Git 常用命令速查手册

Git Common Commands Quick Reference

版本控制必备工具

2025年整理

Git 常用命令速查手册

1. 配置命令

命令	说明
`git config --global user.name "用户名"`	设置全局用户名
`git config --global user.email "邮箱"`	设置全局邮箱
`git config --global core.editor vim`	设置默认编辑器
`git config --global color.ui true`	开启颜色显示
`git config --list`	查看所有配置
`git config user.name`	查看用户名

2. 仓库操作

命令	说明
`git init`	在当前目录初始化Git仓库
`git clone <url>`	克隆远程仓库到本地
`git clone -b <分支名> <url>`	克隆指定分支
`git remote -v`	查看远程仓库地址
`git remote add origin <url>`	添加远程仓库
`git remote remove origin`	删除远程仓库

3. 基本操作

3.1 查看状态

命令	说明
`git status`	查看工作区状态
`git status -s`	简洁模式查看状态
`git diff`	查看未暂存的修改
`git diff --cached`	查看已暂存的修改

3.2 添加与提交

命令	说明
`git add <文件名>`	添加指定文件到暂存区
`git add .`	添加所有修改到暂存区
`git add -A`	添加所有变化（包括删除）
`git commit -m "提交信息"`	提交暂存区的修改
`git commit -am "提交信息"`	添加并提交所有修改
`git commit --amend`	修改最后一次提交

4. 分支操作

命令	说明
`git branch`	列出本地分支
`git branch -a`	列出所有分支（含远程）
`git branch <分支名>`	创建新分支
`git checkout <分支名>`	切换到指定分支
`git checkout -b <分支名>`	创建并切换到新分支
`git switch <分支名>`	切换分支（新命令）
`git switch -c <分支名>`	创建并切换分支（新命令）
`git merge <分支名>`	合并指定分支到当前分支
`git branch -d <分支名>`	删除分支
`git branch -D <分支名>`	强制删除分支

5. 查看历史

命令	说明
`git log`	查看提交历史
`git log --oneline`	简洁显示提交历史
`git log --graph`	图形化显示分支历史

`git log --oneline --graph --all`	显示所有分支图形化历史
`git log -n 5`	显示最近5条提交
`git log --author="用户名"`	查看指定作者的提交
`git log --since="2025-01-01"`	查看指定日期后的提交
`git show <commit>`	查看某次提交的详细信息
`git blame <文件名>`	查看文件每行的修改信息

6. 撤销操作

命令	说明
`git restore <文件名>`	撤销工作区的修改
`git restore --staged <文件名>`	取消暂存文件
`git reset HEAD <文件名>`	取消暂存（旧命令）
`git reset --soft HEAD~1`	撤销最后一次提交，保留修改
`git reset --hard HEAD~1`	撤销最后一次提交，丢弃修改
`git revert <commit>`	创建新提交来撤销某次提交
`git clean -fd`	删除未跟踪的文件和目录

> **提示**
>
> `git reset --hard` 会永久删除未提交的修改，使用前请确认！

7. 远程操作

命令	说明
`git push origin <分支名>`	推送本地分支到远程
`git push -u origin <分支名>`	推送并关联远程分支
`git push origin --delete <分支名>`	删除远程分支
`git pull`	拉取远程更新并合并
`git pull --rebase`	拉取并使用rebase合并
`git fetch`	获取远程更新（不合并）

`git fetch --all`	获取所有远程更新

8. 标签操作

命令	说明
`git tag`	列出所有标签
`git tag <标签名>`	创建轻量标签
`git tag -a <标签名> -m "说明"`	创建附注标签
`git push origin <标签名>`	推送标签到远程
`git push origin --tags`	推送所有标签
`git tag -d <标签名>`	删除本地标签

9. 储藏操作 (Stash)

命令	说明
`git stash`	储藏当前修改
`git stash push -m "说明"`	储藏并添加说明
`git stash list`	查看储藏列表
`git stash pop`	应用并删除最新储藏
`git stash apply`	应用但不删除储藏
`git stash drop`	删除最新储藏
`git stash clear`	清除所有储藏

10. 高级操作

10.1 变基 (Rebase)

命令	说明
`git rebase <分支名>`	将当前分支变基到指定分支
`git rebase -i HEAD~3`	交互式变基最近3次提交
`git rebase --continue`	继续变基
`git rebase --abort`	取消变基

10.2 Cherry-pick

命令	说明
`git cherry-pick <commit>`	将指定提交应用到当前分支
`git cherry-pick -x <commit>`	保留原始提交信息

11. 常用组合命令

常用工作流

```
# 1. 日常开发工作流
git pull origin main           # 拉取最新代码
git checkout -b feature-x      # 创建功能分支
# ... 编写代码 ...
git add .                      # 添加修改
git commit -m "添加功能X"      # 提交修改
git push origin feature-x      # 推送到远程
# ... 创建 Pull Request ...

# 2. 撤销误操作
git reflog                     # 查看操作记录
git reset --hard <commit>      # 回退到指定版本

# 3. 整理提交历史
git rebase -i HEAD~5           # 整理最近5次提交
# 使用 squash 合并提交，reword 修改提交信息
```

12. .gitignore 模板

```
# 编译输出
*.exe
*.dll
*.so
*.class
target/
build/
dist/

# IDE 配置
.idea/
.vscode/
*.iml
*.swp
*.swo
*~

# 依赖目录
node_modules/
vendor/

# 日志和临时文件
*.log
*.tmp
.DS_Store
Thumbs.db

# 环境配置文件（包含敏感信息）
.env
.env.local
config.local.yml
```

13. 快捷别名配置

在 `~/.gitconfig` 中添加别名，提高操作效率：

```
[alias]
    st = status
    co = checkout
    br = branch
    ci = commit
    lg = log --oneline --graph --decorate --all
    last = log -1 HEAD
    unstage = restore --staged
    undo = reset HEAD~1 --mixed
```

14. 文件状态说明

状态	说明
`Untracked`	未跟踪，Git未管理的新文件
`Modified`	已修改，文件被修改但未暂存
`Staged`	已暂存，修改已添加到暂存区
`Committed`	已提交，修改已保存到本地仓库