git代码的协同管理工具

代码管理工具作用

\*防止代码丢失，做备份

\*代码版本的管理，可以进行多个节点的备份，在多个版本之间跳跃。

\*可以方便的将代码在多人之间进行共享传输

\*多人开发时有各种模式可以方便代码管理

什么是git

是一个开源的分布式版本控制系统，可用于高效的管理大小项目。

git分布式和svn集中式

分布式：每个节点都保存完成的代码，没有明确 的中央服务器，节点之间项目推送下载代码完成代码共享代码。

集中式：代码集中管理，每次完成的代码上传到中央管理器，然后在统一从中央管理器下载代码使用。

git特点

\*git可以管理各种文件，特别是代码项目，躲在\*nix系统中使用

\*是分布式管理，不同于集中式，这是git和svn的核心区别

\*git可以更好地支持分支，方便多人协同工作

\*git分布式代码更安全，有全球唯一的commit版本号

\*git是开源的系统

\*使用git可以脱网工作，且数据传输速度较快

git安装

linux

sudo apt-get install git

windows:ms

git配置命令

git config

配置级别1.系统中所有的用户都可以使用该配置

命令：git config --system

配置文件：/etc/gitconfig

2.当前用户可使用该配置

命令：git config –global

配置文件：~/.gitconfig

3.当前项目可以使用改配置

命令：git config

配置文件：project/.git.config

配置内容;

1. 配置用户名

e.g.配置用户名为Tedu

sudo git config --system user.name Tedu

1. 配置用户邮箱

e.g.配置邮箱

git config --global user.email [xxxxxxxxxxx@qq.com](mailto:xxxxxxxxxxx@qq.com)

1. 配置编译器

e.g. 配置编译器

git config core.editor sublime

1. 看配置信息

git config --list

git基本命令

初始化仓库

git init

\*在某个目录下初始化仓库后会自动产生。git目录。该目录下工作的所有文档即可以使用git进行管理

查看分支状态：

git status

\*默认工作分支为master可以通过创建新的分支来切换

文件提交

git add [file]

\*将文件提交到暂存区

\*提交内容可以是一个文件，多个文件用空格分开

\*如果是\*表示所有文件，，也可以是目录

删除暂存区某个文件提交记录

git rm --cached Readme.txt

文件同步到本地仓库

git commit -m 'pythonnet file'

\*同步时需要附加一些同步信息，在-m后添加

\*所有对工作区的修改如果想同步到本地仓库，都需要add---》commit

查看commit日志

git log

git log --pretty=online

一些工作区命令

查看本地文件和工作区差异

git diff file

从本地仓库恢复文件

git checkout Readme.txt

丢弃工作区的修改

git checkout -- file

本地仓库文件的移动和删除

移动

git mv <file> dir

删除

git rm <file>

* 用法和mv rm命令相同。 操作之后直接git commit –m’’即可实现工作区和本地仓库同步

版本控制命令

回到之前版本

git reset --hard HEAD^

\*head后的^数量决定了回到上几个版本

回到指定版本

git reset –hard commit\_id

\*使用commit前7位即可

去往更新的版本

1. 查看所有历史版本号

git reflog

1. 使用git reset去往指定版本

\*git reflog会有所有的操作记录，最新的操作时钟在最上边

标签管理

什么是标签：记载当前工作位置添加快照，保存工作状态，一般用于版本的迭代。

创建新的标签

git tag v1.0

\*默认在最新的commit\_ID处打标签

添加标签信息

git tag v1.0 –m ‘message’

指定某个commit\_id处打标签

git tag v0.9 commit\_id

查看标签：

git tag #列出当前标签

git show v1.0 #显示标签具体信息

删除标签

git tag -d v1.0

去往某个标签版本

git reset --hard v0.9

临时工作区操作

创建保存临时工作区

git stash

查看保存的工作区

git stash list

应用哪一个工作区

git stash apply stash@{1}

应用上一个工作区并且删除

git stash pop

删除工作区

git stash drop stash@{0} 删除某一个

git stash clear 删除所有

复习

match对象的使用

属性 pos endpos re

方法 group() groupdict()

flags的应用

re.I re.S re.M re.X

分支操作 uixiang #############################################

什么是分支？

分支即每个人获取原有代码，在此基础上创建自己的工作环境，单独开发，不会影响其他分支的操作，开发完成后在同一合并到主线分支中。

创建分支的好处：安全，不影响其他人工作，自己控制进度

查看当前的分支

git branch

* 前面有\*号的分支表示当前正在工作的分支

创建分支

git branch [branch\_name]

切换工作分支

git checkout [branch\_name]

创建并切换到新的分支

git checkout -b [branch\_name]

分支合并

将某个分支合并到当前分支

git merge [branch]

\*合并过程中如果没有冲突则直接合并后当前分支即为干净的状态

\*如果产生冲突则需要人为选择然后在进行add commit等操作

\*在创建分支前尽量保证当前分支是干净点以减少冲突的发生

删除分支

git branch -d [branch\_name]

强制删除没有合并的分支

git branch -D[branch\_name]

远程仓库

远程仓库“远程主机上的仓库。实际上git是分布式的，每一台主机的git结构都相似，只是把其他主机的git仓库叫做远程而已”

创建共享仓库

1. 创建文件夹

mkdir gitrepo

1. 设置文件夹属主

chown tarena:tarena gitrepo

1. 将该文件夹设置为可共享的git仓库

cd gitrepo

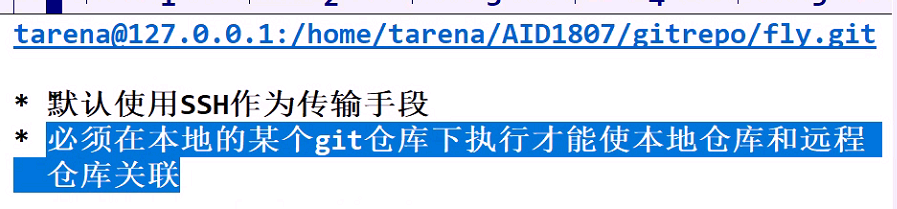
git init --bare tianba.git

1. 设置本地仓库属主

chown -R tarena:tarena tianba.git

连接远程仓库

\*默认使用ssh作为传输手段

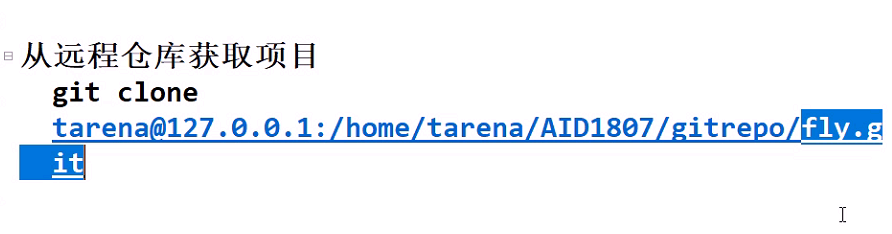


删除远程主机名字

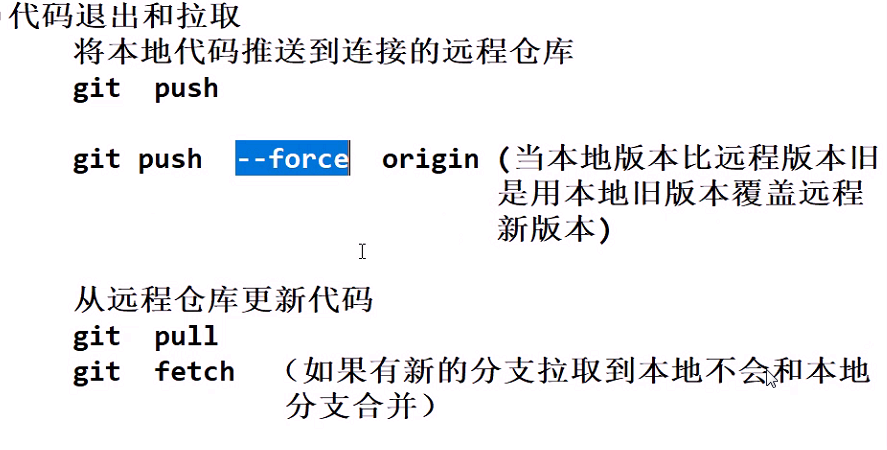
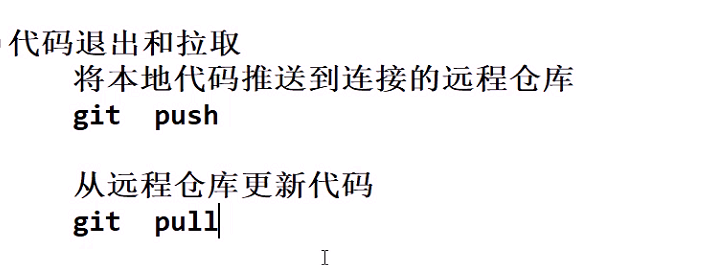
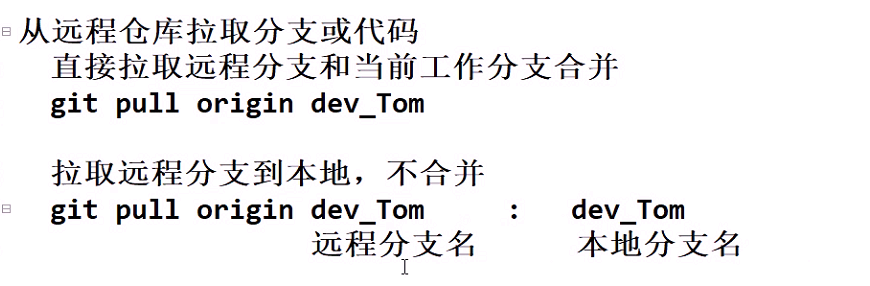
Git remote rm [origin]

将本地分支推送到远程

git push -u origin master



\*在第一次向远程仓库推送时需要加-u选项，以后就不需要了



GitHub

GitHub是一个开源项目社区网站。拥有全球最多的的开源项目。开发者可以注册这个网站建立自己的GitHub仓库，然后就可以本地通过Git像操作远程仓库一样操作GitHub仓库

Git是GitHub唯一指定的代码管理工具

添加ssh秘钥

1. 在本地主机生成ssh密钥对

ssh-keygen

\*默认秘钥对存放在~/.ssh/下

\*生成过程会提示设置密码，如果直接回车则表示不设置密码

2.进入~/.ssh目录赋值id\_rsa.pub公钥内容

3.登录GitHub账号

右上角投下下拉菜单---》srttings -->左侧ssh and gpgkeys --->new ssh key -->填写title，将复制内容加入key文本框。点击add。。

创建新的GitHub仓库

1.右上角+下拉菜单--》new repository --》填写参考名和基本描述，根据情况选择是否添加readme等内容，选择共有还是私有--》点击创建