# git

分布式版本控制系统

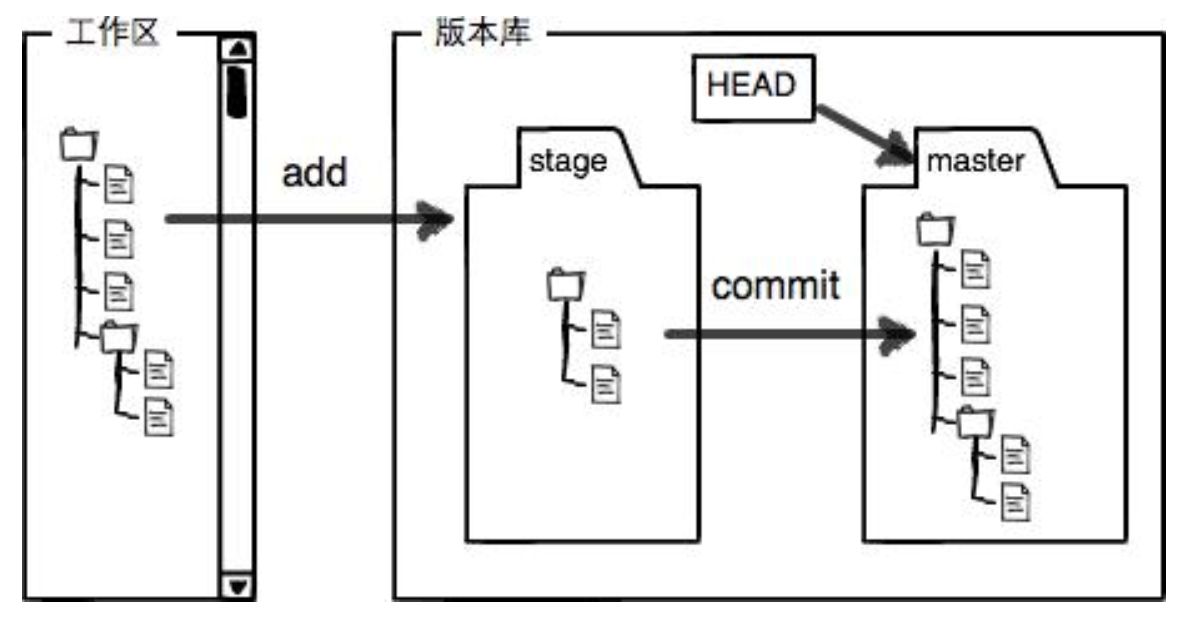
## 概念

git包含工作区、版本库

工作区即仓库目录

仓库目录中包含.git目录，即版本库

版本库包含暂存区stage、分支Master、指向Master的指针HEAD



add操作是将文件保存到暂存区，commit操作是将stage的全部文件提交到master上。

git diff 检测暂存区和工作区的区别

git diff –cached 检测暂存区和分支里的区别

文件修改后如果不使用git add命令添加进缓存区，那么commit时就不会被提交到master。

## 操作

git init 将当前目录初始化为仓库

git add filename 将文件加入到仓库

git commit -m message 提交已加入的文件到仓库

git status 查看仓库文件状态

git diff 查看改变的情况

git log 查看历次提交情况

git reset 回滚历史版本，可以使用HEAD，也可以指定版本号

git reflog 产看历史命令

git checkout -- filename 用版本库的文件替换工作区的文件

git reset HEAD filename 撤回add进暂存区的内容，回到工作区

git rm 删除文件，命令进入暂存区，也需要commit后才能生效，commit后版本库也没有文件了

git remote add origin [git@server-name:path/repo-name.git](mailto:git@server-name:path/repo-name.git)

将本地仓库关联到远程仓库，其中origin是定义的远程仓库名称

git push -u origin master 第一次推送分支的所有内容

git push origin master 推送更新的内容

git clone + url 克隆远程仓库到本地

分支：

git branch 查看当前分支

git branch name 新建分支

git checkout name 切换分支

git branch -d 删除分支

git branch -D 强制删除未合并的分支

分支提交的内容不会影响其他分支

git merge 分支 将指定分支合并到当前分支

两个分支修改了不同的内容后，合并时会起冲突需要手动更改冲突的内容。

git默认为fast forward，merge时会将分支信息全部删除，工作中可以使用git merge –no-ff ，这样会禁用fast forward，保留分支信息

git stash 存储当前分支现场

git stash list 查看存储的分支

git stash apply + 指定分支 将现场恢复

git push origin 将分支push到远程仓库

git checkout -b dev origin/dev clone远程仓库的分支

git pull 从远程仓库中拉取代码

拉取代码时需要建立本地分支与远程分支的链接

git branch --set-upstream-to <branch-name> origin/<branch-name>

拉取时会自动合并，合并时产生冲突需要修复

多人合作流程：

1. push自己的修改
2. 失败则说明远程分支比本地更新，需要pull后合并
3. 合并产生冲突，修复
4. 修复后再次push

tag：默认对本次提交打tag

git tag v1.0 对本次提交打tag

对过往的commit打标签时，可以先通过log查找对应id

git tag -a v0.1 -m "version 0.1 released" 1094adb

tag还可以备注信息

git push origin v1.0 推送tag到远程仓库

* 命令git push origin <tagname>可以推送一个本地标签；
* 命令git push origin --tags可以推送全部未推送过的本地标签；
* 命令git tag -d <tagname>可以删除一个本地标签；
* 命令git push origin :refs/tags/<tagname>可以删除一个远程标签。