INTRODUCTION TO VERSION CONTROL AND GIT

PREPARATION

- Download and install Git
- PC's in EDV Laboratory have a portable Git installation on the Z drive.
- Download and unzip portable Git

WHAT IS VERSION CONTROL?

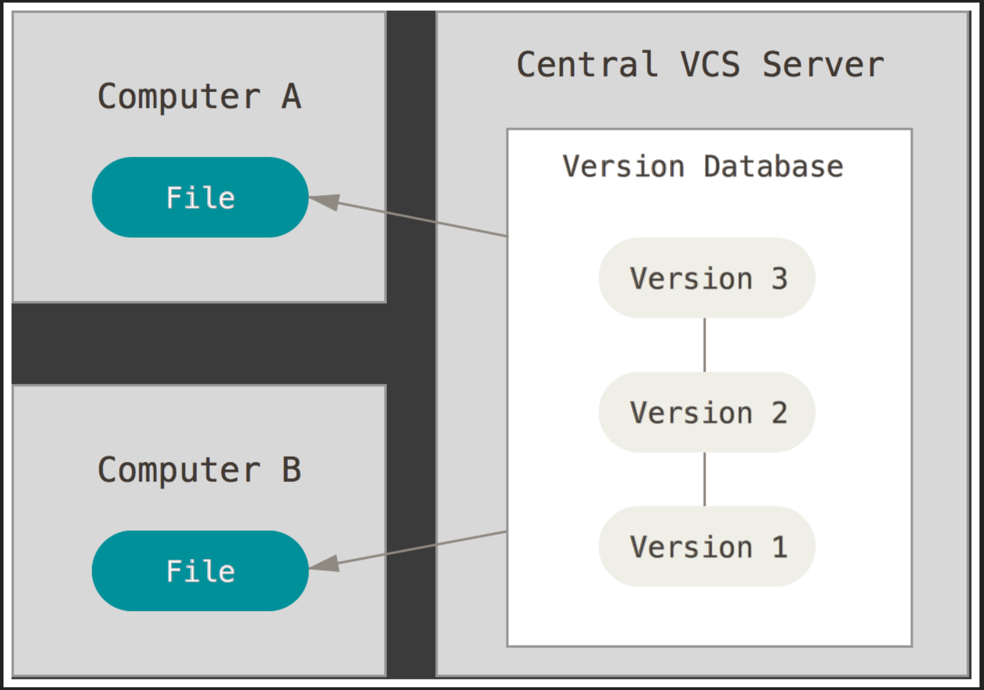
> *Revision control, also known as version control, source control or software configuration management (SCM), is the management of changes to documents, programs, and other information stored as computer files.*
>
> *– Wikipedia*

WHY DO WE NEED VERSION CONTROL?

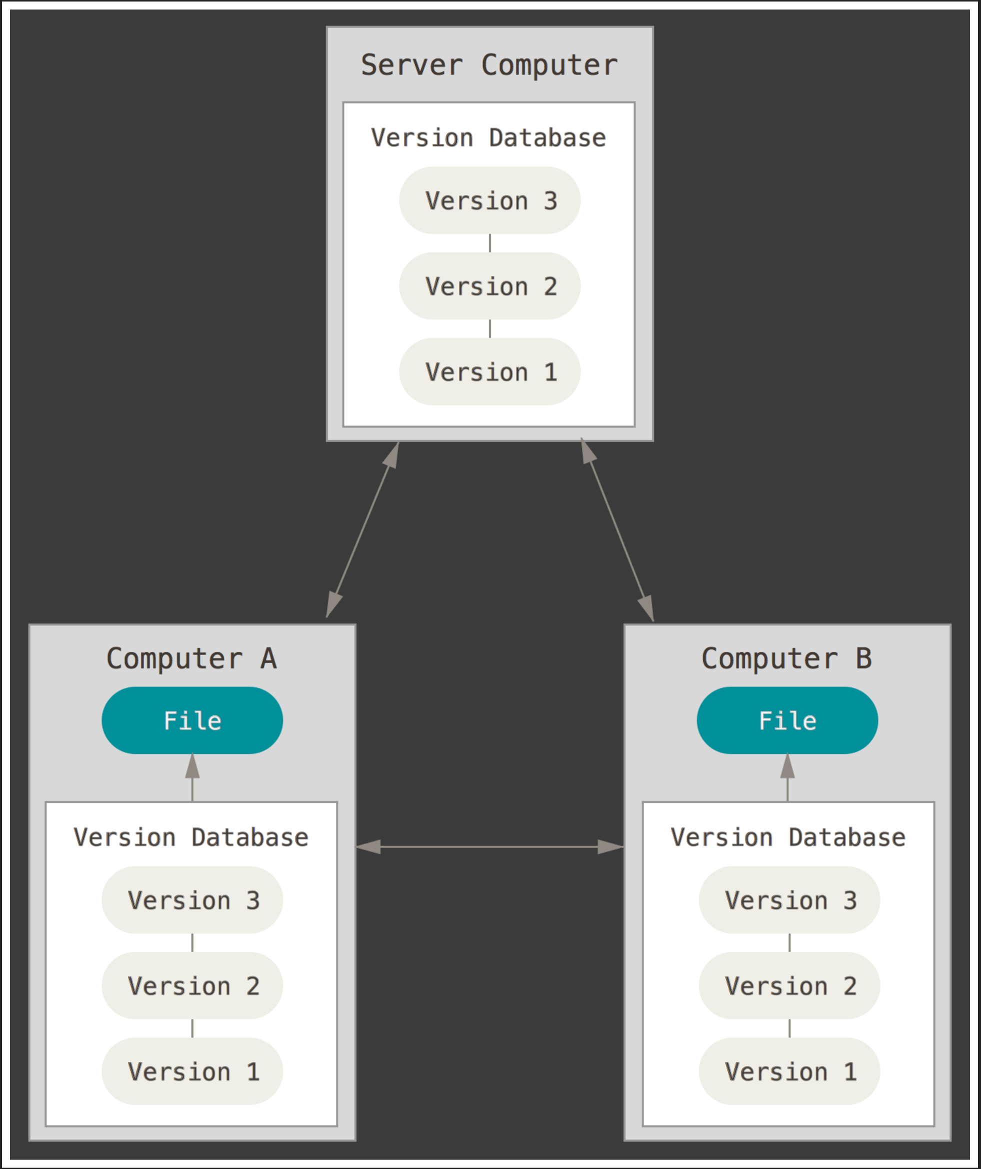
- Reproducibility
  - Track every step of your work
- Peace of mind (backup)
- Freedom (try new stuff without loosing the old)
- Collaboration

DIFFERENT VC APPROACHES

CENTRALIZED (CVS, SVN, ...)



DISTRIBUTED (GIT, MERCURIAL, ...)



WHY GIT?

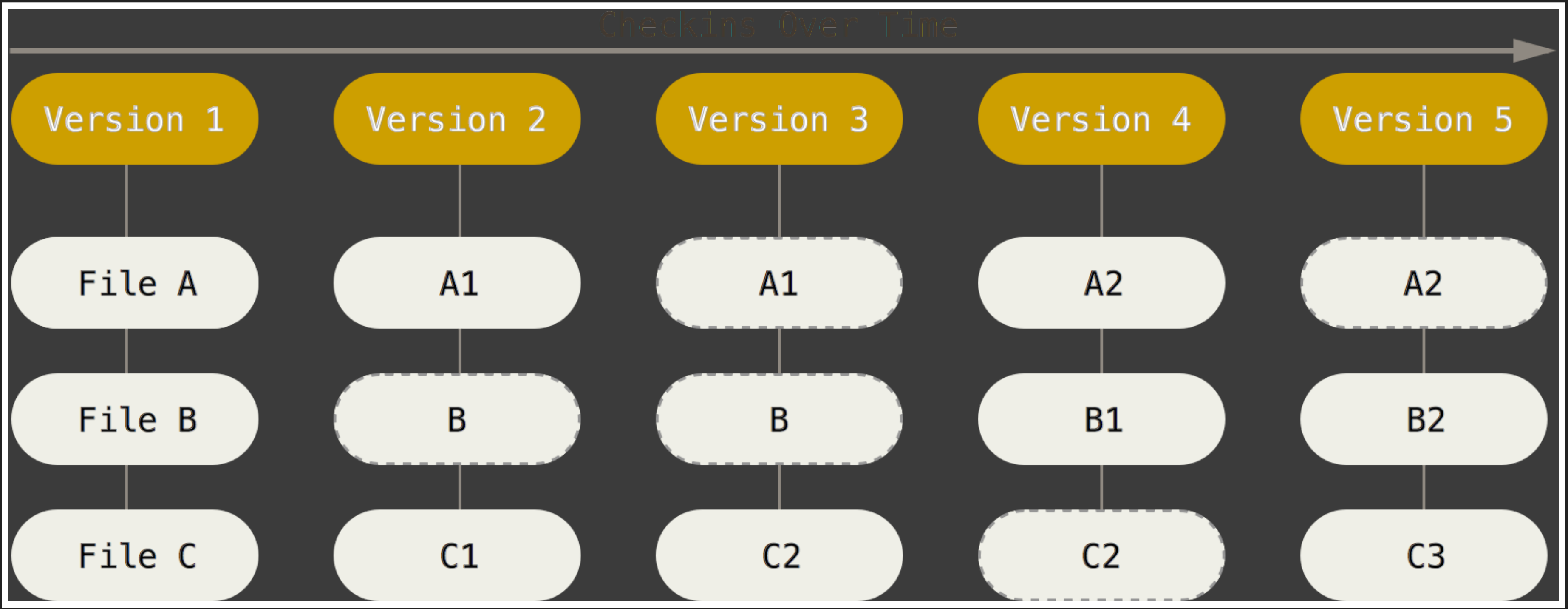
- Fast
- Fully Distributed
- De facto standard for a lot of open source projects (Github)

SHORT HISTORY OF GIT

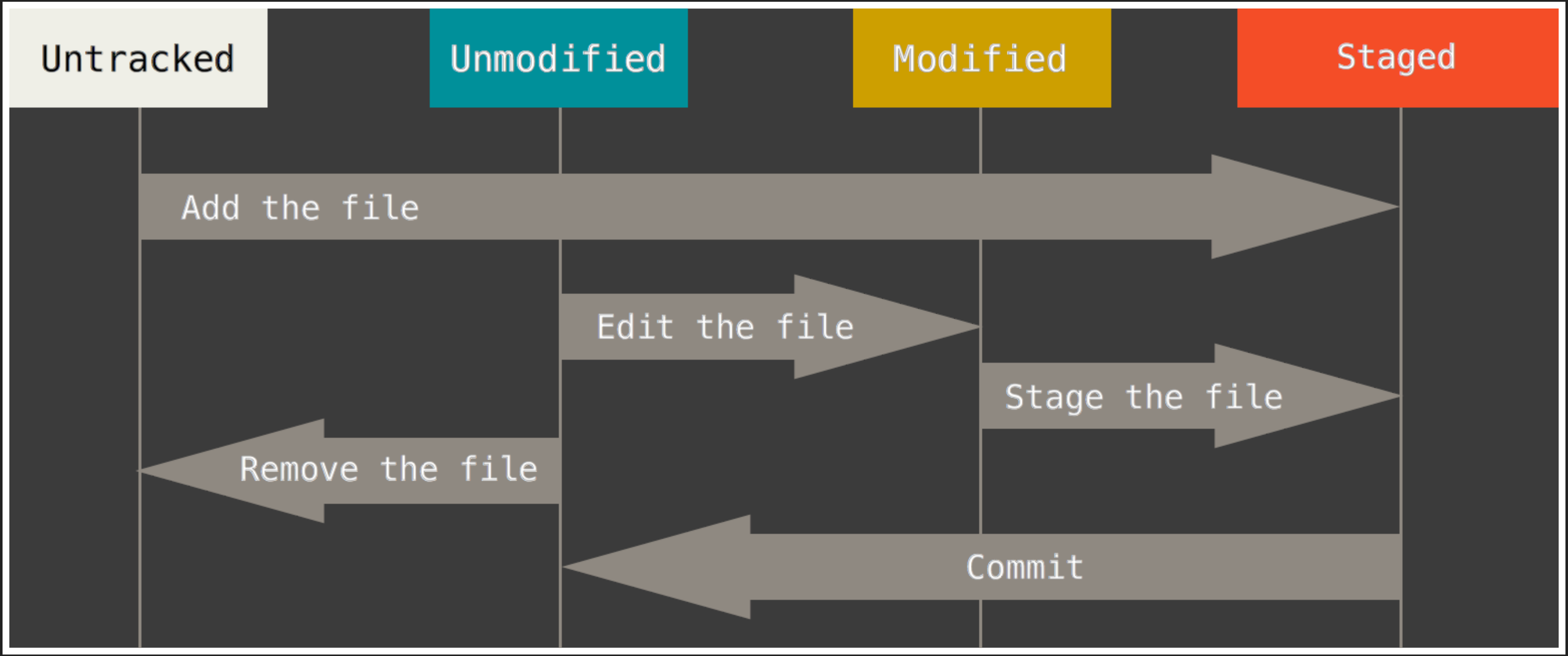
- Developed in 2005
  - for Linux Kernel Development by Linus Torvalds
- Used by Google, Facebook, Microsoft, Twitter, Netflix ….
- Distributed Version Control System

HOW DOES IT WORK

- Git keeps a snapshot of every commited change



FILE LIFECYCLE



LET'S TRY IT

FIRST SETUP. TELL GIT WHO YOU ARE

```
git config --global user.name "Your Name"
git config --global user.email "Your email address"
# can also be set only for current repository
# use these commands on shared computers
git config --local user.name "Your Name"
git config --local user.email "Your email address"
```

IMPORTANT GIT COMMANDS USED

```
git init # initialize a empty repository in current directory
git status # check the status of the repository
git diff # see what has changed in detail
git add file.txt # add file to staging area
git commit # commit the file
git commit -m "commit message" # specify message in command line
git commit -am "commit message" # add and commit modified files
git commit --amend # fix last commit, e.g. forgot file or typo in co
mmit message
git log # see commit history
git log -p # see differences of each commit
git log -2 # see only last 2 commits
git checkout # checkout branch tag or commit
git tag # list tags
git tag -a v1.1 -m "version 1.1" # create tag v1.1 with message "ver
sion 1.1"
git branch # create branch
git merge # merge branches
```

GIT COMMANDS FOR WORKING WITH REMOTE REPOSITORIES

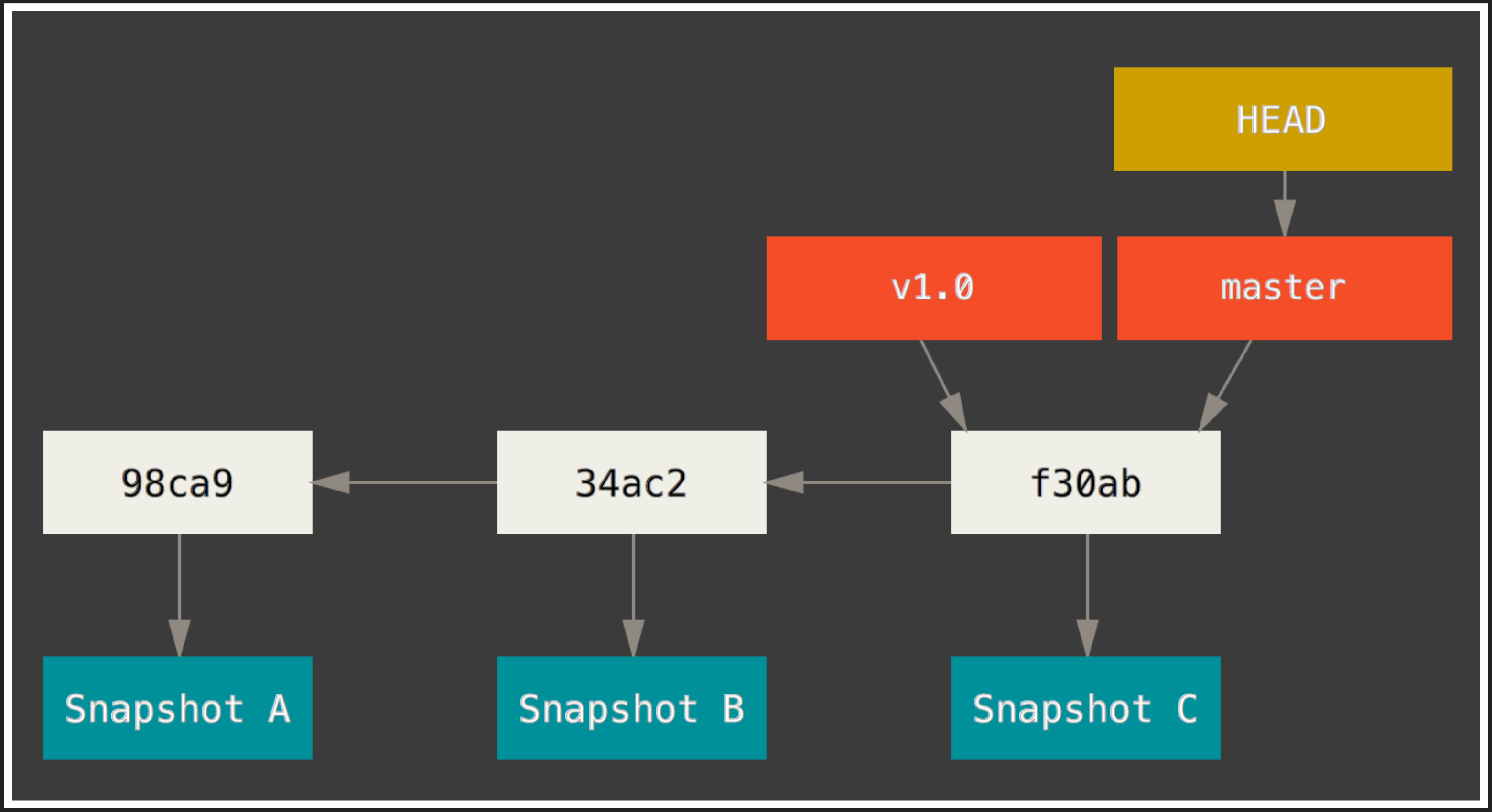
```
git clone url # clone the git repo from the url
git remote add name url # add remote repo at url and give it name "n
ame"
git pull # pull changes from the remote repo
git push # push your changes to the remote repo
```

GIT TAGS

Give a name to a commit to easily get back to it later

GIT BRANCHES

- Useful if developing in parallel or fixing bugs
- master is the default branch.
- Git uses pointers to keep track of branches and tags


HEAD
v1.0
master
98ca9
34ac2
f30ab
Snapshot A
Snapshot B
Snapshot C

CREATING A BRANCH CREATES A NEW POINTER

```
git branch testing
```

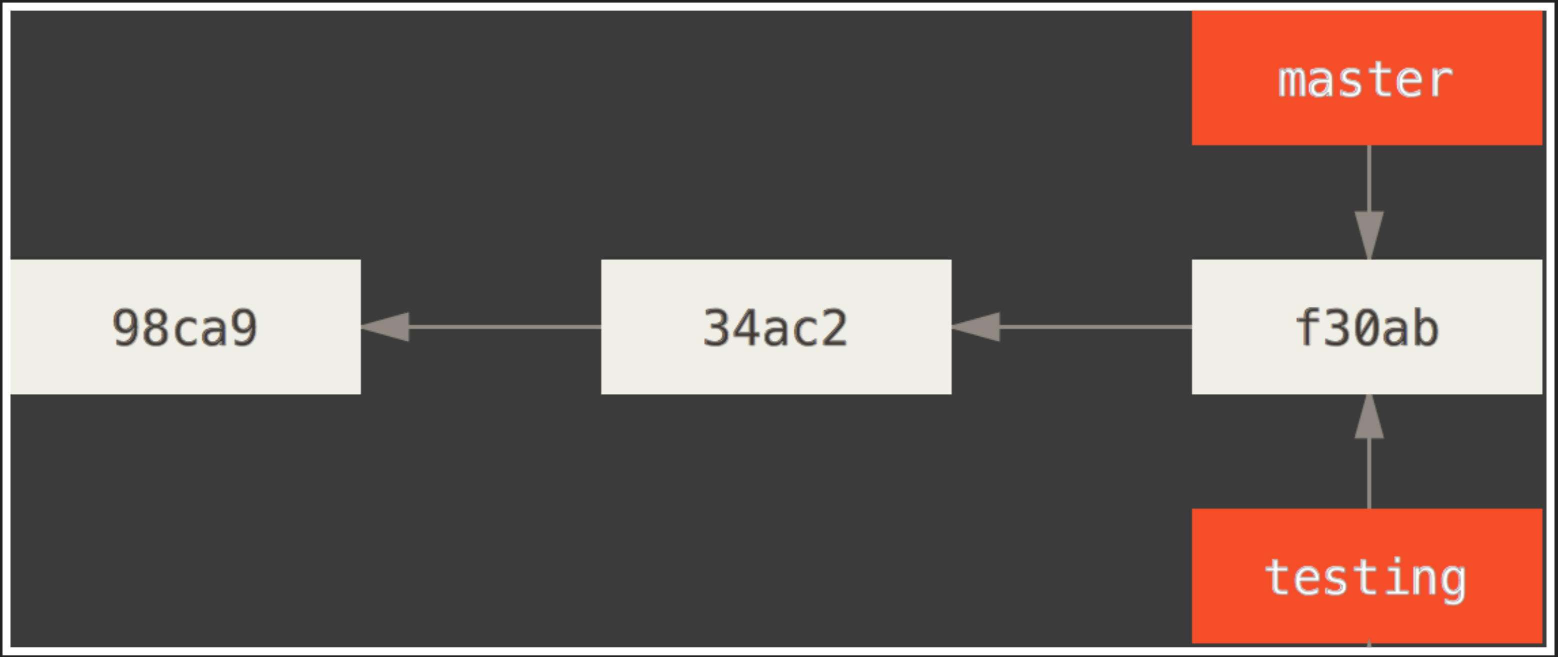
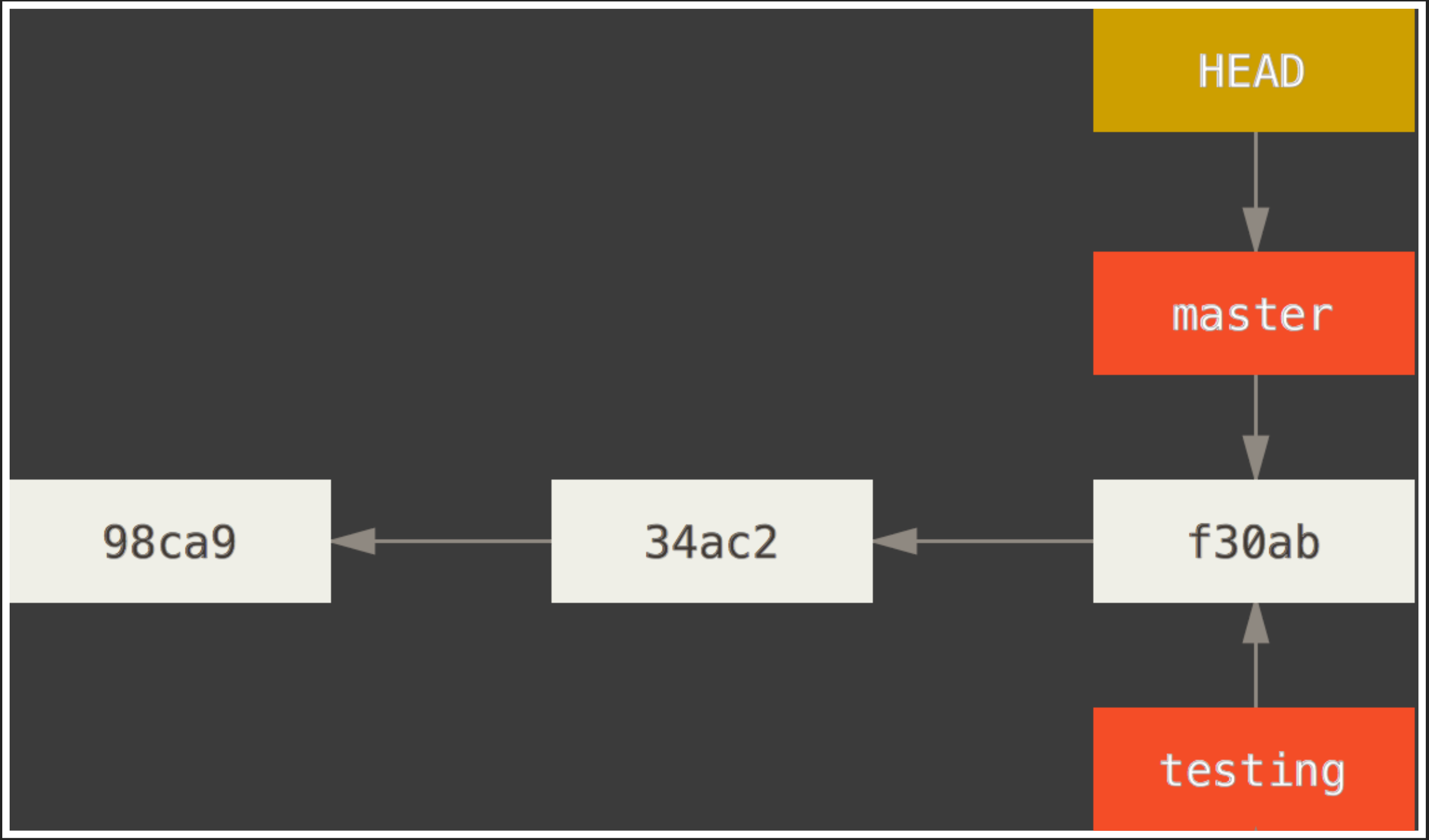


Figure 6: New branch testing

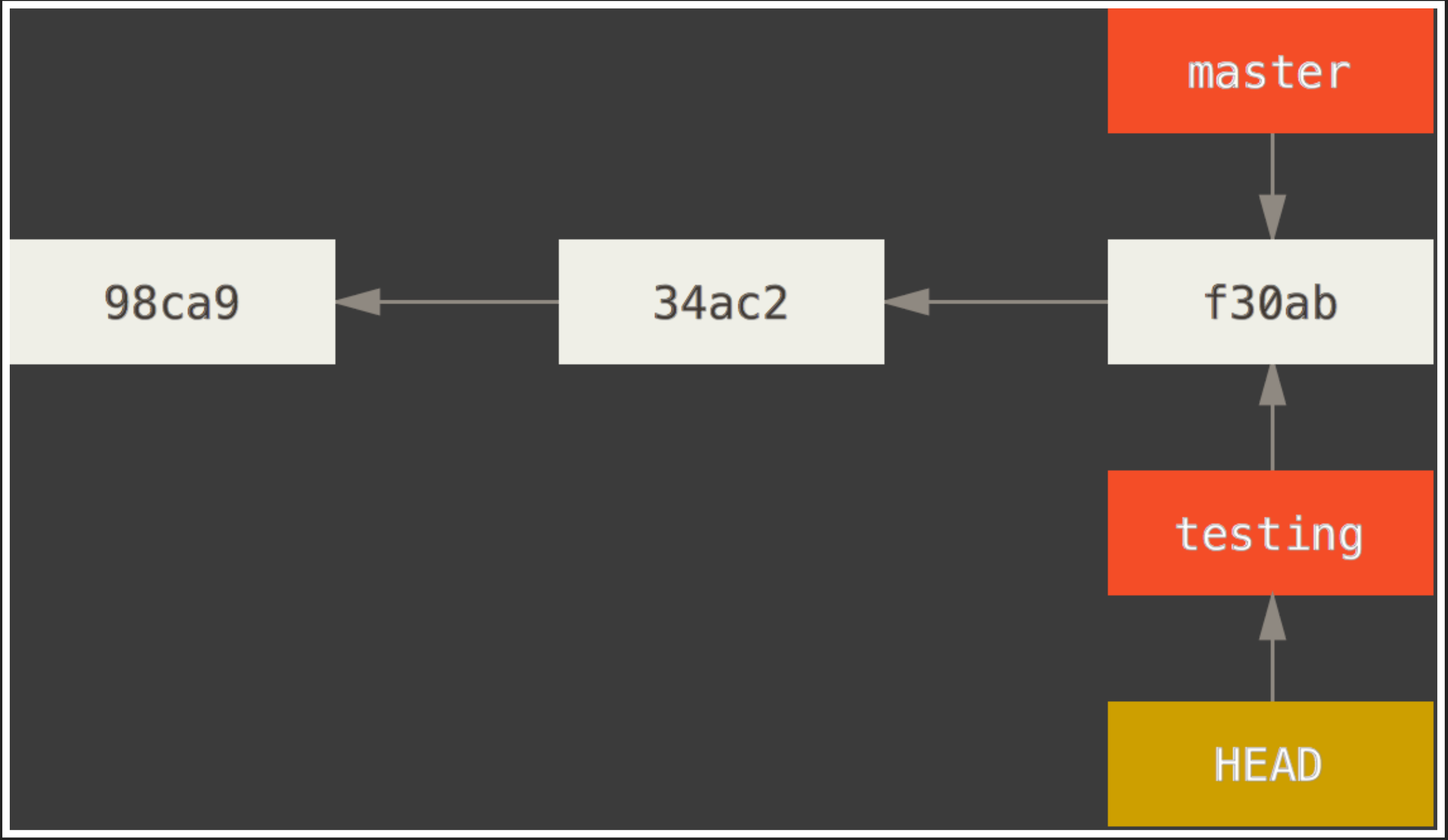
SWITCHING BRANCHES

HEAD pointer is at current position.



SWITCHING BRANCHES

```
git checkout testing
```



MAKING CHANGES TO A BRANCH

```
#edit file
git commit -a -m "made a change"
```

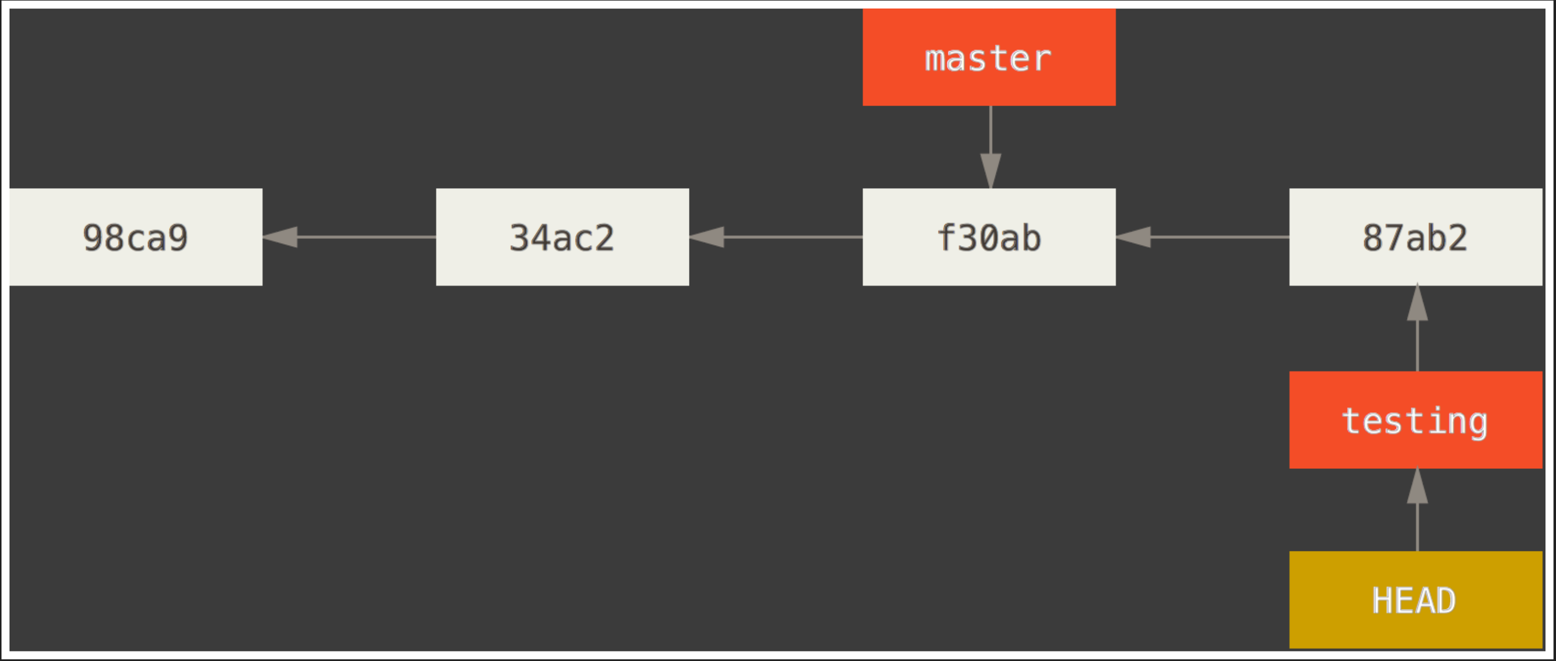


Figure 9: The testing branch moves forward

MERGING BRANCHES

To get changes in one branch also in another branch

```
git checkout master
git merge testing
git branch -d testing # delete testing branch
```

GIT HOSTING SERVICES

http://www.github.com/
: hosts a lot of open source projects

http://www.gitlab.com/
: free private repositories

SHORT GITHUB DEMO

CAVEAT

- only really works for text files
- git hosting services also show differences between image files and other file types
- Not good for big binary files

MORE INFORMATION

- learn Git in 15 Minutes
- Official Documentation
- List of additional Resources
- List of GUI Clients - SourceTree is supposed to be good.

- Images in this presentation are from the ProGit Book