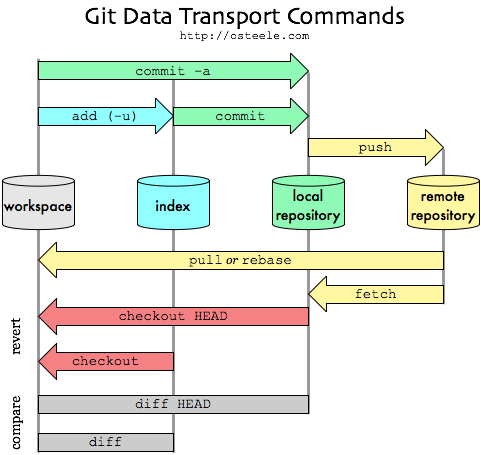
<https://happygitwithr.com/>

<https://stackoverflow.com/questions/3689838/whats-the-difference-between-head-working-tree-and-index-in-git>

GIT Overview

**GIT Workflow Diagram (Code in 4 different places)**



***remote repository****:*Github repository or a remote server hosted in your company. Only when you pull a code initially and pushing the changes when you are done with it.

**local repository:**when you clone a remote Git repo / or create a new repo, the code base is created here in the local repository. All commits you do will come here first

***index: Staging Area —***intermediate place between your working copy of code and your local repository. You use it to stage which files you want to track & commit.

***workspace:***your working directory where you create/edit/delete your files. This code resides in your local machine.

**GIT Work Flow Example**

**Assumption:** I[nstalled Git](http://help.github.com/win-set-up-git/) on your machine and you have the PATH environment variable updated with Git’s location.

**1. Git repository creation**

Open a terminal or command prompt and CD to your project directory. Then give the below command.

c:> cd vraa/projects/helloworld  
C:/vraa/projects/helloworld>git init  
Initialized empty Git repository in C:/vraa/projects/helloworld/.git/

**Summary:** create new local Git repository to track my hello world project.

**2. Git configuration — user name and email (one time)**Next thing is to setup a user name and email to be used in all my git commits. This is a one time activity per Git installation.

C:/vraa/projects/helloworld> git config --global user.name "yourname"  
C:/vraa/projects/helloworld> git config --global user.email "your@mail.com"

**3. Add a file to Git index**Status will tell you the current status of the repository and branch details. In Git you don’t need to check out anything before you start working on it. Just modify the file directly and then commit the changes later.

C:/vraa/projects/helloworld> edit helloworld.txt  
C:/vraa/projects/helloworld> git status  
# On branch master  
#  
# Initial commit  
#  
# Untracked files:  
# (use "git add ..." to include in what will be committed)  
#  
# helloworld.txt  
nothing added to commit but untracked files present (use "git add" to track)C:/vraa/projects/helloworld> git add helloworld.txt  
C:/vraa/projects/helloworld> git status  
# On branch master  
#  
# Initial commit  
#  
# Changes to be committed:  
# (use "git rm --cached ..." to unstage)  
#  
# new file: helloworld.txt

When you say git add [filename] you are asking the git to keep the file in *git index* and track the changes. Simply, you are staging this file for commit. Next time when you do a commit all the files which are in index will be committed.

To discard the changes from the working area

Vidoes : <https://www.youtube.com/watch?v=VUuGa1QFmcA>

Git checkout -- <filename> meaning all the changes done on the file will be gone

Git checkout . : All the changes done working directory will be gone.

Unstaging :

If you make changes to file then it is added to statging area using git add . To Undo staging back to working area please use

Git reset head <file> - to specific file

Git reset head . – all staged file into non staging area.

**Remove files from Commit :**

Git reset <Mode> Head~1

Where mode soft – removed files from commit and put into staging area

Mixed –removed files from commit and put into working area ( this is default option) if you not specify it is Mixed.

Hard- removed files from commit and does NOT put into any place like working area or staging area.

~1- Represents last commit

~3-Represents last three commits

This command does not used to remove particular commit .it go by sequence only from last to first.

**Remove commit from Repo**

Once the commit is pushed to repo using the command Git push origin master

GIT revert <commitname>

Where commitname – which commit want to removes however it maintains history . The file added from the <commitname> will be removed but commitname kept in history and new commit added as apart of removal

**To create branch** : git branch <branchname>

To check branch : git checkout <branchname>

To know the current branch : git branch

**To view all the commits** : got log –oneline or git log

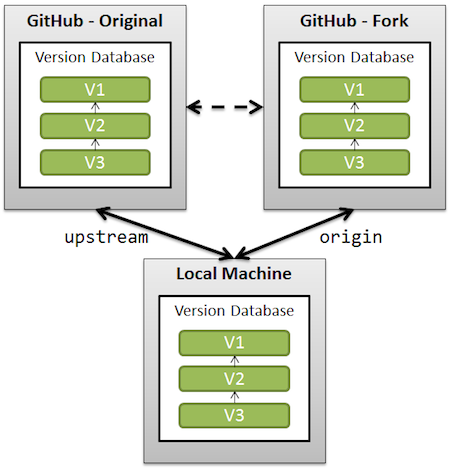
To use git Diffmerge tool : <https://www.youtube.com/watch?v=Tu-CuBqulsA&list=PLH1ul2iNXl7vkfIFF2BxLA5xpkbvWtFWf&index=4>

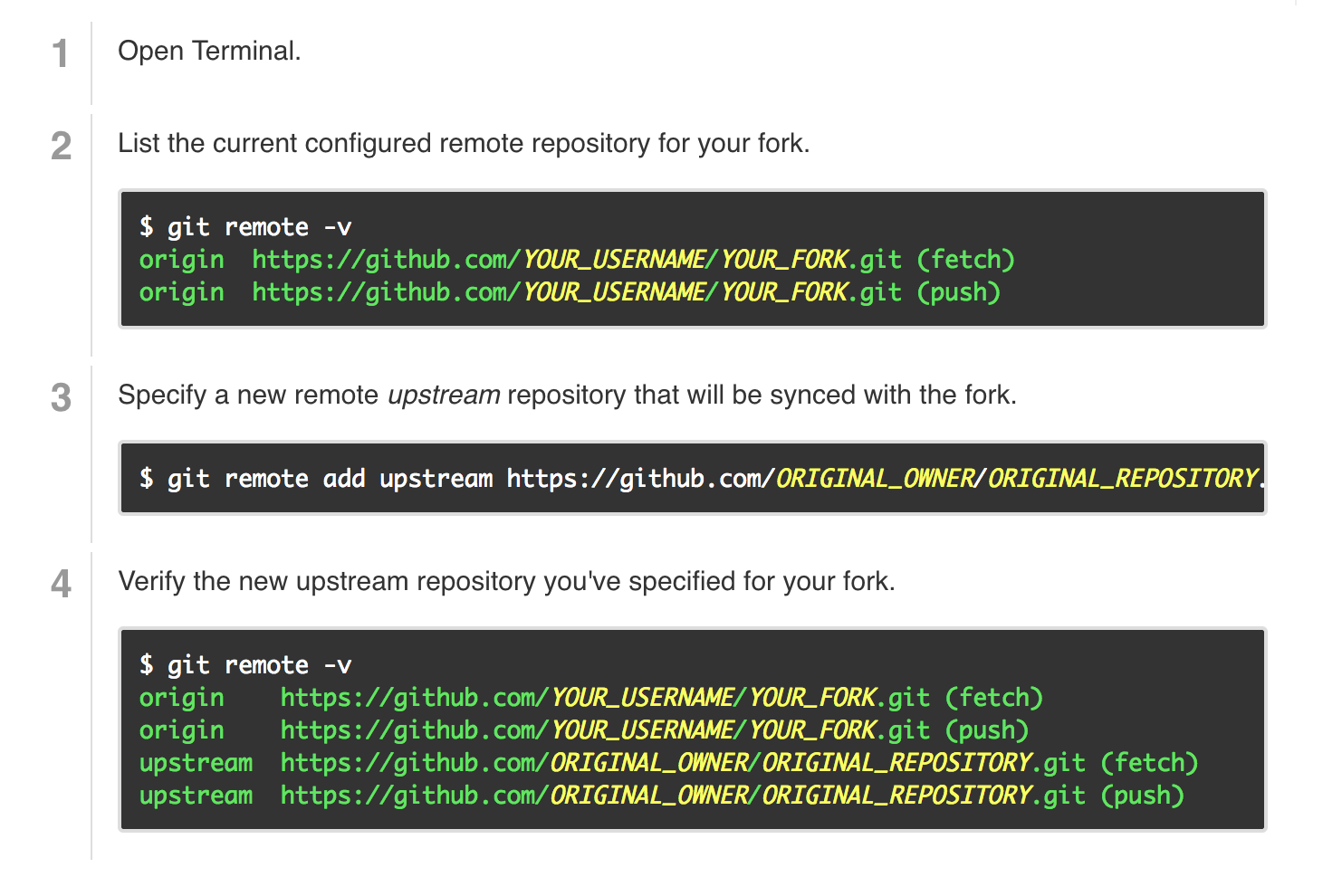
Difference between GIT rebase and GIT merge

Both commands are used to merge the branch but the main difference is that Merge preserves commit history and whereas Rebase does not maintain commit history .

<https://www.quora.com/What-is-the-difference-between-rebase-and-merge-in-Git>

git remote





Git stash: <https://www.youtube.com/watch?v=KLEDKgMmbBI>

Temportaily stores all the modifies files .

To do git stash

Git statsh “ stash Message”

To view git stash list

To get the stash changes into working tree

Git Stash pop – last stashed into working tree

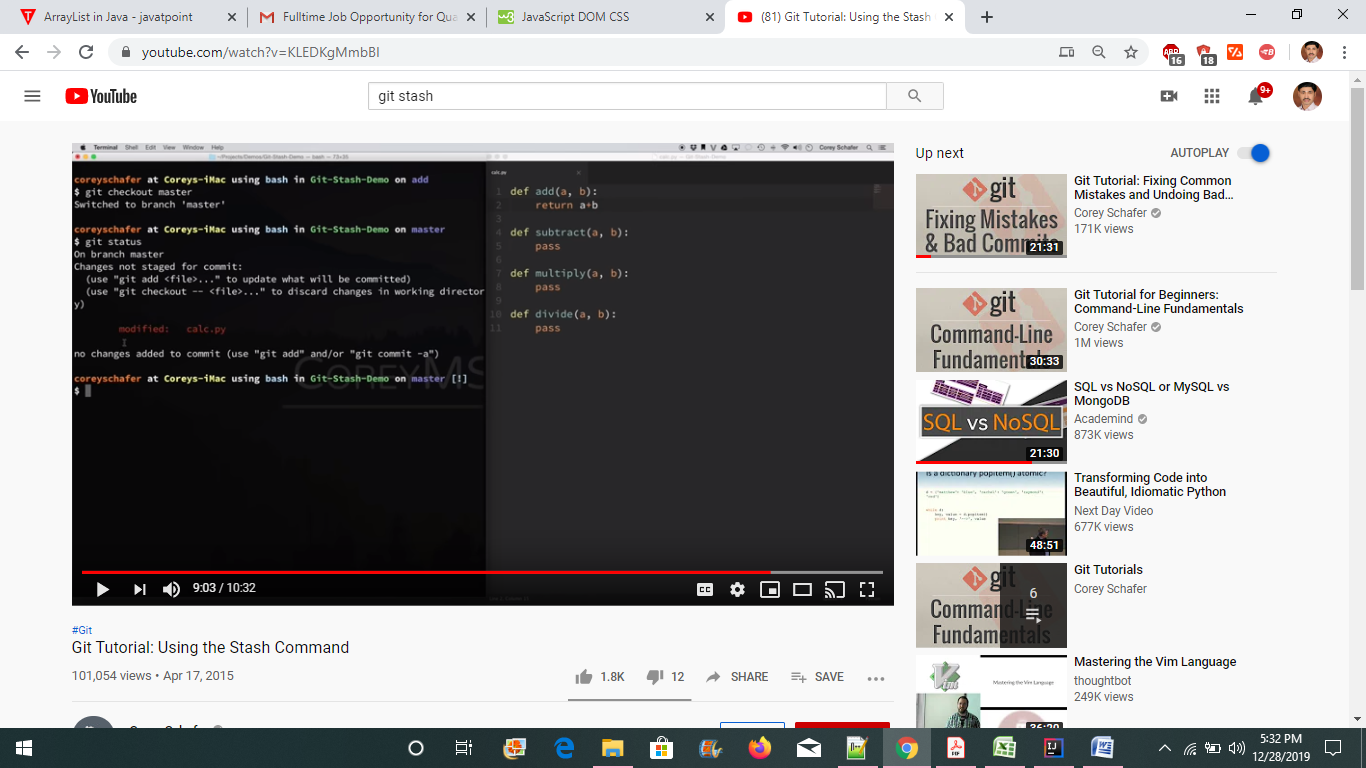
Git drop <stashed> - remove the changes from stash and not moved into working tree

Git stash apply –last stashed changes apply into working treebut stash is maintained in the Git stash list

Git stash clear – will removed all saved stashes .

Git status – shows the filename which has changes42389587

Git diff –shows the changes on the filename instead only Filename



Git diff

