Remotes

Remotes synchronizes the local database with one or multiple external remote databases (repositories). The remote repository must exist before you run remote add with the nickname of remote repo and the URI to access remote repo. By default all recipes call the remote repository origin.

Anyone can clone a public (or their own private) remote repo, a local master of that will be download to the local machine. The local master repo created from the clone will automatically have a fetch and push remote link to the orig repo. With a branch you control and manage the branch, whereas with a fork someone else controls accepting the code back in.

New Repository

Push: Local ---> Remote

To send local repo to github/gitlab create the repo/project but don't create the initial readme file as when using *git push* it should only be used with bare repositories. Since there is no initial file github/gitlab gives extra instructions on how to push

In the local master repo add the link to the remote repo and perform a git push. Can change default remote name of *origin*
macoloco:github_remote (master)$ git remote add *origin* **git@github.com:sjhloco/github_remote.git**

macoloco:github_remote (master)$ git remote -v *The remote links have been added*
origin git@github.com:sjhloco/github_remote.git (fetch)
origin git@github.com:sjhloco/github_remote.git (push)

macoloco:github_remote (master)$ git push -u *origin* **master** *Push the master to the remote (origin)*
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 225 bytes | 112.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:sjhloco/github_remote.git
** [new branch] master -> master*
Branch 'master' set up to track remote branch 'master' from 'origin'.

macoloco:gitlab_remote (master)$ git remote add origin git@gitlab.com:sjhloco/gitlab_testing.git
macoloco:gitlab_remote (master)$ git push -u origin master

Pull: Remote ---> Local

To pull remote repo to github/gitlab create the repo/project and initialize it in github/gitlab by creating a readme file. Click *Clone* and this will provide the download link to be used on the local computer.

When you do the clone it will create the repo with the same, or you can optionally specify a name for the repo to use.
macoloco:git-repos$ git clone git@github.com:sjhloco/github_remote1.git *<repo_name>*
macoloco:git-repos$ git clone git@gitlab.com:sjhloco/githlab_remote1.git *<repo_name>*

Existing Repository

Push changes: Local ---> Remote

Different people can work on their own branch of the remote repo and merge the changes from their branch into the remote repo. Due to this the remote repo could have changed since the local master or branch was created, therefore you should check for and pull any changes made to the Remote to your local repository before you push your changes it.

macoloco:gitlab_remote (master)$ git fetch origin master *Check if anything has changed*
From github.com:sjhloco/github_remote1
** branch master -> FETCH_HEAD*
master

macoloco:gitlab_remote (master)$ git pull origin master *Would pull down and merge any changes*
From github.com:sjhloco/github_remote1
** branch master -> FETCH_HEAD*
Already up to date.

macoloco:gitlab_remote (master)$ git push *By default always pushes the change to the origin repo of a fork*
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 265 bytes | 132.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0)
To github.com:sjhloco/github_remote1.git
6b91571..efd5e2d master -> master

git fetch, *git pull*, *git push* with no other options will merge your checked out, local branch git database with the remote master branch database you created your local branch from. Only really need to specify options if you want to merge your current branch with a different branch than the one you created your local one with originally (multiple remote repos).

Push changes: Branch ---> Remote

Rather than merging the local branch committed changes to the master and pushing from there you can push or pull changes to a remote branch directly from a local branch. Is the same concept as keeping changes in a local branch whilst still working in the project, but that local branch can be extended to a remote of that branch (with same branch name).

macoloco:device_configs (asa)$ git push *First push attempt will not work with normal git push cmd*

fatal: The current branch asa has no upstream branch.
To push the current branch and set the remote as upstream, use
git push --set-upstream origin asa

macoloco:device_configs (asa)$ git push --set-upstream origin *asa* *For all subsequent pushes can use git push*

Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 1.70 KiB | 871.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0)
remote:
remote: To create a merge request for asa, visit:
remote: https://gitlab.com/sjhloco/device_configs/merge_requests/new?merge_request%5Bsource_branch%5D=asa
remote:
To gitlab.com:sjhloco/device_configs.git
** [new branch] asa -> asa*
Branch 'asa' set up to track remote branch 'asa' from 'origin'

Normally you would merge branches into the remote locally.
macoloco:device_configs (work)$ git merge *test1*

Instead of merging two branches locally you perform the merge using the remote web interface (*submit merge request*) which seems better as in GUI is easier to see differences between the branches and make decision on whether to merge.

macoloco:device_configs (work)$ git push

Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 377 bytes | 188.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote:
remote: To create a merge request for work, visit:
remote: ***https://gitlab.com/sjhloco/device_configs/merge_requests/new?merge_request%5Bsource_branch%5D=work***
remote:
To gitlab.com:sjhloco/device_configs.git
a877a7d..caf5d62 work -> work

Your local master repo will now be out of date with the remote as it has merged the branch into the master. Therefore need to pull the changes from the remote master down to the local master repo. Can now also delete the local branch.

macoloco:gitlab_remote (master)$ git pull *Retrieve and merge changes from remote*
macoloco:gitlab_remote (master)$ git branch -d *test1* *Delete local branch*

When working with remotes the log shows if commits were performed on the remote and which branch (origin/)

macoloco:device_configs (work)$ git log --oneline

c61972e (HEAD -> work, origin/work) Deleted random crap *<<< HEAD means last/current commit*
d06e425 (origin/master, origin/HEAD, master) Update hm-vm-esx01.txt *<<< committed in remote master*
a3cabd6 Update README.md
7962d55 (origin/asa) Added hm-vm-esx01 *<<< committed in in remote asa branch*
00dfa72 SH 17/03/19 - Changed hostname
085b757 Added hm-asa-fw1
f9099f5 Initial commit

Forks – Forking a public Repo

It is nothing more than a github/gitlab specific way to clone a someone's remote repo into your github user account. You don't have to contribute back to the original repo, but if you do when you clone it locally need to add an extra remote to original. The *origin* is your remote master and the *upstream* the original remote master.

Within github/gitlab click *Fork* which will automatically create a copy of that repository under your github account. Next download (clone) a local copy of your version of this repository that github/gitlab just created.
macoloco:git-repos$ git clone git@github.com:sjhloco/<repo_name>

macoloco:github_tutorial (gh-pages)$ git remote -v
origin git@github.com:sjhloco/github_tutorial.git (fetch)
origin git@github.com:sjhloco/github_tutorial.git (push)

To complete the fork need to add a remote back to the original owner's repository (name myfriend can be anything)
macoloco:github_tutorial (gh-pages)$ git remote add *myfriend* **git@github.com:kbroman/github_tutorial.git**

macoloco:github_tutorial (gh-pages)$ git remote -v
myfriend git@github.com:kbroman/github_tutorial.git (fetch)
myfriend git@github.com:kbroman/github_tutorial.git (push)
origin git@github.com:sjhloco/github_tutorial.git (fetch)
origin git@github.com:sjhloco/github_tutorial.git (push)

Before making changes to a forked repo you should always pull down and merge any changes from the owners remote repo and push them back to your github repository.
macoloco:github_tutorial (gh-pages)$ git fetch *Check if there were any changes*
macoloco:github_tutorial (gh-pages)$ git pull *myfriend* **master** *Only needed if were changes*
macoloco:github_tutorial (gh-pages)$ git push *Push changes back to your remote copy of repo*

After you make any changes push them back to github. These will go to your version of the repository.
macoloco:github_tutorial (gh-pages)$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 314 bytes | 314.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:sjhloco/github_tutorial.git
515178c..c8fff2b gh-pages -> gh-pages

Now your version of the repo has been changed, if you wanted to also change the original owners repo click *New pull request.* Your friend's repository will be on the left and your repository will be on the right with the changes at the bottom of the page. Click *Create pull request* to send these changes to the owner who can then reject or approve them.

Fork – Handling Pull requests

Once someone submits a pull request you'll get an email about it. Under your repository in Github, *Pull Requests* at the top and click on the particular request. You'll see their comments on the pull request, and can click to see the exact changes.

Can either *Merge pull request*, *add a comment* to send it back to the requester for further changes or *Close* to reject it. Your github repo will now be updated, but if you have a local copy you will need to pull the new changes down to that.
macoloco:github_tutorial (gh-pages)$ git pull

You don't have to use the github website to process pull requests, can be done in the CLI of the in the local master. Need to add a remote to the requesters version of github repo, then Pull his changes and Push them back to your github repo.
macoloco:github_tutorial (gh-pages)$ git remote add *myfriend* **git://github.com/myfriend/the_repo**
macoloco:github_tutorial (gh-pages)$ git pull *myfriend* **master**
macoloco:github_tutorial (gh-pages)$ git push