# git指令

git init 将当前目录初始化成库

git status 查看当前状态

[分支名 +0 ~0 -0 | +1 ~0 -0] 绿色代表暂存区，红色代表工作区，+添加 ~修改 -删除 ！冲突

git add 文件名 添加到暂存区

git add . （.）全部提交

git commit 从暂存区提交版本区（提交时需要注释）

git commit -m “” 直接写注释提交

git commit -a -m “” 简写，可暂存，版本一同提交

对比

git diff 工作区和暂存区的对比

git diff --cached 暂存区和版本区的对比

git diff --staged 同上

git diff 分支名 工作区和版本区对比

撤销

git reset HEAD 文件名 暂存区撤回工作区

git checkout -- 文件名 将工作区文件还原成暂存区/版本区

git commit -m “” --amend 撤回上次提交，合并本次一起提交

删除

git rm 文件名 删除暂存区文件（工作区已经删除的情况下）

git rm -f 文件名 强制删除暂存区和工作区文件

git rm --cached 文件名 只删除暂存区文件

恢复

commit\_id 是版本的id，使用git log 打印日志 在 commit 后的一串字符，复制一部分就可以作为id 使用。

git checkout commit\_id 文件名 还原某文件

git reset --hard commit\_id 还原某版本

git reset --hard HEAD^ 还原成上一版本

git reset --hard HEAD~2 还原成上2版本

git reflog 我的操作行为（有 commit\_id 记录 上面二方法还原）

同步到远程仓库（先放到版本区，在提交远程）

git push 仓库名称 分支名 同步到仓库

git remote 查看仓库名称

git remote add “” 改仓库名

git remote -v 查看仓库地址

用户信息

git config --global user.name 查看用户名

git config --global user.name “名称” 修改用户名

git config --global user.email “邮箱” 修改邮箱

克隆下载

git clone 地址 克隆项目

地址如下

同步

冲突：不拉取，并且同时修改一个文件，并上传，再拉取时就会有冲突

解决冲突，1.拉取同步远端文件2.查看冲去3.合并冲突4.手动删除文件中不要部分5.提交

git fetch 拉取同步

git diff 分支名 仓库名/分支名 查看冲突区别

git merge 仓库名/分支名 合并冲突文件

git pull （常用） 直接拉取还原成远端文件，不会显示冲突

分支（版本库的操作）

git branch 查看分支

git branch 分支名 创建分支

git checkout 分支名 切换分支

git checkout -b 分支名 创建并切换到新分支

git merge 分支名 合并

git branch --merged 查看当前分支所合并的分支

git branch --no-merged 查看当前分支没有合并的分支

git branch -d 分支名 删除分支（已经合并的）

git branch -D 分支名 强制删除分支

合并时出现冲突，手动删除不要的代码，再次提交

git push 仓库名称 分支名 同步分支到仓库或github上操作（下面）

git tag 版本名 同步版本（打标签）或github操作（下面）

git tag 查看当前版本

仓库名称 origin