**GIT and GITHUB**

**GIT:**

* Version control software
* Open source
* git-scm.com

**Goals of GIT**

Simple

speed

Non linear

distributed

Large

**GitHUB:**

Collaboration software, hosts GIT repositories

Repository to store the files/projects/code

Entire linux code runs on GITHUB

**GITHUB Workflow:**

**Branching:**

Create a branch on master replica, as the changes should not be done on master

Master should always has to be deployable

Branch is exact copy of master

**Commits**

sort of making changes to branches.

**Pull request**

Open pull request on github

Comparing the change with master and show other people about your changes

**Collaboration:**

People may give suggestions or go ahead

**Merge**

Commits the branch

**Cloning Repository:**

Cloning is for making local copy from remote copy in GITHUB and working on a project

GITHUB is remote repository

Entire project can be copied to locally and work on it without internet access

Multiple people can work on GITHUB

When we push our changes to github, other users can pull the request and see them

GIT commands:

git clone <https://github.com/username/repository>

git branch feature-branch

git checkout featurebranch

git status

git add index.html

git commit

git push

git fetch

git merge

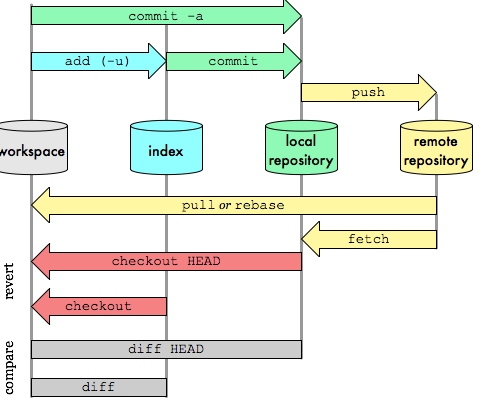
git pull

git diff

git stash

git rebase

**GIT Work flow**



With every commit we make there will be Commit id

When the commit was made date and time (parent commit)

Before the commit what was the previous commit etc

It is a 40 character SHA1 hash includes Bobs (reduced version of the code , files) and metadata

Git commit has 2 steps

2 step commit

Git add 🡪 which files you want to add

Staging files will be added to staging area

Git commit 🡪 all files will be included in github with 40 character sha1 hash

Git merge:

Remote- git hub

Locally – git merge

Merge conflicts:

When there is a change in same line in 2 branches

Try git status

And open the file in editor

Keep the change you want to keep

Rebase:

Git rebase to change the history

Below are commits

A B C D

E F

The changes look like A E B F C D

But if you want the history to be like

A B C D E F

Use rebase

How to undo last commit:

git revert

creating a new commit with opposite changes

other commands:

git reset

git commit-amend

git cherry-pick

**Command line:**

To clone the project

Create user id and password on the git hub site

Then create a new repository on github

Then clone that repository using

git clone [https://github.vp.com/user1-git/EXCITE](https://github.ibm.com/santosh-chennuri/EXCITE)

git config --global user.email [santoshdevops1@gmail.com](mailto:santoshdevops1@gmail.com)

git config --global user.name santosh

from a linux machine:

1. add the ssh key of the linux machine to git hub

Go to [https://github.vp.com/settings/ssh](https://github.ibm.com/settings/ssh)

Click on Ssh keys

Add a new key

Copy key from machine and paste it on github

[root@reviewb .ssh]# ***git clone ssh://github.vp.com/santosh-chennuri/EXCITE***

Cloning into 'EXCITE'...

remote: Counting objects: 15, done.

remote: Total 15 (delta 0), reused 0 (delta 0), pack-reused 15

Receiving objects: 100% (15/15), done.

[root@reviewb .ssh]# ls -lrt

total 16

-rw-------. 1 root root 405 Nov 29 12:08 authorized\_keys

-rw-r--r--. 1 root root 405 Dec 29 09:53 id\_rsa.pub

-rw-------. 1 root root 1675 Dec 29 09:53 id\_rsa

-rw-r--r--. 1 root root 3428 Jan 1 08:28 known\_hosts

drwxr-xr-x. 3 root root 72 Jan 1 08:30 EXCITE

[root@reviewb .ssh]#

sangit@VP089-PBXCEE7 MINGW64 ~ (master)

$ cd EXCITE/

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ pwd

/c/Users/VP\_ADMIN/EXCITE

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ ls -lrt

total 1

-rw-r--r-- 1 sangit 197121 27 Nov 15 05:56 README.md

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ cat README.md

# EXCITE

Sample project

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git status

On branch master

Your branch is up-to-date with 'origin/master'.

nothing to commit, working tree clean

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

Modify the read me file

And git status shows the uncommitted change in red

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git status

On branch master

Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: README.md

no changes added to commit (use "git add" and/or "git commit -a")

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

Add a new file to the directory and git status shows it

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git status

On branch master

Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: README.md

Untracked files:

(use "git add <file>..." to include in what will be committed)

file-added

no changes added to commit (use "git add" and/or "git commit -a")

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git add README.md

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git status

On branch master

Your branch is up-to-date with 'origin/master'.

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: README.md

Untracked files:

(use "git add <file>..." to include in what will be committed)

file-added

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git add file-added

warning: LF will be replaced by CRLF in file-added.

The file will have its original line endings in your working directory.

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ echo $?

0

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git status

On branch master

Your branch is up-to-date with 'origin/master'.

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: README.md

new file: file-added

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git commit -m "made the changes "

[master f74d55f] made the changes

2 files changed, 2 insertions(+)

create mode 100644 file-added

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git commit

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

nothing to commit, working tree clean

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

To add all the files at once

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git add --all

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

To commit all the files at once

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ pwd

/c/Users/VP\_ADMIN/EXCITE

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git commit --all

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

nothing to commit, working tree clean

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

Example: git commit –all –m “adding all the files”

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git commit --all -m "commiting all the changes"

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

nothing to commit, working tree clean

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git log

commit f74d55fd058515567a34f8d53c7a64baeb63e81c

Author: sangit1 <sanchenn@in.vp.com>

Date: Tue Nov 15 06:08:20 2016 +0530

made the changes

commit 4ae15896e770acb13543f739d6d9641aeb5854c2

Author: User1 K. Git <user1.git@in.vp.com>

Date: Tue Nov 15 05:55:39 2016 +0530

Initial commit

commit f74d55fd058515567a34f8d53c7a64baeb63e81c 🡪 40 character sha1 code commit code

push the changes to master

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git push

Counting objects: 4, done.

Delta compression using up to 4 threads.

Compressing objects: 100% (2/2), done.

Writing objects: 100% (4/4), 366 bytes | 0 bytes/s, done.

Total 4 (delta 0), reused 0 (delta 0)

To https://github.vp.com/user1-git/EXCITE

4ae1589..f74d55f master -> master

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

To create a new branch and switch to to it:

git checkout -b <branch>

To switch to a branch

git checkout <branch>

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git checkout -b branch1

Switched to a new branch 'branch1'

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git branch

\* branch1

master

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$

**Change a file in the branch**

And add

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git add --all

warning: LF will be replaced by CRLF in file-added.

The file will have its original line endings in your working directory.

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git status

On branch branch1

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: file-added

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git branch

\* branch1

master

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git push --set-upstream origin branch1

Total 0 (delta 0), reused 0 (delta 0)

To https://github.vp.com/user1-git/EXCITE

\* [new branch] branch1 -> branch1

Branch branch1 set up to track remote branch branch1 from origin.

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git commit --all -m "committing changes to branch called branch1"

[branch1 8ab53f1] committing changes to branch called branch1

1 file changed, 1 insertion(+)

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git status

On branch branch1

Your branch is ahead of 'origin/branch1' by 1 commit.

(use "git push" to publish your local commits)

nothing to commit, working tree clean

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git push --set-upstream origin branch1

Counting objects: 3, done.

Delta compression using up to 4 threads.

Compressing objects: 100% (3/3), done.

Writing objects: 100% (3/3), 347 bytes | 0 bytes/s, done.

Total 3 (delta 0), reused 0 (delta 0)

To https://github.vp.com/user1-git/EXCITE

f74d55f..8ab53f1 branch1 -> branch1

Branch branch1 set up to track remote branch branch1 from origin.

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (branch1)

$ git checkout master

Switched to branch 'master'

Your branch is up-to-date with 'origin/master'.

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git pull

remote: Counting objects: 1, done.

remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (1/1), done.

From https://github.vp.com/user1-git/EXCITE

f74d55f..fa3eba8 master -> origin/master

Updating f74d55f..fa3eba8

Fast-forward

file-added | 1 +

1 file changed, 1 insertion(+)

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git branch --delete branch1

Deleted branch branch1 (was 8ab53f1).

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git branch --delete branch1

Deleted branch branch1 (was fa3eba8).

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$ git branch

\* master

sangit@VP089-PBXCEE7 MINGW64 ~/EXCITE (master)

$

***How to merge branch with the master:***

git checkout -b br2 🡪 creates a new branch br2

create a file, touch filebr2

modify the file filebr2

git add filebr2

git commit filebr2

git status

move back to master

git checkout master

git merge br2 🡪 merges branch2 with the master brahnch

git push 🡪 will push all changes to github/remote repository

To configure the linux client

Install git on linux machine with internet access

Make sure it pings

[root@rscthydnet1 .ssh]# git clone ssh://github.vp.com/user1-git/EXCITE

Cloning into 'EXCITE'...

remote: Counting objects: 15, done.

remote: Compressing objects: 100% (11/11), done.

remote: Total 15 (delta 0), reused 10 (delta 0), pack-reused 0

Receiving objects: 100% (15/15), done.

[root@rscthydnet1 .ssh]# ls -lrt

total 16

-rw-r--r--. 1 root root 409 Nov 15 12:44 id\_rsa.pub

-rw-------. 1 root root 1679 Nov 15 12:44 id\_rsa

-rw-r--r--. 1 root root 191 Nov 15 12:53 known\_hosts

-rw-r--r--. 1 root root 138 Nov 15 13:07 config

drwxr-xr-x. 3 root root 68 Nov 15 13:09 EXCITE

[root@rscthydnet1 .ssh]# cd EXCITE/

[root@rscthydnet1 EXCITE]# ls -lrt

total 12

-rw-r--r--. 1 root root 47 Nov 15 13:09 README.md

-rw-r--r--. 1 root root 59 Nov 15 13:09 Gittest.txt

-rw-r--r--. 1 root root 73 Nov 15 13:09 file-added

[root@rscthydnet1 EXCITE]#

[root@rscthydnet1 EXCITE]#

**Staging** is a step before the commit process in **git**. That is, a commit in **git** is performed in two steps: **staging** and actual commit. As long as a changeset is in the **staging area**, **git** allows you to edit it as you like (replace **staged** files with other versions of **staged** files, remove changes from **staging**, etc.)

git pull -> gets all latest commits to local repos, to working copy

git push 🡪 copies local files to remote repos

git fetch 🡪 copies remote commits to local repos, but not to working copy

git remote update 🡪 updates local copy with remote changes

workspace

index

local repos

remore repos

***#git fetch***

***#git diff origin***

**diff --git a/file b/file**

**index 332e8fc..3d5eaf4 100644**

**--- a/file**

**+++ b/file**

@@ -4,4 +4,3 @@ hi

abc

xyz

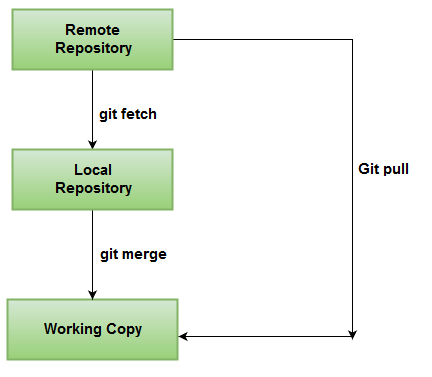
this is for pull

-this is for push

***git merge***

***git pull = git fetch + git merge***

***its always recommended to do git fetch , review changes and then do merge***



To delete the local branch use:

$ git branch -d branch\_name

To delete the branch from the remote repository:

git push origin --delete <branch\_name>

Master is local branch

Origin is remote brach

The below a and b does same thing

**a)**

mkdir repo

cd repo

git init

git remote add origin git://github.com/cmcculloh/repo.git

git fetch --all

git pull origin master

**b)**

git clone git://github.com/cmcculloh/repo.git

git stash - temporary storage box for the changes you are not ready to commit

git branch

git stash

git checkout -b newbr

change something save changes

git stash --> puts all changes in a box

there is something wrong with master , so we need to go back to master and do some changes

git checkout master --> correct it

then go back to branch newbr

git stash pop --index

git diff

create a fle and modify

git add .

git diff --cached --> difference between staging and committed version

git diff - working and staged version

git commit

git diff HEAD - working dir and commit dir

**Git rebase:**

To check the history of commits:

git log --graph --all –oneline

**To check files in a branch:**

$ git ls-tree -r --name-only master

build.xml

cloud.png

file

file123

file12345

file123890

file89089

fle67890898989

index.html

index.html.j2

$ git ls-tree -r --name-only branch123

build.xml

cloud.png

file

file123

file12345

file123890

file89089

fle67890898989

index.html

index.html.j2

**to find out difference between 2 files in 2 diff branches**

ADMINIBM@IBM847-PC0BY40U MINGW64 /123/sample (branch123)

$ git diff master:file branch123:file

**To remove a file**

git rm -f file

git reset HEAD file 🡪 unstage changes

git rebase:

D 🡪 E

A 🡪 B 🡪 C

**F**

**Git clone**

**Git add .**

**Git commit .**

git remote add origin <https://github.com/santoshdevops/sample2.git>

git push -u origin master

**add 2 files and commit**

$ git log --graph --all --oneline

\* aa4d11c Second

\* 5525681 first