

git

Git Introduction

Tao YANG



"FINAL".doc



FINAL.doc!



FINAL_rev.2.doc



FINAL_rev.6.COMMENTS.doc



FINAL_rev.8.comments5.
CORRECTIONS.doc



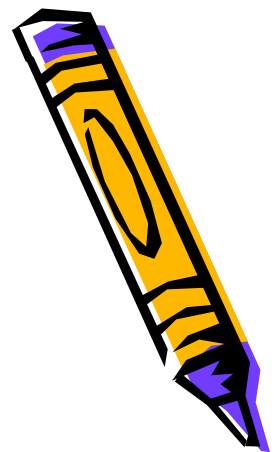
FINAL_rev.18.comments7.
corrections9.MORE.30.doc

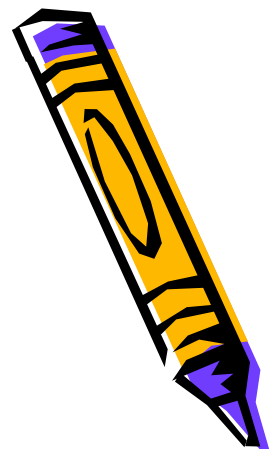


FINAL_rev.22.comments49.
corrections.10. #@\$%WHYDID
ICOMETOGRADSCHOOL????.doc

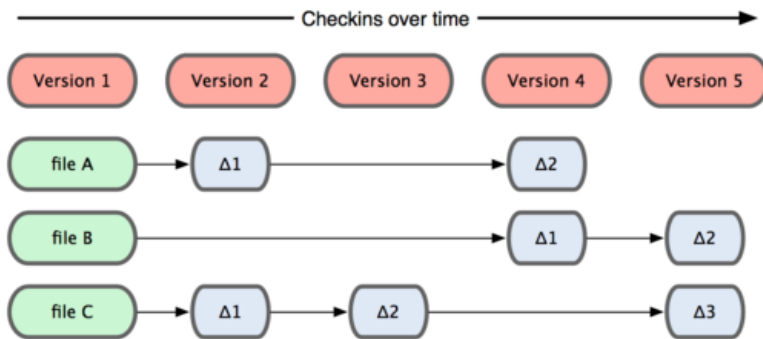


JORGE CHAN © 2012

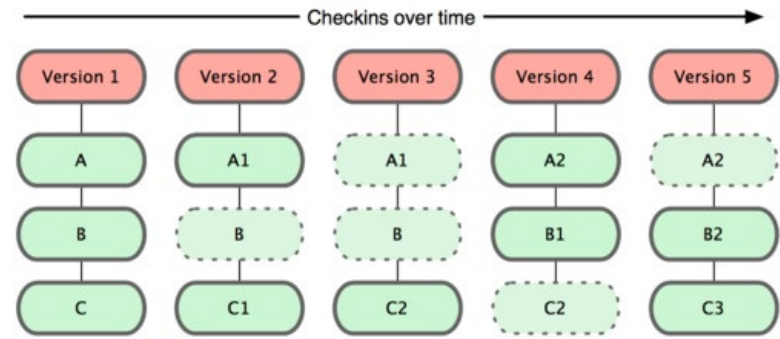




Subversion



Git



Collaboration Versioning Rolling Back Understanding

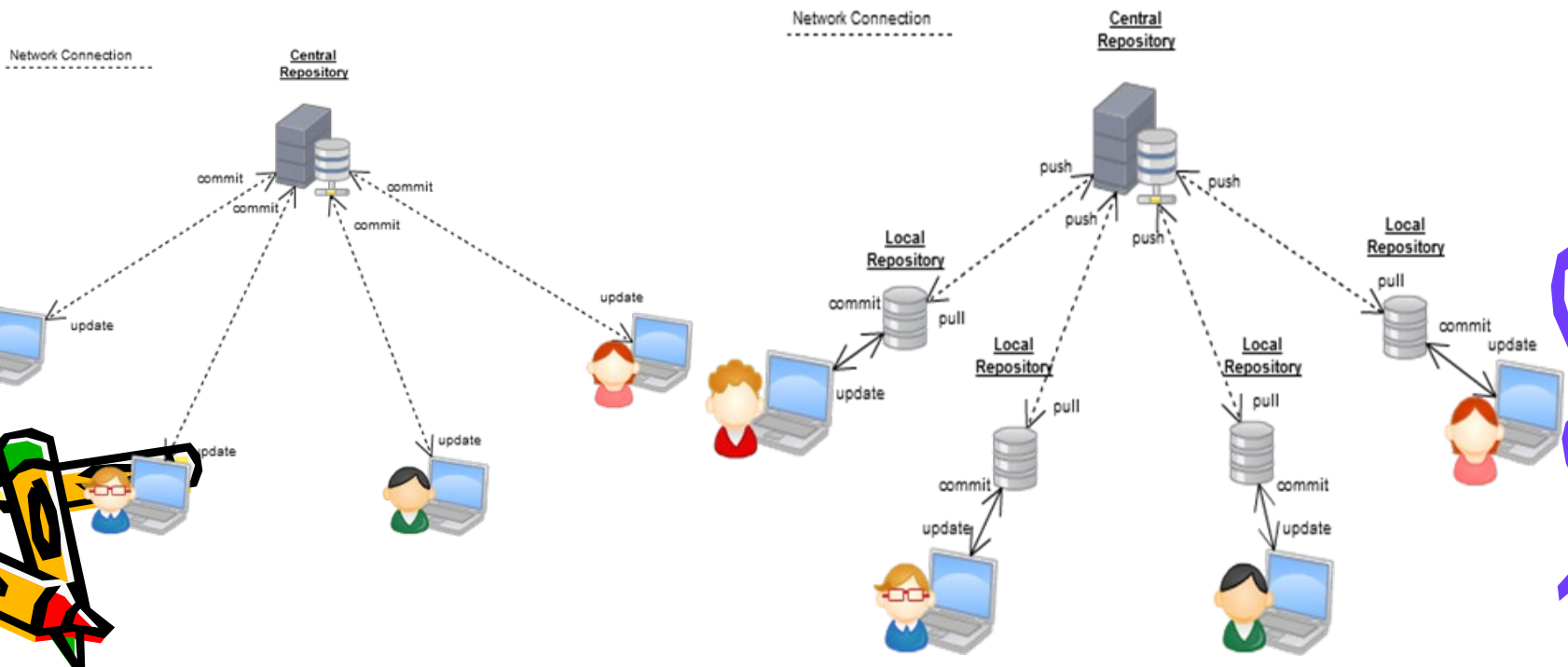
GIT



Version Control Systems



- Some well-known version control systems are **CVS**, **Subversion**, **Mercurial**, and **Git**
- Git has many advantages over earlier systems such as CVS and Subversion



Introduce yourself to Git

- Local:

```
git config user.name "name"
```

```
git config user.email "email-address "
```

Or edit `.git/config`

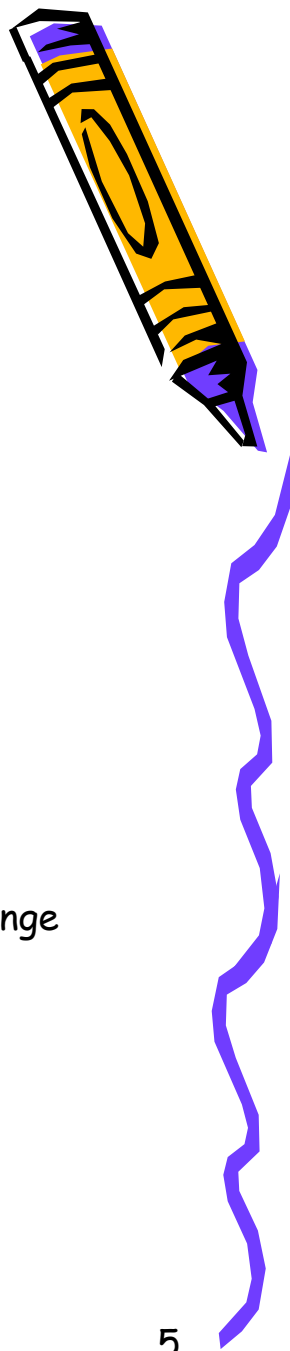
- Global:

--global after config


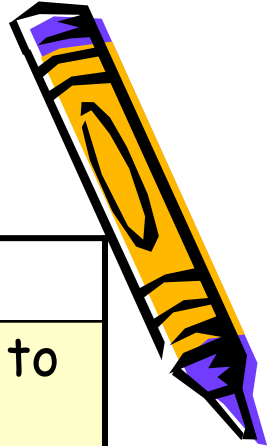
You only need to do this once

If you want to use a different name/email address for a particular project, you can change it for just that project

- `cd` to the project directory
- Use the above commands, but leave out the `--global`



Git Command



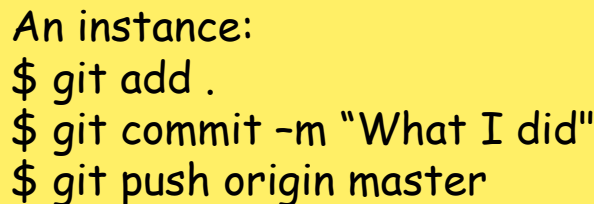
command	description
<code>git clone url [dir]</code>	copy a git repository so you can add to it
<code>git add files</code>	adds file contents to the staging area
<code>git commit</code>	records a snapshot of the staging area
<code>git status</code>	view the status of your files in the working directory and staging area
<code>git diff</code>	shows diff of what is staged and what is modified but unstaged
<code>git help [command]</code>	get help info about a particular command
<code>git pull</code>	fetch from a remote repo and try to merge into the current branch
<code>git push</code>	push your new branches and data to a remote repository
others: <code>init, reset, branch, checkout, merge, log, tag</code>	

Typical workflow

- **git pull *remote_repository***
 - Get changes from a remote repository and merge them into your own repository
- **git status**
 - See what Git thinks is going on
 - Use this frequently!
- Work on your files
- **git add --all (or just changes)**
 - **git hash-object -w xxxx**
 - **git update-index --add --cacheinfo 100644 SHA1 xxxx**
- **git commit -m "What I did" git commit -a -m "What I did"**
 - **git write-tree**
 - **git commit-tree**



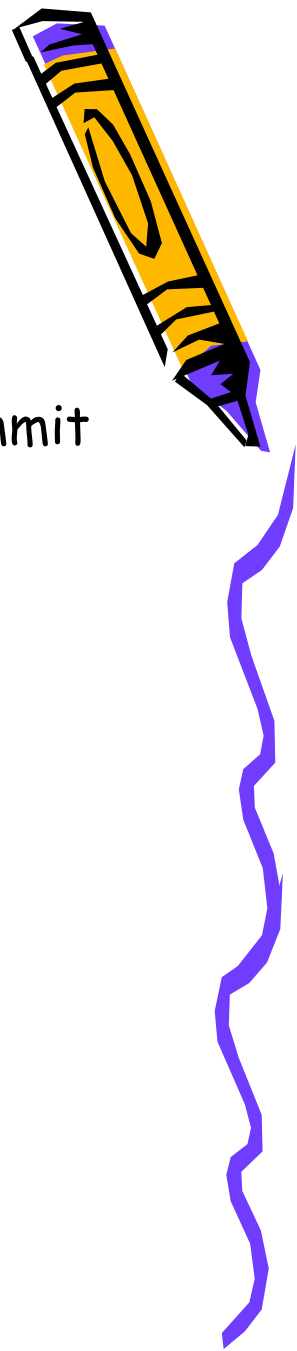
Mode:
100 regular file
644 Permission
755=rwxr-xr-x, 644=rw-r--r--



An instance:
\$ git add .
\$ git commit -m "What I did"
\$ git push origin master



git push
gitignore

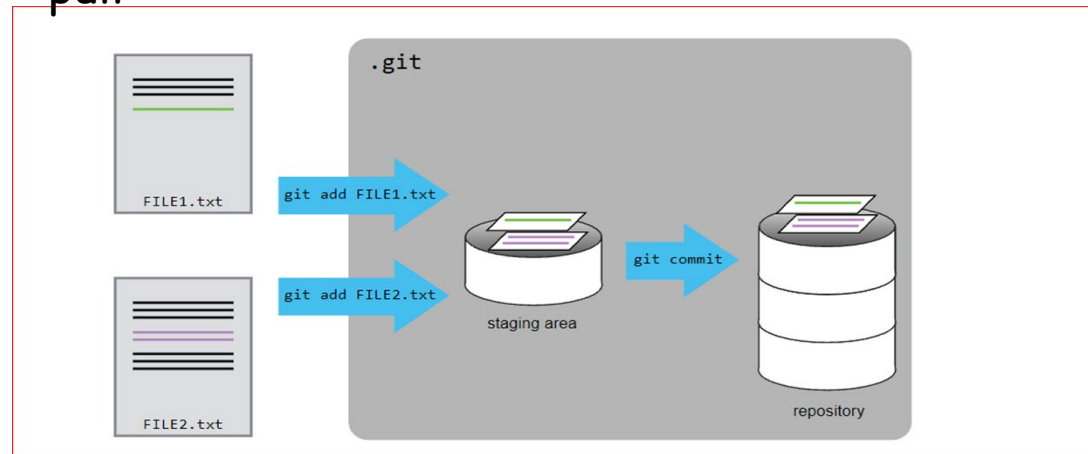
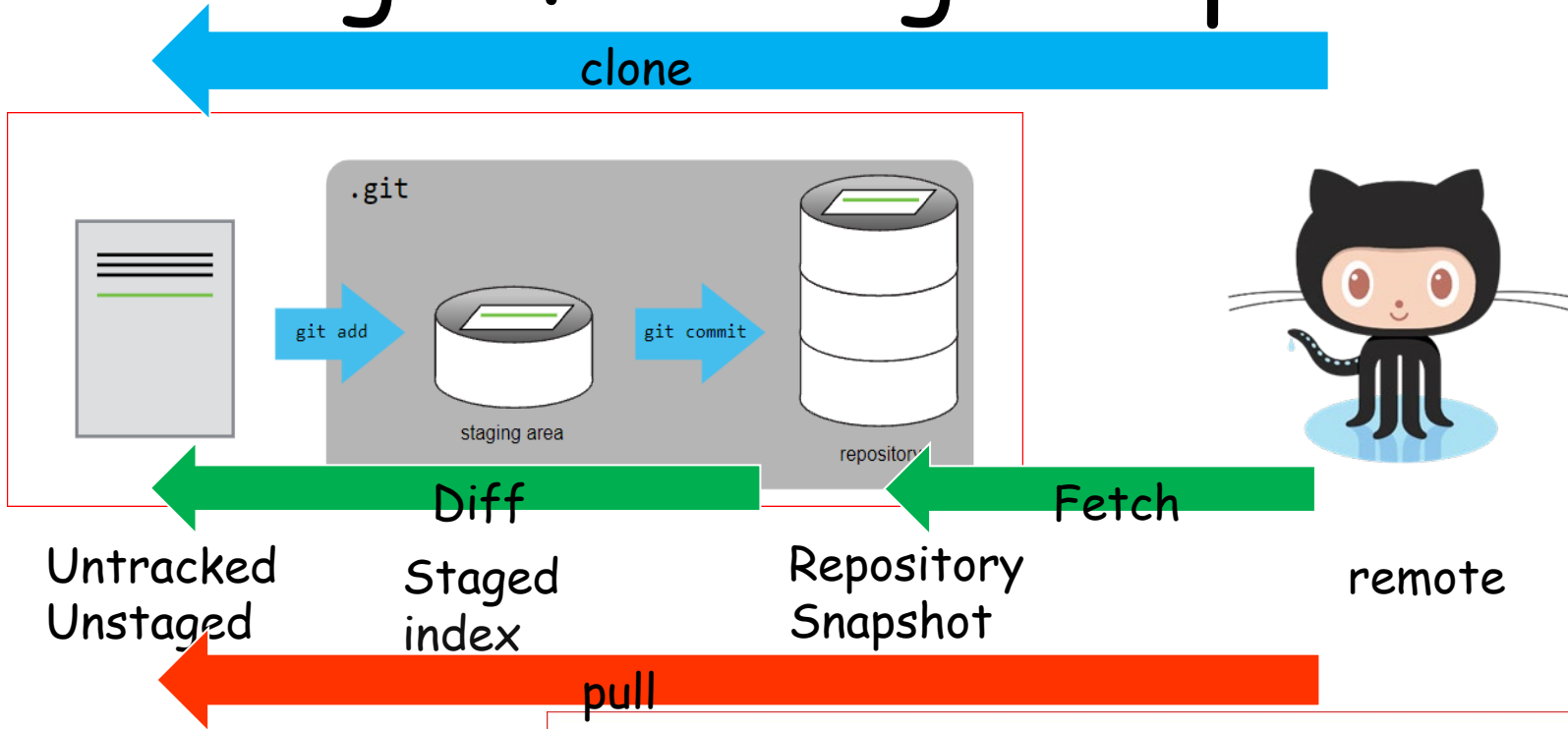


Choose an editor

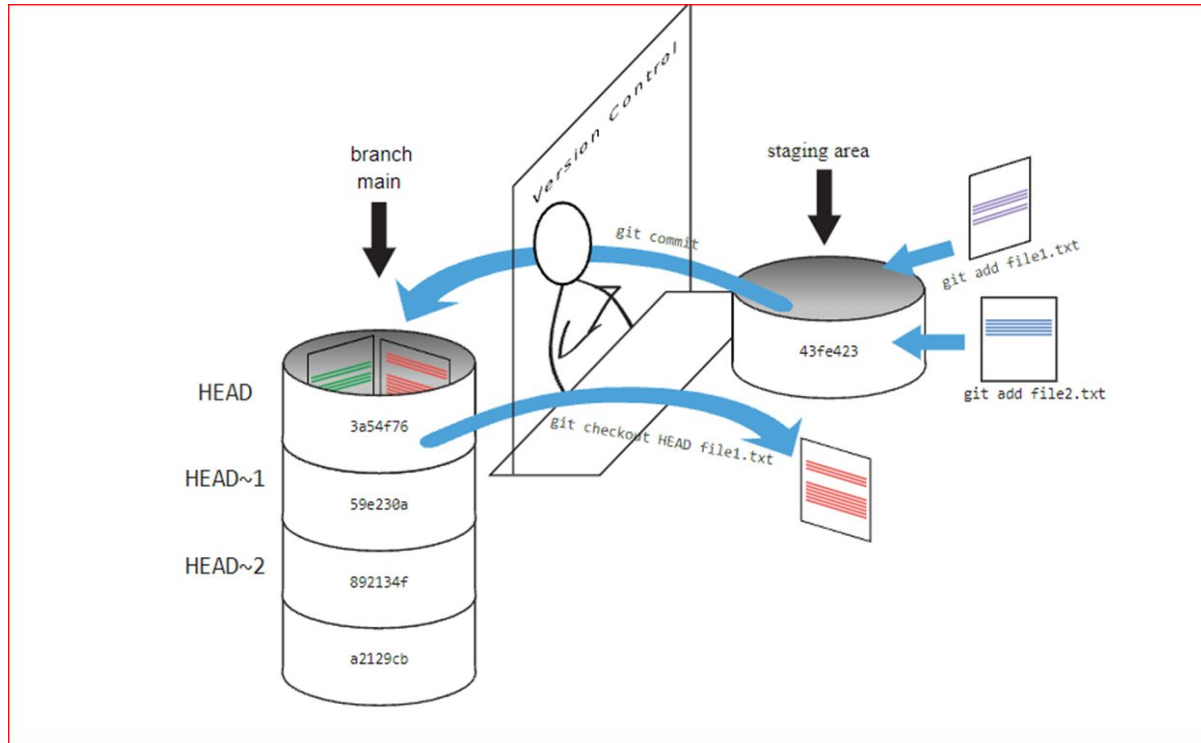
- When you "commit," git will require you to type in a commit message
- For longer commit messages, you will use an editor
- The default editor is probably **vim**
- To change the default editor:
 - **git config --global core.editor /usr/bin/vim**
- You may also want to turn on colors:
 - **git config --global color.ui auto**
- See your options:
 - **git config -l**



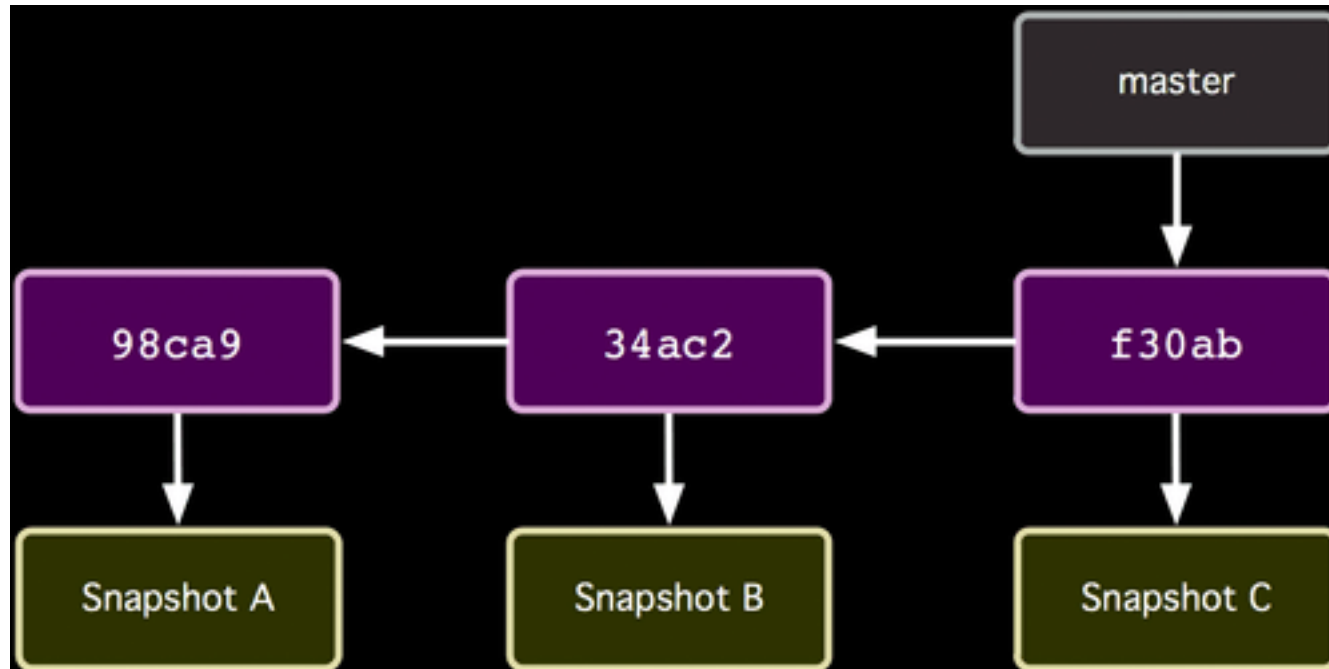
Adding a file to git repository

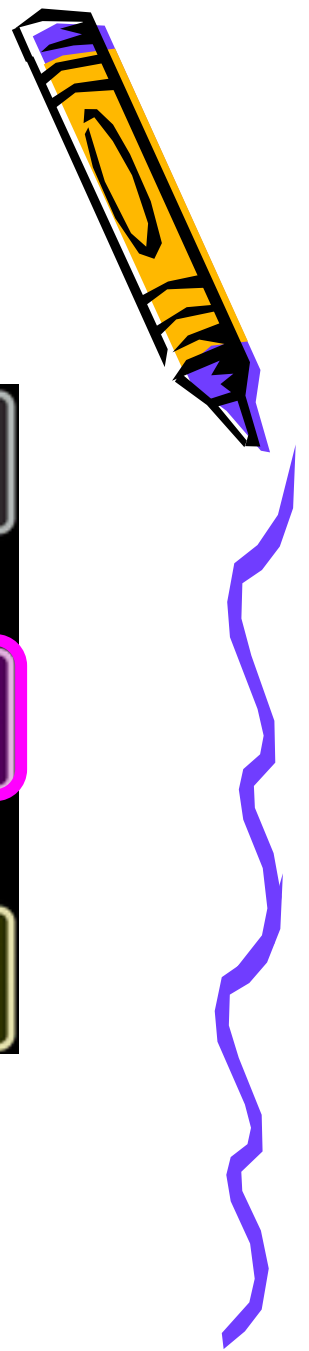


Adding a file to git repository



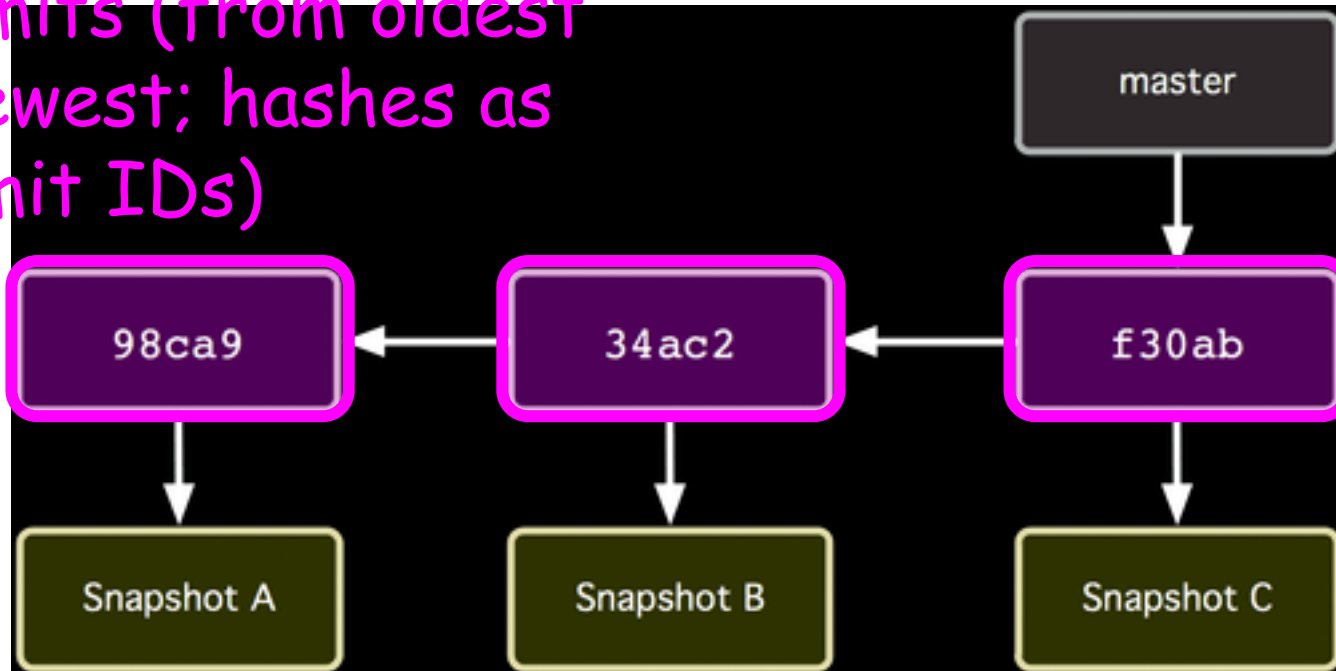
Repo Organization



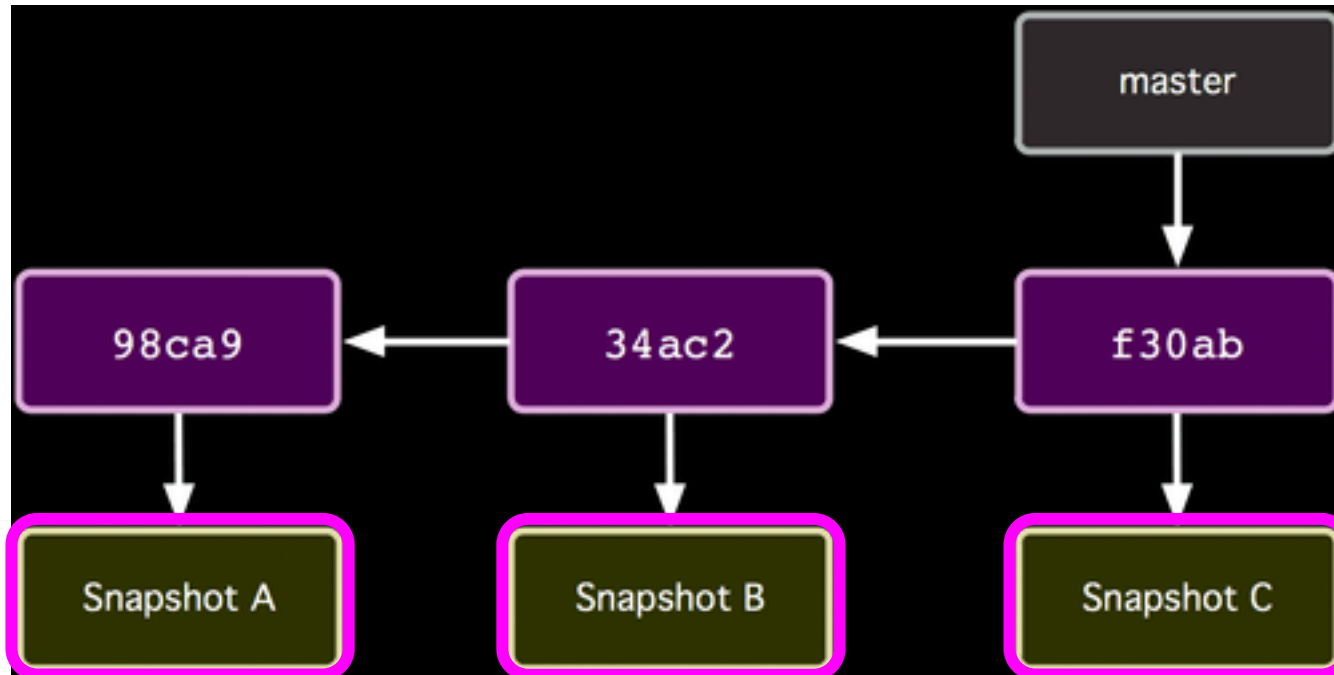


Repo Organization

Commits (from oldest to newest; hashes as commit IDs)



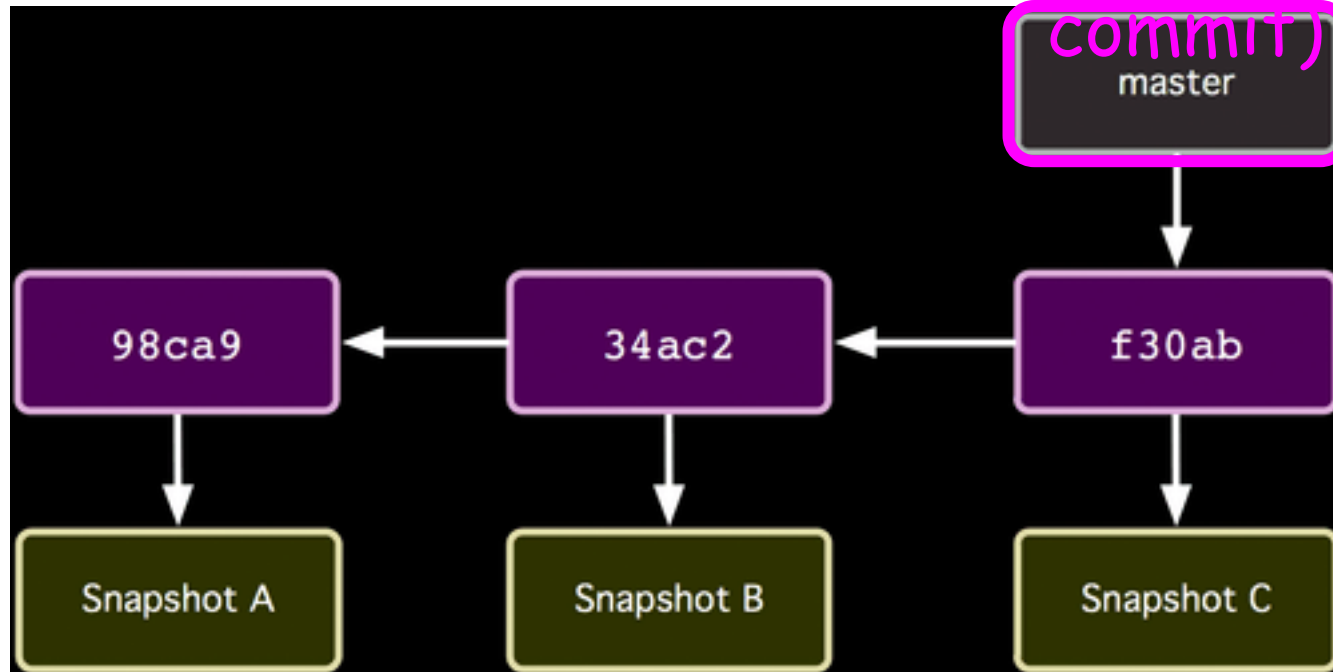
Repo Organization



Snapshot of all
files at each
commit

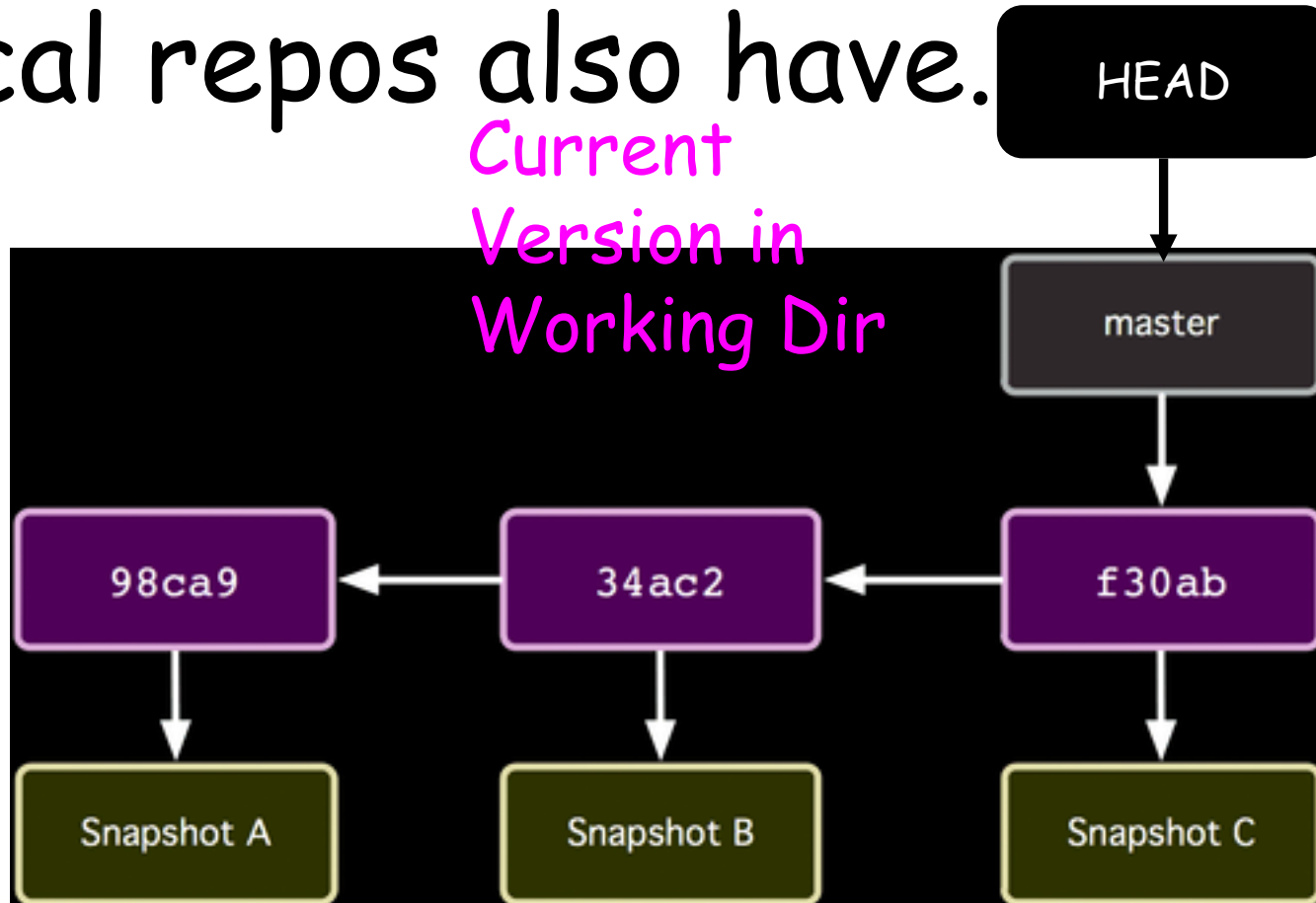
Repo Organization

Branch (last
commit)



Local repos also have.

Current
Version in
Working Dir



Git Tools



Windows Git Clients:

- Sourcetree
- GitHub for Windows
- Tortoise Git

Mac Git Clients:

- GitUp
- GitBox
- Git-Xdev





Helpful gitBash commands

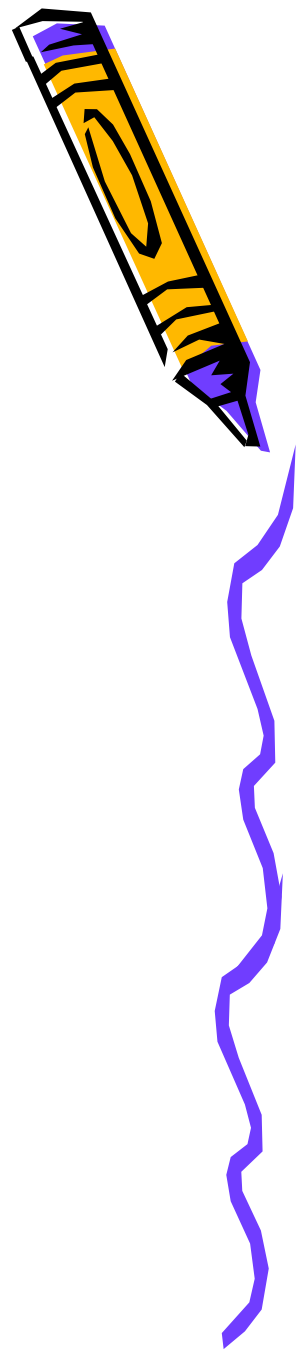
- Show staged differences: `git diff -- cached`
- Show status : `git status`
- Make branches: `git branch xxxx -d xxxx -D xxxx`
- Show branches: `git branch`
- See history: `git log`
- Checkout a branch: `git checkout branch git checkout -b xxxx`
- Fetch so you can look but maybe not take: `git fetch`
- Pull will fetch and merge with what you have: `git merge`

Also File, Commit,
branch



Git log commands

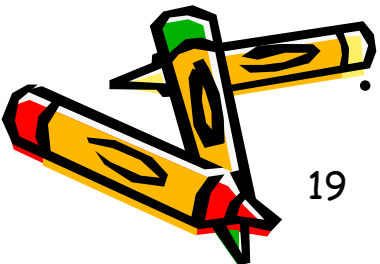
- `git log`
 - `--pretty=oneline --max-count=2 git log`
 - `--pretty=oneline --since='5 minutes ago' git log`
 - `--pretty=oneline --until='5 minutes ago' git log`
 - `--pretty=oneline --author=<your name> git log`
 - `--pretty=oneline -all`



Git log pretty

- `git log --pretty=format:"%h %ad | %s%d [%an]" --graph --date=short`

- `--pretty="..."` defines the output format.
- `%h` is the abbreviated hash of the commit
- `%d` commit decorations (e.g. branch heads or tags)
- `%ad` is the commit date
- `%s` is the comment
- `%an` is the name of the author
- `--graph` tells git to display the commit tree in the form of an ASCII graph layout
- `--date=short` keeps the date format short and nice



Good aliases

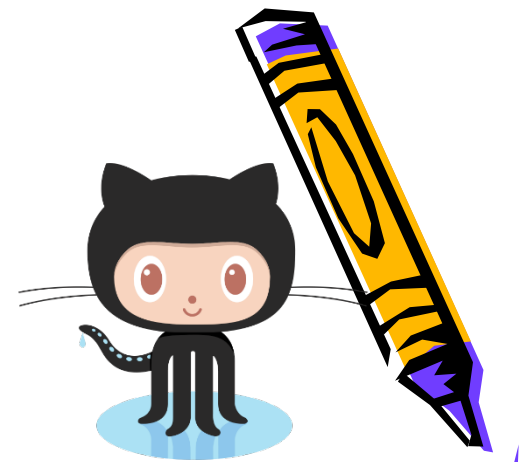
- `alias gs='git status '`
- `alias ga='git add '`
- `alias gb='git branch '`
- `alias gc='git commit'`
- `alias gd='git diff'`
- `alias go='git checkout '`
- `alias gk='gitk --all&'`
- `alias gx='gitx --all'`
- `alias got='git '`
- `alias get='git '`

Or

```
$ git config --global alias.co checkout  
$ git config --global alias.br branch  
$ git config --global alias.ci commit  
$ git config --global alias.st status
```

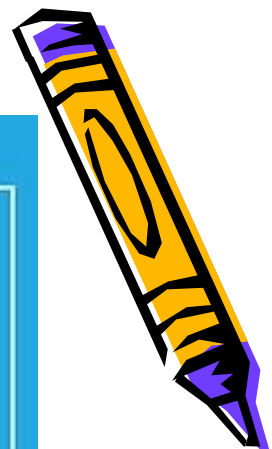
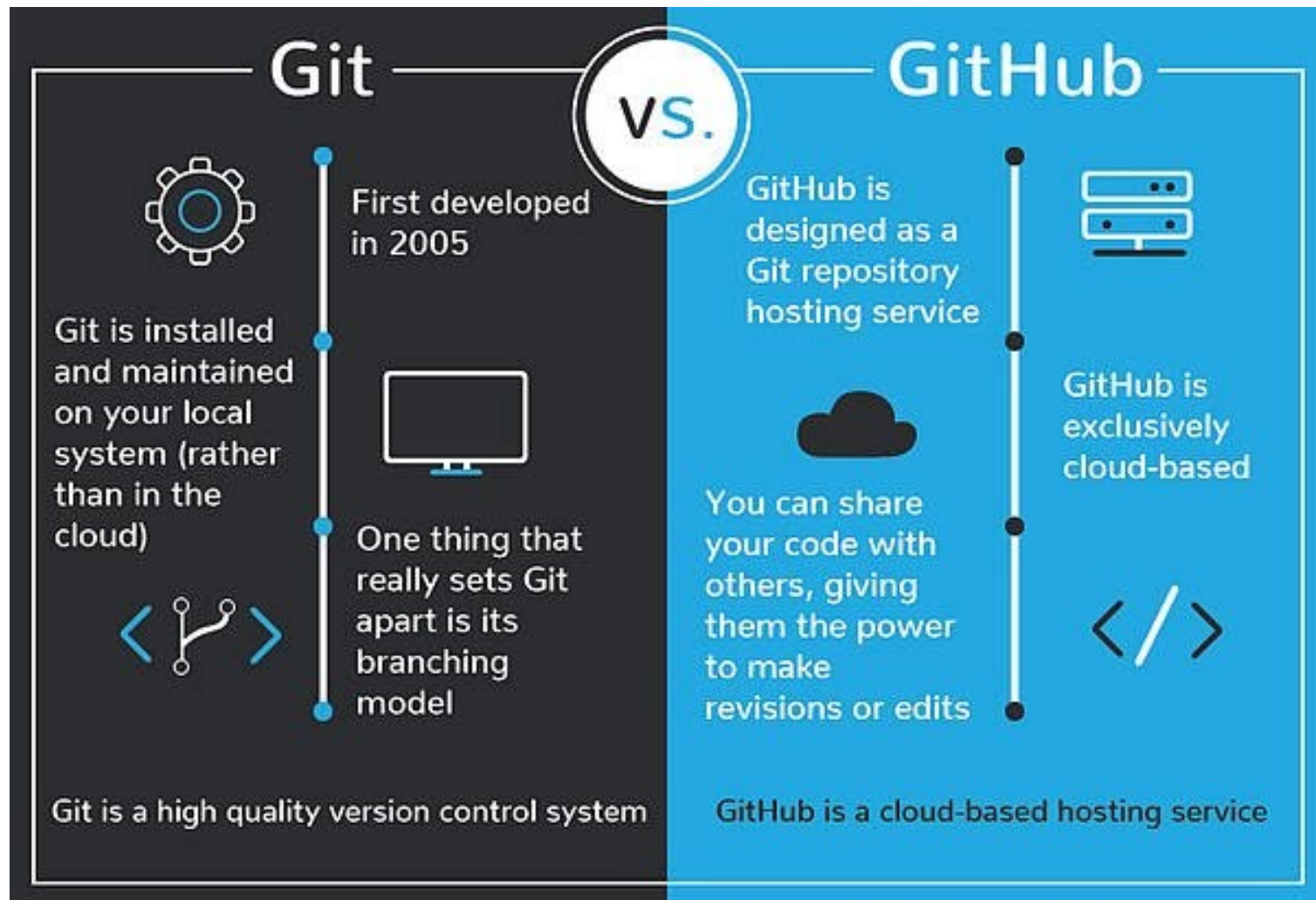


What is Github?

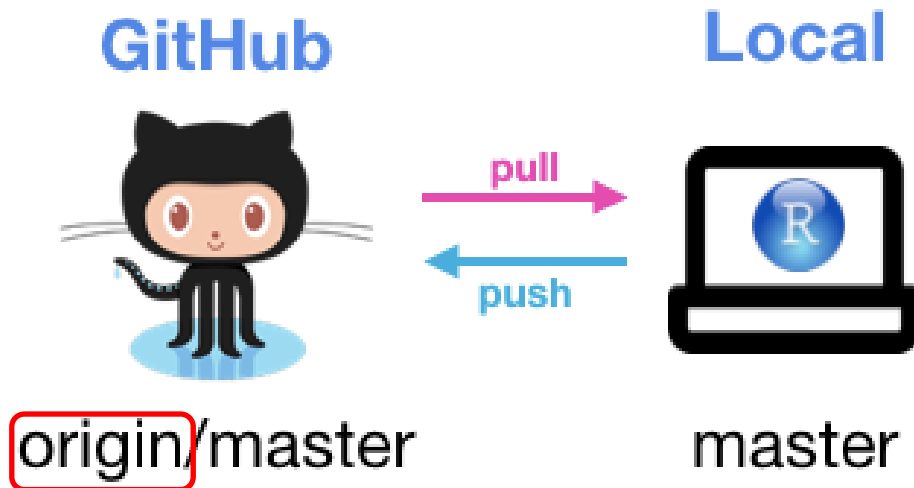


- GitHub was developed by Chris Wanstrath, P. J. Hyett, Tom Preston-Werner and Scott Chacon using Ruby on Rails, and started in February **2008**
- It is a subsidiary of **Microsoft**, which acquired the company in **2018** for \$7.5 billion
- GitHub.com is **a site for online storage of Git repositories.**





Thank You



More see:

<http://www.ruanyifeng.com/blog/2018/10/git-internals.html>

