

# git r' done!

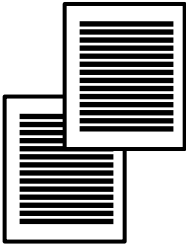
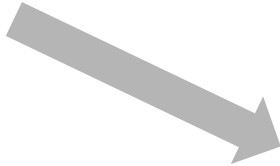


*A brief introduction to  
distributed version control with git*

Oscar Nierstrasz — University of Bern — [scg.unibe.ch](http://scg.unibe.ch)

**Why git?**

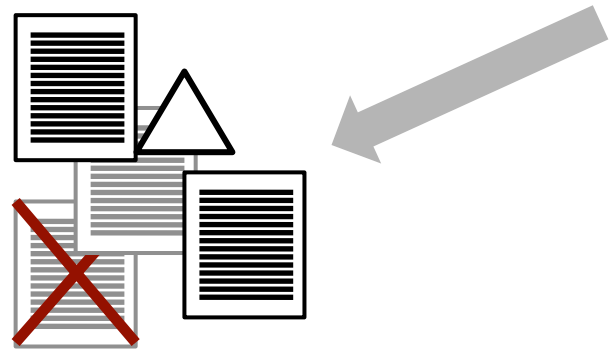
**Bob**



**Bob**



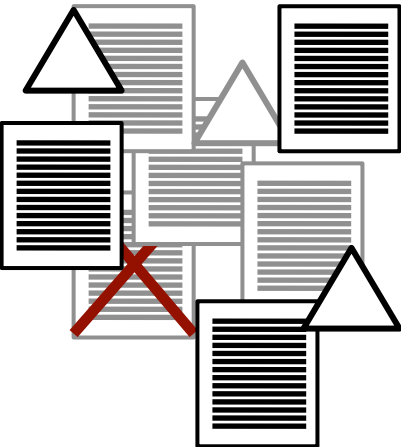
**Carol**



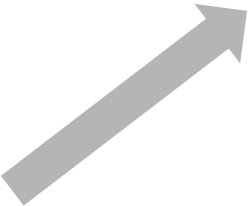
Bob



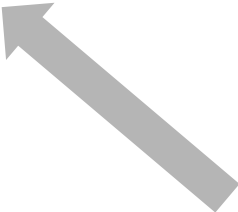
Carol

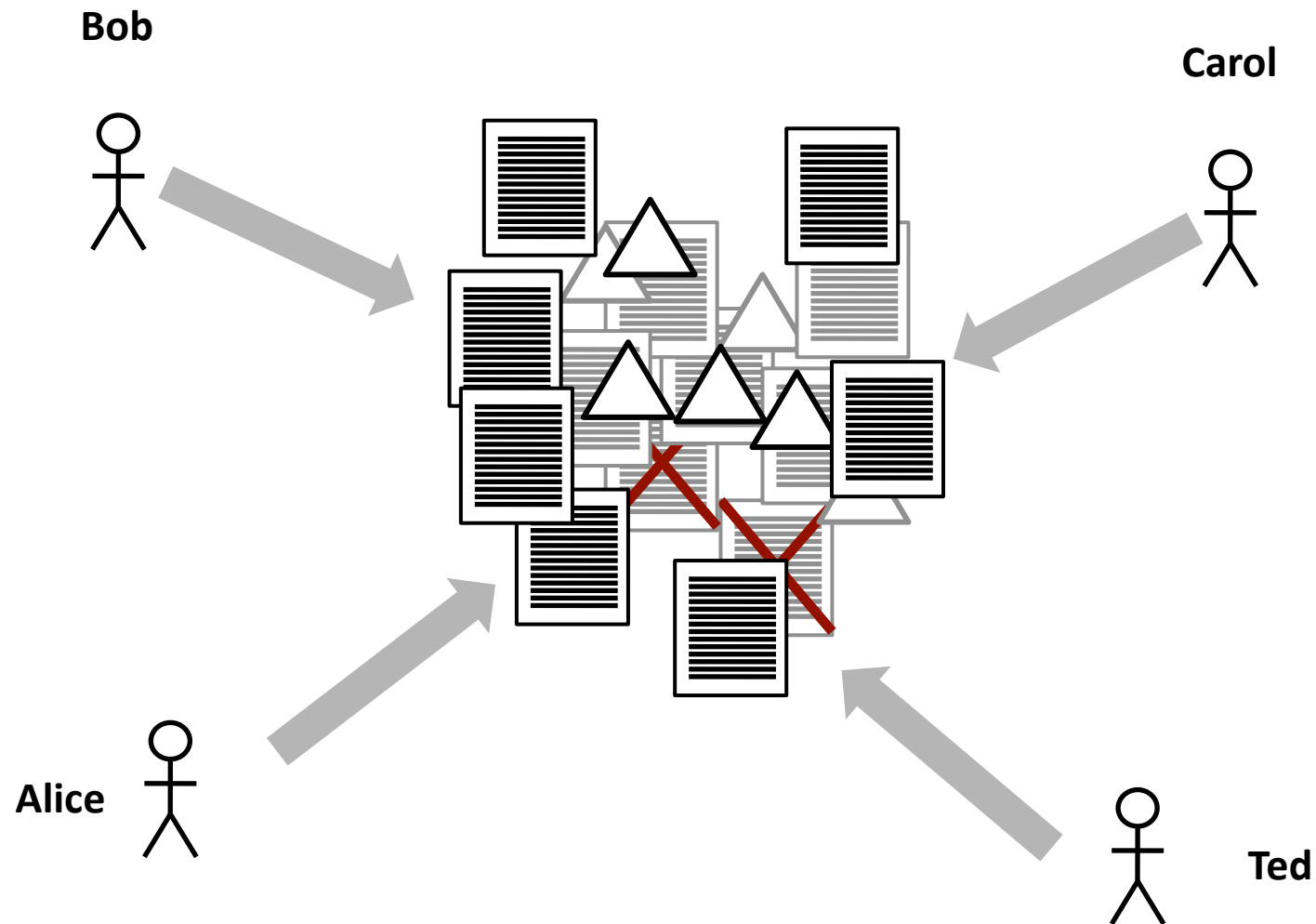


Alice



Ted

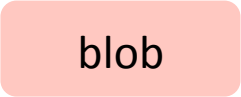




**A recipe for disaster!**

# The git object model

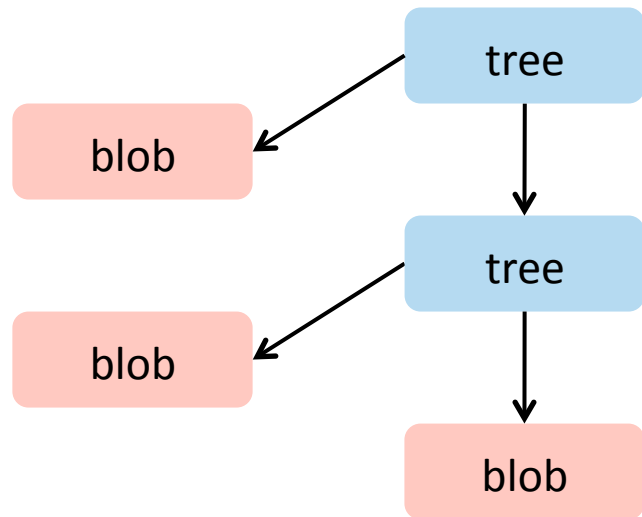
A “blob” is *content* under  
version control (a file)



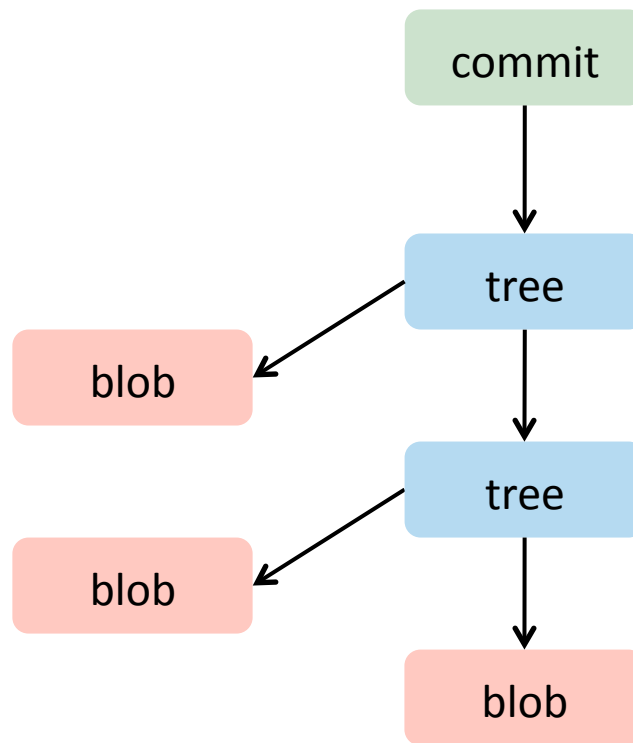
blob



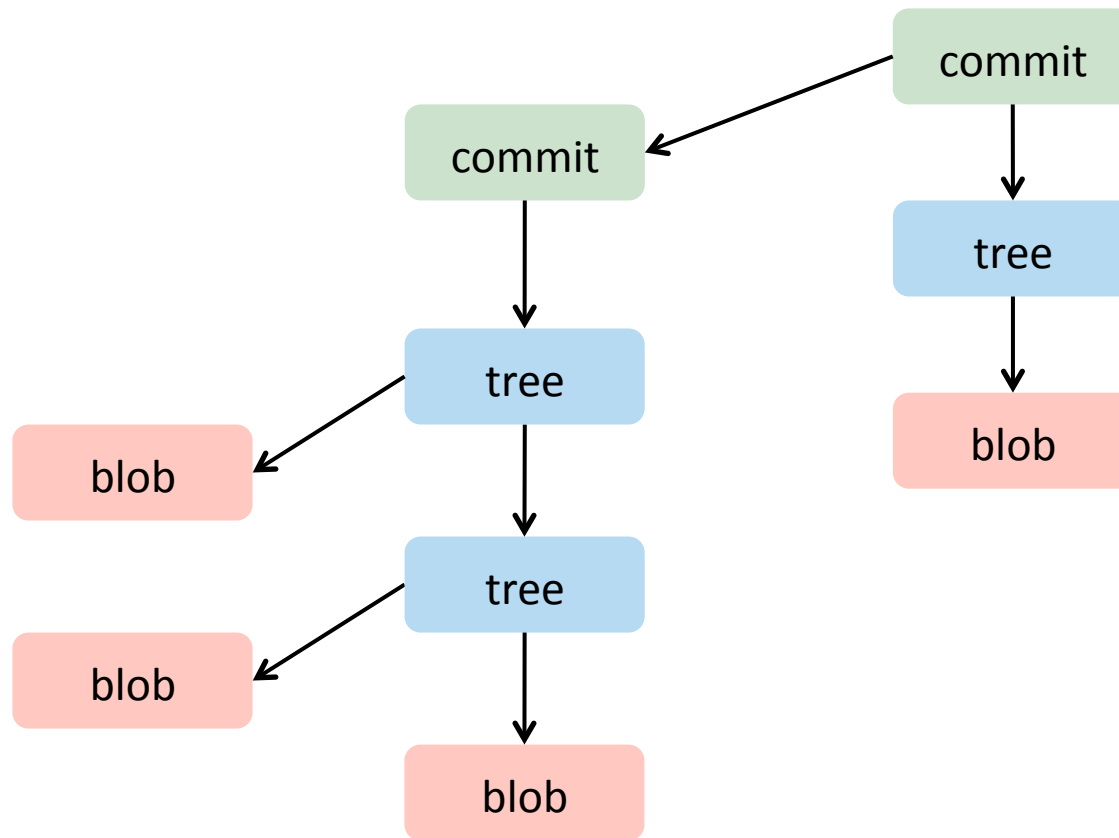
You can have *trees* of blobs  
(directories of files)



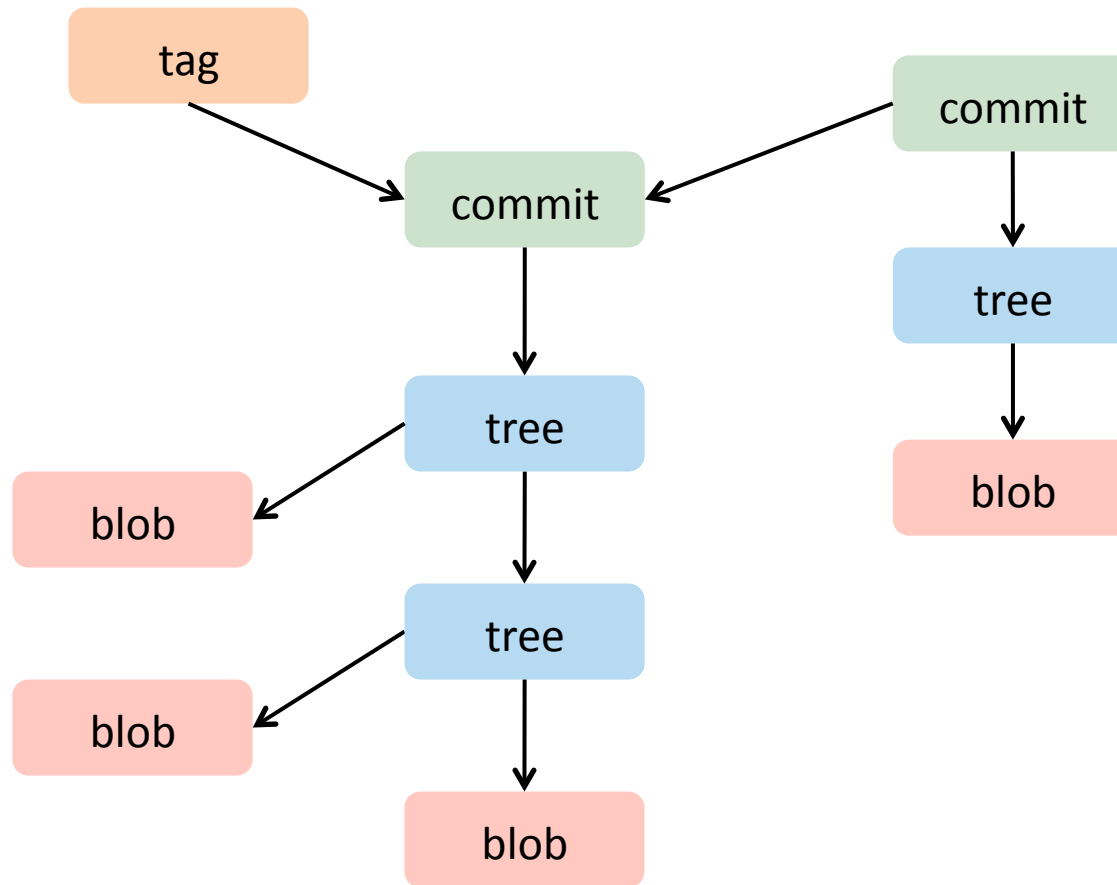
A “commit” is a tree of blobs  
(a set of changes)



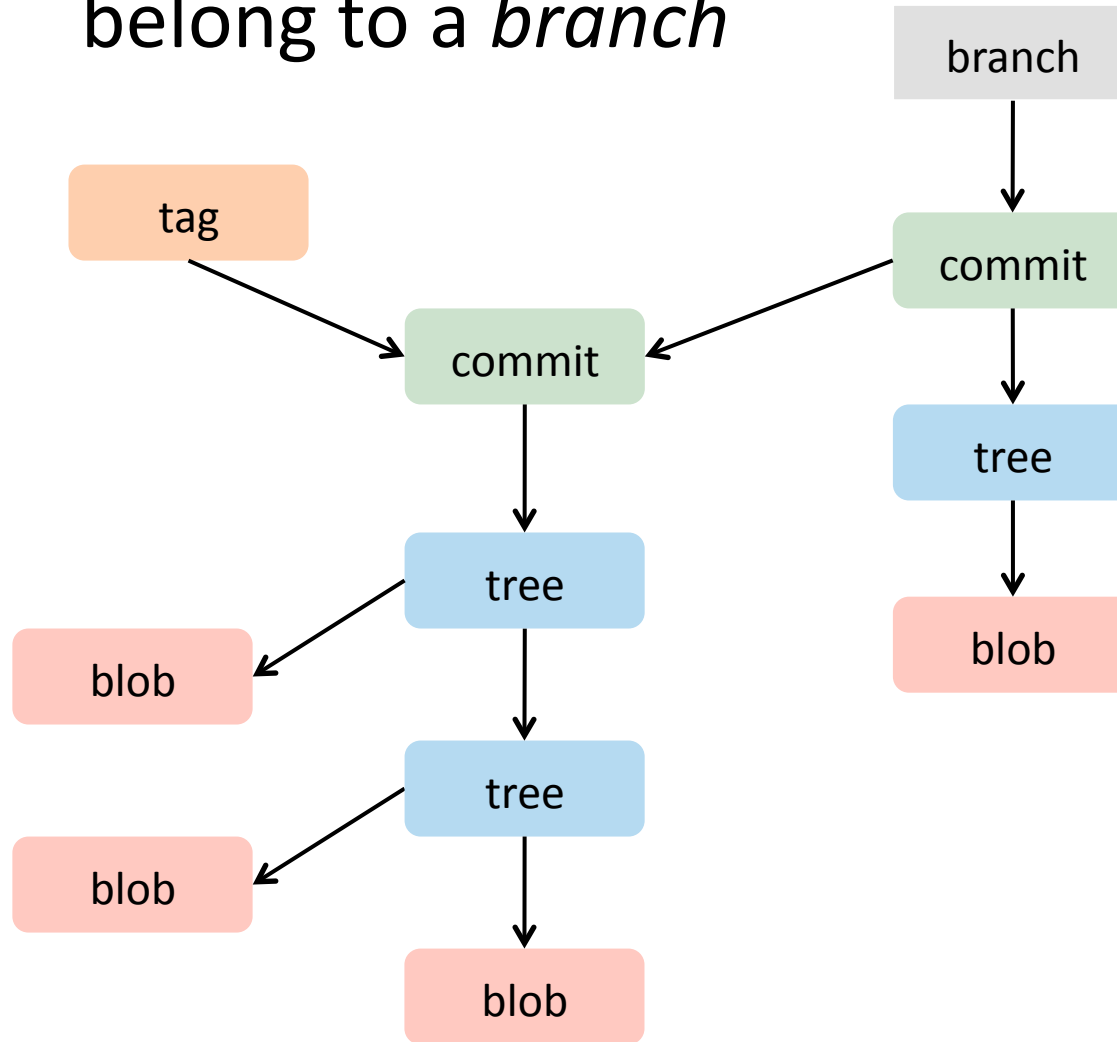
Most commits modify  
(or merge) earlier commits



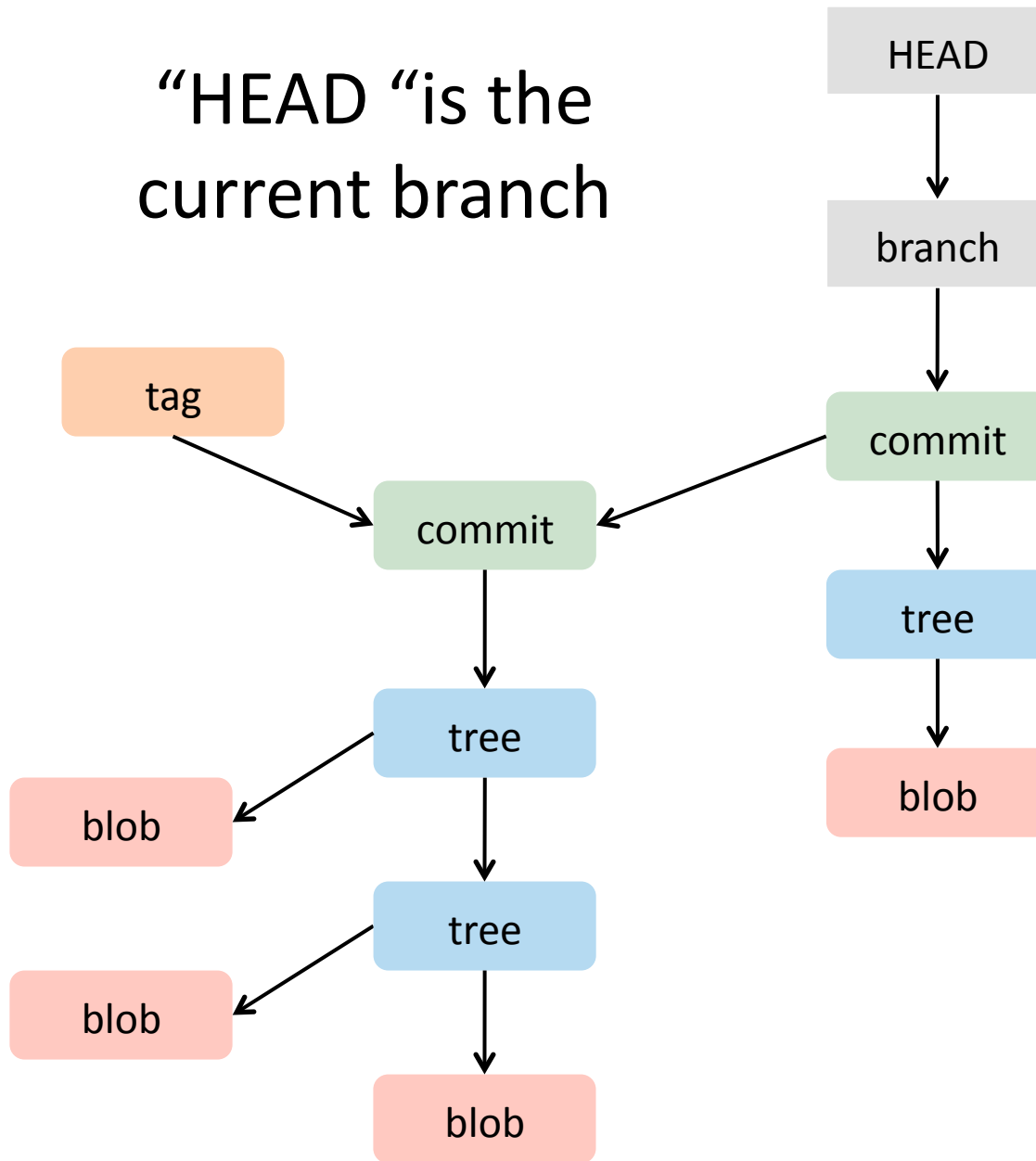
You can “tag” an interesting commit

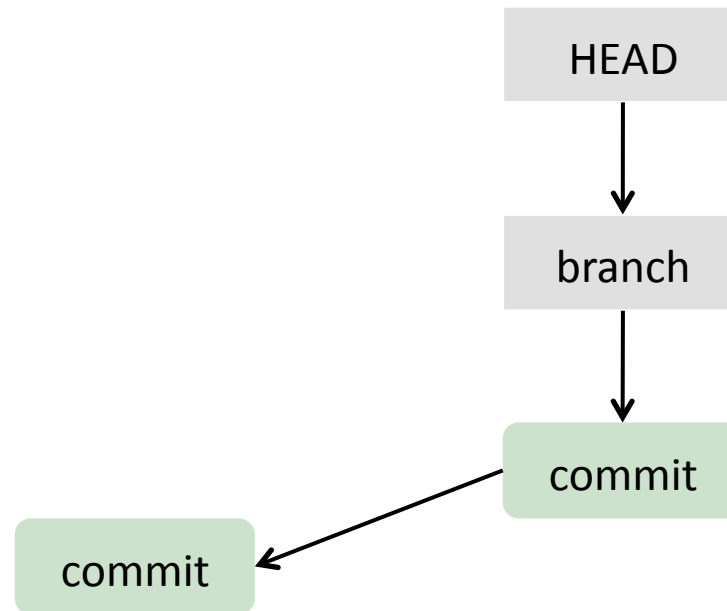


A graph of commits may  
belong to a *branch*



“HEAD “is the  
current branch





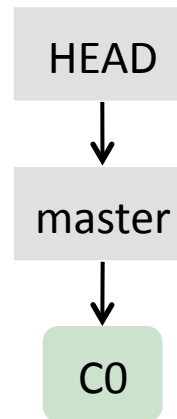
Let's focus on *commits*  
and *branches*

# Basic git



## Create a git repo

```
mkdir repo  
cd repo  
git init
```



Tell git to “stage”  
changes

HEAD

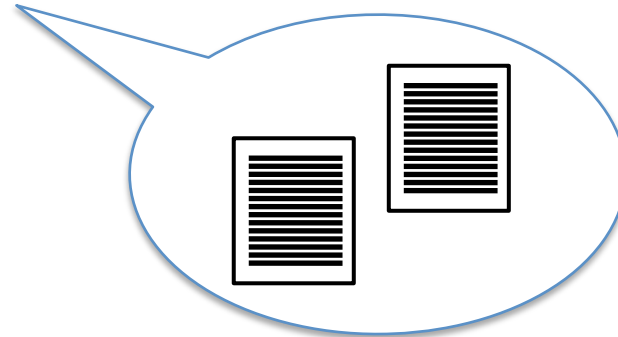


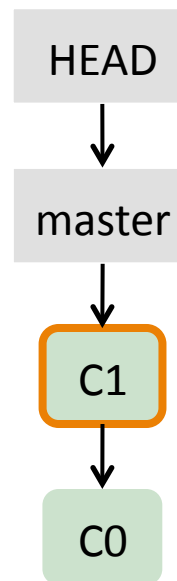
master



C0

```
git add ...
```





Commit your  
changes

```
git commit ...
```

**Collaborating**



John

Local repo

Jane



Local repo

Public repo

master



C1



C0



John



Jane

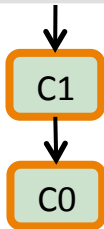
Local repo

Public repo

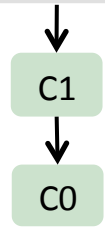
Local repo

**git clone ...**

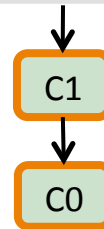
master



master



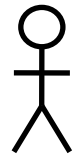
master



**git clone ...**

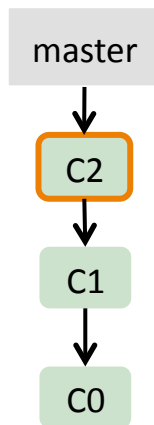


John



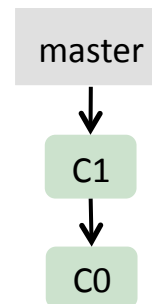
Jane

Local repo

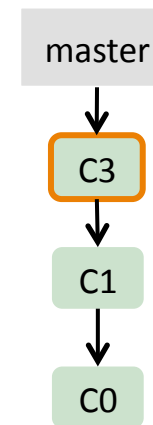


```
git add ...  
git commit ...
```

Public repo



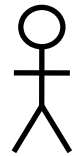
Local repo



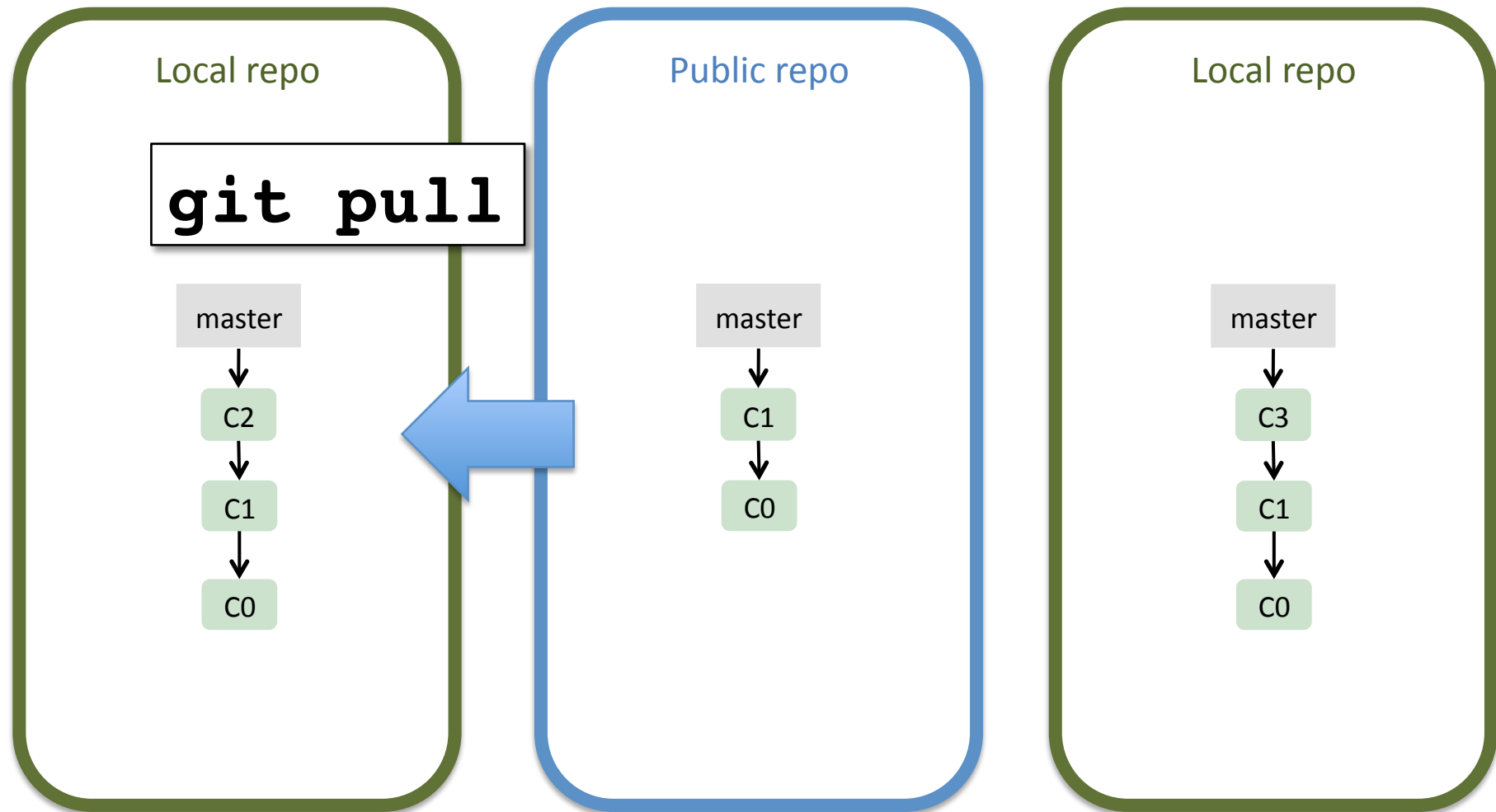
```
git add ...  
git commit ...
```



John



Jane

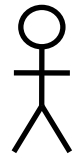


(nothing new to pull)





John



Jane

Local repo

**git push**

master



C2



C1



C0



Public repo

master



C2



C1



C0

Local repo

master



C3



C1

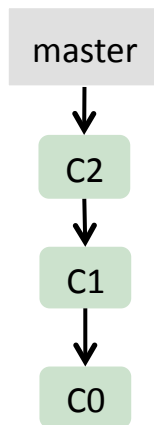


C0

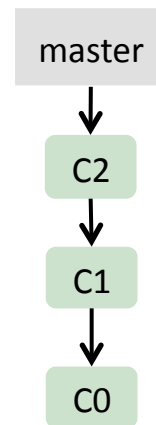


John

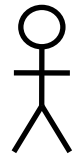
Local repo



Public repo

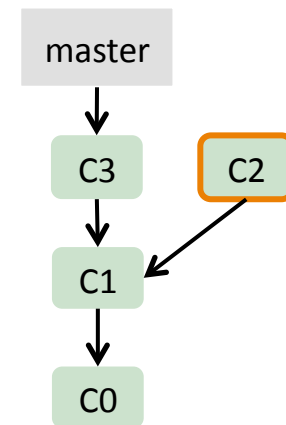


Jane



Local repo

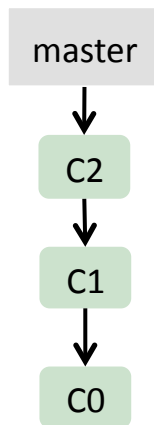
**git fetch**



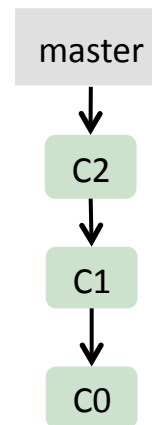


John

Local repo



Public repo

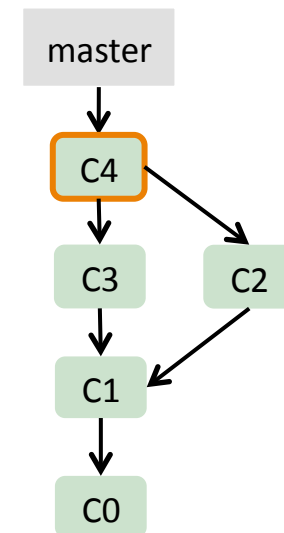


Jane



Local repo

**git merge**

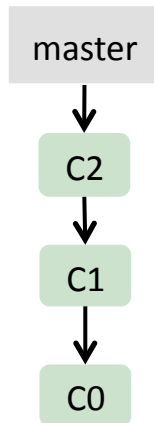


**NB: git pull = fetch + merge**

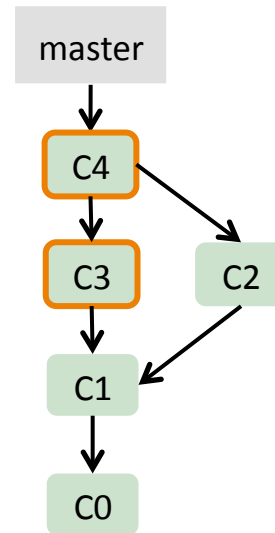


John

Local repo



Public repo

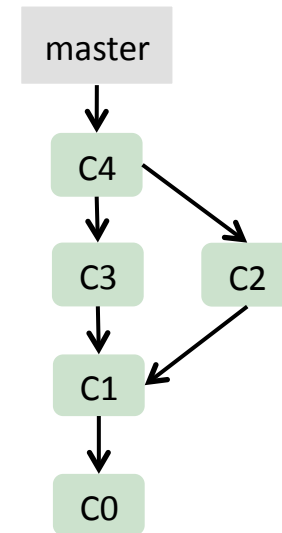


Jane



Local repo

**git push**





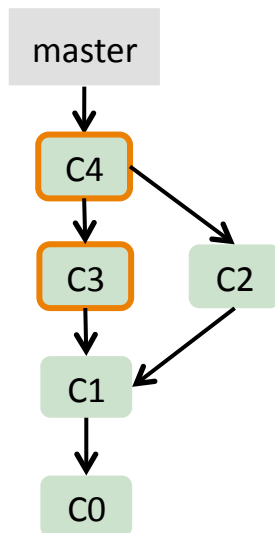
John



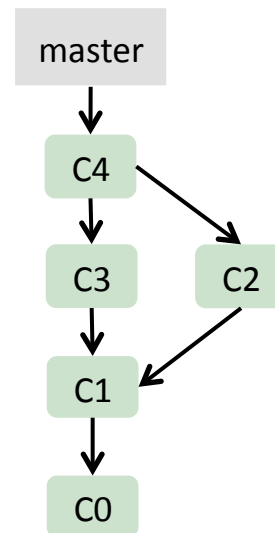
Jane

Local repo

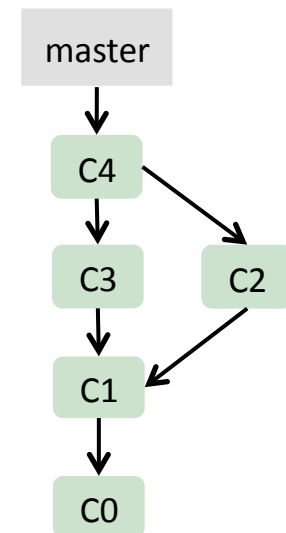
**git pull**



Public repo

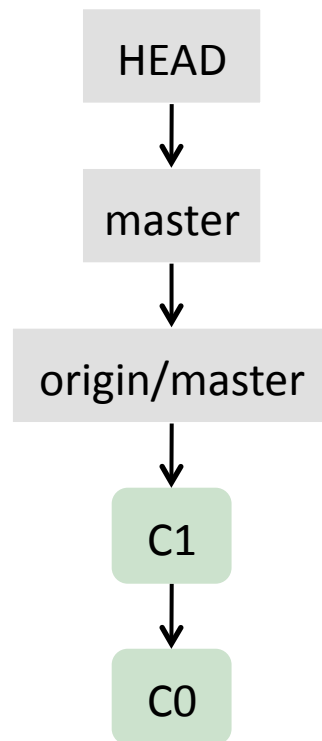


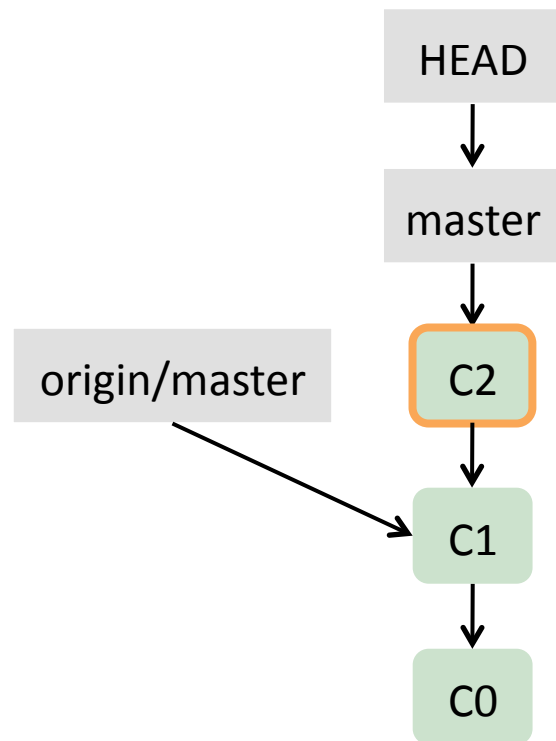
Local repo



# **Branching and merging**

“origin” refers to the  
remote repo

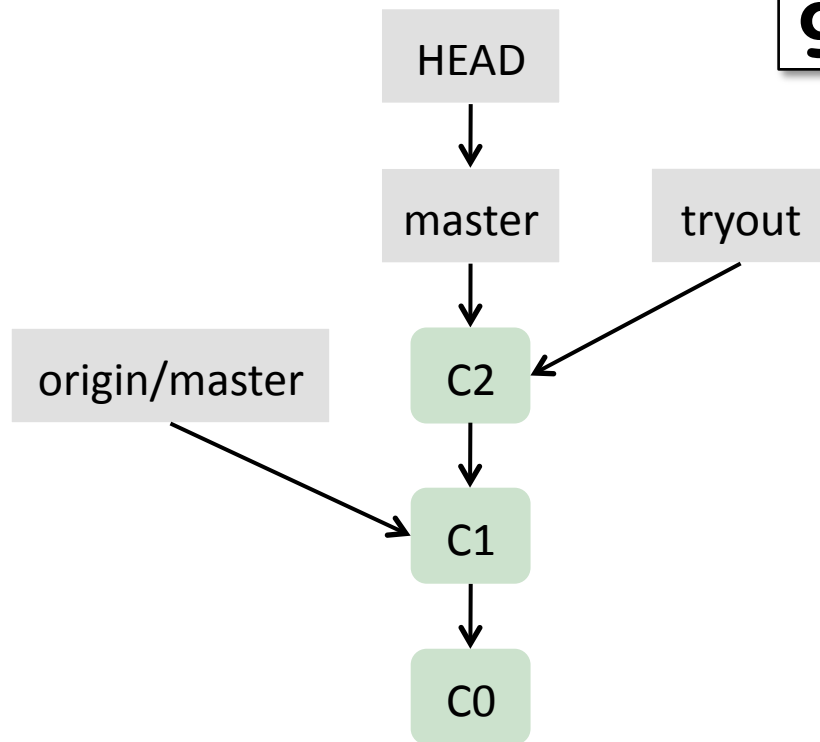




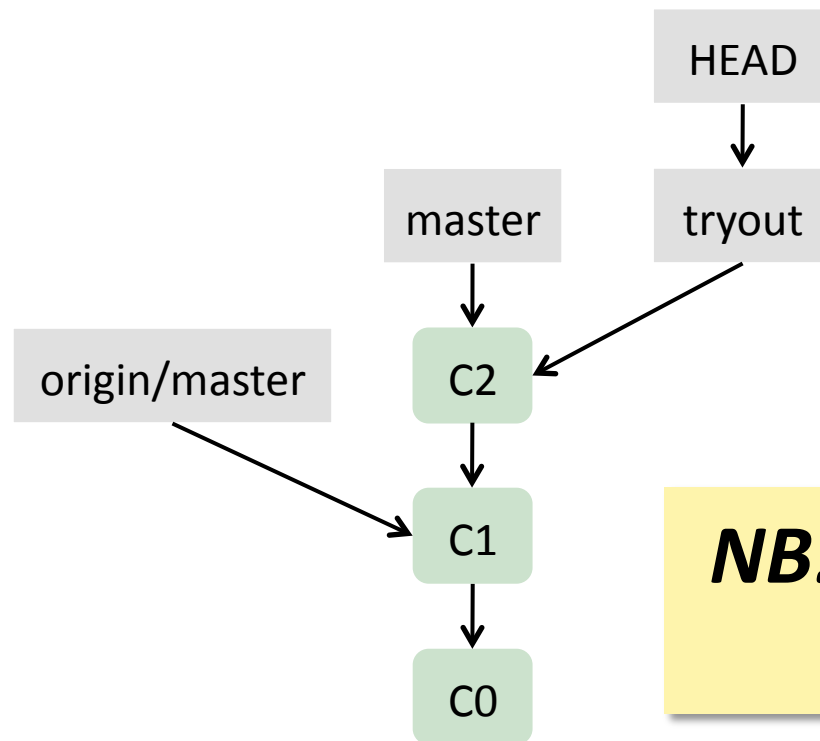
...  
**git commit** ...



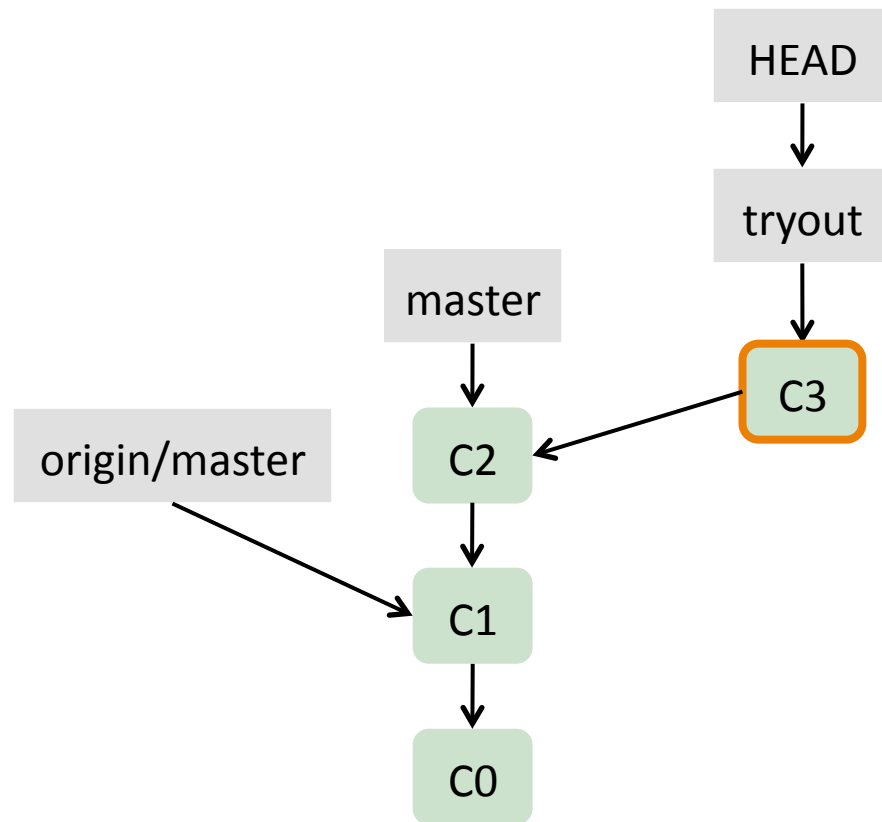
# git branch tryout



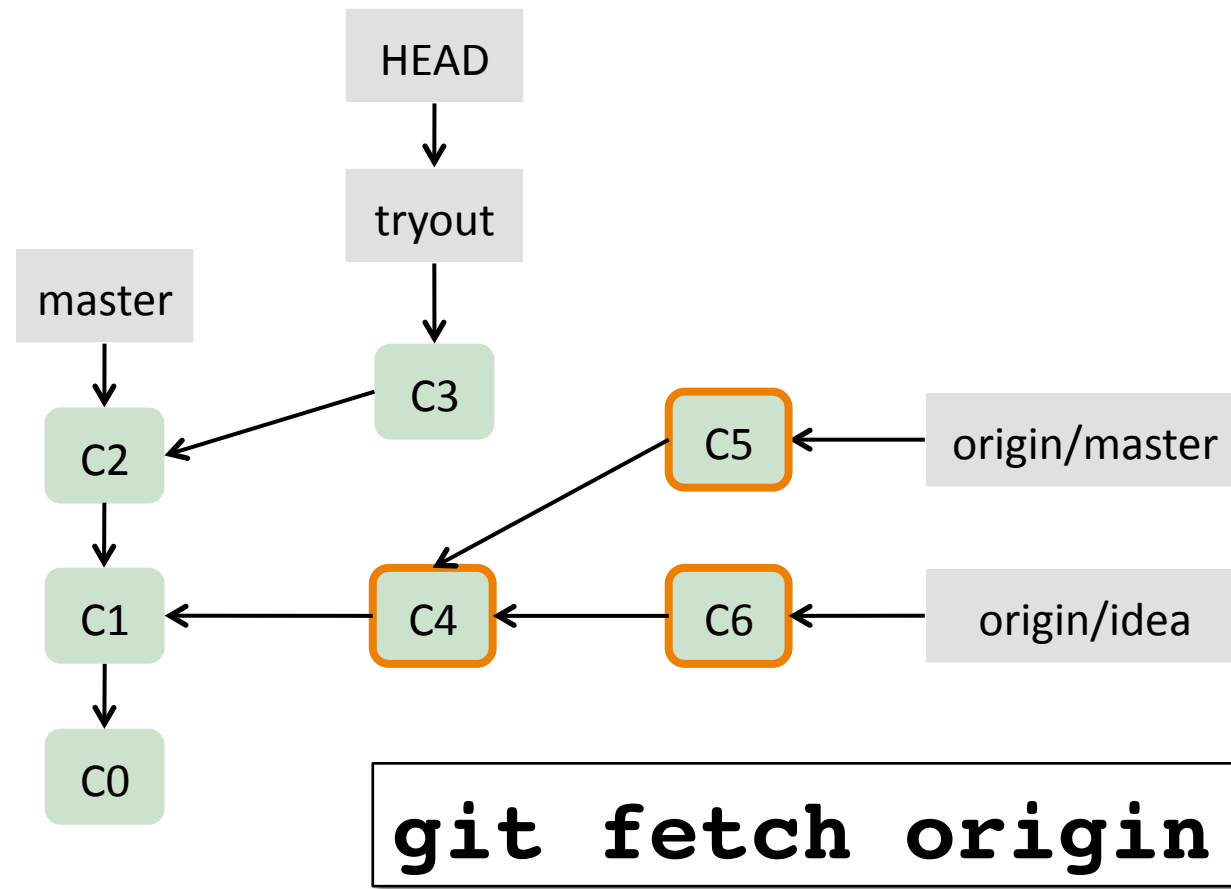
# git checkout tryout



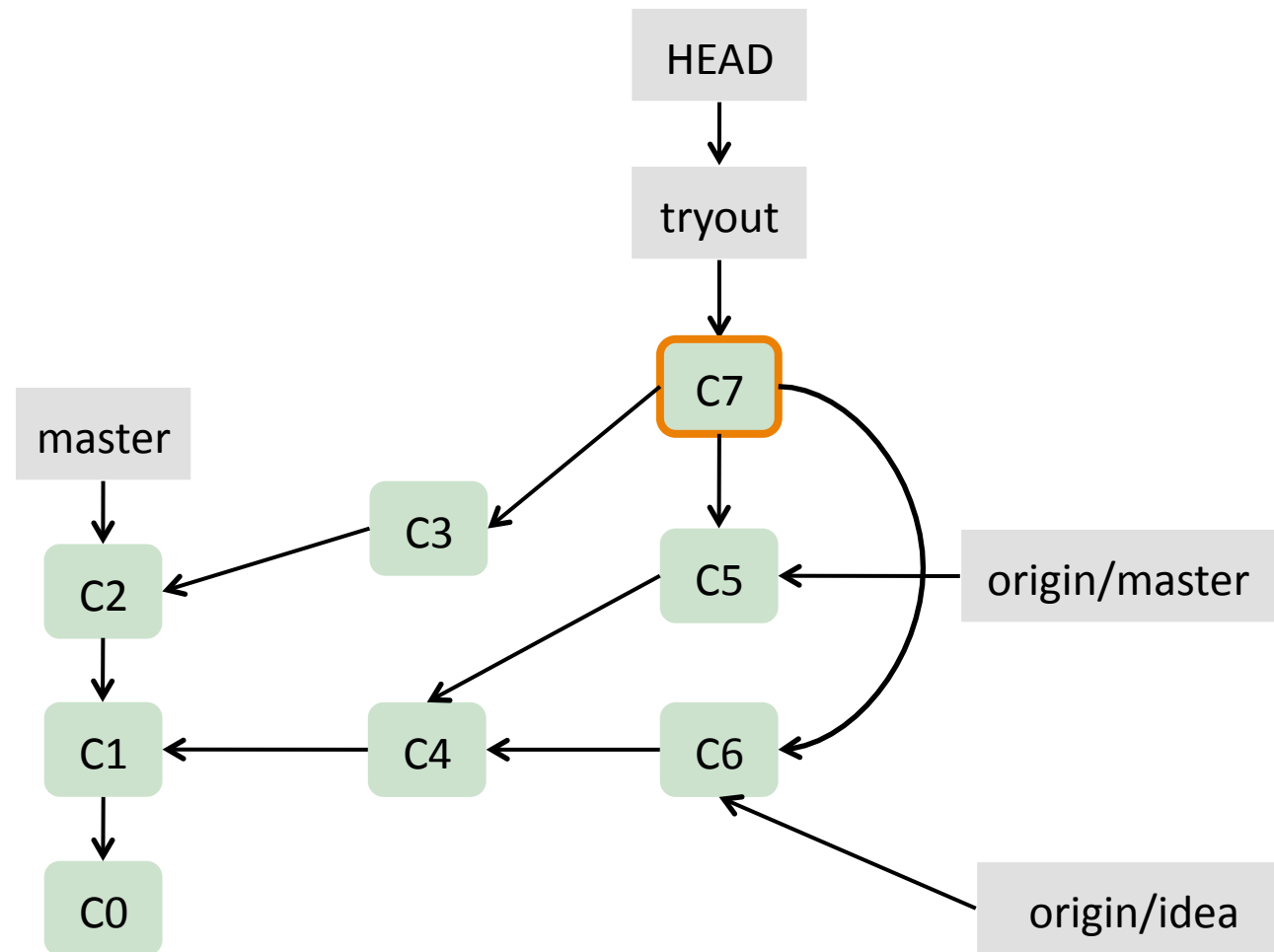
***NB:* `git checkout -b ...`  
= branch + checkout**



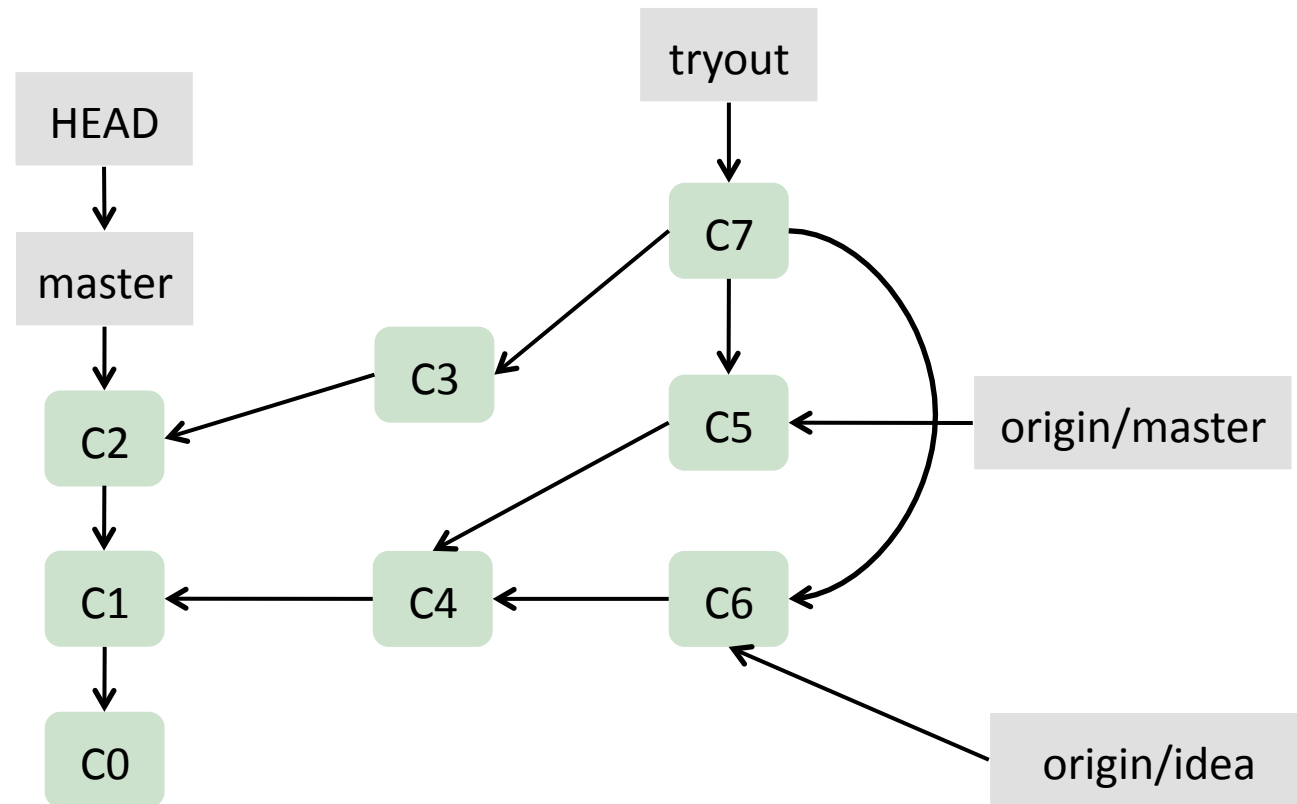
**git commit ...**



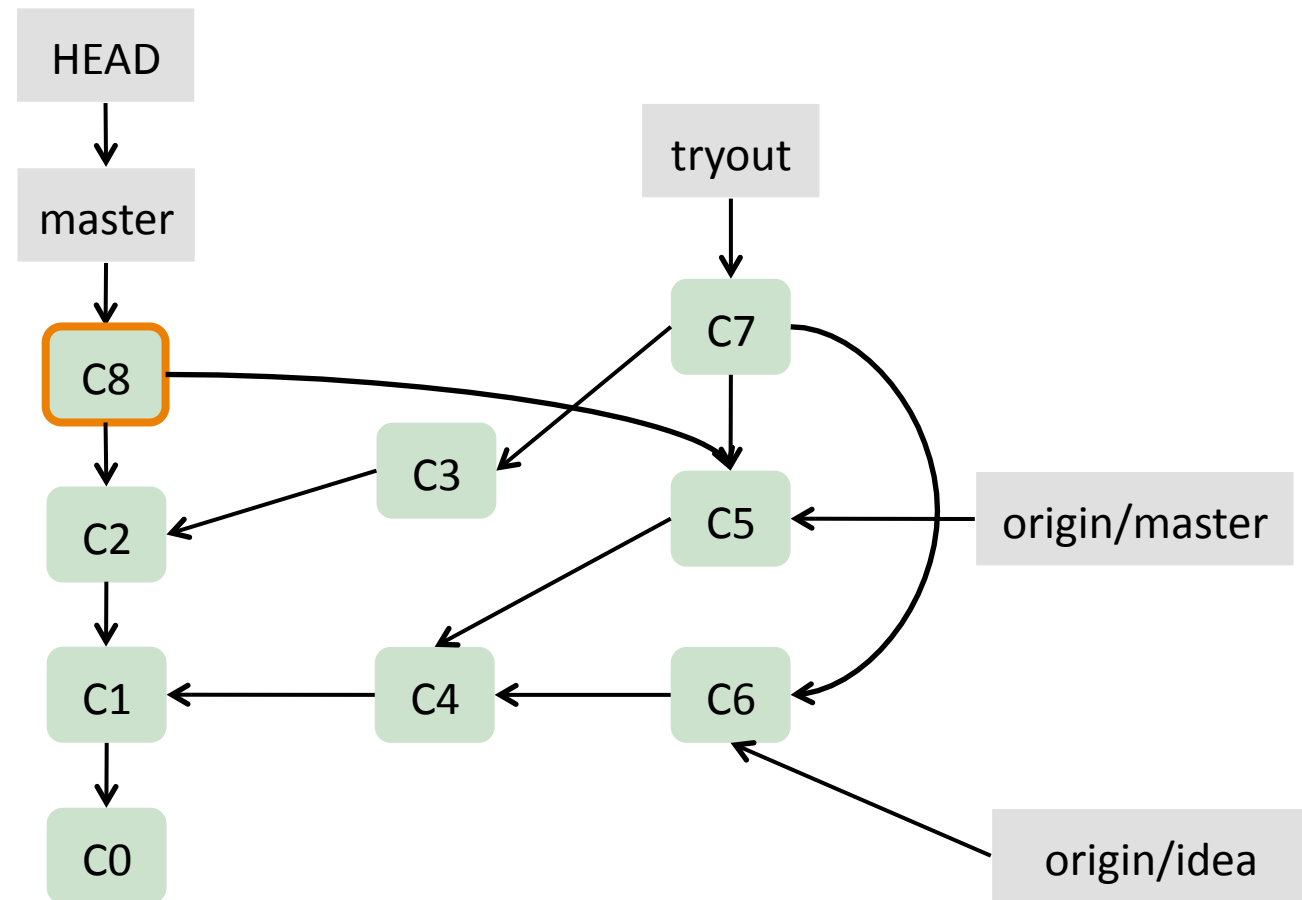
**git merge origin/master origin/idea**



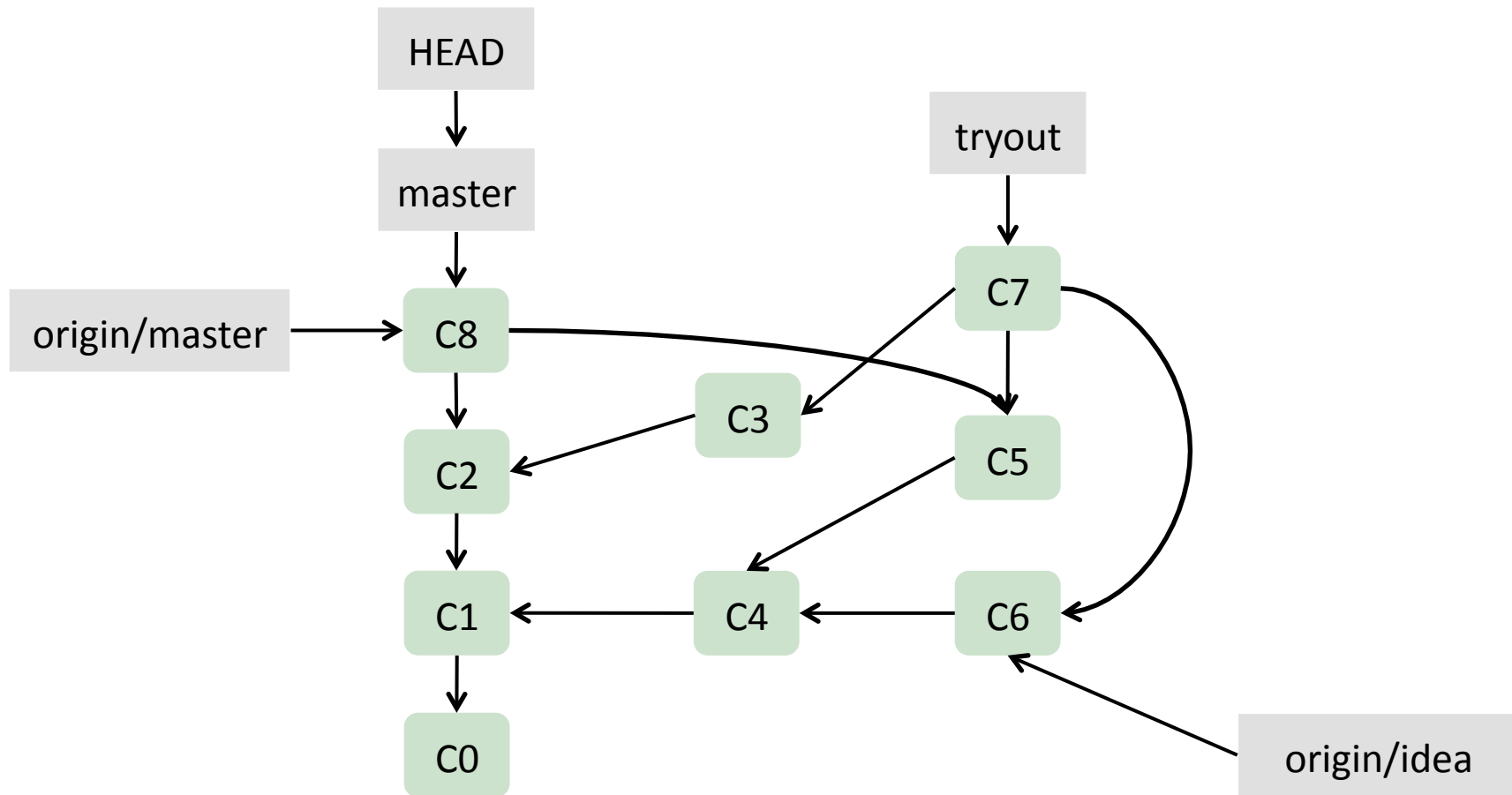
# git checkout master



# git merge



# git push





**More to git**

# More to git ...

- Merging and mergetool
- Squashing commits when merging
- Resolving conflicts
- User authentication with ssh
- gitx and other graphical tools
- git configure — remembering your name
- git remote — multiple remote repos
- github — an open source public repo
- ...

# Resources



<http://git-scm.com/>



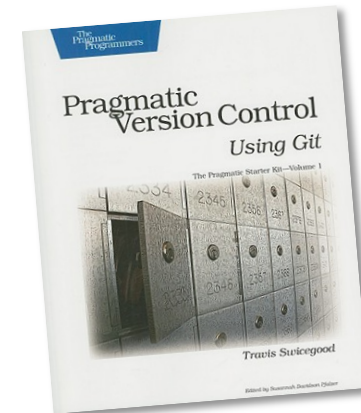
<http://book.git-scm.com/index.html>



<https://github.com/>

Getting Git  
Scott Chacon

<http://www.slideshare.net/chacon/getting-git>



<http://oreilly.com/>