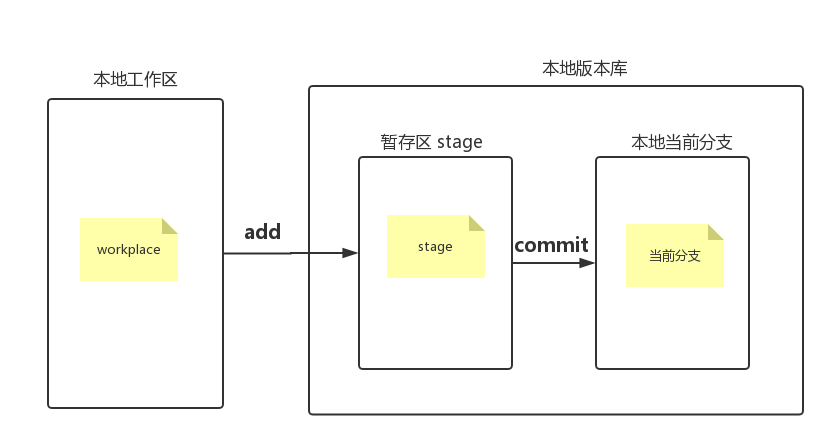
工作原理



通过上图大概可以看出git的工作原理：

本地工作区：随意的一个文件都可以被认为是工作区，一般来说指的是我们IDE中的代码。

暂存区：通过git add命令可以把本地代码提交到本地版本库的暂存区中。

当前分支：通过git commit命令可以把暂存区的代码提交到本地当前分支管理中并生成commitId，然后就可以通过git push提交到远端的分支上了

一、git常用命令总结：

1.1、删除本地/远程分支

git branch -d 分支名 删除本地分支

git push --delete origin 分支名 删除远程分支

1.2、修改提交人的姓名和邮箱

全局替换：

git config --global user.name 用户名

git config --global user.email 用户邮箱

1.3、查看当前分支状态

git status

1.4、查看当前所有分支

git branch

1.5、从当前分支上创建新分支

1、更新当前分支的代码 git pull origin 分支名

2、创建新的分支 git checkout -b 新分支名

此时，当前分支的代码就是基于新分支的，新分支代码还是在本地创建的

3、将新分支代码推送到romote: git push origin 新分支名

4、本地和远端建立连接：git branch --set-upstream-to origin/远端分支名 本地分支名

1.6、分支合并到master上

举个例子：比如我们现在在dev分支上开发代码呢，开发完之后需要把本地代码提交到远端，执行了下列命令：

git add .

git commit -m 'push\_description'

git push origin dev

通过上面的命令我们就把dev本地的代码提交到了远端，接着我们要把dev分支的代码合并到master上，该怎么做呢？

首先切换到master分支上：

git checkout master

然后执行

git pull origin master

保证本地的master代码是最新的，接着把dev分支的代码合并到master上：

git merge dev

通过上面这个命令就可以在本地把dev分支的代码合并到master了，然后执行git status查看当前分支状态：

On branch master

Your branch is ahead of 'origin/master' by 465 commits.

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

nothing to commit, working tree clean

上面说明本地master的代码是超前远端master的代码的，所以还需要把本地master的代码同步到远端，执行：

git push origin master

这样就把本地master最新代码同步到远端了。

1.7、提交代码到远端

提交本地所有代码到暂存区 git add .

将本地暂存区的改动提交到本地版本库 git commit -m ‘message’

将本地代码提交到远端 git push origin 分支名 如果在当前分支 可以直接用git push

查看本地日志 git log

版本回退 git reset

显示工作目录和暂存区状态 git status

1.8、git rebase 和 git merge的用法

假设一个项目是由A和B两个人开发，基于C2代码建立的分支feature1.0，如图：

接着A提交了两次代码（C3、C4）到远端feature1.0，此时B提交了两次代码（C5、C6）到本地（注意是本地），此时如果B想提交代码到远端，应该首先通过git pull拉取一下分支上的代码和B本地的修改进行一次合并（注：git pull= git fetch+ git merge git pull -r =git fetch+ git rebase ）,但是如果忘了调用git pull，直接调用git merge可以合并代码：

从图上看可以看到先分叉又合成一条线了，并且C7是合并代码新生成的一次commitId,如果你想让历史分支看起来想没有经过任何合并一样，可以使用git rebase:

看上去就是一条线顺下来的，并且少生成一次commitId，rebase过程中如果有冲突，git会停止rebase并提示去解决冲突，解决完冲突后，可以通过git add更新内容索引，接着可以执行git rebase --continue继续执行；如果想撤销rebase，可以通过git rebase --abort

rebase和merge也可以操作两个不同的分支：

https://www.cnblogs.com/0201zcr/p/5752771.html

https://www.cnblogs.com/marblemm/p/7161614.html

https://blog.csdn.net/wh\_19910525/article/details/7554489

1.9、将当前分支的某一次代码提交到另一个分支上去

比如需要将A分支的某一次提交到B分支上去，首先在A分支通过git log发现提交过的commitID, 接着将分支切到B分支上git checkout B,最后执行git cherry-pick commitId 将A分支上相对应commitId的代码提交到B分支上了，更多信息见：

https://www.cnblogs.com/yangmingyu/p/6904683.html

1.10、打TAG

1、列本地已有标签 git tag

2、新建含附注的标签 git tag -a 标签名 -m '标签注释'

-a annotated的缩写 -m message缩写

3、查看相应版本信息 git show 标签名

4、把标签推送到远端 git push origin 标签名

一次性推送所有本地新增标签到远端 git push origin --tags

5、删除本地tag git tag -d 标签名

更多: https://git-scm.com/book/zh/v1/Git-基础-打标签

1.11、撤销本地工作区所有改动

如果只撤销某个文件的改动 git checkout xxx(文件名)

如果想撤销本地所有工作区的改动 git checkout .

1.12 git revert & git reset

git reset: 某个版本分支之后的代码不要了， 回滚版本，指针Head向前移，指向千面某个指定版本

git revert: 回滚版本，但是Head指针是向前移动的。

具体请移步：https://blog.csdn.net/yxlshk/article/details/79944535

二、实战

1、本地与remote相应分支没有关联上

Can't update:no tracked branch No tracked branch configured for branch 分支名 To make your branch track a remote branch call, for example, git branch --set-upstream-to origin/分支名 分支名

没有关联到远端分支，直接按提示执行git branch --set-upstream-to origin/分支名 分支名就可以进行关联了

2、替换本地commit时的信息

本地执行了git commit -m 'message'命令，发现message信息写错了，这时候想替换message备注信息，可以直接使用git commit --amend -m 'New commit message'来修改message信息

3、本地撤回提交

本地已经执行过git add 和git commit命令，也就是说已经将代码提交到本地暂存区并且提交到本地版本库了，突然发现有一个文件不需要提交，这时候就需要执行版本回退了，版本回退分为两种：

git reset --soft:

回退到某个版本，只回退了commit的信息，本地的代码还是存在的，如果想再次提交，再次commit就可以了

git reset --hard:

彻底回退到某个版本，本地的代码也会会退到指定版本的代码，撤销的commit中的更改也会被冲洗掉

举个栗子：

我执行了git commit -m 'message'之后，代码推到本地版本库中并生成了一个commit\_id，通过git log可以看到，如下图：



PS：如果想退出查看Git log,可以直接按Q键退出

可以看到我们最新一次的commit是E，但是我发现有个文件提交错了，不想把这个文件push到远端，那么可以直接通过执行git reset --soft df4fb2929c41c29bae36b9fcb7234d164673564b 回退到前一个版本D，之前提交的代码还可以在本地找到，修改之后就可以重新commit了；这里如果提交到本地版本E中的代码都没有用，当回退到D时，我希望直接把没用的代码直接冲掉，那么可以直接执行git reset --hard df4fb2929c41c29bae36b9fcb7234d164673564b,这样在回退到D的同时，之前提交的代码也就被冲洗掉了