## Git简介

最先进的分布式版本控制系统

在repository中才有git 环境可以使用git语句操作，在repository外面可以使用Linux命令操作文件或文件夹 eg: rm rf /path 删除文件夹 rm –f/path/file.txt\

使用github概念

Git init ，git config –global user.name “” git config –global user.email “”

Master为主分支名 master branch

Origin 为远程仓库

创建remote repository只能在github 网页上创建

多人协作，使用branch 提交到master 选择merge

直接将项目克隆到本地，会自动创建目录,或者直接将本地项目初始化成repository，修改后提交，要提交要先连接远程仓库git remote add origin <http://path/to/remote/repository/repository.git，git> add,git commit –m “”,git push origin master，系统会自动识别修改部分，建立分支后在master 使用git merge branch 整合

Branch只能看到自己分支中文件，包括本地的repository中的文件也会因为分支不同而显示的文件不同，

Ssh keygen 秘钥用于对应本地仓库和远程仓库 设置秘钥才能上传

ssh -keygen -T ras -C [youremail@example.com](mailto:youremail@example.com)

ssh-agent –s查看秘钥是否启用

ssh –T [git@github.com](mailto:git@github.com) 验证后才能上传

remote repository operation

查看 git remote –v

复制：git remote clone

添加 git remote add name url

删除 git remote rm name

修改 git remote set-url --push name newurl

Create && clone

Create new local repository：git init

Clone local repository：git clone /path/to/repository

Clone remote repository:git clone [username@host://path/to/repository.git](mailto:username@host://path/to/repository.git) 将remote repository复制进本地的repository 没有相应目录则自动创建

Add && remove

Add changes to INDEX: git add <filename>

Add all changes to INDEX: git add \*

Remove/delete: git rm <filename>

Commit && synchronize

Commit changes: git commit –m “Commit message”

Connect local repository to remote repository: git remote add origin <server>

Push changes to remote repository: git push origin master

Update local repository with remote changes: git pull

Branches

Create new branch: git checkout –b <branch> eg:git checkout –b <feature\_x>

Switch to master branch: git checkout master

Delete branch: git branch-d <branch>

Push branch to remote repository: git push origin<branch>

拉取 git pull remotename localbranchname

推送 git push remotename localbranchname

git push origin test:master //提交本地test分支作为远程master分支

git push origin test:test //提交本地test分支作为远程test分支

Merge

Merge changes from another branch: git merge<branch>

View changes between two branches: git diff <source\_branch> <target\_branch> eg:git diff feature\_x feature\_y

Tagging

Create tag : git tag<tag> <commit ID> eg:git tag 1.0.0 1b2x3dfdf

Get commit IDs: git log

Retore

Replace working copy with latest from HEAD : git check –filename

<>括号不用加

# 删除 untracked files

git clean -f

# 连 untracked 的目录也一起删掉

git clean -fd

# 连 gitignore 的untrack 文件/目录也一起删掉 （慎用，一般这个是用来删掉编译出来的 .o之类的文件用的）

git clean -xfd

# 在用上述 git clean 前，墙裂建议加上 -n 参数来先看看会删掉哪些文件，防止重要文件被误删

git clean -nxfd

git clean -nf

git clean -nfd

The most commonly used git commands are:

add Add file contents to the index

bisect Find by binary search the change that introduced a bug

branch List, create, or delete branches

checkout Checkout a branch or paths to the working tree

clone Clone a repository into a new directory

commit Record changes to the repository

diff Show changes between commits, commit and working tree, etc

fetch Download objects and refs from another repository

grep Print lines matching a pattern

init Create an empty Git repository or reinitialize an existing one

log Show commit logs

merge Join two or more development histories together

mv Move or rename a file, a directory, or a symlink

pull Fetch from and integrate with another repository or a local branch

push Update remote refs along with associated objects

rebase Forward-port local commits to the updated upstream head

reset Reset current HEAD to the specified state

rm Remove files from the working tree and from the index

show Show various types of objects

status Show the working tree status

tag Create, list, delete or verify a tag object signed with GPG

usage: git branch [options] [-r | -a] [--merged | --no-merged]

or: git branch [options] [-l] [-f] <branchname> [<start-point>]

or: git branch [options] [-r] (-d | -D) <branchname>...

or: git branch [options] (-m | -M) [<oldbranch>] <newbranch>

Generic options

-v, --verbose show hash and subject, give twice for upstream branch

-q, --quiet suppress informational messages

-t, --track set up tracking mode (see git-pull(1))

--set-upstream change upstream info

-u, --set-upstream-to <upstream>

change the upstream info

--unset-upstream Unset the upstream info

--color[=<when>] use colored output

-r, --remotes act on remote-tracking branches

--contains <commit> print only branches that contain the commit

--abbrev[=<n>] use <n> digits to display SHA-1s

Specific git-branch actions:

-a, --all list both remote-tracking and local branches

-d, --delete delete fully merged branch

-D delete branch (even if not merged)

-m, --move move/rename a branch and its reflog

-M move/rename a branch, even if target exists

--list list branch names

-l, --create-reflog create the branch's reflog

--edit-description edit the description for the branch

-f, --force force creation (when already exists)

--no-merged <commit> print only not merged branches

--merged <commit> print only merged branches

--column[=<style>] list branches in columns

usage: git remote [-v | --verbose]

or: git remote add [-t <branch>] [-m <master>] [-f] [--tags|--no-tags] [--mirror=<fetch|push>] <name> <url>

or: git remote rename <old> <new>

or: git remote remove <name>

or: git remote set-head <name> (-a | --auto | -d | --delete |<branch>)

or: git remote [-v | --verbose] show [-n] <name>

or: git remote prune [-n | --dry-run] <name>

or: git remote [-v | --verbose] update [-p | --prune] [(<group> | <remote>)...]

or: git remote set-branches [--add] <name> <branch>...

or: git remote set-url [--push] <name> <newurl> [<oldurl>]

or: git remote set-url --add <name> <newurl>

or: git remote set-url --delete <name> <url>

-v, --verbose be verbose; must be placed before a subcommand