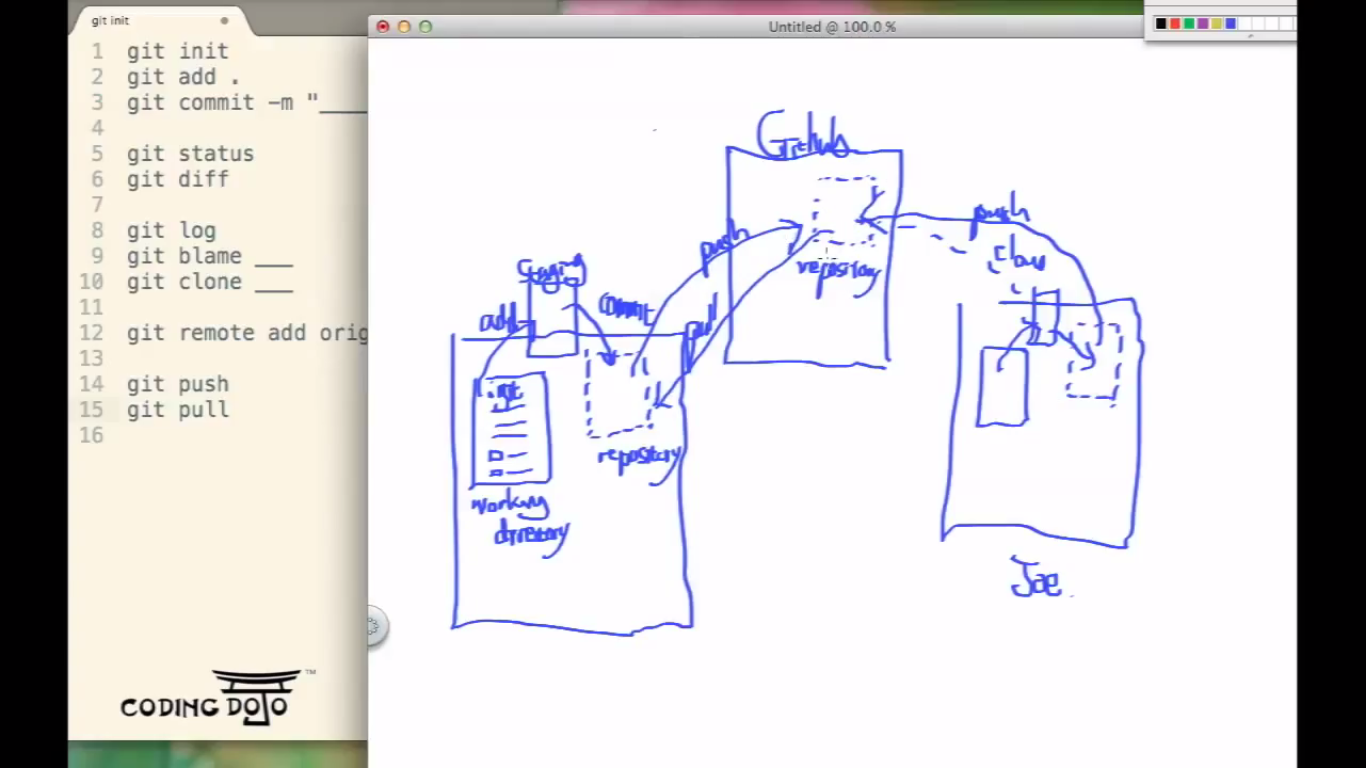
Git help log

Man git-log

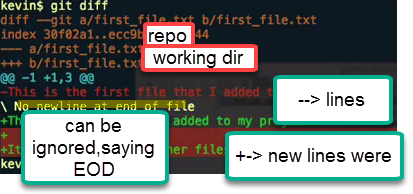
Learn Everything about Git and GitHub in 7 minutes



Git diff---(make diff. I have changed)

Compares what is in repo(haed is pointing to with that of in working dir.

-->repo, ++-> working dir



Git rm:uses rm-rf (shift+del)

Git mv:

Rename filee in working dir: Shows file deleted+ new untracked file

Git blame

**Learn Git in 20 Minutes**

.git->stores changesets,branches,all tracking info..

SVN puts .svn folders in all dir.Git puts 1 .git ,at top level for tracking

.gitignore-> \*.log( we can see msg ignored files while doing commit)

Git commit –a –m `msg`

Gip and send to others, so they can have access to repo…as long as they have git installed.

Git blame filename

Untrack-Stag

Change1->(red)-> add(green)->change2(red)

Git status shows both as untracked and added.If we commit without adding 2nd 🡪 will take only 1st time changes

**Branching:**

* Allows to work on copy of code, inline with main code without effecting main line directly.
* We create branches, while working on bugfix/new feature.
* We can work on our own branches,and everyone can ccontinue to work their work without effecting being effected by ur work.
* Later merge changes back to main line.(while merging we should be on main branch(source))

**Merge with conflits**

Original🡪change1

Git checkout master

Change1

Change2 -🡪commit

Git checkout branch1

Change1

Change3

Git merge branchname--🡪>>>Conflit

Git branch –D brname(Delete branch)

**STASH ( to make change persist b/w branch switching---without making commit)**

**Git stash pop ---check it**

It takes dirty stage of branch inc. stag modifications and changes,that can bereapplied at any time

I have made changes in a branch and didn’t commit.. And want to switch to other BR to make some bug fix/enhance …File/changes will be lost.

Git stash apply

**Origin**: alias for repo we have cloned

Origin is like a pointer ,which points to repo

We need to keep upto date by: **fetching and pulling**

Git fetch will go out to server and get any changes made since u last cloned/fetched.

Fetch will pull data into ur local repo.but will not merge.Should merge manullay

Git pull:Automatically fetch and merge changes from remote BR to local BR

Push : guit push **origin master**:

We are telling git to push our changes to remotye repo**(alias ORIGIN) and commit to master** BR

**Git remote add**

**We can add many repo. And pull/push changes**

**Basic Git Administration -1**

**1.)Create git hub a/c.Create a repo compu**

**2.)I have files of snapshot view locally**

**Push all files to github compu**

**Authentication(ssh/https)**

**Ssh-keygen(generate ssh key)**

**GIT**

**Version:**

**Review history log of changes made**

**Keep track of versions and changes**

**Compare**

**Move back and forth at point in change process**

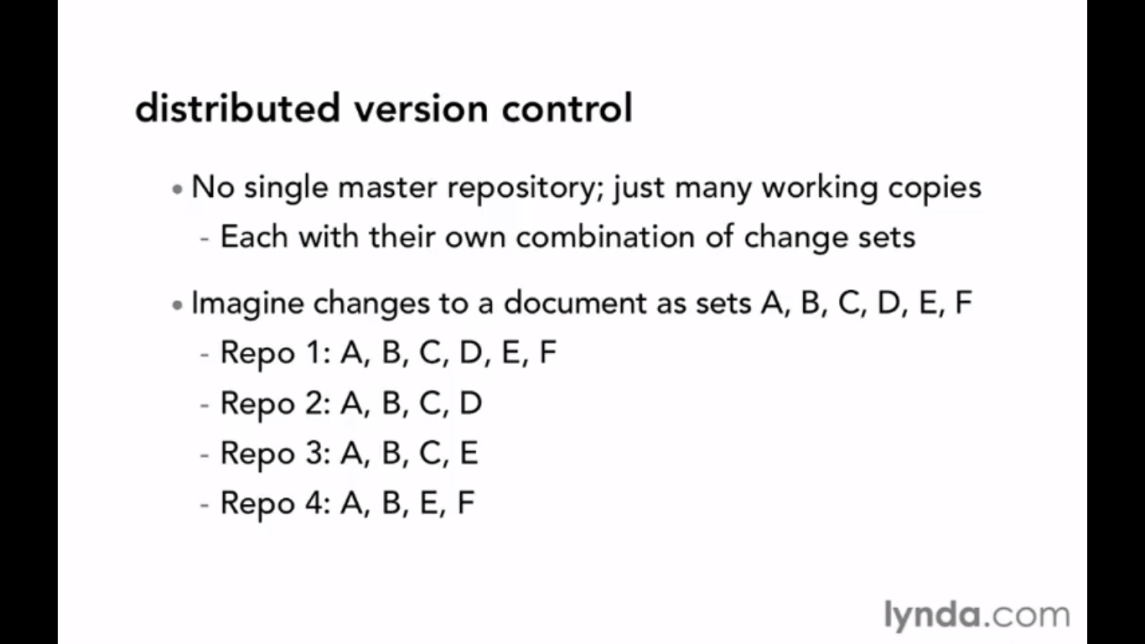
**Non-source code:**

**Work..track change**

**Photoshop history**

**Ctrl+Z**

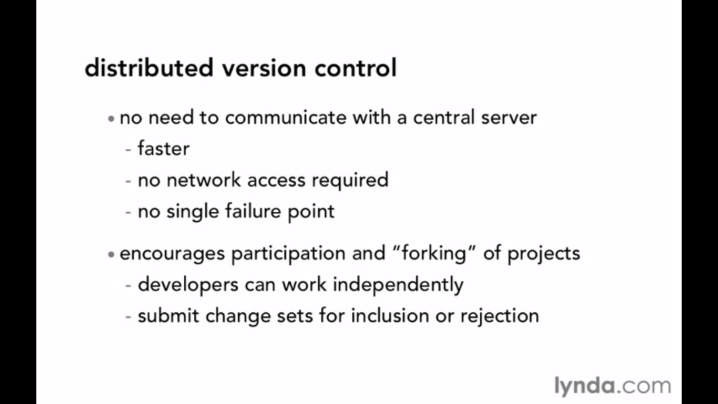
****

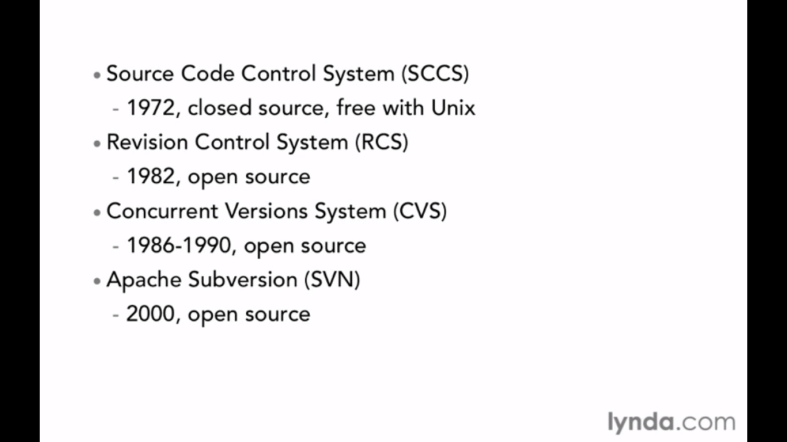


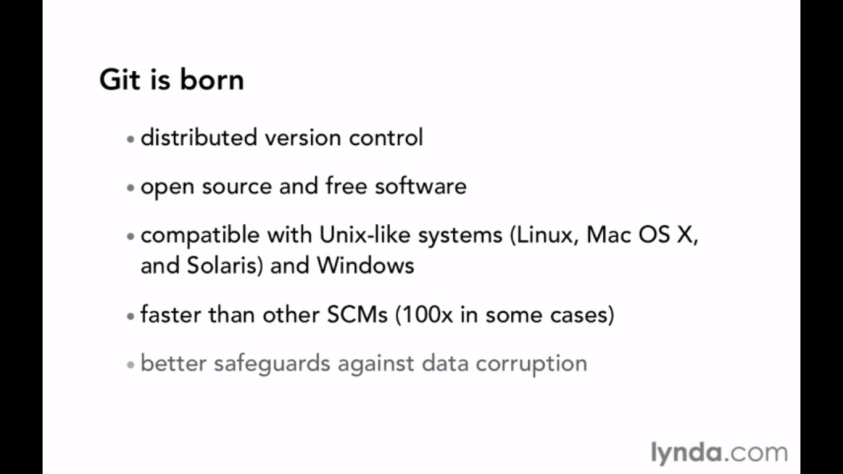
**We can have 3/4 masters ones have diff.versions and features in them.**

**No need to sync with each other.**

**We can add changeset G in repo-3 and share with repo4 without going to central server.**







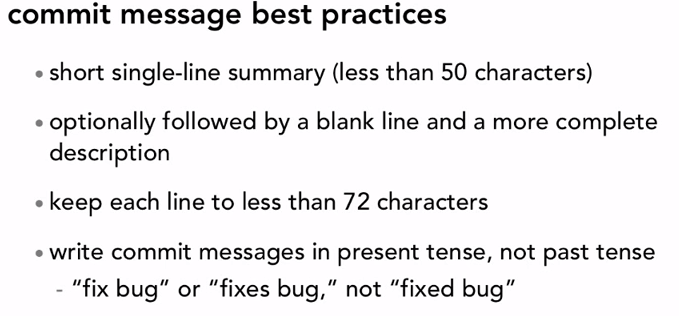
Git-completion bash

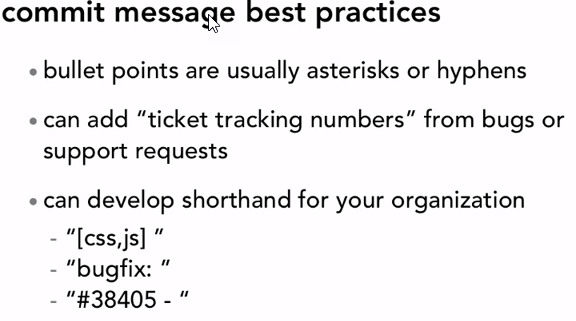
Git has bash completion Auto for linux..For windows, download git-completion.bash and Add to bash profile

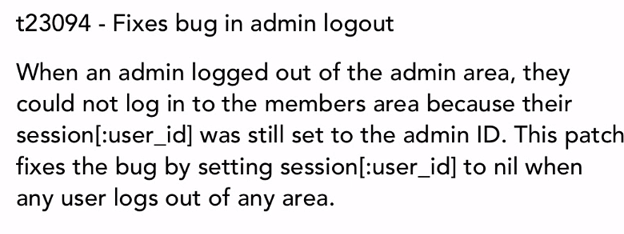


Git help command (or) man git-log

f-> forward,b->backward,space->1 pg down







Git log –n 5 (recent 5 changes)

Git log –author=”kelvin”

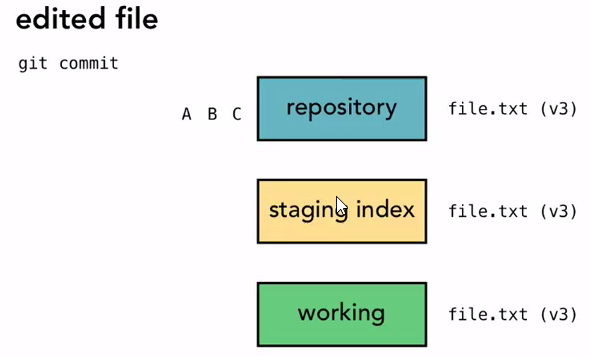
Git log –until=2012-06-14

Git log –grep=”COMMIT message”-🡪made changes to java script,can use grep to search

|  |  |
| --- | --- |
|  |  |

We can make 10 diff. changes in our working copy.But want only commit 5 files.

We can commit only 5 files using 1 changeset

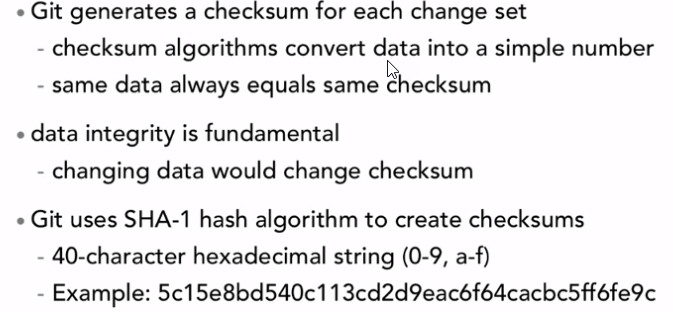


A,B,C -> changesetss

They doesn’t refer to change to single file at all.

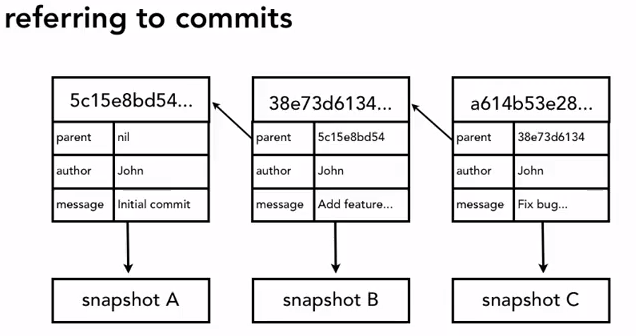
A->changes to 5 files,B->changes to 2 files,C-> new 3 files

A,B,C are snapshots that we made



**SHA/CommintID are** Unique to changes we commit

**PACKAGE TOGETHER AS A SNAPSHOT ----git commit-m (file1,2,3,4,5,…)**



How series of commits are linked together :

To know sequence of comiits-> each of metadata points to other metadata snapsho**t**

**HEAD**

**Points to last commit**

**Git status:** diff. b/w working dir,staging and repo

Working dir.clean->Nothing in stading dir.

work dir exactly matches with repo.

Head ptr in main BR. points to same as repo.

**Unstage file** : Git reset HEAD file**(((UNDO CHANGES TO STAGING INDEX))**

1. IF we unstage change will be in working dir.
2. When we use???::If we have 5/6 file changes made And assemble them to COMMIT in staging area.But there is extra file we added.So we ne ed to unstage that 1 file and leave other file in stage area .NOW COMMIT
3. Git reset HEAD file:what it means?

Go and look at head ptr(head ptr points to last commit) of master br. And reset urself to be same as what it has.

**Git checkout file:**  to discard changes made to working dir.---(((UNDO CHANGES TO WORKING DIR\_))

***Want my file/dir to look like that of repo.***

Checkout is used for both files/dir and BR

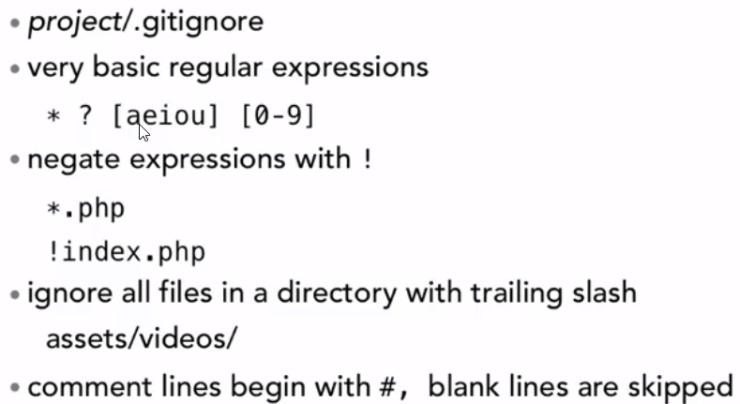
What happens if same name of files/dir AND BR?

Use: git checkout - - index.html (for files/dir)

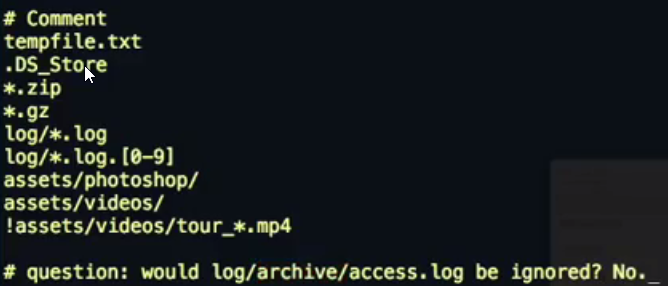
**UNDO CHANGES to repo/UNDO CHANGES TO COMMITS**

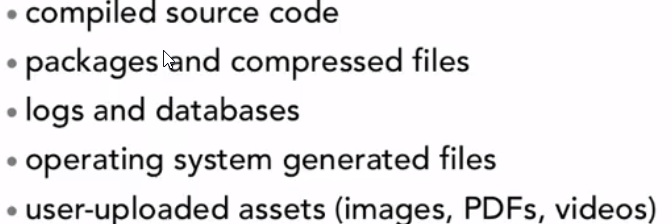
**We only have ability to amend the last commit(head id pointing to)**

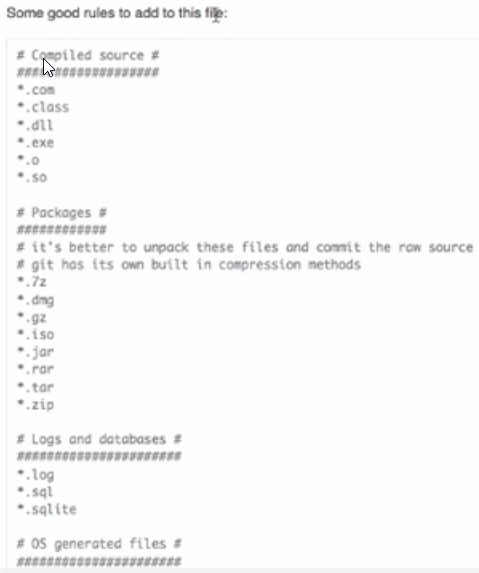
**.gitignore**



[**https://github.com/github/gitignore**](https://github.com/github/gitignore)







**Removing untracked files:**  
git clean –n(list untracked files)

git clean –f(**delete untracked files and not from staging area…Does rm -rf)**

**Ignoring tracked files(which are in repo. Or have versions/added to src ctrl)**

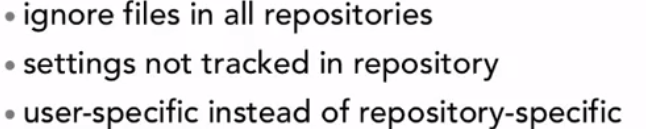
Git keeps tracks of version files,even we add in .gitignore

Git rm file(remove file from repo)

**Git rm - - cached file**(remove file from tracking/staging area..not from repo)

**Globally ignore files:**

Diff. project members want to ignore specific files. If we commit .gigignore there will be different for users. Or will be in red .





**BRANCHES**

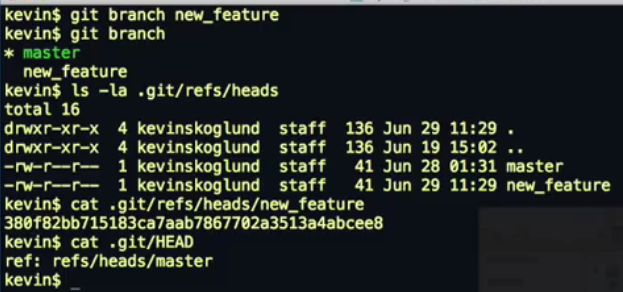
**Git branch:shows list of Branches**

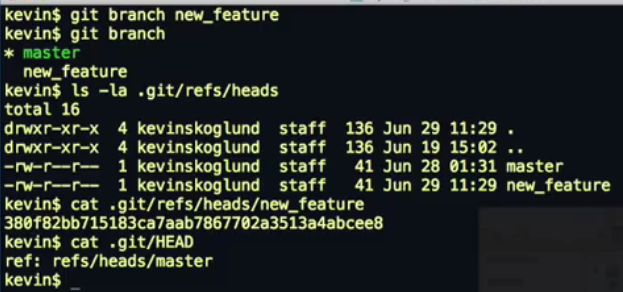
**Ref. of head ptr is stored in cat .git/HEAD----(master)---Ref. to one of branches**

**Ls -la .git/refs/head ---(master)**

**.git/refs/heads/master ---SHA …Contains SHA that points to commit**







**Git status –oneline**

Git show HEAD (or SHA)

Git checkout -b BR (create and checkout branch at same time)

(MAKE SURE FROM WHICH BR ,U CREATED NEW BR)

Git log –graph --online –decorate –all

Git checkout->Tells git get latest versions of that BR and make working dir exactly like it(BR)

Working dir mostly should be clean to switch

Modified file in newBR

Git checkout master…(error ur local changes ould be overwritten by commit)

**Options:**

1. git checkout –file filename (undo change in working dir)
   1. Git stash
   2. Git commit

**CompareBranches**

**Givesdiff.b/wtipofmastra`nd newBR …NowicanseewhatdiffB/W2versions.**

**CompareBRwithrangeoperator (..)**

