

配置工具

```
$ git config --global user.name " [na me] "
```

对 commit 操作设置关联用户名

```
$ git config --global user.email "[email address]"
```

对 commit 操作设置关联的邮箱地址

创建仓库

```
$ git init [project-name]
```

创建一个本地的仓库，并设置名字

```
$ git clone [url]
```

下载一个项目以及它所有的版本历史

更改

```
$ git status
```

列出所有新建或者更改的文件，这些文件需要被commit

```
$ git diff
```

展示那些没有暂存文件的差异

```
$ git add [file]
```

将文件进行快照处理用于版本控制

```
$ git diff --staged
```

展示暂存文件与最新版本之间的不同

```
$ git reset [file]
```

将文件移除暂存区，但是保留其内容

```
$ git commit -m"[descriptive message]"
```

将文件快照永久地记录在版本历史中

批量更改

```
$ git branch
```

列出当前仓库中所有的本地分支

```
$ git branch [branch-name]
```

建立一个新分支

```
$ git checkout [branch-name]
```

切换到一个特定的分支上并更新工作目录

```
$ git merge [branch-name]
```

合并特定分支的历史到当前分支

```
$ git branch -d [branch-name]
```

删除特定的分支

同步更改

```
$ git fetch [remote]
```

下载远程仓库的所有历史

```
$ git merge [remote]/[branch]
```

合并远程分支到当前本地分支

```
$ git push [remote] [branch]
```

上传所有本地分支commit到GitHub上

```
$ git pull
```

下载书签历史并合并更改

停止追踪

```
*.log |build/ |temp-*
```

文本文件.gitignore可以防止一些特定的文件进入到版本控制中

```
$ git ls-files --others --ignored --exclude-standard
```

列出所有项目中忽略的文件

重构文件

```
$ git rm [file]
```

从工作目录中删除文件并暂存此删除

```
$ git rm --cached [file]
```

从版本控制中移除文件，并在本地保存文件

```
$ git mv [file-original] [file-renamed]
```

改变文件名并准备commit

保存临时更改

```
$ git stash
```

临时存储所有修改的已跟踪文件

```
$ git stash pop
```

重新存储所有最近被stash的文件

```
$ git stash list
```

列出所有被stash的更改

```
$ git stash drop
```

放弃所有最近stash的更改



查阅历史

\$ git log

列出当前分支的版本历史

\$ git log --follow [file]

列出文件的版本历史，包括重命名

\$ git diff [first-branch]...[second-branch]

展示两个不同分支之间的差异

\$ git show [commit]

输出元数据以及特定commit的内容变化

撤销commit

\$ git reset [commit]

撤销所有 [commit] 后的的commit，在本地保存更改

\$ git reset --hard [commit]

放弃所有更改并回到某个特定的commit



By liuy

cheatography.com/liuy/

Published 13th December, 2019.

Last updated 19th December, 2021.

Page 2 of 2.

Sponsored by **Readable.com**

Measure your website readability!

<https://readable.com>