1. **删除本地分支和远程分支**
2. 切换到要操作的项目文件夹   
   命令行 : $ cd <ProjectPath>
3. 查看项目的分支们(包括本地和远程)   
   命令行 : $ git branch –a
4. 删除本地分支   
   命令行 : $ git branch -d <BranchName>
5. 删除远程分支   
   命令行 : $ git push origin --delete <BranchName>
6. 查看删除后分支们   
   命令行 : $ git branch –a
7. [**git 本地给远程仓库创建分支**](https://www.cnblogs.com/Amos-Turing/p/6285390.html)
8. 本地创建分支dev

git branch dev

1. 把本地分支提交到远程仓库

 git push origin dev

1. **Git拉取远程分支并创建本地分支**
2. **方法1**

git checkout -b 本地分支名x origin/远程分支名x

使用该方式会在本地新建分支x，并自动切换到该本地分支x。

1. **方法2**

git fetch origin 远程分支名x:本地分支名x

使用该方式会在本地新建分支x，但是不会自动切换到该本地分支x，需要手动checkout。

1. **强制覆盖本地代码**

方法1:git强制覆盖：

     git fetch --all

    git reset --hard origin/master

    git pull

方法2:git强制覆盖本地命令（单条执行）：

    git fetch --all && git reset --hard origin/master && git pull

1. **查看最近三次提交以及修改了哪些文件**

**git log -3 --stat**

1. **Git diff 用法**
2. **比较工作目录和暂存区某个文件的修改**

**git diff [--options] [--] [<path>...]**

1. **比较文件系统上的两个文件**

**git diff --no-index [--options] [--] [<path>...]**

**注：当运行命令处于工作区，但是有一个文件在工作区之外或者命令运行在工作区之外，可以删掉-no-index选项**

1. **比较暂存区和已经提交的文件的比较**

**git diff [--options] --cached [<commit>] [--] [<path>...]**

1. **比较工作区和已经提交文件的比较**

**git diff [--options] <commit> [--] [<path>...]**

1. **比较任意两次提交下某个文件的差异**

**git diff [--options] <commit> <commit> [--] [<path>...]**