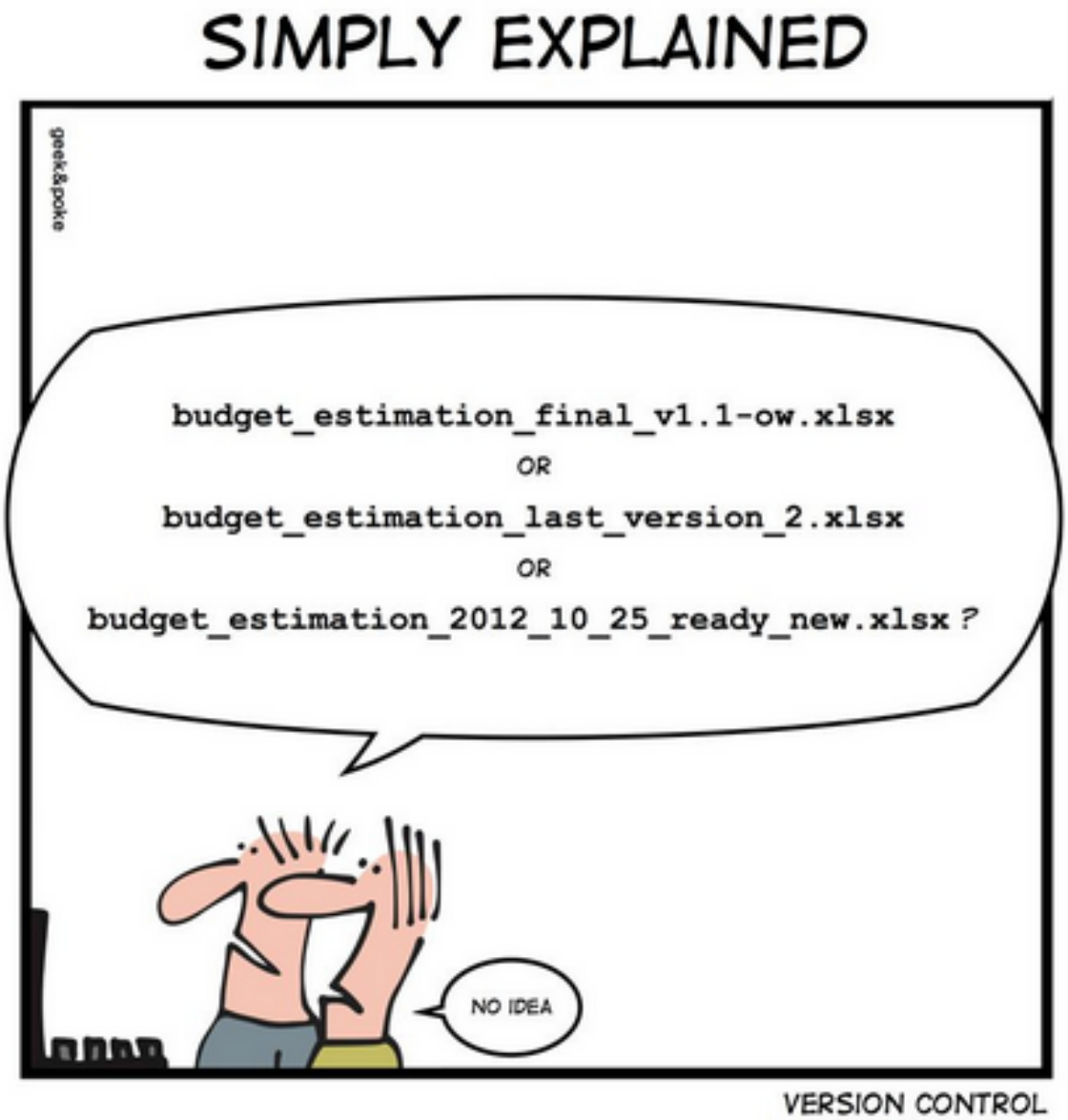
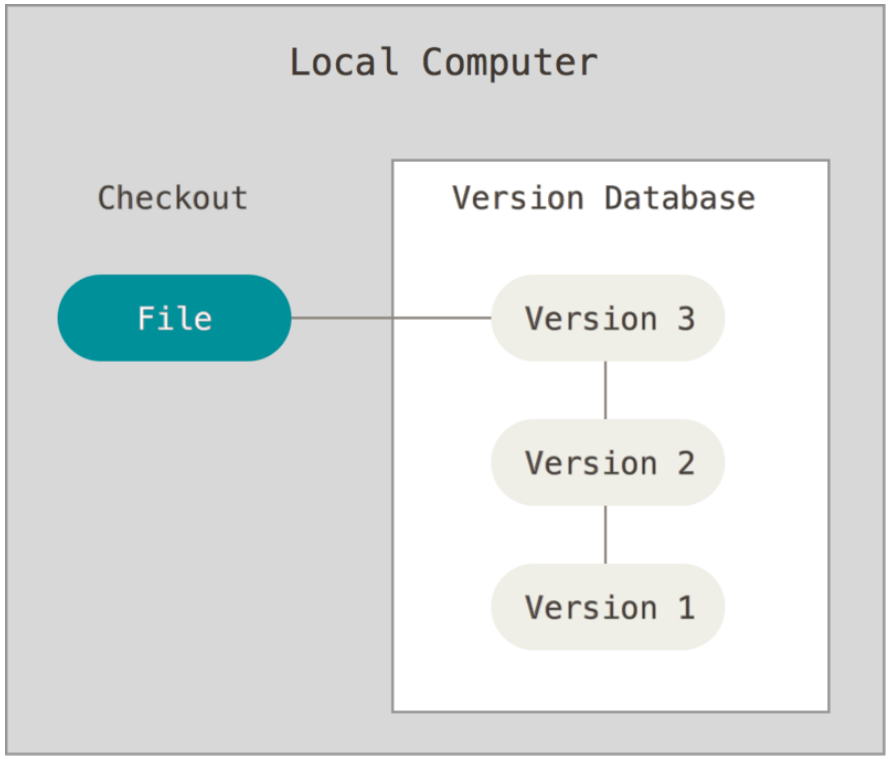
Version Control with Git

What is version control?



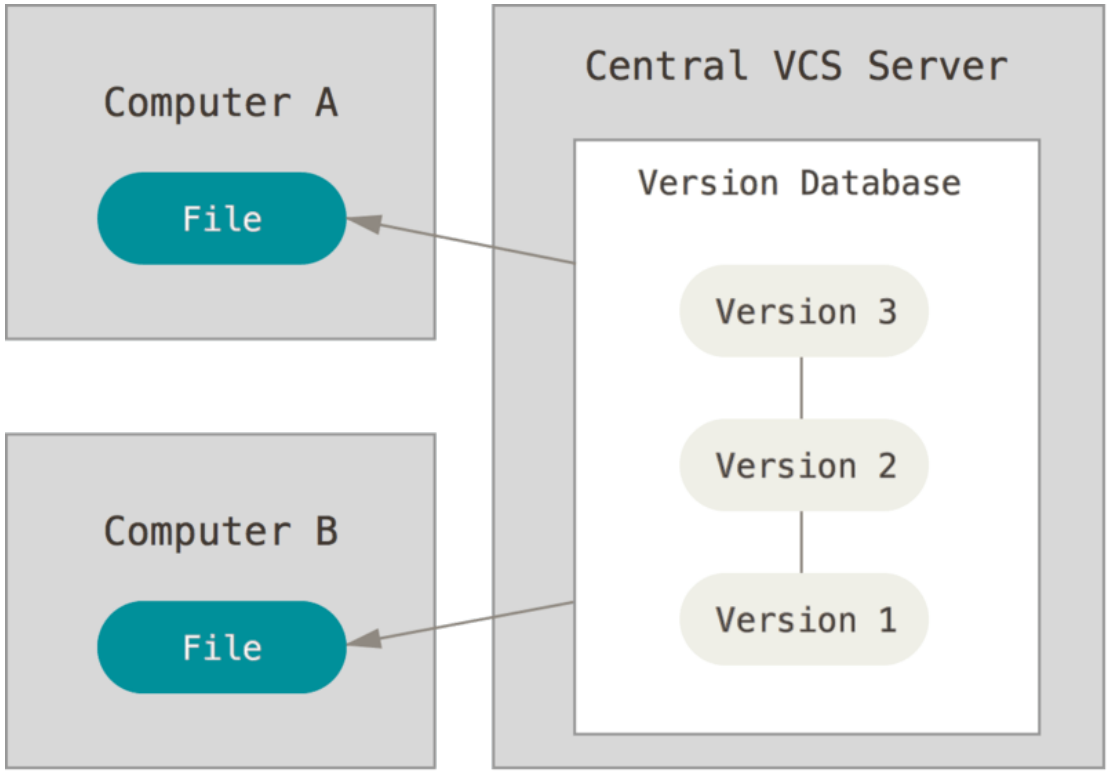
What is version control?

A version control system:



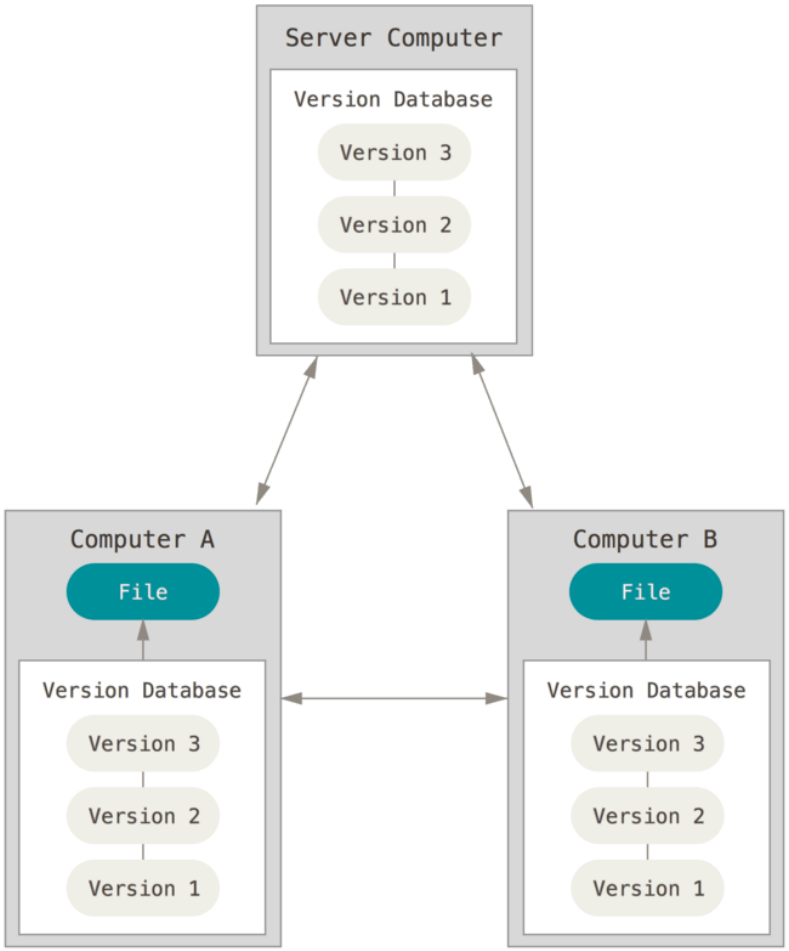
Centralized version control systems

In a central VCS there is **one** repository containing the master version (the "trunk") of the source code.



Example: SCCS, RCS, CVS, SVN

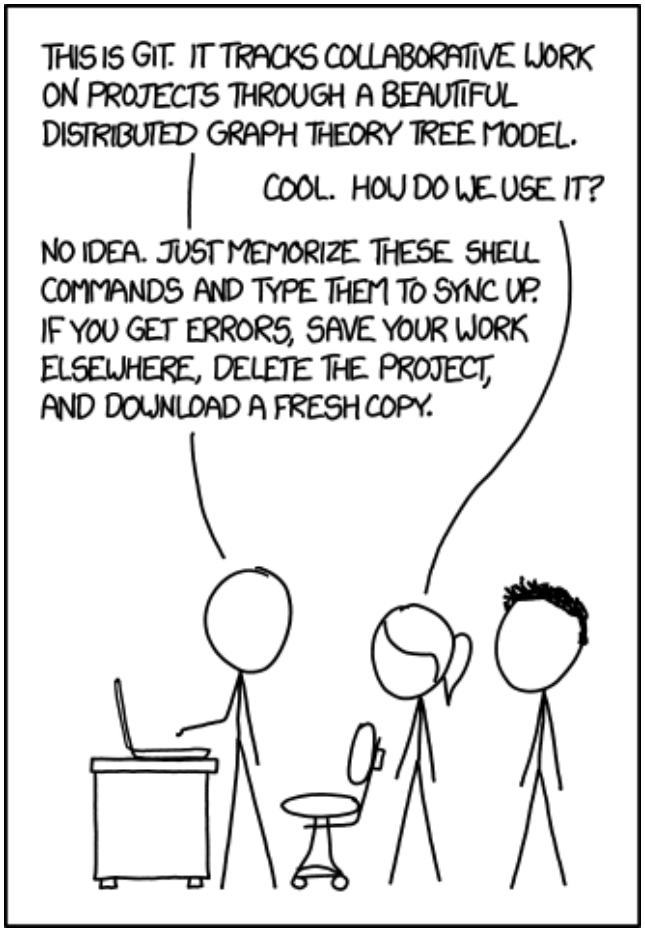
Distributed version control systems



Example: Git, Mercurial

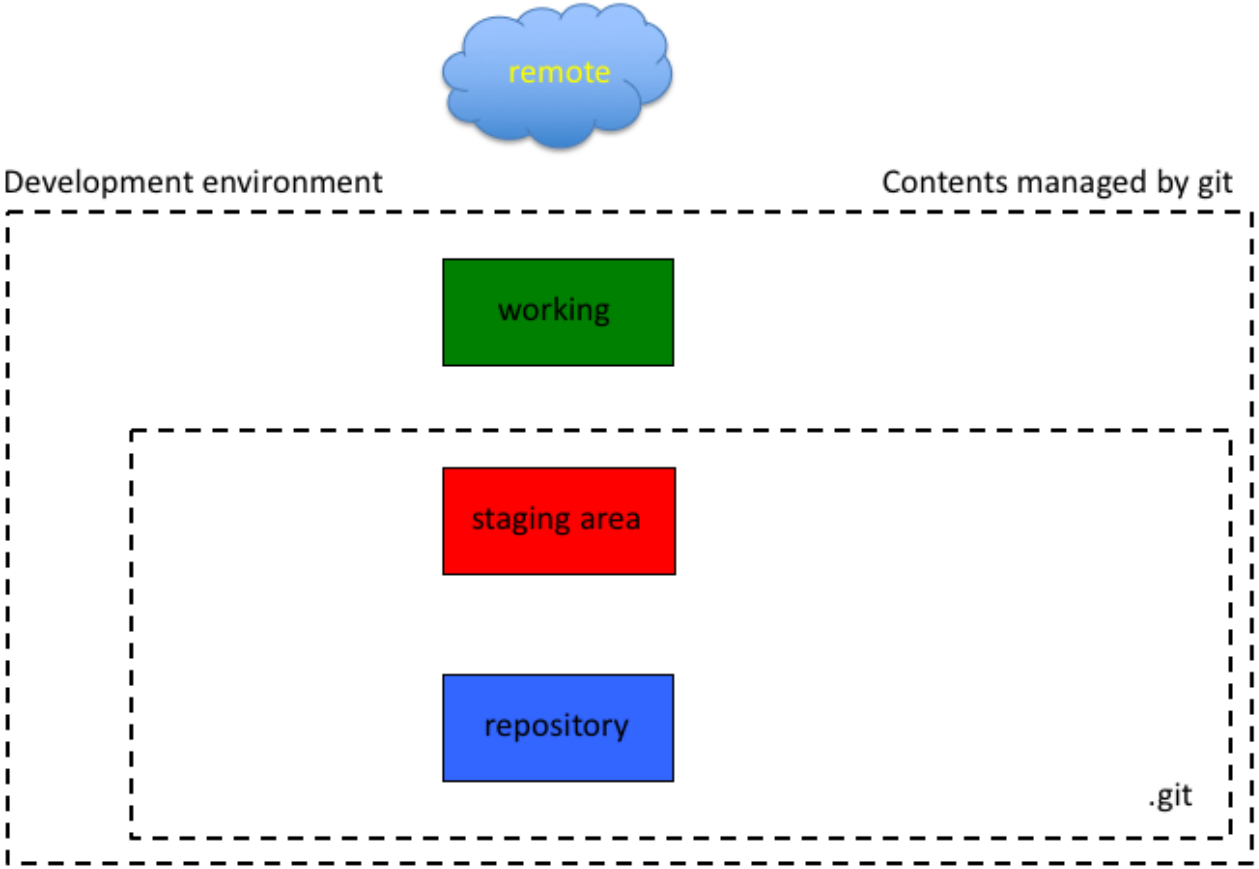
The Git concept

What is Git?[1]

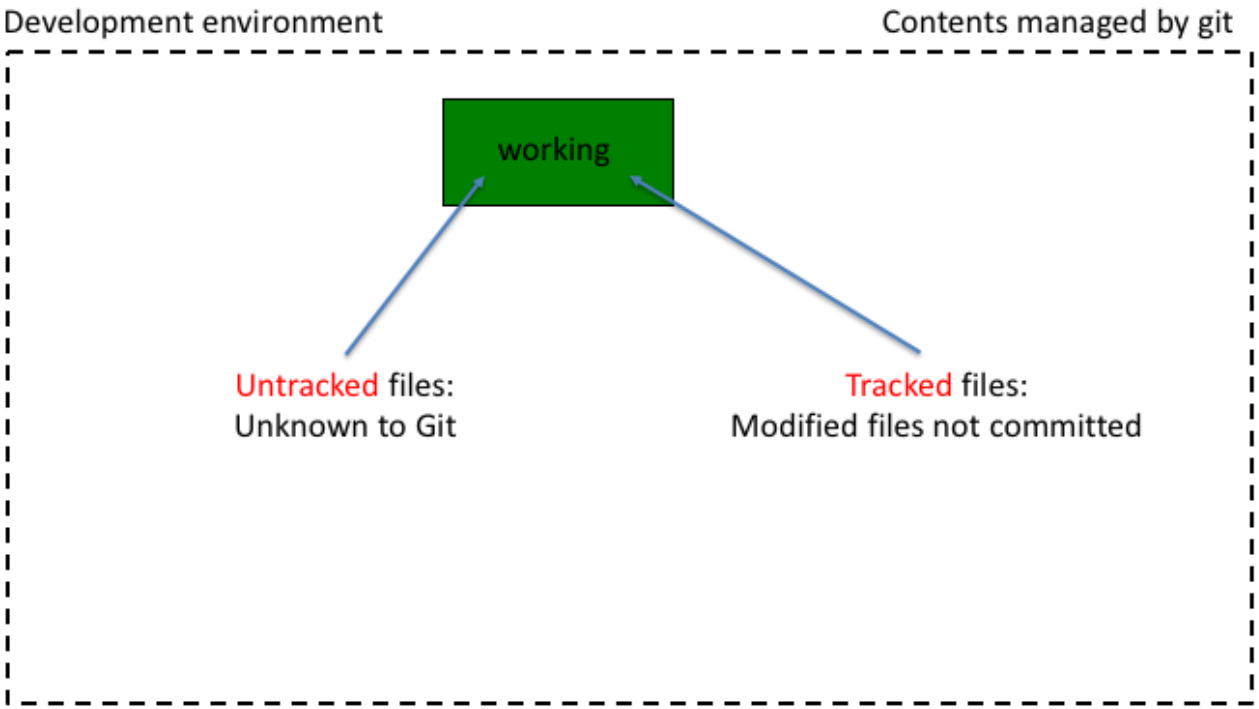


[1] https://git-scm.com

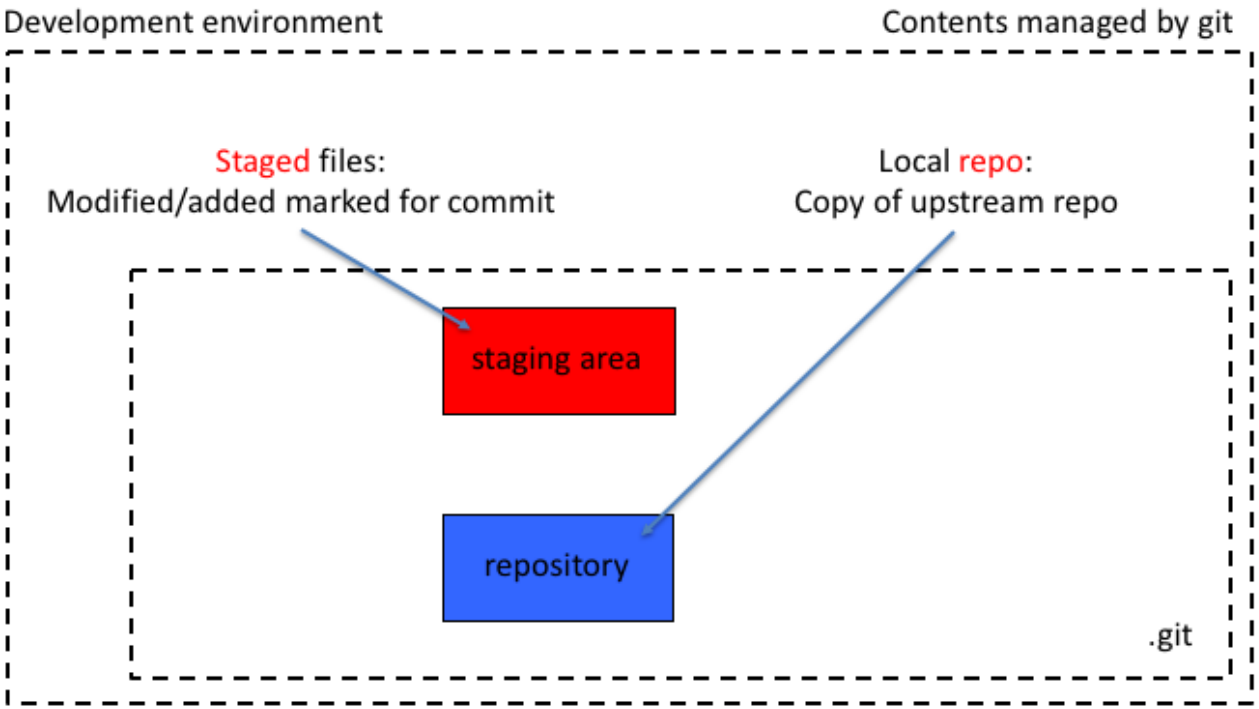
The Git Setup



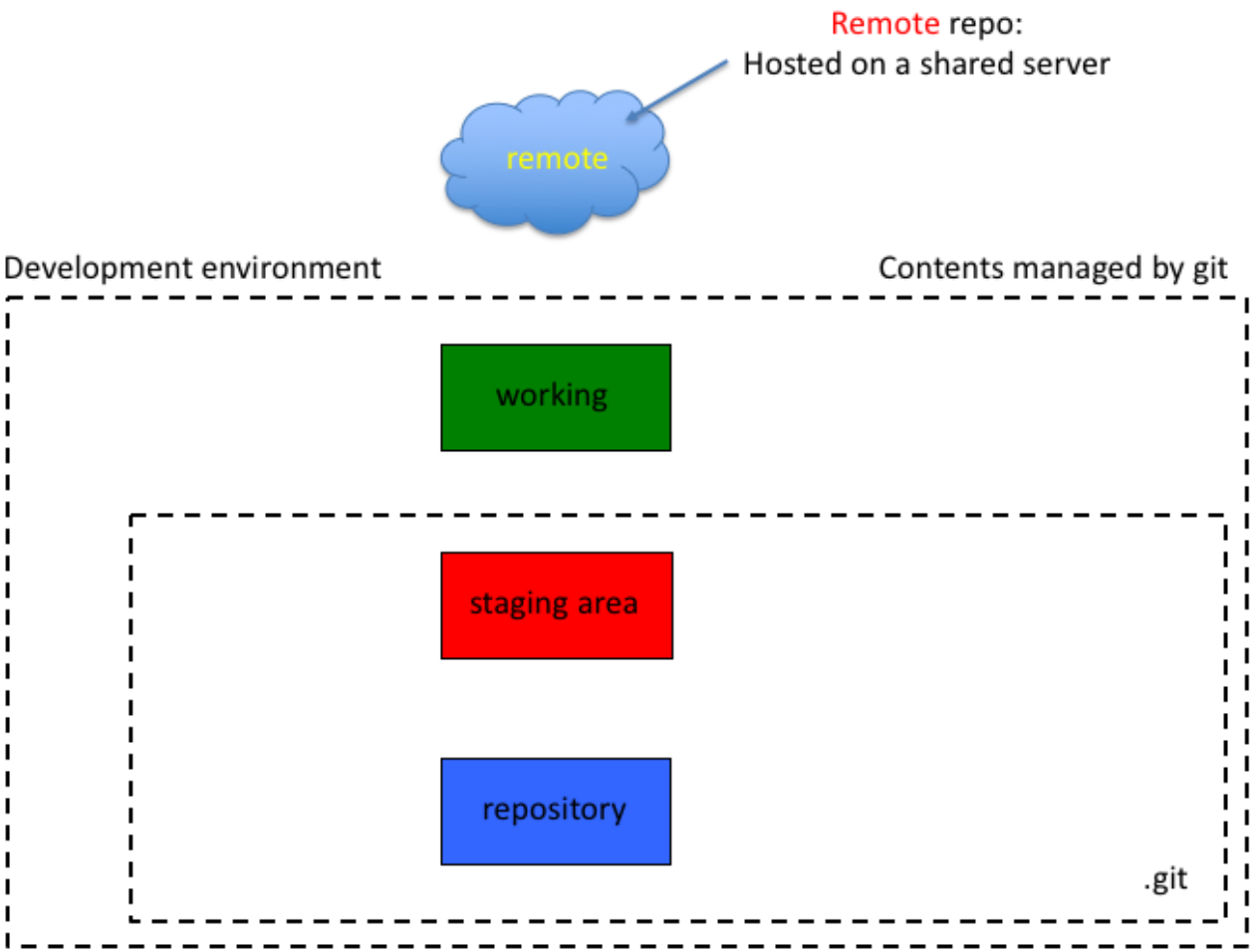
The Git Setup



The Git Setup

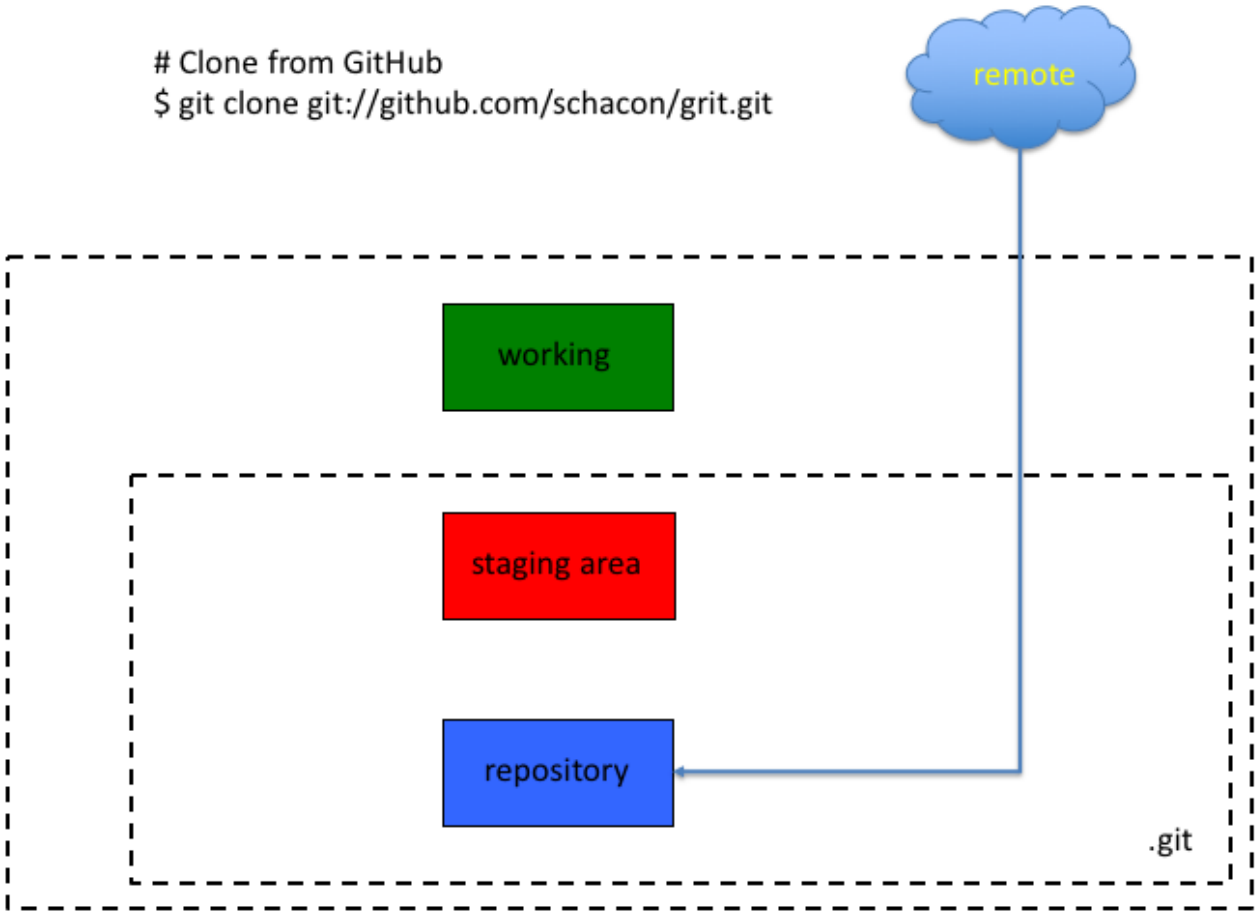


The Git Setup



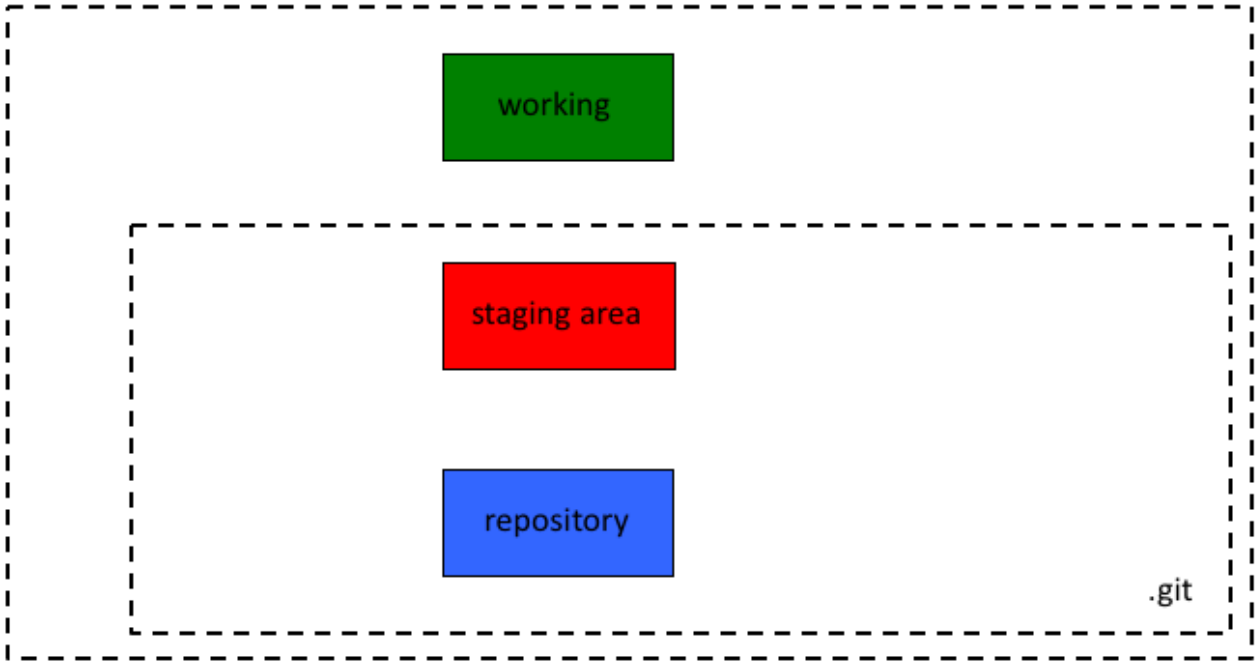
Setting up a Git repo

```
# Clone from GitHub
$ git clone git://github.com/schacon/grit.git
```



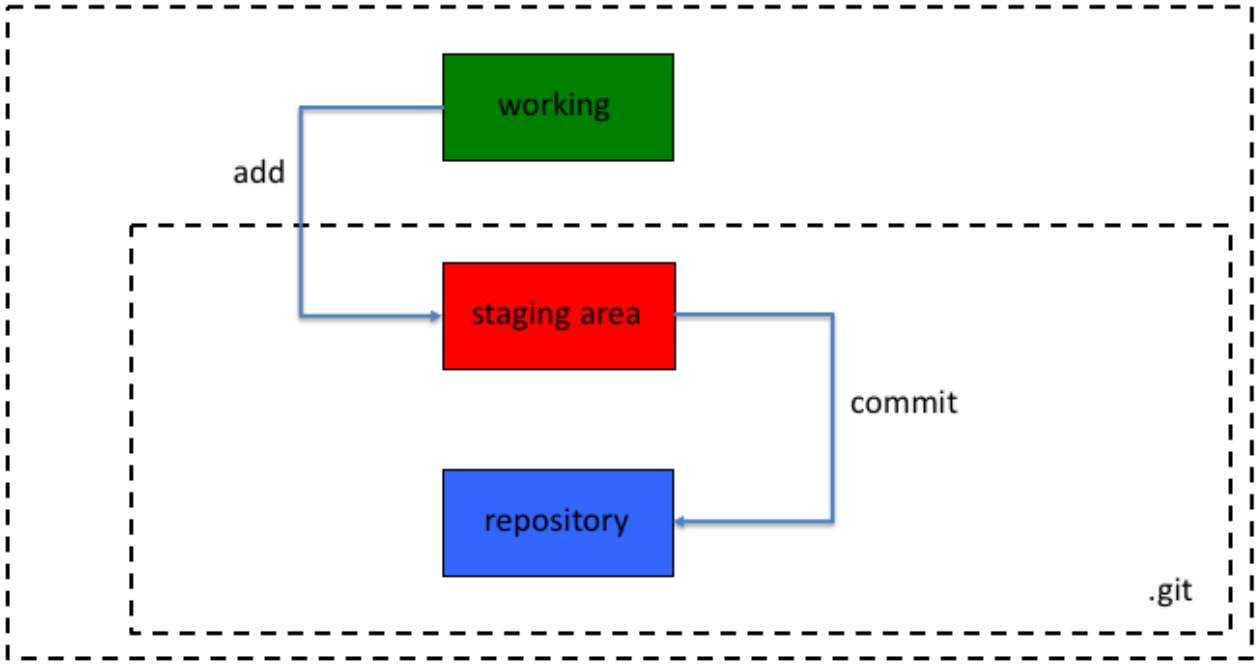
Setting up a Git repo

```
$ mkdir myproj
$ cd myproj
# Put myproj under version control
$ git init
Initialized empty Git repository in /some/path/myproj
```



Making changes in the working directory

```
# Add a file
$ git add README
# Save a file
$ git commit –m 'Add a file' README
# Show commit log
$ git log
```

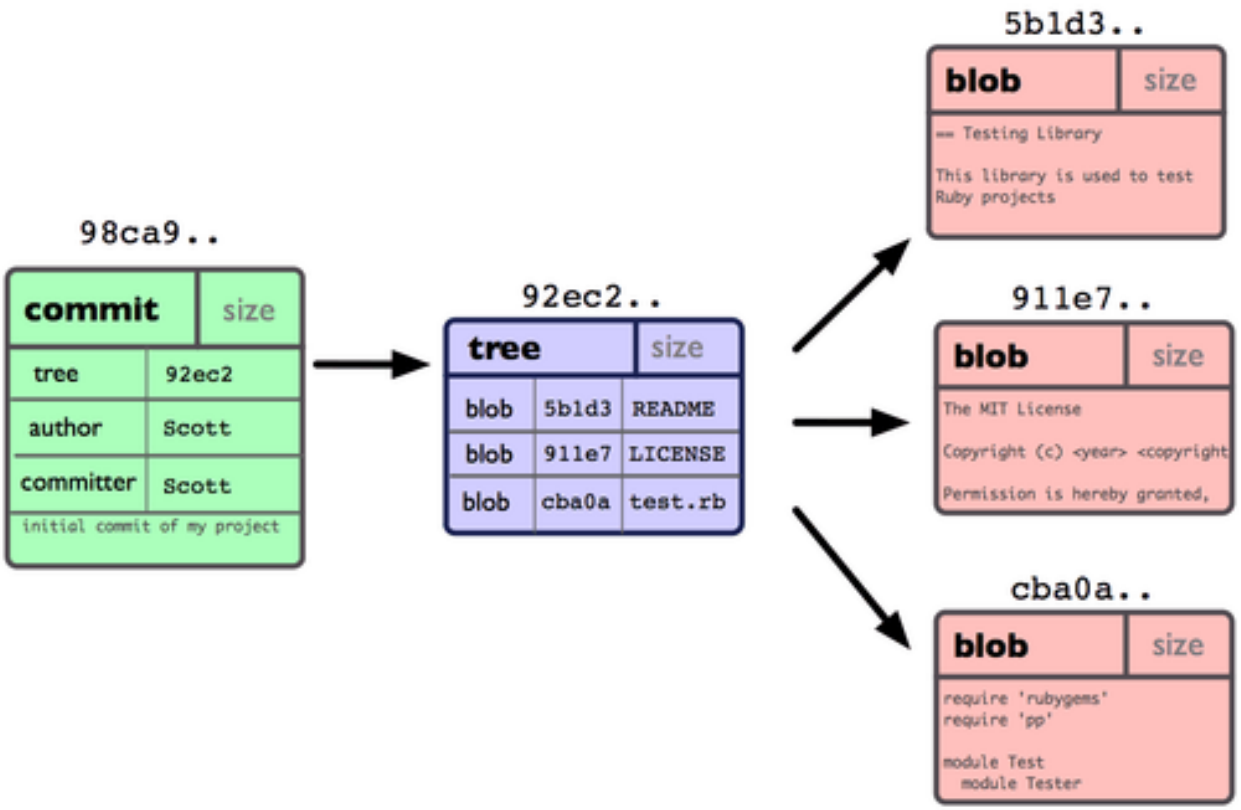


Git internals

For each change set, i.e. each commit:

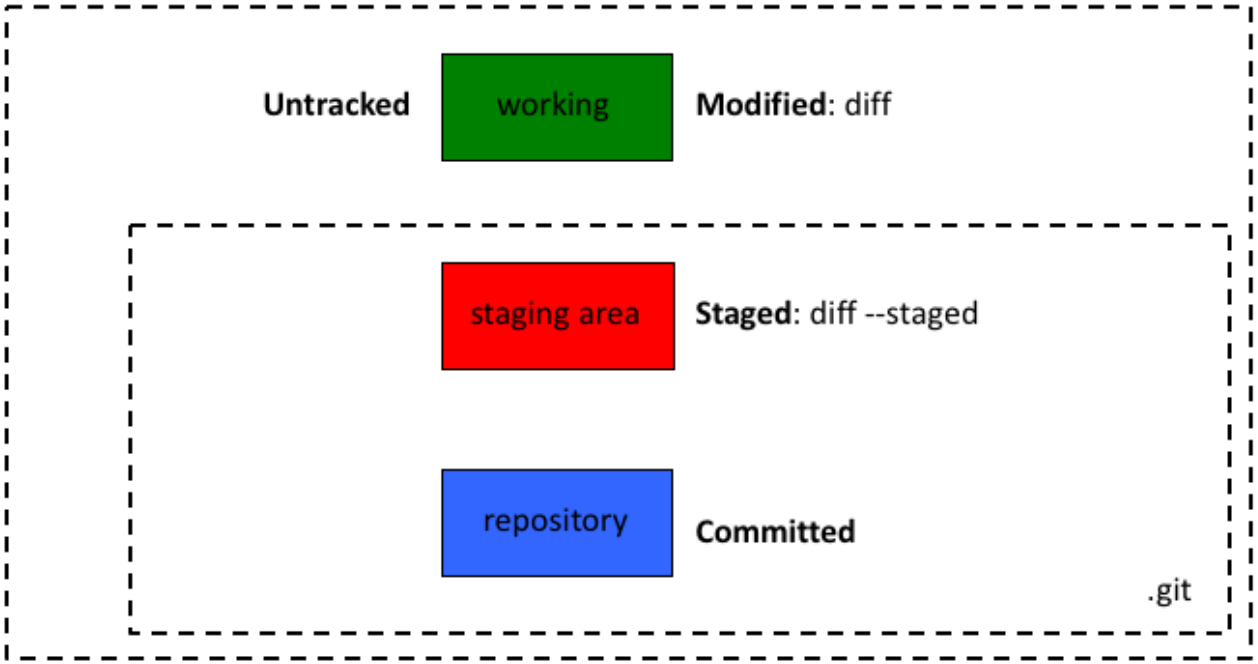
- Git generates a **checksum**
  - Git uses SHA1 algorithm to create checksums. SHA1 guarantees complete uniqueness.
  - 40-character hexadecimal string
  - e.g. 1fbb5af06b9e4facff4170fc687ecdd143daad50
- Git uses SHA1 string to organize database of references
- Fortunately, one can use a subset of SHA1 string (typically 4-8 characters)

Git internals



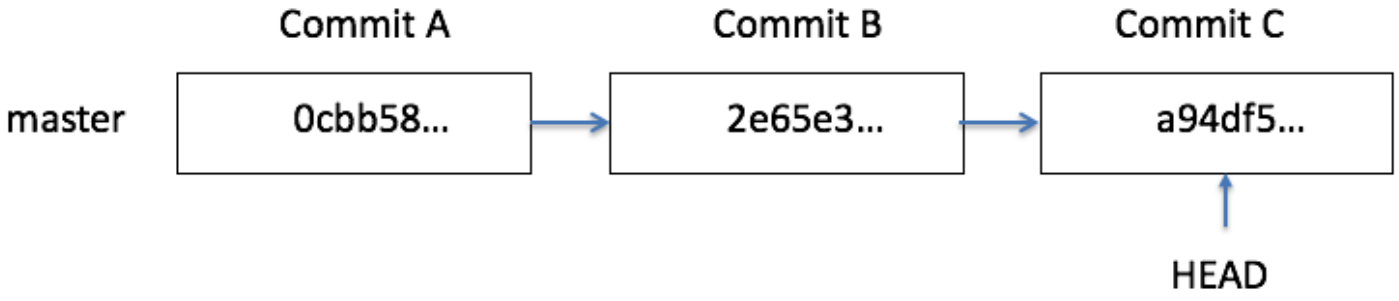
File Status

```
# Show commit log
$ git log
# Show status of repo
$ git status
# Differences
$ git diff
$ git diff --staged
```



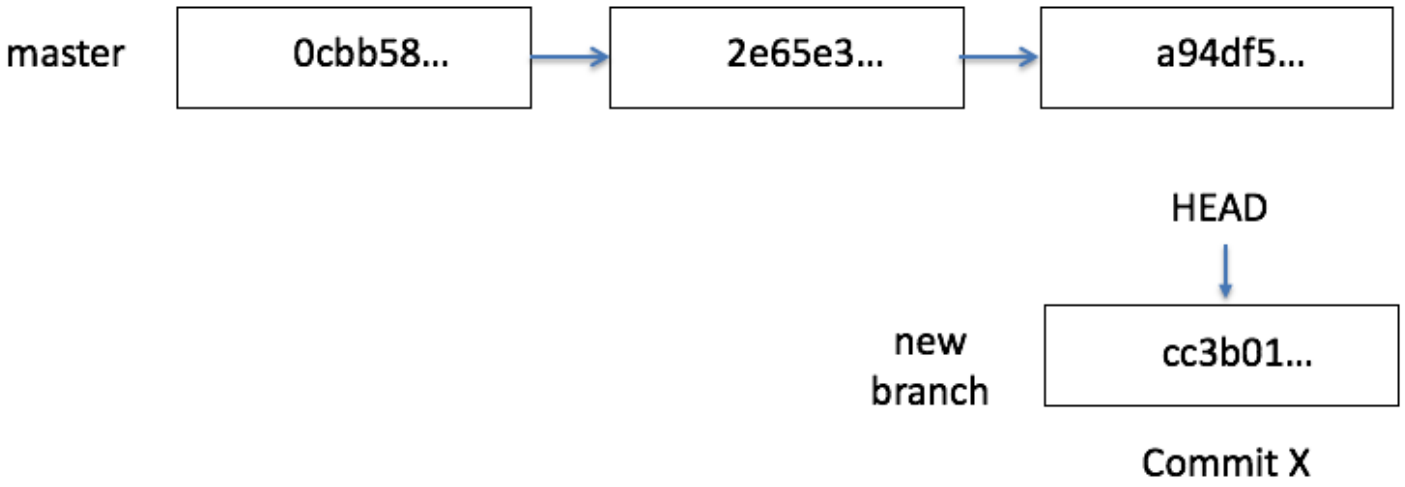
Branching

Commits are repository snapshots



Branching

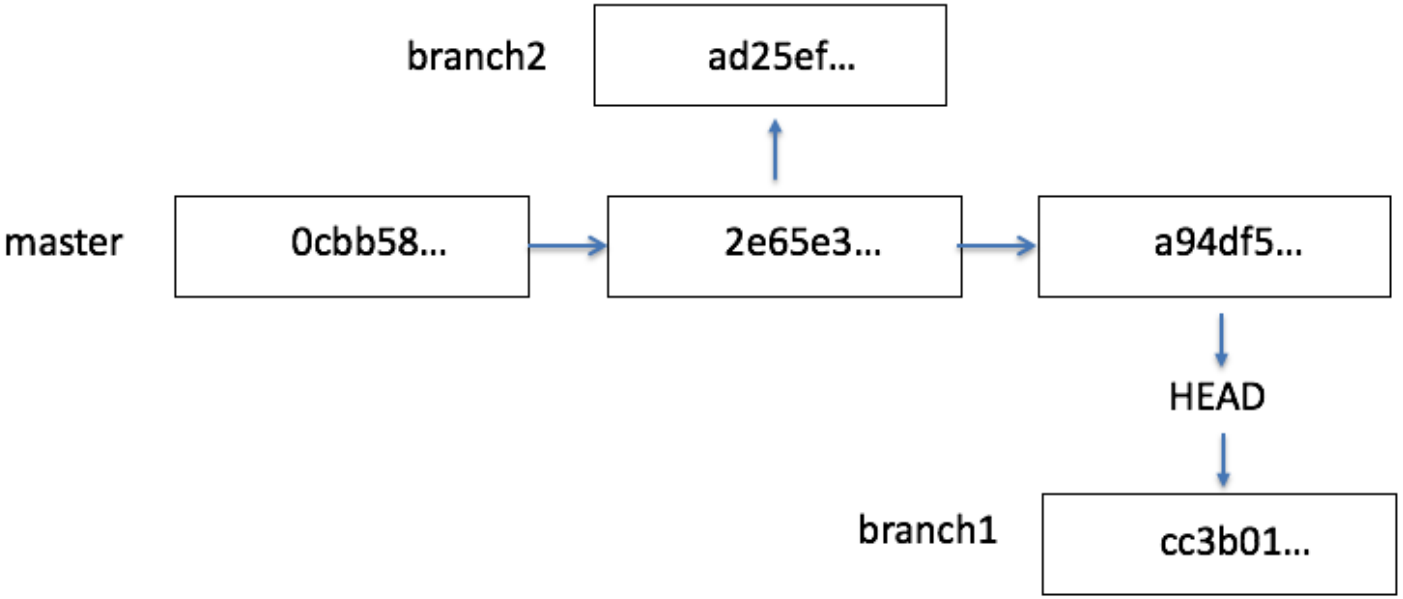
A branch is a pointer to commit - not a full copy.



HEAD points to tip of current branch in repository

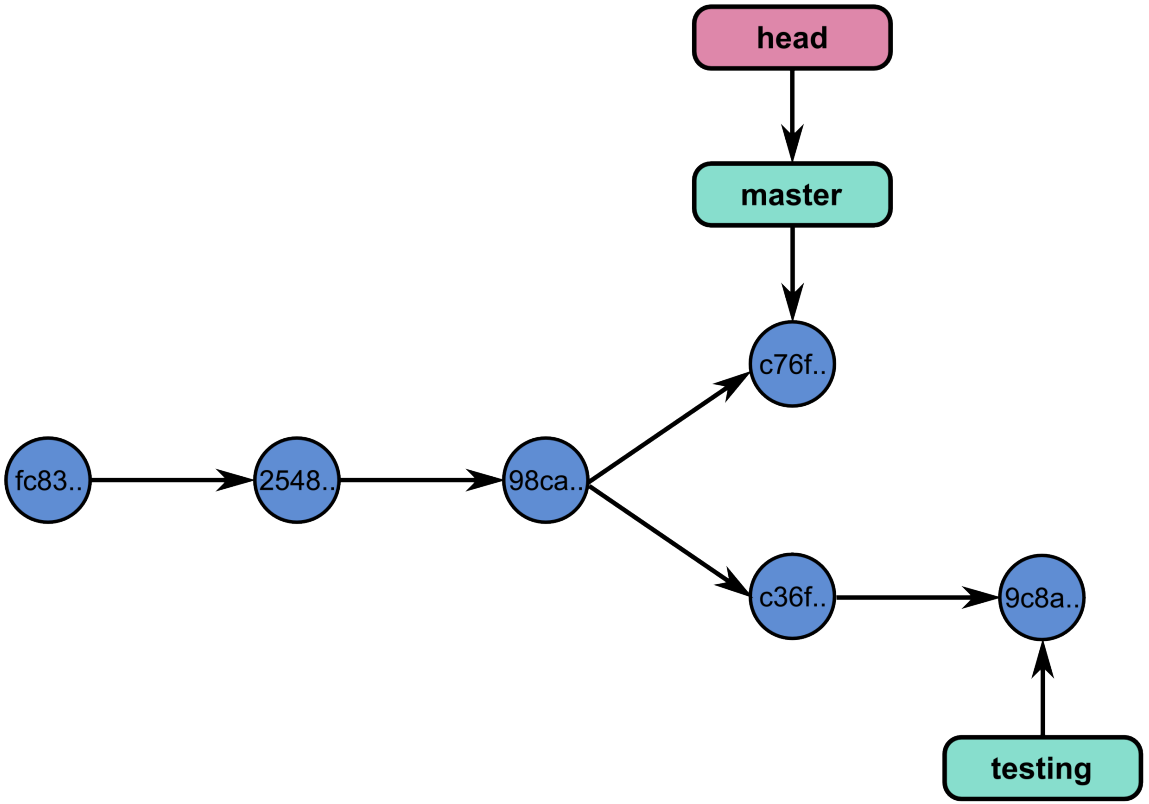
Branching

Can have many branches



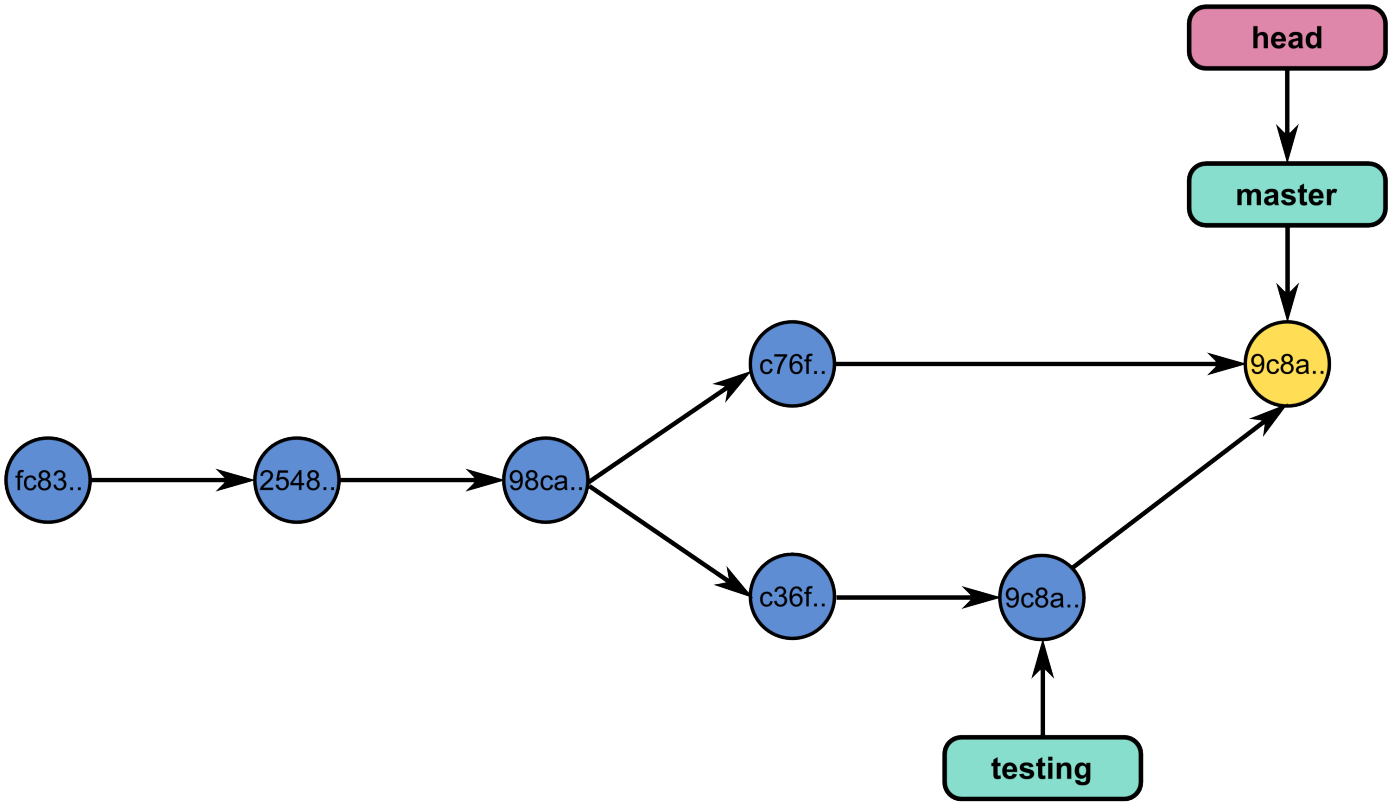
Merging

We have two branches

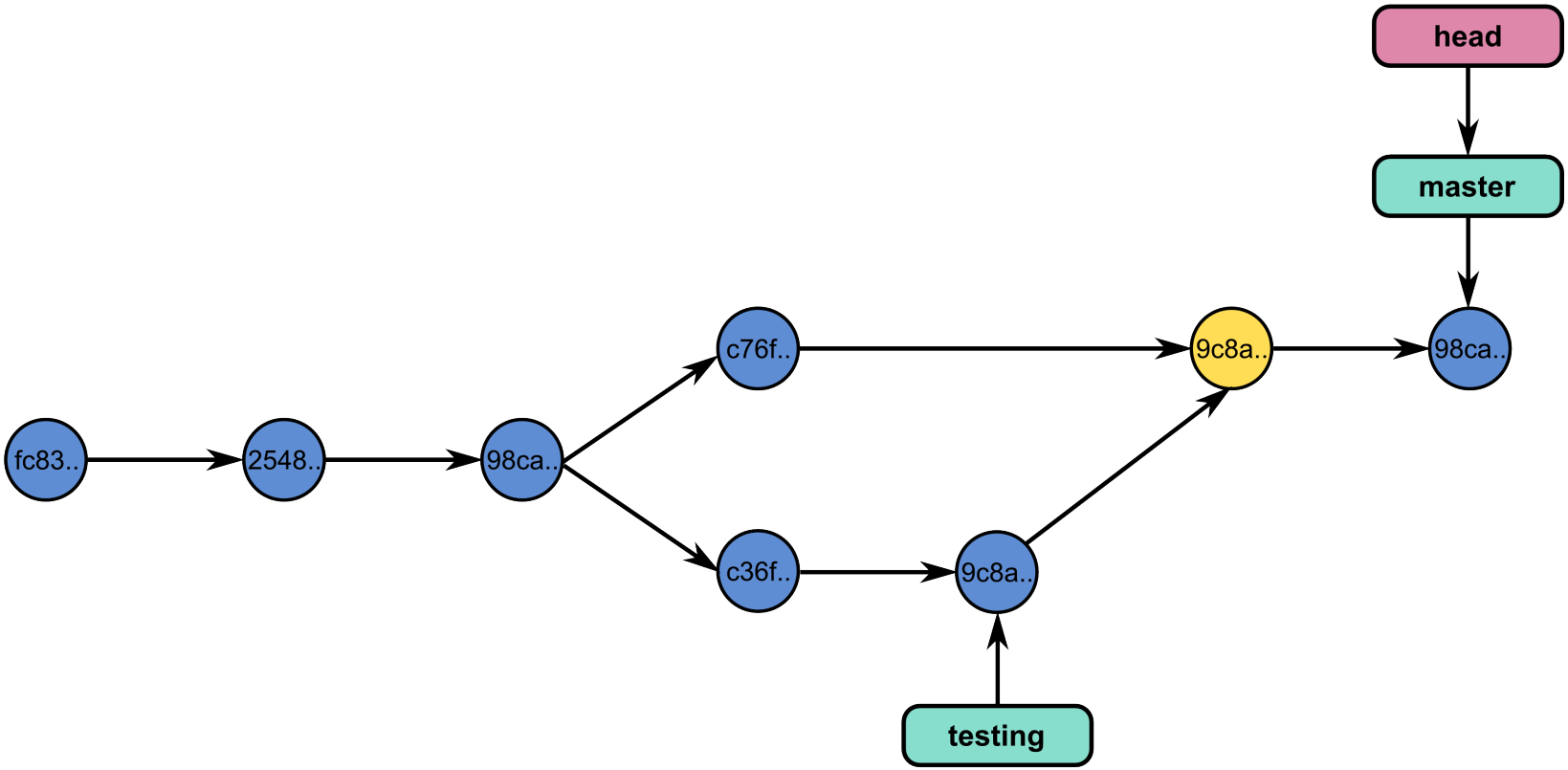


Merging

Merge testing into master



Merging



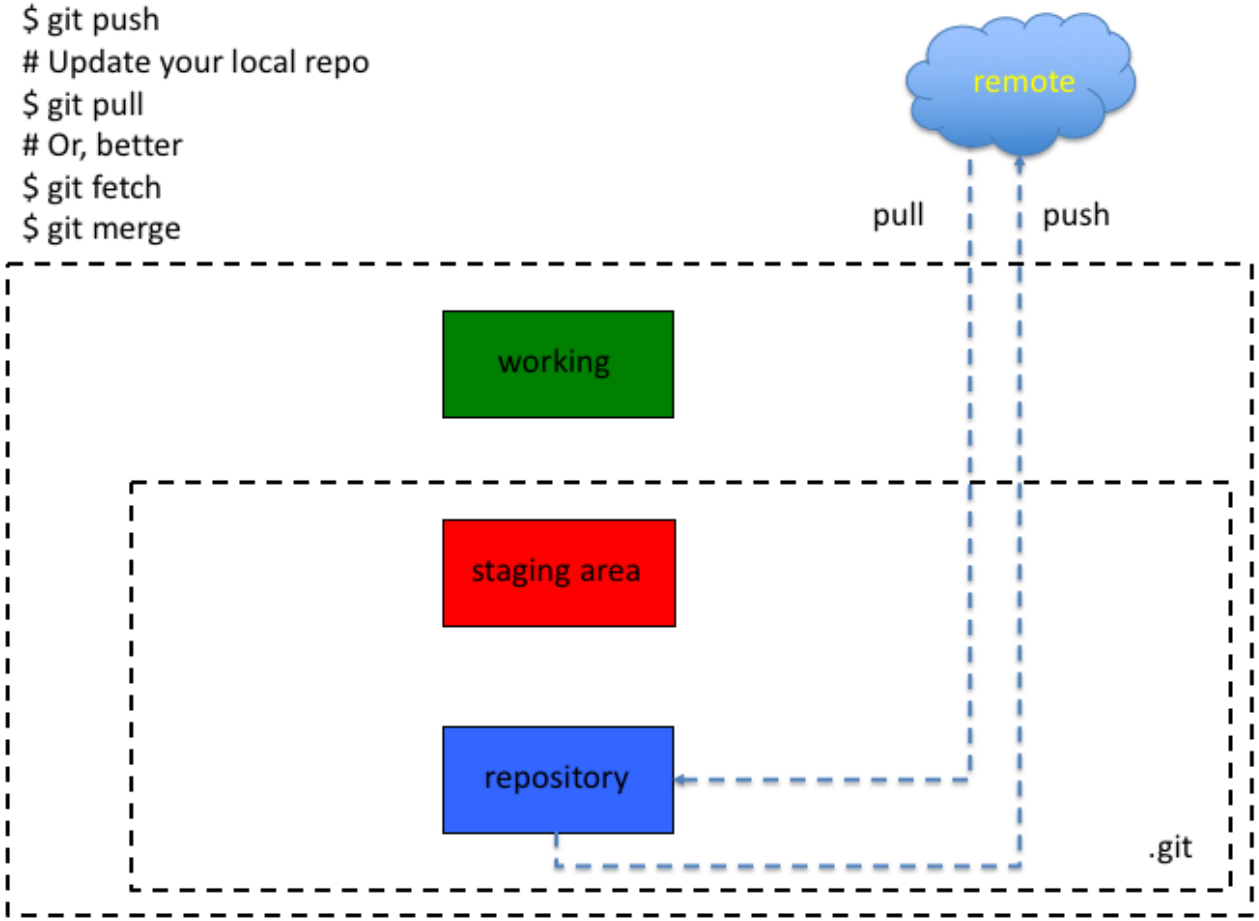
Merge conflicts

When a conflict emerges, you must manually edit the files

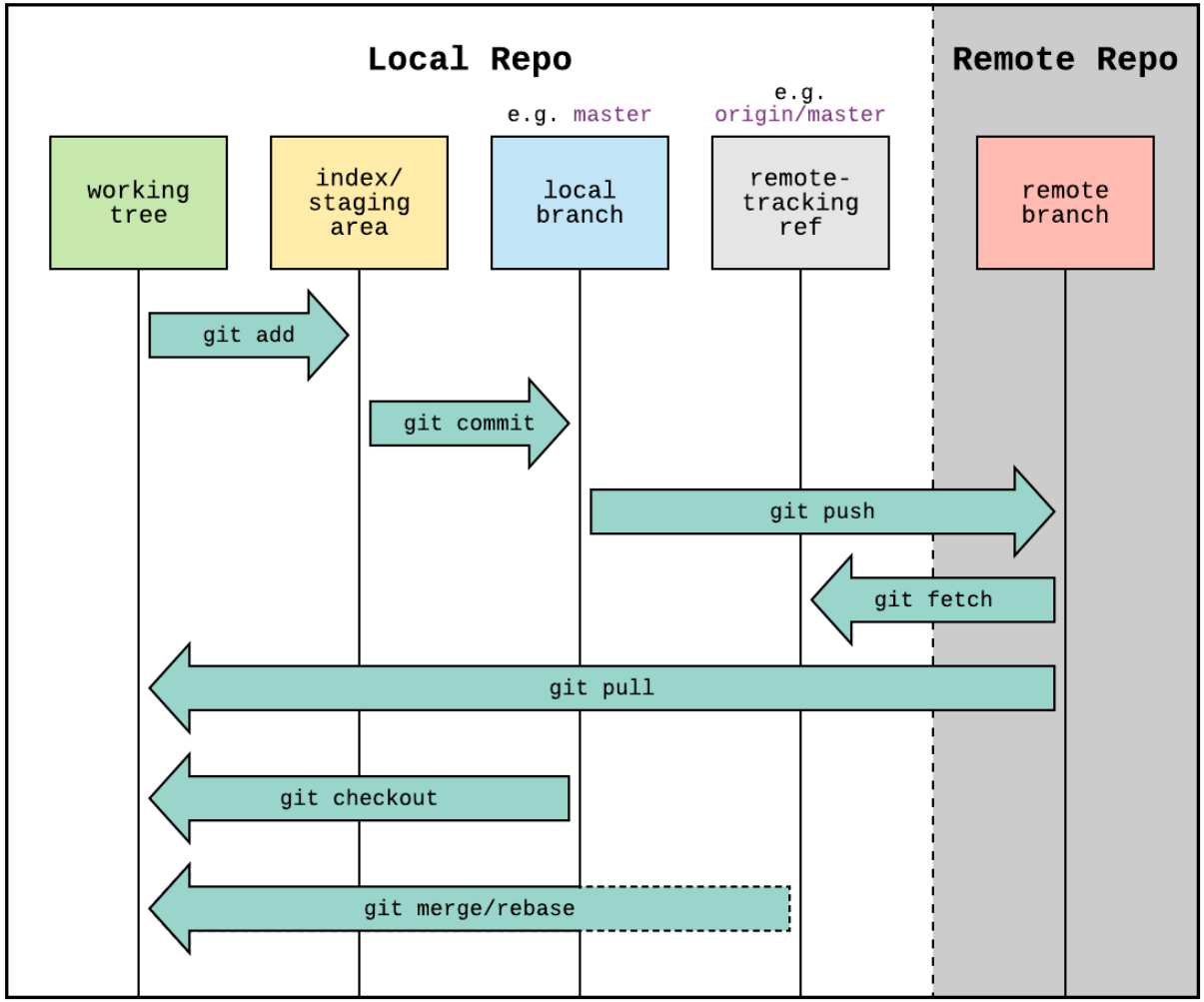
```
<<<<<<< HEAD:calc.py
print 'the average is', sum(x) / float(len(x))
print 'sumsqdiffs is', sum(diffs)
=======
avg = sum(x) / float(len(x))
sumsqdiffs = sum(diffs) / float(len(x))

print 'average is', avg, 'and sumsqdiffs is', sumsqdiffs
>>>>>>> edgier:calc.py
```

Syncing your repo

```
# Send changes to remote
$ git push
# Update your local repo
$ git pull
# Or, better
$ git fetch
$ git merge
```



Git workflow



Resources

- Git Community Book
- Pro Git
- A Visual Git Reference
- Git Cheat Sheet

Exercise 1

1. create a new directory, open it and perform a “git init“ to create a new git repository.
2. run “git config –global [options]“ to set up user configurations
3. create a file and run “git add“ and “git commit“ command
4. edit the file and run “git diff“
5. run “git add“ and “git commit“ command, again
6. type “git log –oneline –graph –decorate“ What do you see?

Exercise 2

- create a new branch “testing“.
- switch to that branch.
- check what branch you are in using “git branch“.
- create a python script that prints the first 10 integers, and commit it.
- look at the history of your repository.
- switch to the branch “master“, and look again at the history. What do you see?
- merge “testing“ into “master“, and look again at the history. What do you see?

The End

Questions?