**Git学习笔记**

**- 分布式版本系统**

1、创建版本库

(1) mkdir learngit

cd learngit

1. git init

-> 将这个目录变成git可以管理的仓库

-> 在当前目录下就会多一个隐藏目录 .git , 可以用 ls -ah命令查看

(3) 在learngit目录下创建文件: readme.txt

(4) 用 git add 告诉Git，把文件添加到仓库

git add readme.txt

(5) 用 git commit 告诉Git，把文件提交到仓库

git commit -m "wrote a readme file"

注：可以git add多个文件后，一次提交

2、对已经提交的文件进行修改并提交

git status: 随时掌握工作区的状态

git diff: 可以查看修改内容

-- 示例：git diff readme.txt

-- 用git diff HEAD -- readme.txt 可以产看工作区和版本库里面最新版

本的区别

还是使用 git add / git commit进行添加、提交

3、查看历史记录

git log: 显示从最近到最远的提交日志

git log --pretty=oneline: 缩少显示的从最近到最远的提交日志的信息

注：每提交一个新版本，实际上Git就会把他们自动串成一条时间线

4、版本回退

注：用HEAD表示当前版本，上一个版本就是HEAD^，上上一个版本就是 HEAD^^ ... 可以写成HEAD~100 (Git底层有志向当前版本的HEAD指针)

git reset: 用来将当前版本回退到上一个版本

- git reset --hard HEAD^

若版本回退后后悔了，只要知道那个版本的commit id(利用git commit --pretty=oneline可以看到)就可以用：

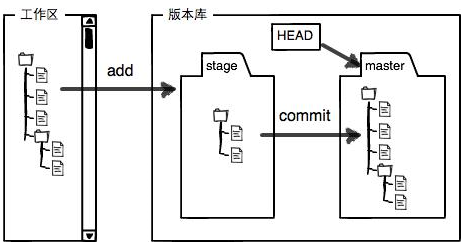
git reset --hard 3628164(commit id)

git reflog: 用来记录你的每一次命令

5、工作区和暂存区

工作区：就是在电脑上能看到的目录 (eg: learngit文件夹就是一个工作区)

版本库：工作区有一个隐藏的目录.git，就是Git的版本库



Git的版本库中存了很多东西，其中最重要的就是称为stage的暂存区，还

有Git自动创建的第一个分支master，以及指向master的指针HEAD

注：git add 就是把文件修改添加到暂存区

git commit提交更改，就是把暂存区中的内容提交到当前分支

**注：创建Git版本库是，Git就会自动创建唯一一个master分支，git commit**

**就是往master分支上提交更改(git版本库 <--> master分支)**

1. 撤销修改

git checkout -- file：可以**丢弃工作区的修改**

示例：

命令git checkout -- readme.txt

-- 把readme.txt文件在工作区的修改全部撤销

这里有两种情况：

1. 一种是readme.txt自修改后还没有被放到暂存区，现在，撤销修改就回 到和版本库一模一样的状态；
2. 一种是readme.txt已经添加到暂存区后，又作了修改，现在，撤销修改 就回到添加到暂存区后的状态。

**总之，就是让这个文件回到最近一次git commit或git add时的状态**

若修改后已经git add到暂存区，如何撤退？

1. 用git reset HEAD File可以把暂存区的修改撤销掉，重新放回工作区
2. 用 git checkout -- readme.txt
3. 删除文件

git rm file + git commit : 用来删除从版本库中删除该文件

若删错了(在命令行使用rm 删除文件)：

git checkout -- file： 将误删的文件恢复到最新版本

- 问题：可能会丢失最近一次提交后所修改的内容

1. 远程仓库

Git本地仓库和GitHub仓库之间的传输是通过SSH加密的

1. 关联远程库(EG:github上的learngit)

git remote add origin git@github.com:dujiaojiao/learngit.git

github用户名 要关联的仓库

1. 第一次推送master分支的所有内容

git push -u origin master

1. 此后本地每一次提交，只要有必要，就可以使用 git push origin master

推送最新修改

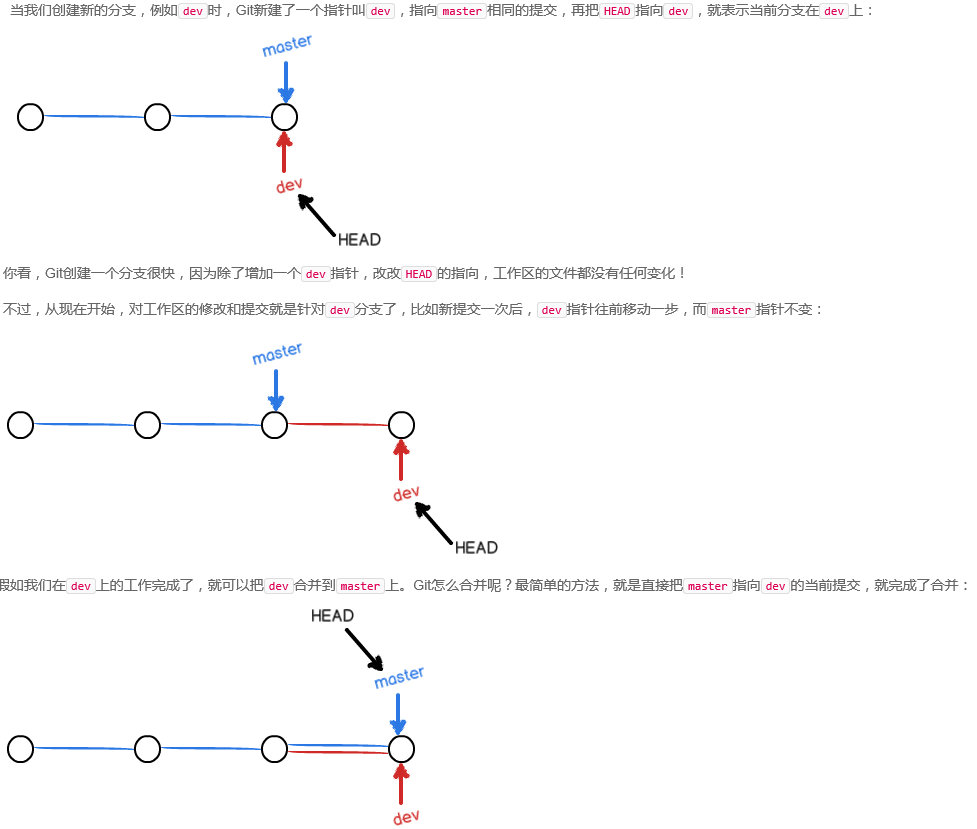
1. 从远程库克隆
2. 在github上创建一个新的仓库(EG: gitskills)，勾选 Initialize this

Repository with README,这样github会自动为我们创建一个README.md

1. 用命令git clone 克隆一个本地库

git clone git@github.com:dujiaojiao/gitskills.git

1. 创建并合并分支



1. 创建dev分支，然后切换到dev分支

git checkout -b dev

相当于

git branch dev

Git checkout dev

1. 用git branch命令查看当前分支，带\*的时当前分支
2. 在dev分支上对readme.txt进行修改并提交
3. 切换到master分支 git checkout master
4. 在master分支上查看readme.txt并未有在dev分支上修改的内容
5. 将dev分支的工作成果合并到master分支上,master就可见dev上的修改

git merge dev

1. 删除dev分支

git branch -d dev

git log --graph：可以看到分支合并图

git log --graph --pretty=oneline --abbrev-commit

注：当两个分支都对同一个文件进行了修改并提交，git无法自动合并分支，产

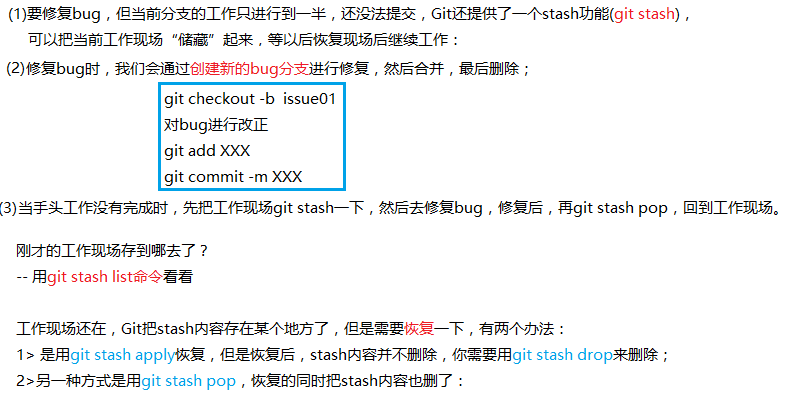
生冲突，需要手动解决并提交

Git用<<<<<<<，=======，>>>>>>>标记出不同分支的内容

注：合并分支时，加上--no-ff参数就可以用普通模式合并，合并后的历史有分 支，能看出来曾经做过合并，而fast forward合并就看不出来曾经做过合 并。

git merge --no-ff -m "merge with no-ff" dev

1. bug分支



12、Feature分支

开发一个新feature，最好新建一个分支；

如果**要丢弃一个没有被合并过的分支**，可以通过git branch -D <name>强行删除。



