目录

[**一、Git是什么？ 1**](#_Toc533705608)

[**二、SVN与Git的最主要区别？ 1**](#_Toc533705609)

[**三、如何在WINDOWS上部署Git? 1**](#_Toc533705610)

[**四、如何操作？ 2**](#_Toc533705611)

[**1.创建版本库 2**](#_Toc533705612)

[**2.版本回退 2**](#_Toc533705613)

[**3.理解工作区与暂存区的区别 2**](#_Toc533705614)

[**4.Git撤销修改和删除文件操作 2**](#_Toc533705615)

[**5.远程仓库 3**](#_Toc533705616)

[**6.分支与合作 3**](#_Toc533705617)

[**7.分支管理策略 4**](#_Toc533705618)

**Git Bash使 用 详 细 教 程**

Author：[涂根华](http://www.cnblogs.com/tugenhua0707/p/4050072.html" \t "_blank)

# Git是什么？

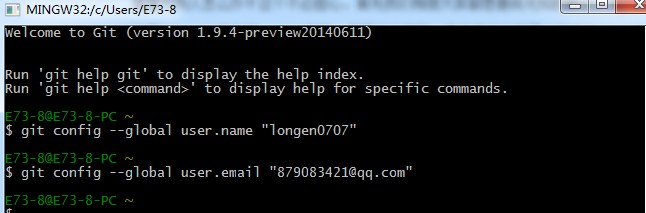
Git是先进的分布式版本控制系统。

# SVN与Git的最主要区别？

SVN是集中式版本控制系统，版本库是集中放在中央服务器的，而干活的时候，用的都是自己的电脑，所以首先要从中央服务器哪里得到最新的版本，然后干活，干完后，需要把自己做完的活推送到中央服务器。集中式版本控制系统是必须联网才能工作，如果在局域网还可以，带宽够大，速度够快，如果在互联网下，如果网速慢的话，就纳闷了。

　　Git是分布式版本控制系统，那么它就没有中央服务器的，每个人的电脑就是一个完整的版本库，这样，工作的时候就不需要联网了，因为版本都是在自己的电脑上。既然每个人的电脑都有一个完整的版本库，那多个人如何协作呢？比如说自己在电脑上改了文件A，其他人也在电脑上改了文件A，这时，你们两之间只需把各自的修改推送给对方，就可以互相看到对方的修改了。

# 如何在WINDOWS上部署Git?



因为Git是分布式版本控制系统，所以需要填写用户名和邮箱作为一个标识。

注意：git config  --global 参数，有了这个参数，表示你这台机器上所有的Git仓库都会使用这个配置，当然你也可以对某个仓库指定的不同的用户名和邮箱。

# 四、如何操作？

## 1.创建版本库

1. $ cd D: //进入D盘
2. $ mkdir testgit //创建项目仓库文件夹（mkdir）
3. $ cd testgit //进入项目仓库文件夹
4. $ pwd //pwd显示当前的目录
5. $ git init //初始化仓库（git init把这个目录变成git可以管理的仓库）

//执行后，testgit目录下会多出一个.git目录，该目录是Git来跟踪管理版本的。

1. $ git add readme.txt //将readme.txt文件添加到暂存区
2. $ git commit -m “readme.txt文件提交” //告诉Git，把文件提交到仓库
3. $ git status //查看git当前状态
4. $ git diff readme.txt //diff指令查看文件改了什么内容

## 2.版本回退

1. $ git log //log指令显示从最近到最远的日志
2. $ git log --pretty=oneline //简化显示日志信息
3. $ git reset --hard HEAD^ //版本会操作，回退到上上个版本，HEAD^改成HEAD^^
4. $ git reset –hard HEAD~100 //回退到前100个版本
5. $ git reflog //获取仓库所有版本号
6. $ git reset –hard 版本号 //回退到某个版本号的版本

## 3.理解工作区与暂存区的区别

　工作区：就是你在电脑上看到的目录，比如目录下testgit里的文件(.git隐藏目录版本库除外)。或者以后需要再新建的目录文件等等都属于工作区范畴。

版本库(Repository)：工作区有一个隐藏目录.git,这个不属于工作区，这是版本库。其中版本库里面存了很多东西，其中最重要的就是stage(暂存区)，还有Git为我们自动创建了第一个分支master,以及指向master的一个指针HEAD。

## 4.Git撤销修改和删除文件操作

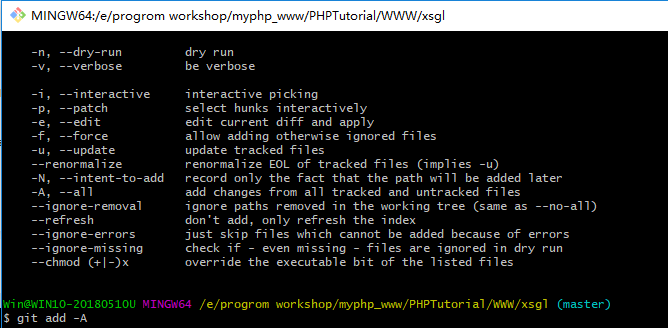
1. $ git checkout --readme.txt //把readme.txt文件在工作区做的修改全部撤销

// readme.txt自动修改后，还没有放到暂存区，使用 撤销修改就回到和版本库一模一样的状态。

//另外一种是readme.txt已经放入暂存区了，接着又作了修改，撤销修改就回到添加暂存区后的状态。

注意：命令git checkout -- readme.txt 中的 -- 很重要，如果没有 -- 的话，那么命令变成创建分支了。

1. $ rm b.txt //删除文件b.txt
2. $ git checkout – b.txt //回复b.txt文件
3. $ git add -A //-A提交所有变化 -u提交被修改和被删除文件，不包括新文件
4. $ git commit -a -m “delete a useless file” //提交所有新文件
5. $ git push //上传更新到git版本库



## 5.远程仓库

1. $ ssh-keygen -t rsa -C [youremail@example.com](mailto:youremail@example.com) //创建SSH Key
2. $ git remote add origin <https://github.com/wsderss/xsgl.git> //设置联系的远程仓库
3. $ ssh -T [git@github.com](mailto:git@github.com) Attempts to ssh to github //测试ssh连接
4. $ git remote -v //查看当前的remote url
5. $ git push -u origin master //把本地仓库分支master内容推送到源仓库去并关联
6. $ git remote set-url origin <https://github.com/wsderss/xsgl.git> //设置源仓库地址
7. $ git pull –rebase origin master //本地与源仓库代码合并 注：pull=fetch+merge
8. $ git push origin master //把本地仓库分支master内容推送到源仓库去
9. $ git clone <https://github.com/wsderss/xsgl.git>

## 6.分支与合作

1. $ git checkout -b BaseFramework //创建并切换分支

//git checkout命令加上 -b表示创建并且换，相当于一下两命令

1. $ git branch BaseFramework //创建名为BF的分支
2. $ git checkout BaseFramework //切换到BF分支
3. $ git checkout master //切换到主分支master
4. $ git merge BaseFramework //合并BF分支上的内容到master上（需切换到master执行）
5. $ git branch -d BaseFramework //删除BF分支
6. $ git branch //查看分支，会列出所有分支,当前分支前会加\*号

## 7.分支管理策略

通常合并分支时，git一般使用”Fast forward”模式，在这种模式下，删除分支后，会丢掉分支信息，现在我们来使用带参数 –no-ff来禁用”Fast forward”模式。首先我们来做demo演示下： $ git merge –no-ff -m “注释” BaseFramework

1. $ git log –graph –pretty==oneline –abbrev-commit

分支策略：首先master主分支应该是非常稳定的，也就是用来发布新版本，一般情况下不允许在上面干活，干活一般情况下在新建的dev分支上干活，干完后，比如上要发布，或者说dev分支代码稳定后可以合并到主分支master上来。

1. $ git stash //将当前的工作现场隐藏起来
2. $ git stash list //查看隐藏前的工作现场
3. $ git stash apply //恢复隐藏前工作现场，但不删除stash内容
4. $ git stash drop //删除隐藏的工作现场
5. $ git stash pop //恢复现场的同时删除stash内容
6. $ git push orgin BaseFramework //推送BF分支到远程中
7. $ git checkout -b BaseFramework origin/BF //创建远程origin的BF分支到本地来
8. $ git branch –set-upstream BaseFramework origin/BaseFramework
9. $ git pull //合并

1. 首先，可以试图用git push origin branch-name推送自己的修改.

2. 如果推送失败，则因为远程分支比你的本地更新早，需要先用git pull试图合并。

3. 如果合并有冲突，则需要解决冲突，并在本地提交。再用git push origin branch-name推送。

　　Git基本常用命令如下：

　　mkdir： XX (创建一个空目录 XX指目录名)

　　pwd： 显示当前目录的路径。

　　git init 把当前的目录变成可以管理的git仓库，生成隐藏.git文件。

　　git add XX 把xx文件添加到暂存区去。

　　git commit –m “XX” 提交文件 –m 后面的是注释。

　　git status 查看仓库状态

　　git diff XX 查看XX文件修改了那些内容

　　git log 查看历史记录

　　git reset --hard HEAD^ 或者 git reset --hard HEAD~ 回退到上一个版本

　　(如果想回退到100个版本，使用git reset –hard HEAD~100 )

　　cat XX 查看XX文件内容

　　git reflog 查看历史记录的版本号id

　　git checkout -- XX 把XX文件在工作区的修改全部撤销。

　　git rm XX 删除XX文件

　　git remote add origin https://github.com/tugenhua0707/testgit 关联一个远程库

　　git push –u(第一次要用-u 以后不需要) origin master 把当前master分支推送到远程库

　　git clone https://github.com/tugenhua0707/testgit 从远程库中克隆

　　git checkout –b dev 创建dev分支 并切换到dev分支上

　　git branch 查看当前所有的分支

　　git checkout master 切换回master分支

　　git merge dev 在当前的分支上合并dev分支

　　git branch –d dev 删除dev分支

　　git branch name 创建分支

　　git stash 把当前的工作隐藏起来 等以后恢复现场后继续工作

　　git stash list 查看所有被隐藏的文件列表

　　git stash apply 恢复被隐藏的文件，但是内容不删除

　　git stash drop 删除文件

　　git stash pop 恢复文件的同时 也删除文件

　　git remote 查看远程库的信息

　　git remote –v 查看远程库的详细信息

git push origin master Git会把master分支推送到远程库对应的远程分支上

git rebase --abort  想要放弃前rebase操作用

git rebase –continu 冲突已经解决先add冲突文件