First, if you added ssh key

SSH settings for GitHub:

1. Set user name and email for your Git:

$ git config --global user.name "XXXX"

$ git config --global user.email "XXXX@paypal.com"

1. Check if ssh key exists on your device:

cd ~/.ssh

ls

1. Generate ssh key for GitHub:

$ ssh-keygen -t rsa -C “XXXX@paypal.com”

Press Enter 3 times to set the empty password

Your identification has been saved in /home/XXXX/.ssh/id\_rsa. Your public key has been saved in /home/XXXX/.ssh/id\_rsa.pub. The key fingerprint is:

..................

Then you got 2 files: id\_rsa and id\_rsa.pub

1. Add ssh key on GitHub

Copy the code in id\_rsa.pub

Add it to GitHub 🡪 Settings 🡪 SSH and GPG keys

1. Test your key

ssh -T git@github.com

1. `$ git config --global user.name "xuhaiyan"`

`$ git config --global user.email "haiyan.xu.vip@gmail.com"`

2. `$ ssh-keygen -t rsa -C "humingx@yeah.net"`

3. `eval "$(ssh-agent -s)"`

`$ ssh-add ~/.ssh/id\_rsa`

4. `$ clip < ~/.ssh/id\_rsa.pub`

copy \_id\_rsa.pub\_ to github

Clone your repository to your local device:

$ git clone git@github.paypal.com:cdanny/G11N-Parent.git

Add remote repository:

-- origin represents your repo，upstream represents international quality repo

$ git remote add origin git@github.paypal.com:cdanny/G11N-Parent.git

$ git remote add upstream git@github.paypal.com:InternationalQuality/G11N-Parent.git

Remove remote repository:

$ git remote remove origin

Fetch code from upstream (remote) repo:

$ git fetch upstream

Fetch code from origin (local) repo:

$ git fetch origin

\*\* Fetch after clone, and then you can have all the branches of that repo.

About git remote:

$ git remote show all remote repositories

$ git remote –v show all remote repositories with respective address

$ git remote show [remote-name] show remote repo info

$ git remote add [short-name] [url] add new remote repo

$ git remote rename [old-name] [new-name] rename existing remote repo

$ git remote rm remove existing remote repo

Push for the first time, makes your new branch show up on GitHub:

$ git push --set-upstream origin [branch-name]

Delete local branch:

$ git branch -d [branch name]

Force delete local branch:

$ git branch -D [branch name]

Delete GitHub branch in command line:

$ git push origin --delete [branch name]

$ git push origin --delete [branch name] # Git version 1.7.0 or newer

$ git push origin :[branch name] # Git versions older than 1.7.0

Rename local branch:

$ git branch -m [old-name] [new-name]

Create a new local branch with latest code from develop branch:

$ git checkout -b [branch name] upstream/develop

or

git branch [branch-name] up/dev

git checkout [branch-name]

Normal steps of commit:

1. $ git gui

(new commit / commit)

Or

$ git add . – add all finel

1. $ git commit –m “first commit”
2. $ git fetch upstream

$ git rebase upstream/develop

$ git push origin [branch name] -f (force commit)

$ git stash

$ git stash pop

$ git cherry-pick [SHA1 ID]

$ git pull == fetch + merge

$ git pull --rebase == fetch + rebase

Reset back to previous version:

$ git reflog -...

$ git reset HEAD...

Merge 2 (or more) commits to 1 commit:

1. $ git rebase -i HEAD~2
2. modify 2nd commits from pick to squash, save and quit (:wq)
3. remove 2nd commit message, save and quit (:wq)
4. git push origin [branch-name] -f

Clean screen:

$ reset

Show commit history:

$ git reflog show

Modify commit author:

1. $ git log -2
2. copy the value of 2nd SHA1
3. $ git reset --mixed [SHA1]
4. commit again

Alternatively (modify commit author):

git commit --amend --author="Author Name <email@address.com>"

git fetch XXXX

git rebase repo/branch

Conflict

git add .

git rebase --continue

e7b50a6a