# Git分布式版本控制

git的使用场景：备份、代码还原、协同开发、追溯问题代码的编写人和编写时间。

# Git初始化配置：

设置用户信息：

git config --global user.name “luqing”

git config --global user.email [“luqing@qq.com”](mailto:\“luqing@qq.com\”)

查看配置信息：

git config --global user.name

git config --global user.email

有些常用的指令参数非常多，每次都要输入好多参数，我们可以使用别名。

在用户目录下创建.bashrc文件：

touch ~/.bashrc

在文件中输入的内容如下：

#用于输出git提交日志

alias git-log='git log --pretty=oneline --all --graph --abbrev-commit'

#用于输出当前目录所有文件及基本信息

alias ll='ls -al'

查看提交记录：

git log [option]

·options

--all 显示所有分支

--pretty=oneline 将提交信息显示为一行

--abbrev-commit 使输出的commitId更简短

--graph 以图的形式显示

# 2、Git常用指令

·git init 初始化当前目录为一个仓库

·git add . 将所有修改添加到暂存区（.代表将所有修改添加，也可以替换 成单个文件）

·git commit -m ‘注释内容’ 将暂存区中的内容提交到本地仓库

·git status 查看修改状态

·git reset --hard commitID 版本切换，也可以说是版本回退，commitID可以使用

git-log 或 git log指令查看

·git reflog 这个指令可以看到已经删除的提交记录

·git branch 查看本地分支

·git branch 分支名 创建本地分支

·git checkout 分支名 切换分支

·git checkout -b 分支名 创建并切换分支

·git merge 分支名称 合并分支

·git branch -d 分支名称 删除分支时，需要做各种检查（不能删除当前 分支）

·git branch -D 分支名称 不做任何检查，强制删除

·git remote add origin URL 添加远程仓库git remote add <远端名称> <仓 库路径>

远端名称，默认是origin 取决于远端服务器 设置

仓库路径，从远端服务器获取

·git remote 查看远程仓库

·git push --set-upstream origin master 推送到远端的同时并建立起和远端分支的关 联关系，这必须是本地分支与远端分支同名才 可，下一次直接git push即可

·git branch -vv 查看本地分支与远程分支的关联关系

·git clone <仓库路径> 从远端仓库克隆

·git fetch 将仓库里的更新都抓取到本地，不会进行合并

·git pull 将远端仓库的修改拉到本地并自动进行合并， 等同于fetch+merge

# 3、在远端创建一个分支

1）在本地创建并切换到新分支

·git checkout -b <分支名>

2）推送本地分支到远程仓库

·git push -u origin <分支名> #-u表示设置上游跟踪，也就是将本地分支与远程分支关 联起来（后续可直接用git push/pull）

3）验证分支是否创建成功

·git branch -r #查看运程分支列表