# 1. 安装Git

安装省略，安装完后

git config --global user.name "Your Name"git config --global user.email "email@example.com"

因为Git是分布式版本控制系统，所以，每个机器都必须自报家门：你的名字和Email地址。

# 2. 创建版本库

非常简单，首先，选择一个合适的地方，创建一个空目录

初始化一个Git仓库，使用git init命令。

添加文件到Git仓库，分两步：

* 第一步，使用命令git add <file>，注意，可反复多次使用，添加多个文件；
* 第二步，使用命令git commit，完成。

# 3. 版本库状态

git status命令可以让我们时刻掌握仓库当前的状态

git diff顾名思义就是查看difference，显示的格式正是Unix通用的diff格式。

* 要随时掌握工作区的状态，使用git status命令。
* 如果git status告诉你有文件被修改过，用git diff可以查看修改内容。
* git diff 显示当前工作区的文件和stage区文件的差异
* git diff --staged 显示stage区和HEAD的文件的差异
* git diff HEAD 显示工作区和上次递交文件的差异

# 4. 回退

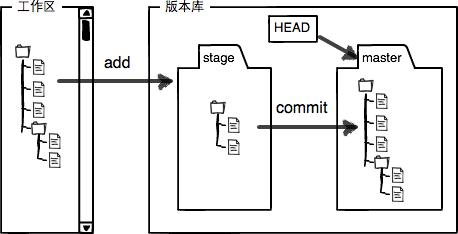
在Git中，用HEAD表示当前版本，也就是最新的提交3628164...882e1e0，上一个版本就是HEAD^，上上一个版本就是HEAD^^，

HEAD指向的版本就是当前版本，因此，Git允许我们在版本的历史之间穿梭，使用命令git reset --hard commit\_id。

穿梭前，用git log可以查看提交历史，以便确定要回退到哪个版本。

要重返未来，用git reflog查看命令历史，以便确定要回到未来的哪个版本。

# 5. 暂存区



git add 将文件加到暂存区

git commit 将文件加到工作区

暂存区是Git非常重要的概念，弄明白了暂存区，就弄明白了Git的很多操作到底干了什么。

现在，你又理解了Git是如何跟踪修改的，每次修改，如果不add到暂存区，那就不会加入到commit中。

# 6. 修改和删除

总结：

1) git rm+ git commit －m “remove file” 则会删除master里的内容；已经提交了不合适的修改到版本库时，想要撤销本次提交，即版本回退，不过前提是没有推送到远程库

删除已经提交到master：利用git reset －－hard HEAD^可以将删除的master从回收站恢复过来；

2) rm file+git add file：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想丢弃修改，分两步，第一步用命令git reset HEAD file，就回到了场景1，第二步按场景1操作。

删除提交到stage，但未提交到master：（1）利用git reset HEAD －－ file （与git reset HEAD file效果一样？）从master中拷贝到stage中；

（2）最后再用git checkout －－ file 从stage中拷贝到工作目录中。

3)rm file：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时，用命令git checkout -- file。

git checkout --其实是用版本库里的版本替换工作区的版本，无论工作区是修改还是删除，都可以“一键还原”。

删除未提交到stage

git checkout －－ file

rm file(类似create file)->git rm file(相当于git add file)->git commit file -m ""(和添加相同)

另外一个总结：

一、rm只删除工作区信息；

使用 git checkout -- file即可恢复；

二、git rm 删除工作区和stage信息；

1、使用git reset HEAD -- file即可恢复；

2、恢复步骤：

1）先使用git reset HeAD -- file从master中把已删除信息copy到到stage区；

2）再使用git checkout -- file从stage区把已删除信息copy到工作区。over！！！

3）这时你就看到删除的文件又回来啦。啦啦啦！

三、git commit －m “file”删除工作区，stage和master信息；

1、使用git reset --hard HEAD^即可恢复。

2、恢复步骤：

1）先使用git reset --hard HEAD^从master回收处恢复到master中；

2）再使用git reset HeAD -- file从master中把已删除信息copy到到stage区；

3）最后，使用git checkout -- file从stage区把已删除信息copy到工作区。

4）这时你就看到删除的文件又回来啦。啦啦啦！

# 7.远程仓库

7.1 关联远程仓库(先有本地库，后有远程库的时候，如何关联远程库)

你已经在本地创建了一个Git仓库后，又想在GitHub创建一个Git仓库，并且让这两个仓库进行远程同步，这样，GitHub上的仓库既可以作为备份，又可以让其他人通过该仓库来协作

首先，登陆GitHub，然后，在右上角找到“Create a new repo”按钮，创建一个新的仓库如learngit：

目前，在GitHub上的这个learngit仓库还是空的，GitHub告诉我们，可以从这个仓库克隆出新的仓库，也可以把一个已有的本地仓库与之关联，然后，把本地仓库的内容推送到GitHub仓库。

我们根据GitHub的提示，在本地的learngit仓库下运行命令：

 git remote add origin <https://github.com/Ritali/learngit.git>

或者 git remote add origin git@github.com:Ritali/learngit.git

（Git支持多种协议，默认的git://使用ssh，但也可以使用https等其他协议，但通过ssh支持的原生git协议速度最快）

下一步，就可以把本地库的所有内容推送到远程库上：

git push -u origin master

此后，每次本地提交后，只要有必要，就可以使用命令git push origin master推送最新修改；

7.2 从远程库克隆(先创建远程库，然后，从远程库克隆)

git clone [git@github.com](mailto:git@github.com):Ritali/gitskills.git