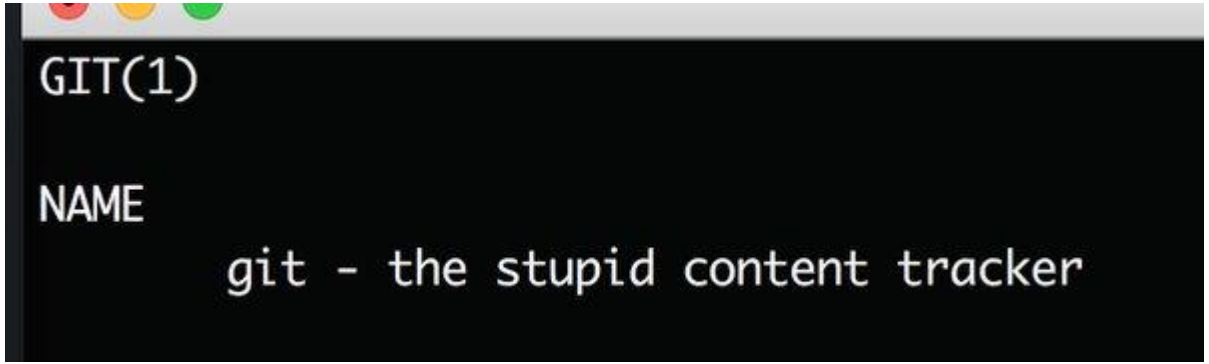
GIT

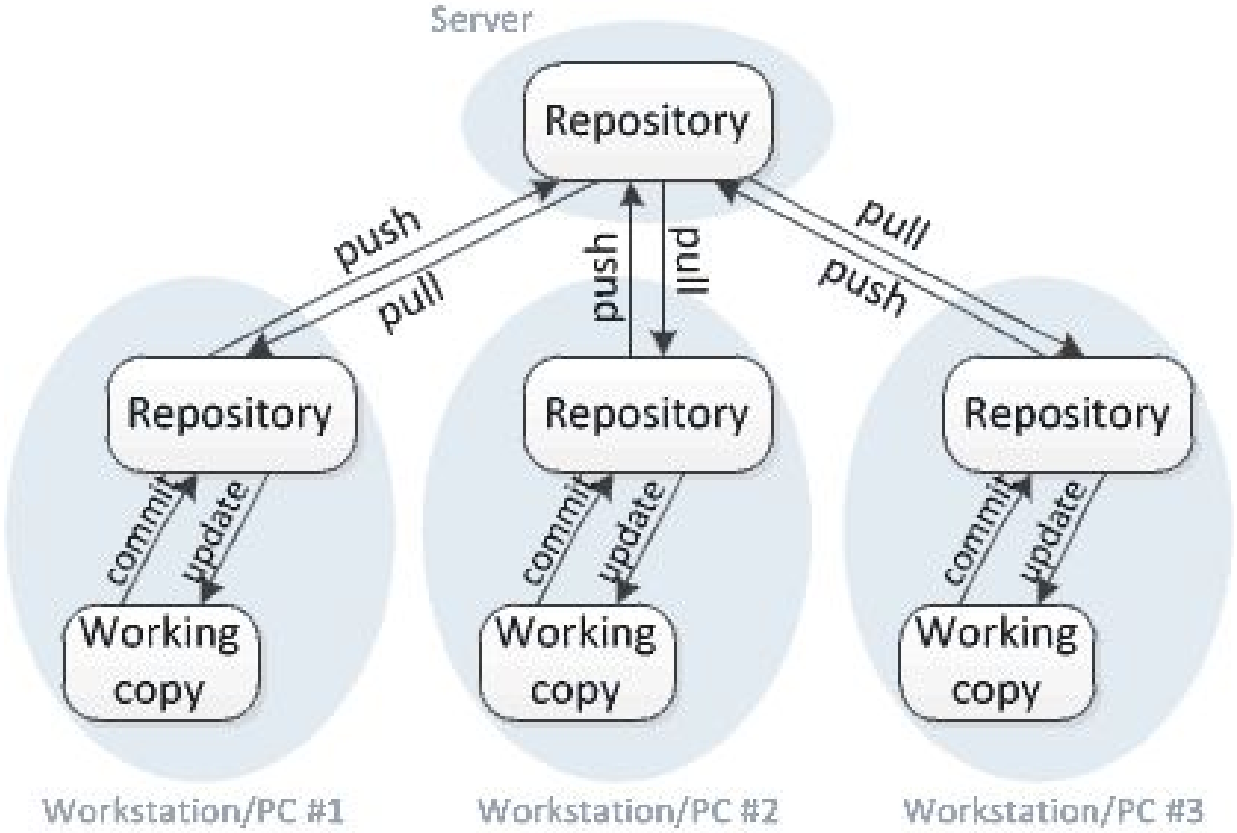
Jana Kostičová, 25.4.2016

Basic features

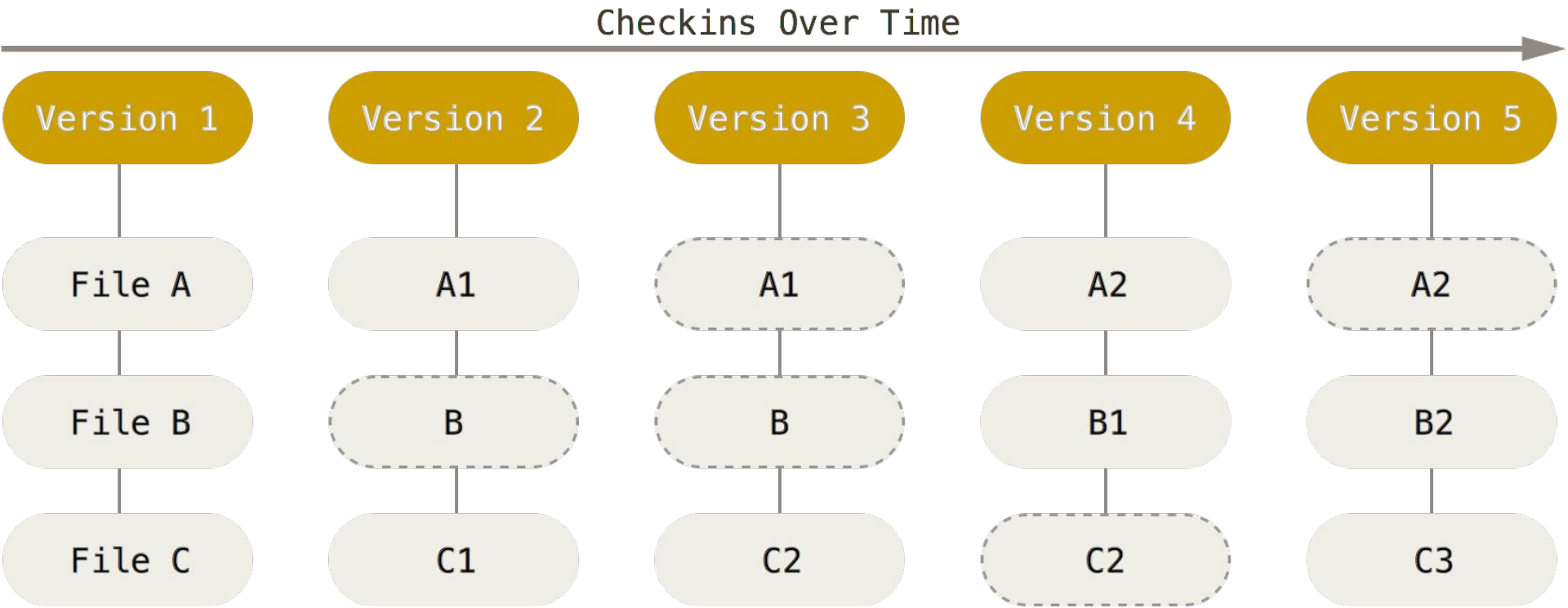


- Distributed version control
- Developed in 2005, originally for Linux kernel development
- Free, GNU General Public License version 2
- Available for Linux, BSD, Solaris, OS X, Microsoft Windows
- Installation: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

Distributed version control

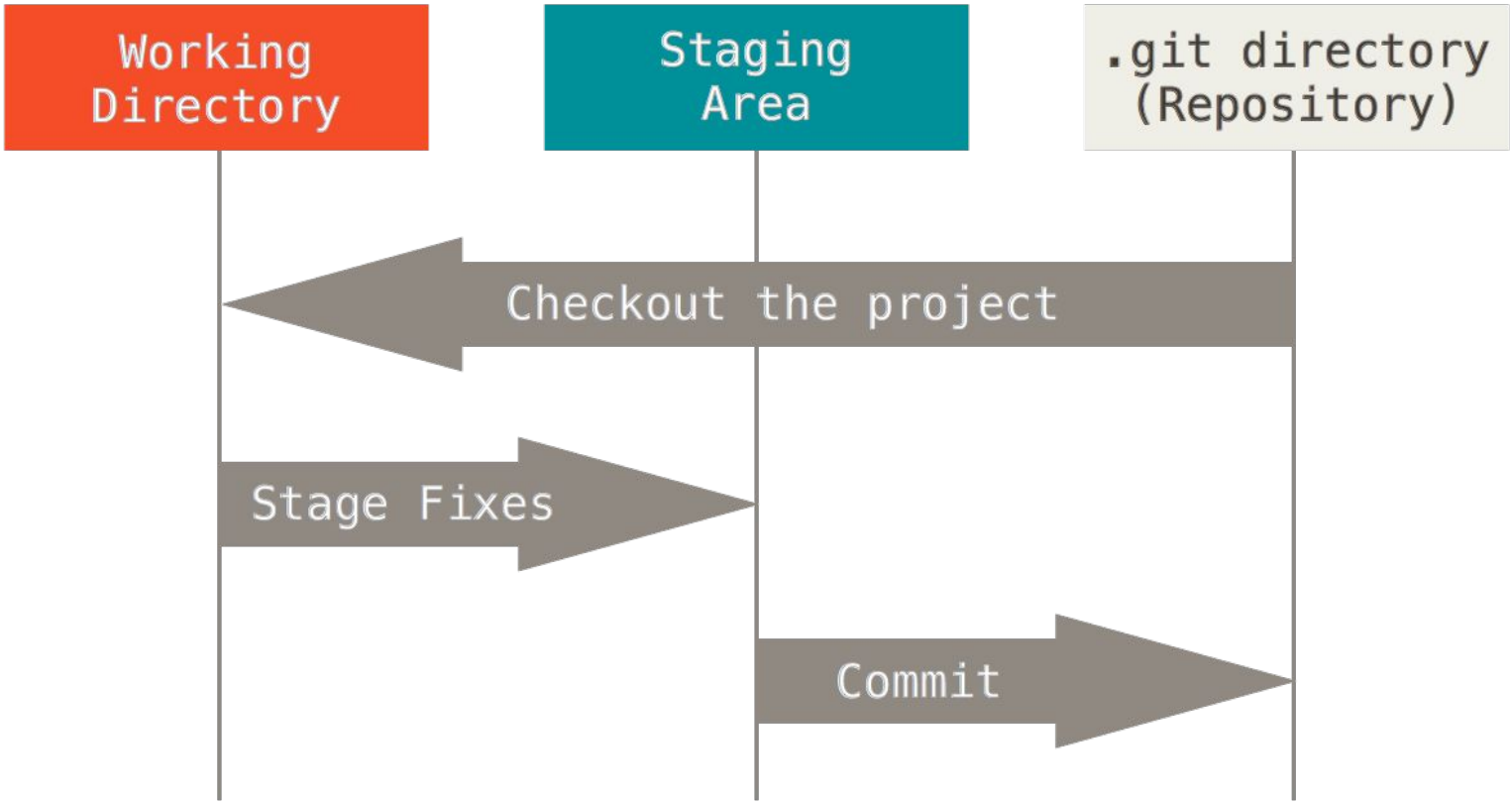


Storing data as snapshots

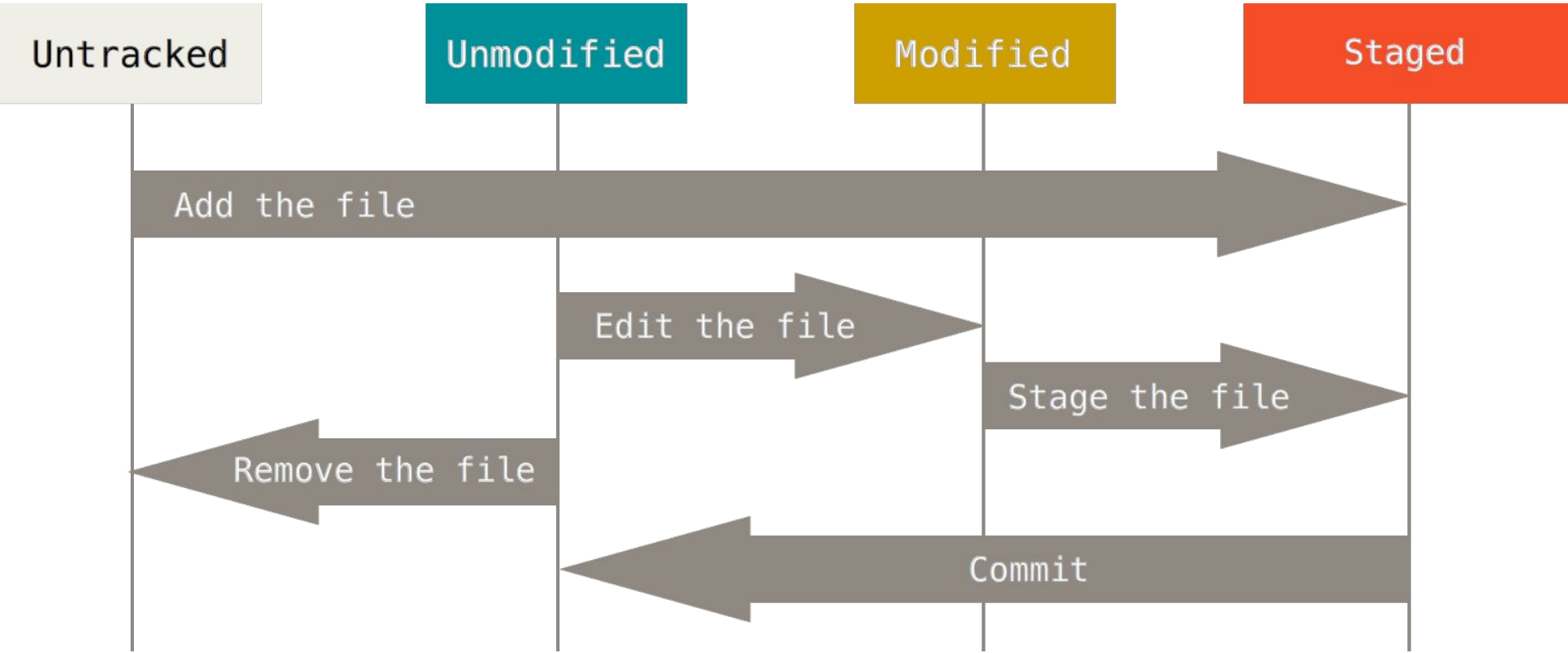


Basic snapshotting

Areas


Working Directory
Staging Area
.git directory (Repository)
Checkout the project
Stage Fixes
Commit

Working directory - file states



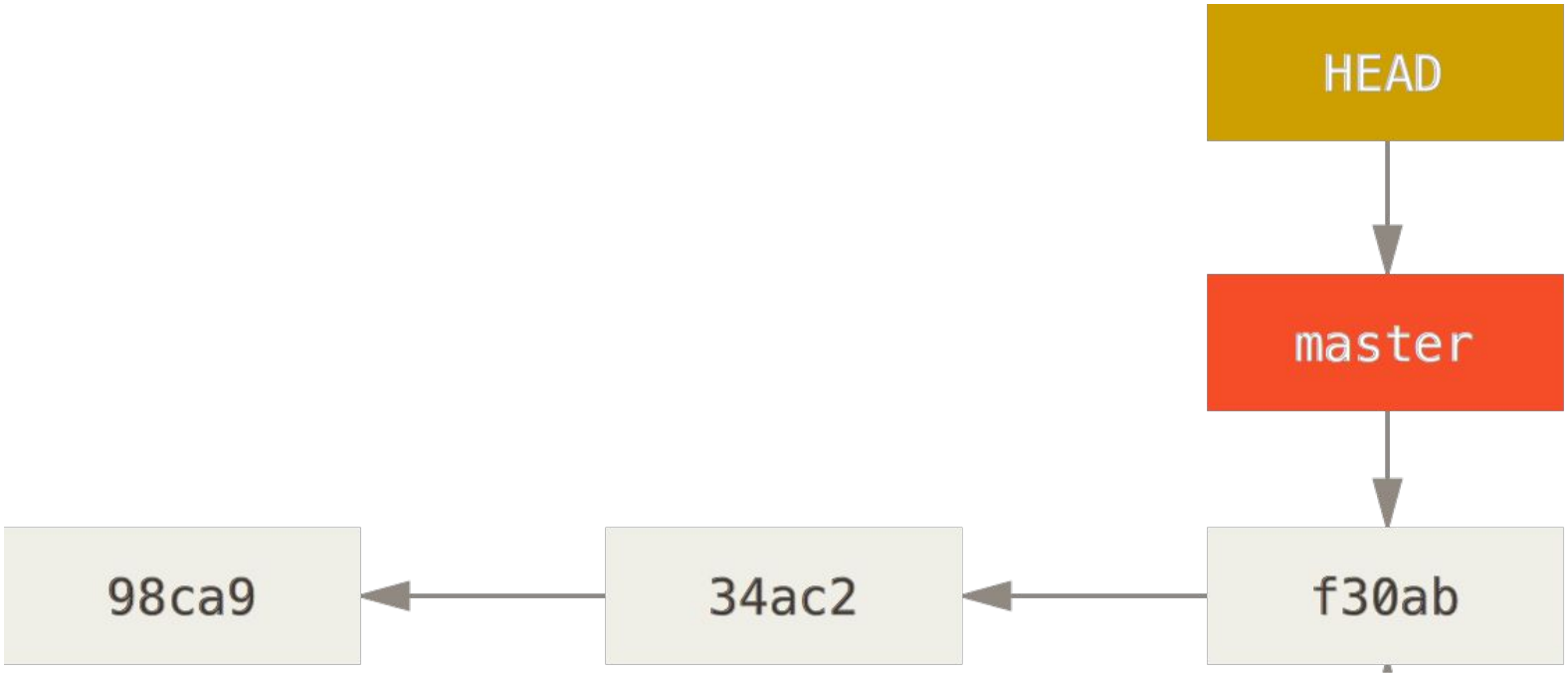
Basic snapshotting - summary

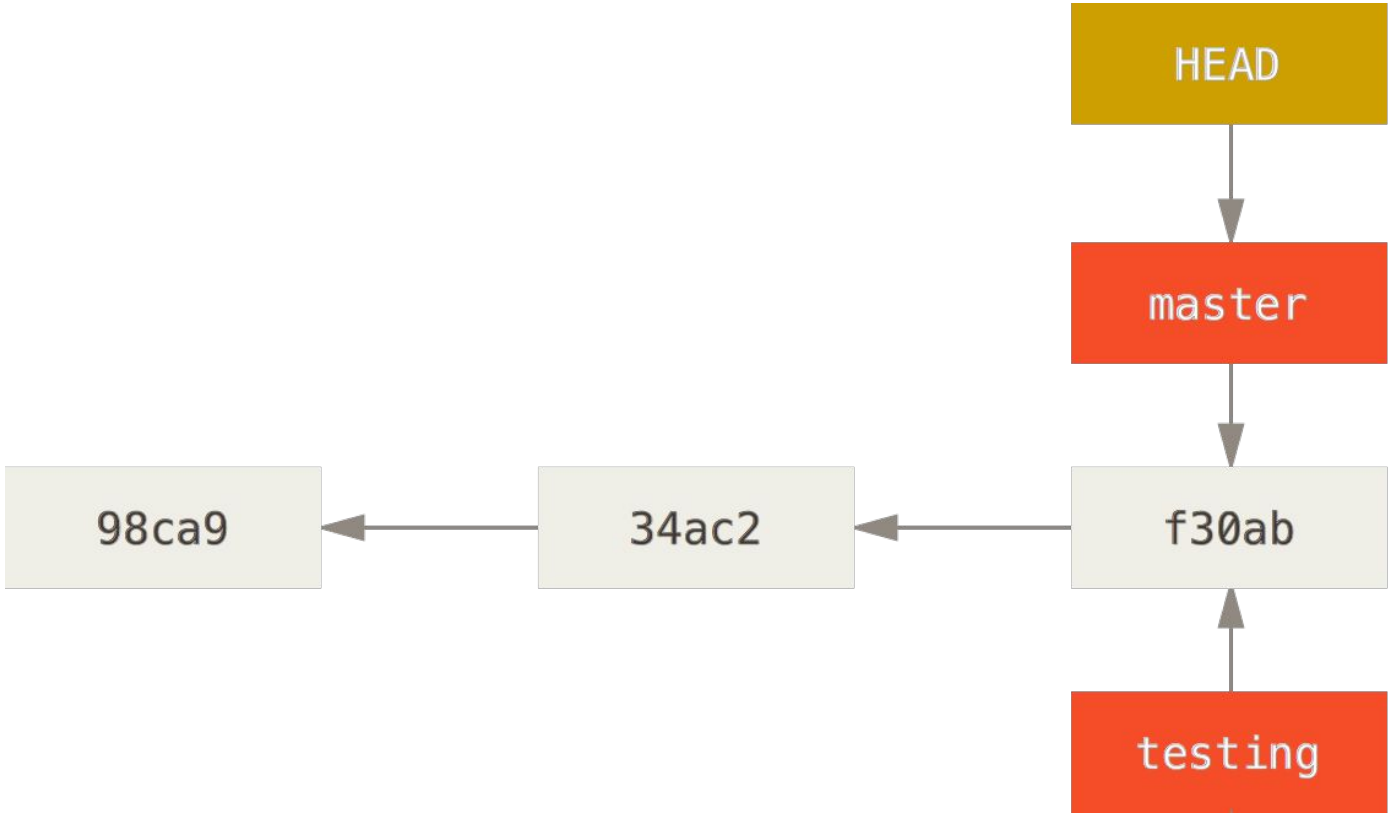
- **git init::** creates a new (empty) local repository
- **git add [path]:** adds new/modified files to the staging area (next commit)
- **git commit -m “[message]”:** commits staged files into the local repository with the specified commit message
- **git status:** shows status of the files in the working directory
- **git log**: displays history of commits in reverse chronological order: author’s name and e-mail, the date written and the commit message
- **gitk, gitk --all**: simple built-in gui for viewing history

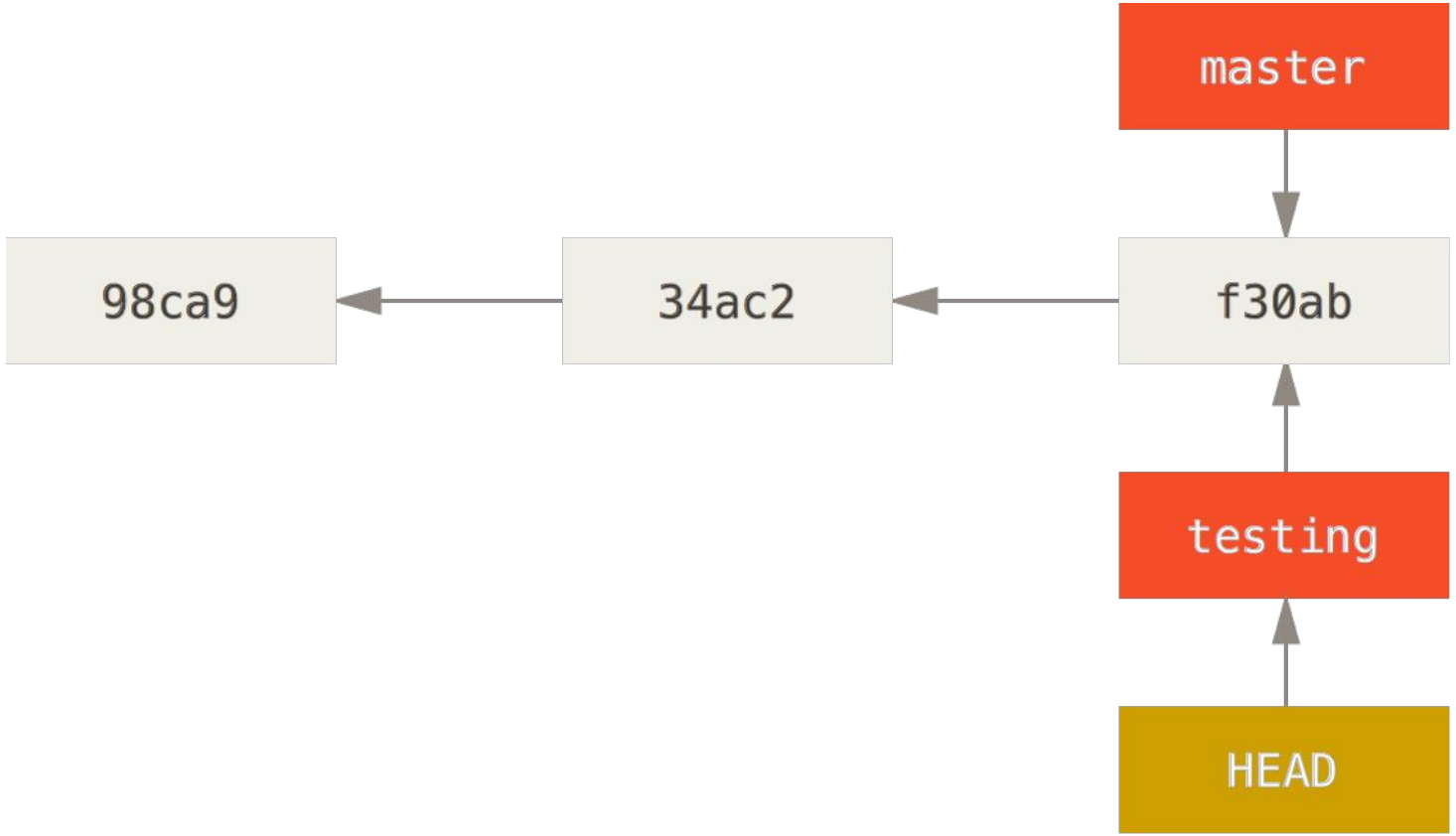
Basic snapshotting - other useful commands

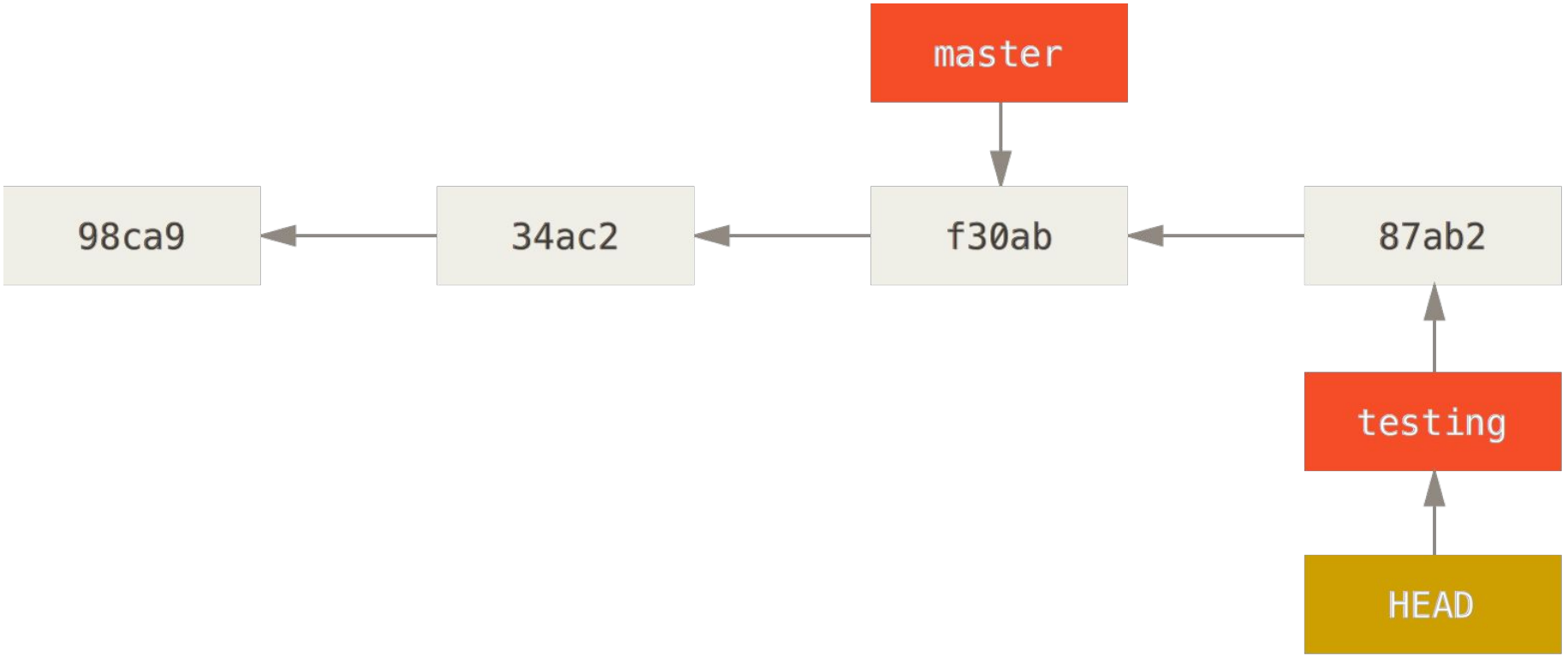
- **git commit -a:** shortcut, automatically stage all tracked, modified files and commits them
- **git status -s (=status --short):** shows short version of status
- **.gitignore:** file which defines file types ignored by git
- **git rm:** removes files
- **git mv:**: moves/renames files
- **git gui**: simple built-in gui for snapshotting
- **git log -p, -- [filename], -N, --stat, --pretty=oneline, --pretty=format: [format], --graph, --since, -S:** various log formats and filters
- **git diff:** compares the working directory to the staging area
- **git diff --staged (=git --cached):** compares staged changes to the last commit

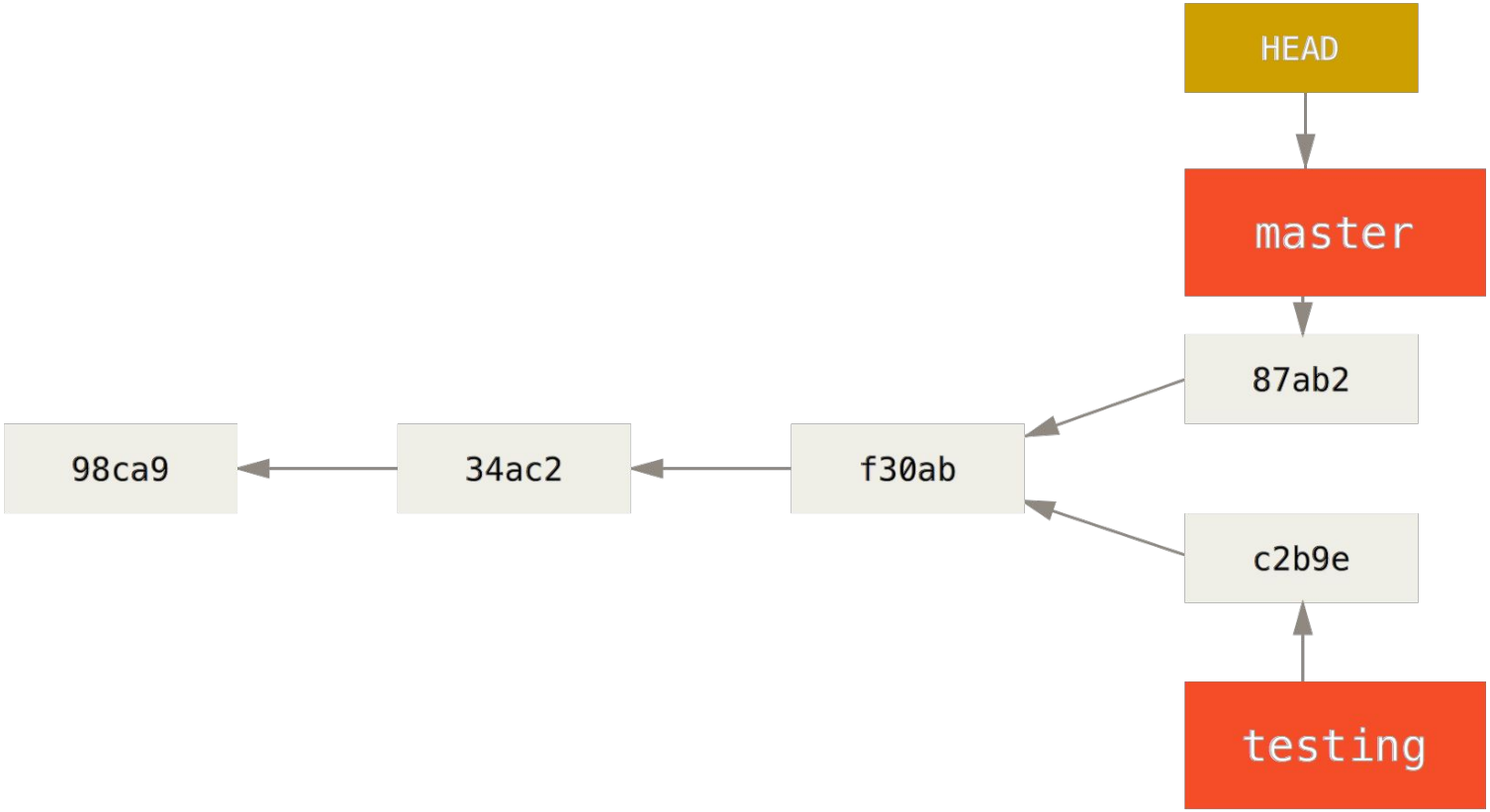
Branching

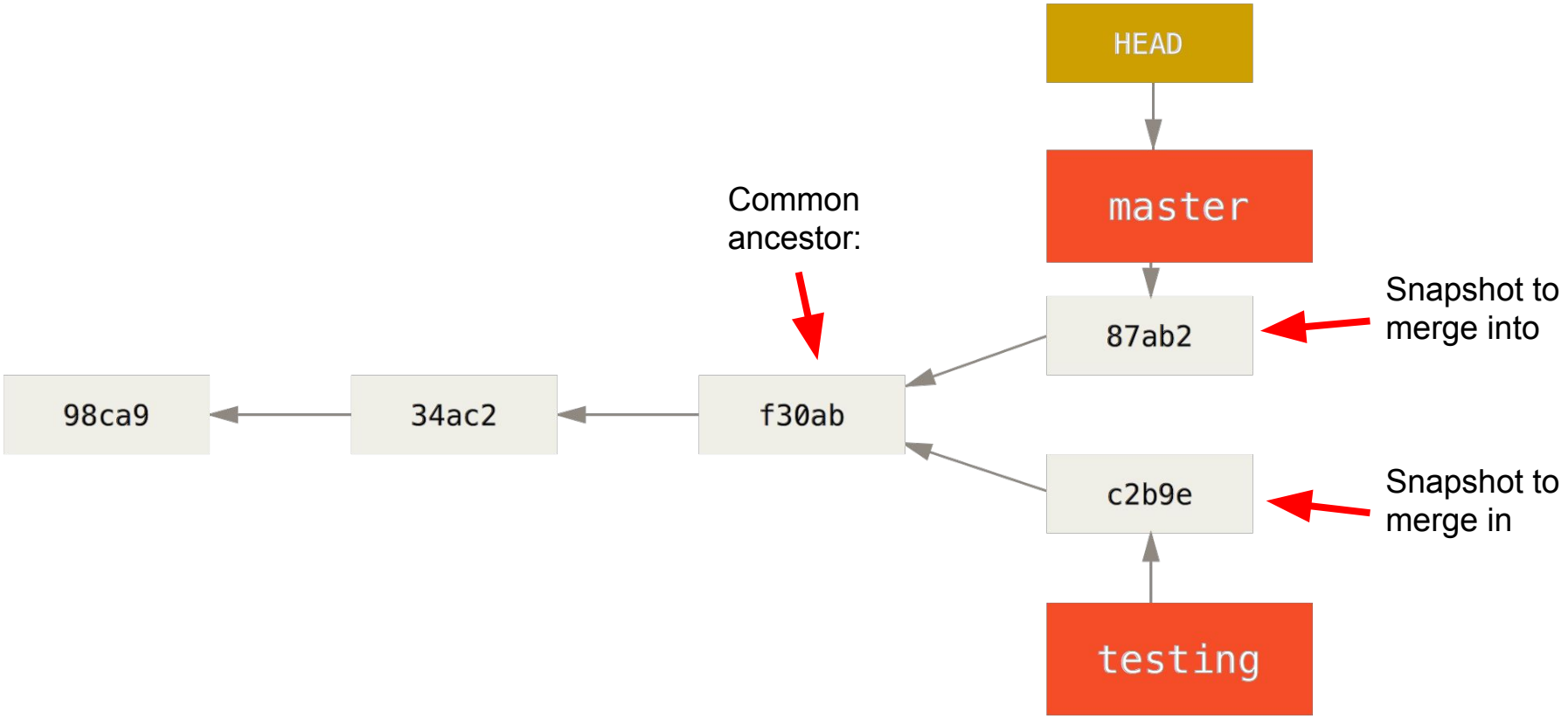

HEAD
master
98ca9
34ac2
f30ab

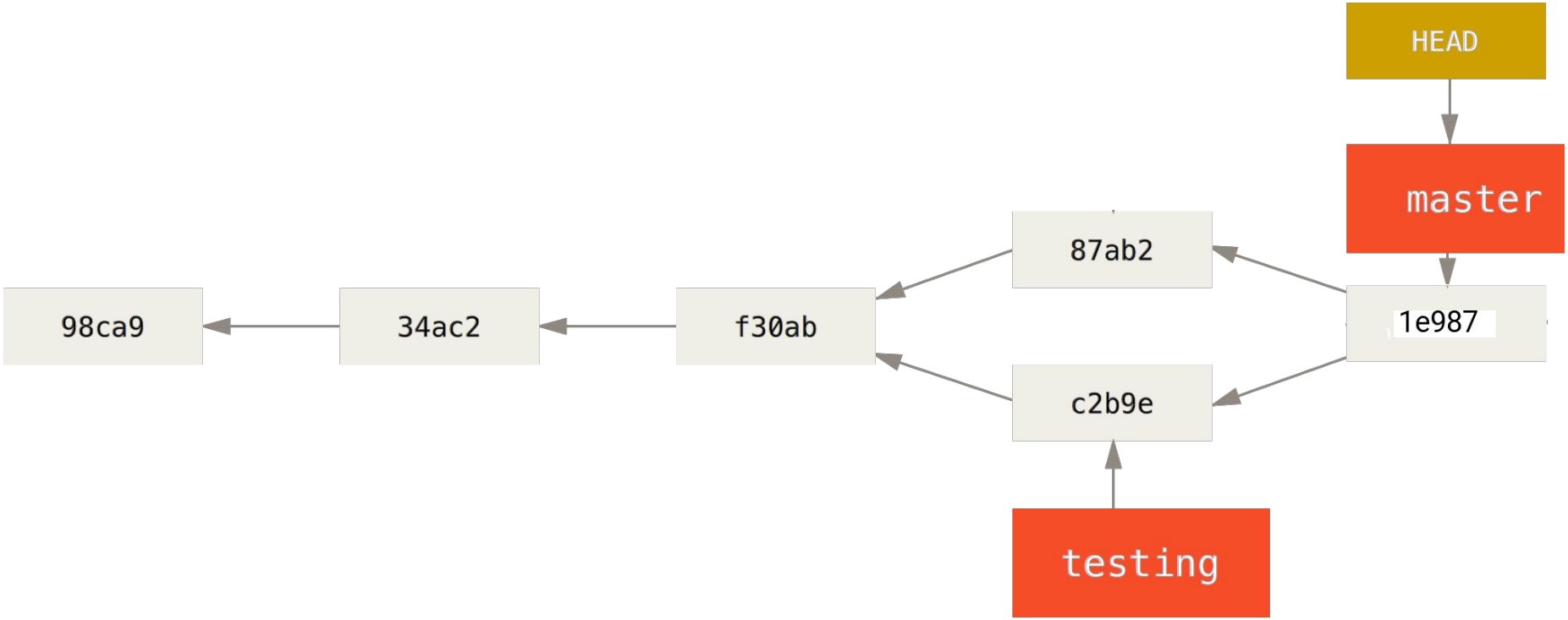

HEAD
master
98ca9
34ac2
f30ab
testing


master
98ca9
34ac2
f30ab
testing
HEAD


master
98ca9
34ac2
f30ab
87ab2
testing
HEAD


HEAD
master
87ab2
98ca9
34ac2
f30ab
c2b9e
testing


HEAD
master
Common
ancestor:
Snapshot to
merge into
87ab2
98ca9
34ac2
f30ab
c2b9e
Snapshot to
merge in
testing


HEAD
master
87ab2
98ca9
34ac2
f30ab
1e987
c2b9e
testing

Branching - summary

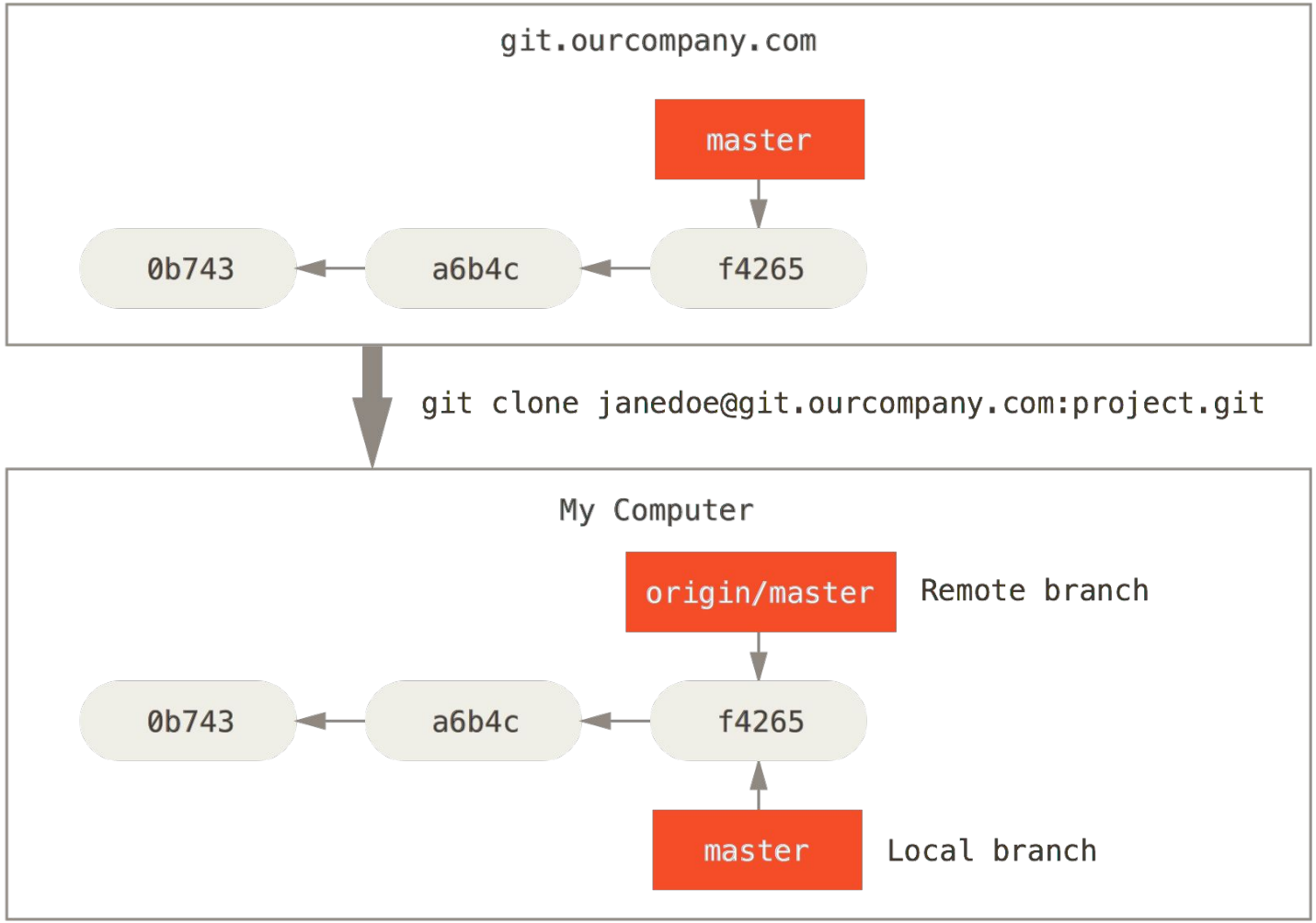
- **git branch:** lists all branches (-v, --merged, --no-merged)
- **git branch [branchname]:** creates new branch
- **git checkout [branchname]:** switches to a branch “branchname”
- **git merge [branchname]:** merges branch “branchname” into current branch
- **git add [path]:** stages fixed files
- **git commit:** finalizes merge - allows that only if all conflicted files have been fixed and staged
- **git branch -d [branchname]:** deletes branch “branchname”
- **git tag -a [tagname] -m [message]:** tags last commit

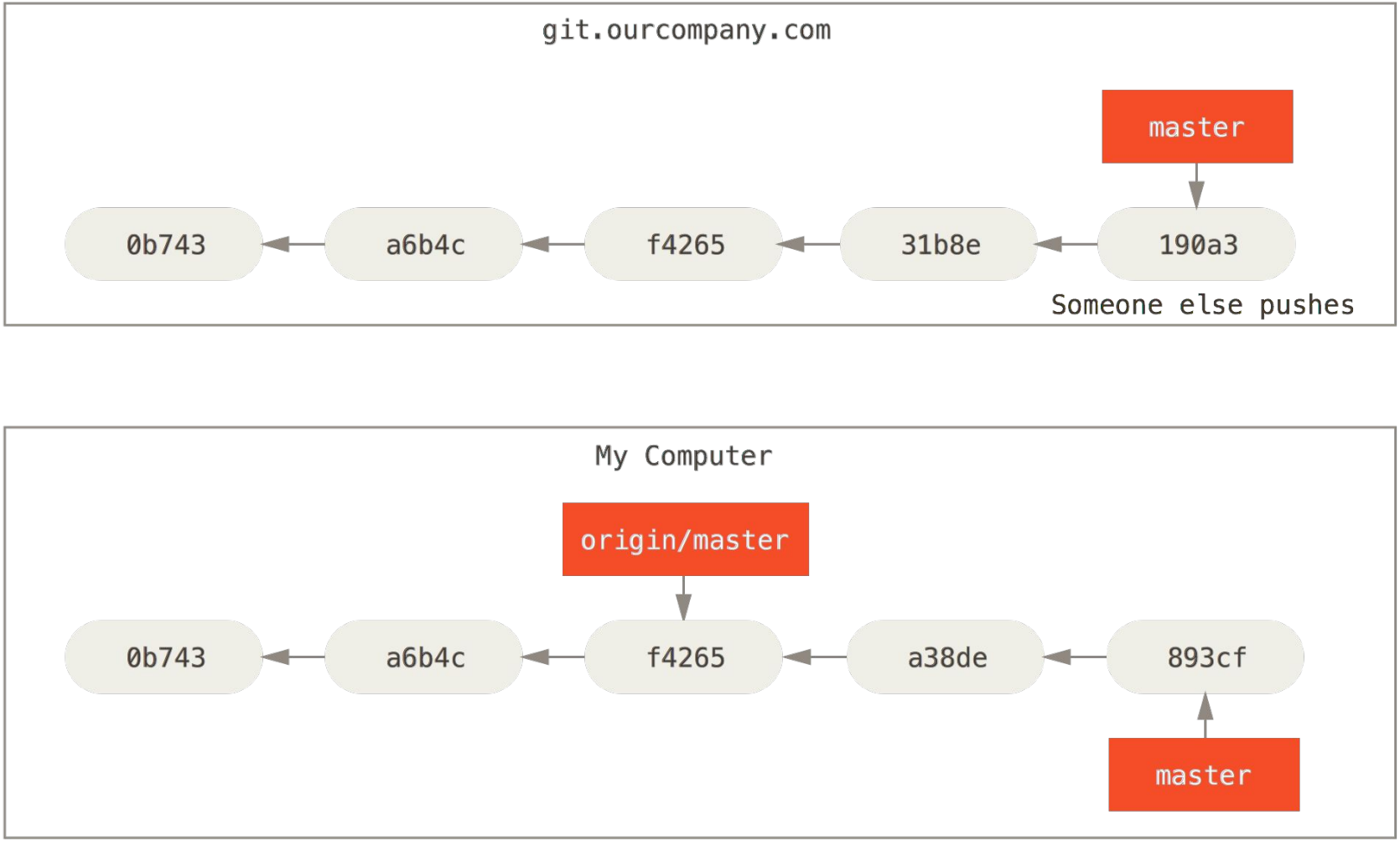
Branching - other useful commands

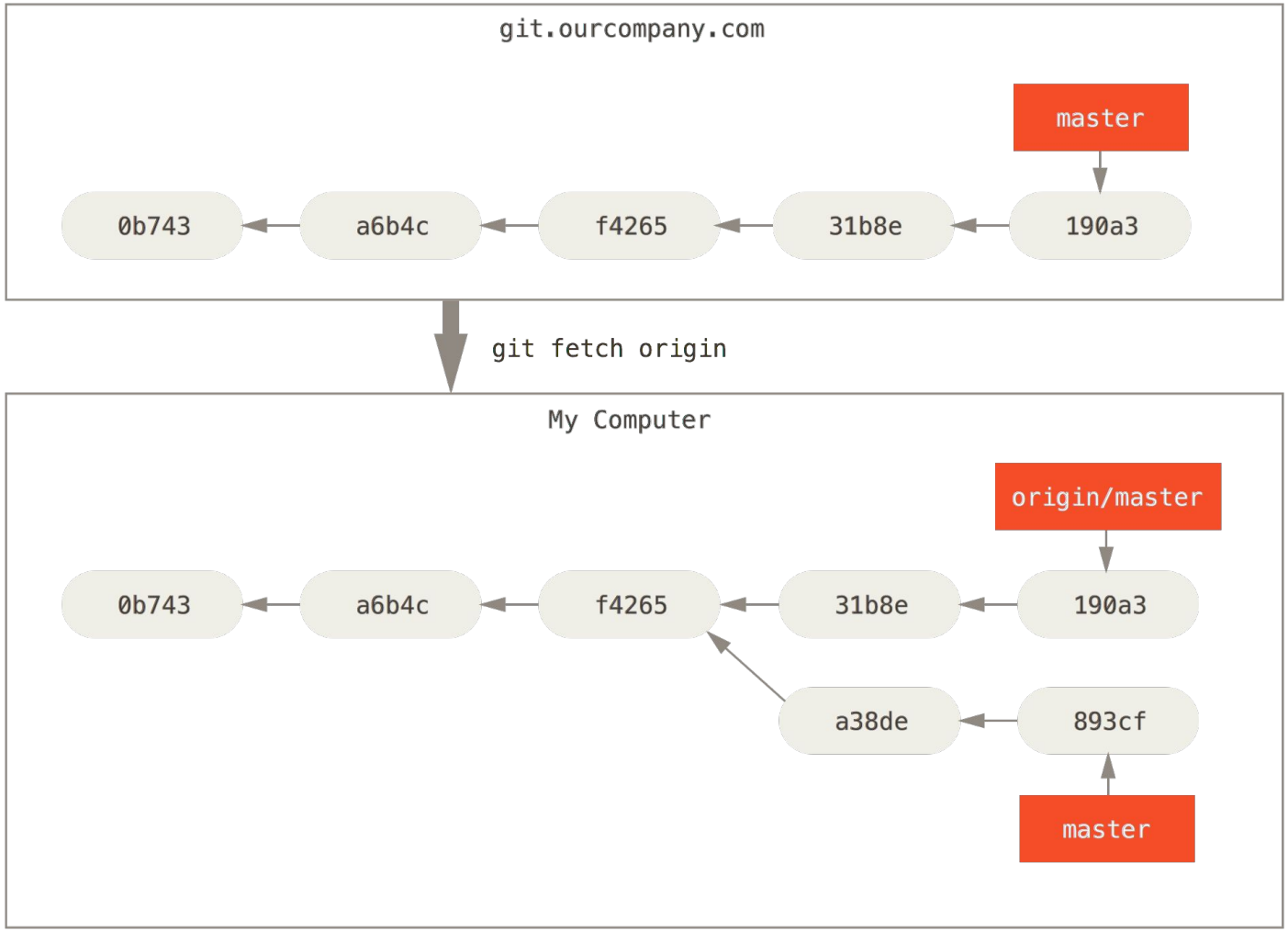
- **git checkout -b [branchname]:** shortcut, creates a new branch “branchname” and switches to its context
- **git tag:** lists all defined tags
- **git tag -a [tagname] [commit] -m [message]:** tags specified commit
- **git show [tagname]:** shows detailed information on given tag

- **Rebasing** - an alternative to merge

Remote repositories


git.ourcompany.com
master
0b743
a6b4c
f4265
git clone janedoe@git.ourcompany.com:project.git
My Computer
origin/master
Remote branch
0b743
a6b4c
f4265
master
Local branch


git.ourcompany.com
master
0b743
a6b4c
f4265
31b8e
190a3
Someone else pushes
My Computer
origin/master
0b743
a6b4c
f4265
a38de
893cf
master


git.ourcompany.com
master
0b743
a6b4c
f4265
31b8e
190a3
git fetch origin
My Computer
origin/master
0b743
a6b4c
f4265
31b8e
190a3
a38de
893cf
master

Remote repositories - summary

- **git clone [url]:** gets a copy of an existing Git repository
- **git branch -vv:** shows list of local branches and corresponding upstream branches (if the local branch is a tracking branch)
- **git fetch [remote-name]:** downloads data to the local repository (without merging it)
- **git push [remote-name] [branch-name]:** uploads local branch to the branch in the remote repository

Remote repositories - other useful commands

- **git clone [url] [dir]:** downloads the repository into specified directory
- **git pull [remotename]:** fetches data from the server and automatically tries to merge it into the current code
- **git remote:** lists the shortnames of all specified remote handles
- **git remote -v:** lists also URLs
- **git remote show [shortname]:** shows detailed information on given remote repository
- **git ls-remote --heads [remote-name]:** lists all remote branches

Other commands

Undoing things

- **git commit --amend:** adds more files or modifications to the last commit and/or changes commit message of last commit
- **git reset HEAD [file]:** unstages file
- **git checkout -- [file]:** reverts changes in file

Other useful commands:

- **git revert [commit]:** undoes the changes introduced by given commit and appends a new commit with the resulting content
- **git reset --hard HEAD~1:** removes last commit
- **git reset --hard [commit]:** resets the branch to the specified commit (all following commits are removed)

(warning: “git reset --hard” command will delete all changes in the working directory)

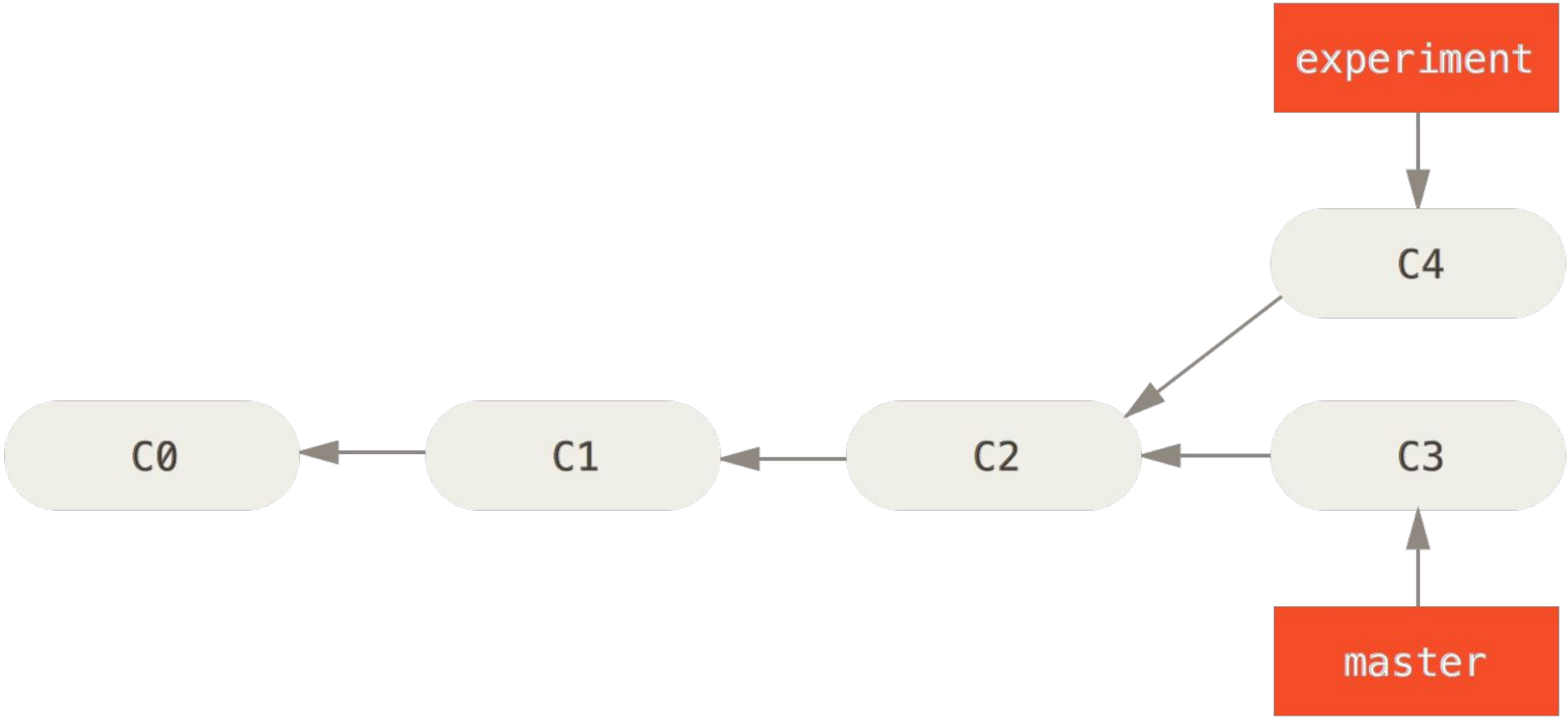
Setup, help

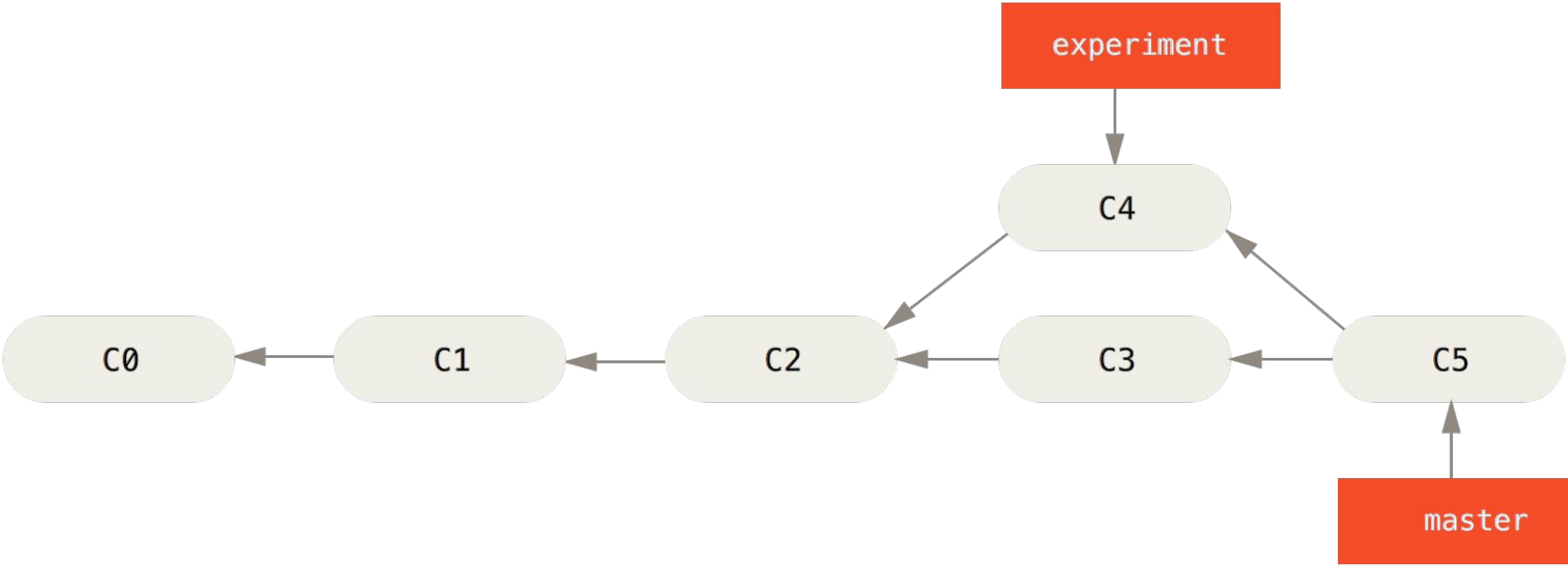
- **git config**: various settings, e.g.
  --global user.name "John Doe"
  --global user.email johndoe@example.com
  --global alias.co checkout
- **git help [cmd]:** help
- **man git [cmd]:** help

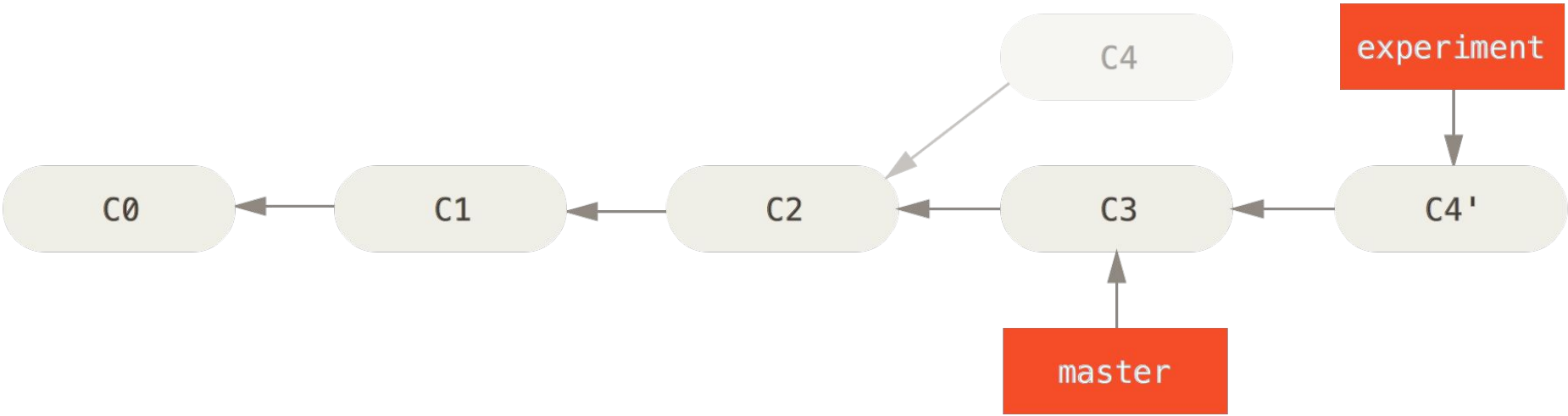
References

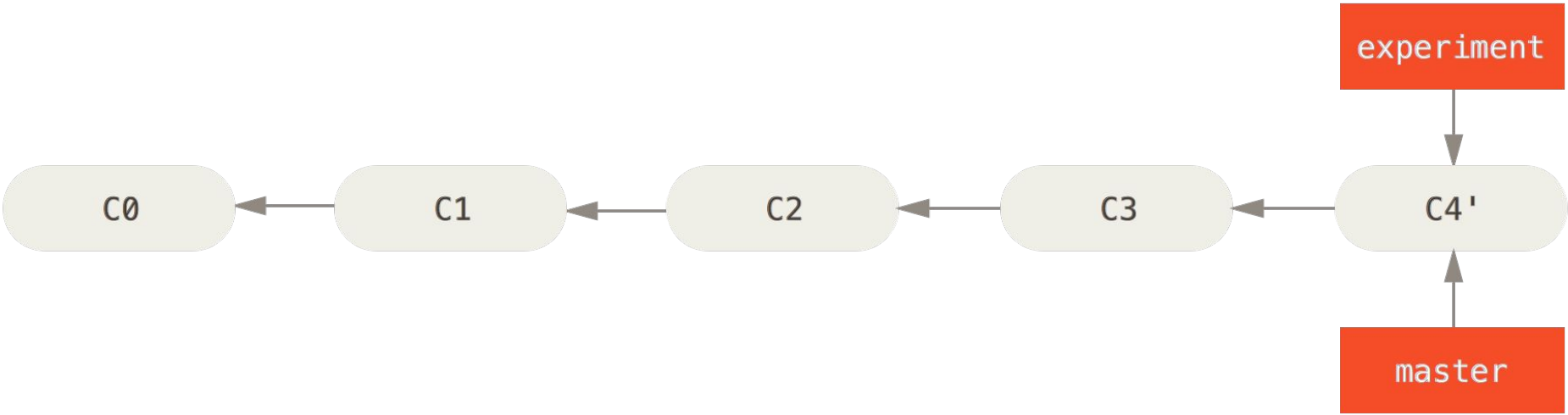
- https://git-scm.com/book/en/v2
- https://git-scm.com/docs
- https://www.atlassian.com/git/

Rebasing


experiment
C4
C0
C1
C2
C3
master


experiment
C4
C0
C1
C2
C3
C5
master


C4
experiment
C0
C1
C2
C3
C4'
master


experiment
C0
C1
C2
C3
C4'
master

Rebasing - summary

- **git rebase [branch2]:** replays all (separate) commits of the current branch (branch1) to the branch2 and makes branch1 to point to the resulting commit
- **git checkout [branch2], git merge [branch1]:** finalizes merge

Other useful commands:

- **git rebase --onto [branch1] [branch2] [branch3]:** checks out the branch3, figures out the commits from the common ancestor of branch 2 and branch3, and then replays them onto branch1
- **git rebase -i:** interactive rebase
- **git cherry-pick [commit1] [commit2], ..:** applies the changes introduced by given commits one by one, recording a new commit for each