2018-10-24

1、创建初始账户

在git bash 中执行

git config –global user.name “XXXX”

git config –global user.emai [lj.bao@huahengweld.com](mailto:lj.bao@huahengweld.com)

2、创建仓库，目的 供git管理的目录

（1）直接将你的项目所在的目录变为git仓库

进入你的项目所在目录，在git bash中cd 到你的项目所在目录执行git init （执行完毕，生成.git 文件，表示成功），此后可以执行 git add file 及 git commit –m “注释，说明本次提交的内容或者其他等”

（2）创建空目录，设置为git仓库，然后在增加项目内容。

在git bash 中cd到某个目录，执行 mkdir testgit，生成一个testgit 的文件夹，再执行git init,

其他操作同上。

这里的仓库都是指在我本地的一个工作区。

3、区域概念

当你将某个目录git init 意味着这个目录变为你的工作区，此时如果工作有文件为add,执行git status 会发现该文件或者文件夹的目录是Untrack 状态。

当你git add file ,则意味着这个文件保存在了git 内在的暂存区，此时这个文件属于 unstage,等待commit。

当你git commit 后，此时git status 发现 会提示你没有可以去提交的文件，此时文件均属于stage状态，已经被提交到某个分支（后续讲分支的概念，现在默认提交到master的分支下）

4、修改程序，查看修改内容

git status 随时用来跟踪工作区的文件状态

git diff git status告诉你有文件被修改过，用git diff可以查看修改内容。

-------早上早起学习的感觉棒棒哒。------后续要坚持，早睡早起早上学习。git

5、git log 查看修改记录

git log 查看修改记录，时间由近及远。主要会提示你commit 时的备注



如果觉得显示的内容太多，眼花缭乱，可以 --pretty=oneline

2种写法 git log pretty=oneline git log –oneline





其中这里一大串16进制的数字表示‘commit id ’ 即所谓的版本号，与SVN不同没有使用1,2,3，主要是考虑到git的分布式属性及后续多人同在一个版本工作会出现冲突，因此就有了这么个sha1 计算出来的数字。

如果想反向排序，从一个文件一开始到最新的commit 状态

git log –revise –oneline



6、搜索日志

1> git log --author [yourname]

筛选出yourname 提交的所有日志，yourname可以保护通配符（\*） （疑问：每个电脑只能有一个name么，同一个电脑可以使用不用的name去提交等操作么？）

$ git log --author ljbao

$ git log --author lj\*

2> git log –grep [keywords]

从提交的关键字抓取匹配的commit项



3> git log –p –-[file]

根据文件名获取仅和这个文件有关的提交

***待验证该功能***

4> git log --oneline --before={} --after={}

查看某个时间阶段的commit记录， 前 before，since，后 until，after

git log --oneline --before={3.weeks.ago} --after={2018-10-24}

 

6、版本回退

git reset –hard commitid

这里的commitid 是指提交时生成的16进制的数，可以通过git log –oneline 去查看



此时版本已有最新的哈哈 回退到 验证回退功能

如果 出现上述 unlink of file 的提示，这说明你的文件现在处于打开状态，回退需要在关闭当前文件或者程序的状态。

git log --oneline 发现只能查到这个版本之前的commit记录，那要是想回退刚刚我们回退之前的版本怎么办呢？ 别担心，再git这里你可以吃到后悔药。git reflog



你可以查看到整个版本的全部信息，这个时候你在使用git reset --hard commitID,即可回到回退之前的版本



版本回退总结：

1、Head 就是当前的版本。git 允许我们在版本的历史之间来穿梭，git reset –hard commitid

2、git log –oneline 查看提交历史记录，以便于回退到哪个历史版本中。

3、git reflog 查看命了历史，以便于回到未来的某个版本。