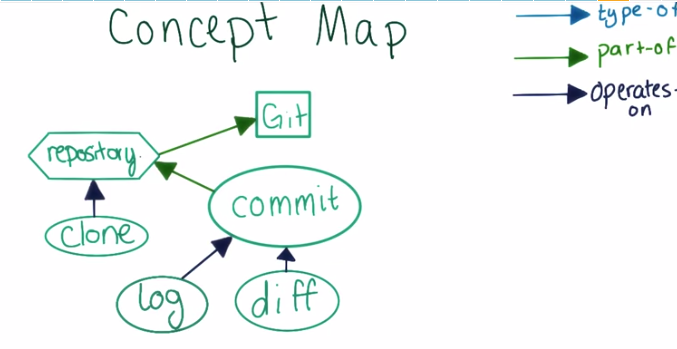
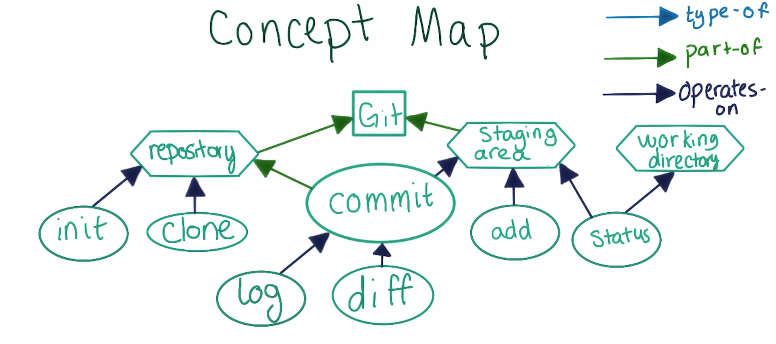
1. For angular to run, after clone a repo, we should do:
   1. sudo npm install
   2. cd into where bower.json located
   3. bower install
2. To set up git repo, in the project directory, do
   1. git init (this could be offline)
   2. git status(optional, see what hasn’t been added)
   3. git add FileName
      1. or git add . (add all)
3. To see what changes you have made,
   1. git status
   2. red: things that changed but not added
   3. green: things that changed and added
4. removing a file that has done git inti is recommended to remove through command
   1. git rm filename
   2. git status
   3. git commit –m “some commit”
5. to see the commit history
   1. git log
6. to compare the difference between two commit
   1. git log (see the commit history)
   2. copy the older and newer commit id something like 1231u209410jhf0wf
   3. **git diff** xsfnaksldfn(old) slkdfjklasdjflkas(new)
   4. then see the ‘+’ and ‘-’ for changes
   5. where ‘-’ stands for deletions and ‘+’ stands for addition
   6. if you don’t see the colors, type git config –global color.ui auto
   7. \*becaue git diff operates on commits
7. to see what origin the project has
   1. git remote show origin
8. After renaming your repo in github, the origin doesn’t change, so you should
   1. git remote set-url origin NEWurl
   2. or remove it first
   3. git remote rm origin
   4. git remote add origin NEWurl
9. to go back to the older commit, like undo, do this
   1. git checkout CommitID
   2. this not only allows us to go back do some changes but also allows us to take the risk of modifying a lot of code because we can go back.

General notes

1. **Concept Map, relationships**
   1. GoogleDoc, DropBox, Git are type of version controls
   2. (suggested map)



1. **When to commit?**
   1. We don’t want to commit too much, as this would make commit history hard to keep track of. We don’t want to commit too little, as this would make each commit large and hard to keep track as well.
   2. Solution: one commit per logical change. For example, a typo fixed, a feature fixed/create, a class/struct set up.
   3. Don’t worry, as you commit more, you know what should commit.
2. **When to use diff?**
   1. When working with people on the same project, we can use git diff to see the version.
   2. When making changes to a project, if we forget, we can use git diff to see the differences from the commits.
3. **More on relationships**
   1. ****git init – connect working directory to repository
4. **git diff for working directory, staging area, and repository**
   1. git diff compares all files in working directory and staging area. (things added but not committed)
   2. git diff --staged compares staging area with repository (the most recent commit)
   3. git diff commitID1 commitID2 compares the two commits