**Git常用命令**

1. **git clone**：clone一个git仓库
2. **git config：**我们可以通过git config来配置用户名和邮箱地址，便于我们将代码提交到远程仓库，具体格式如下：

|  |
| --- |
| Dockerfile git config --global user.name '你的用户名' git config --global user.email '你的邮箱'  // 使用缓存来记住密码，git 会将认证凭据**缓存在内存中一段时间**，以便在下次访问同一仓库时不必再次输入用户名和密码。 git config --global credential.helper cache |

1. **git add：**将文件添加到缓存 git add . // 添加所有文件到缓存
2. **git status：** 命令来查看相关文件的状态
3. **git commit：**将缓存区内容添加到仓库中，可以在后面加-m选项，以在命令行中提供提交注释

git commit -m "第一次版本提交"

1. **git reset HEAD：**取消已缓存的内容

|  |
| --- |
| JavaScript git reset HEAD test.txt // 执行完之后，再使用commit提交时，test.txt文件不会被提交 git reset --soft HEAD~1 // 撤销最后一次提交（但保留更改） git reset --hard HEAD~1 // 撤销最后一次提交并丢弃更改 |

1. **git branch：**查看/创建分支

|  |
| --- |
| JavaScript git branch // 查看当前分支 git branch branchNAme // 创建分支名为branchName的分支 |

1. **git checkout branchName**：切换到分支名为branchName的分支

git checkout -b branchName：创建分支名为branchName的分支并切换至该分支

1. **git merge**：合并分支

git merge branchName 将 branchName分支合并到当前分支

1. **git remote add**：添加远程仓库

git remote add [别名] [远程url] 将远程仓库添加到本地，并命名为[别名]

git remote：查看当前有哪些远程仓库

1. **git fetch：**拉取远程仓库
2. **git submodule update --init --recursive：**确保在该项目下的子模块(独立的仓库)被正确初始化和更新，通常情况下，只有在克隆父项目之后首次使用子模块时才需要初始化它们。

git submodule update --remote --recursive从远程仓库中获取更新，并将这些更新应用到当前仓库中的所有子模块，同时也会更新子模块的子模块，直到所有的子模块都被更新到最新版本。

1. **git clean -dfx**：删除工作目录中所有未跟踪的文件和忽略文件
2. **git checkout -- <file>：**撤销工作区中的更改
3. **git reset <file>：**撤销暂存区的更改（恢复工作区）
4. **git diff <branch1>..<branch2>：**比较两个分支之间的差异
5. **git log --grep="search\_text"：**根据提交信息查找提交
6. **git branch -d <branch>：**删除一个分支

|  |
| --- |
| 当一个分支被推送并合并到远程分支后，-d 才会本地删除该分支。如果一个分支还没有被推送或者合并，那么可以使用-D强制删除它。 |

1. **git push <remote> --delete <branch>：**删除远程分支

|  |
| --- |
| JavaScript git push origin --delete branch\_test // 删除远程分支branch\_test |