sourcetree 所在目录：C:\Users\hua\AppData\Local\Atlassian\SourceTree

linux命令:

pwd显示当前的目录

ll 展示下面的文件

cd ..上一级目录

创建目录: mkdir 目录名字

ls -a 显示此目录下的所有目录和文件，列出目录下的所有文件，包括以 . 开头的隐含文件。

cat 文件名 :查看文件内容

git命令:

1.进入要初始化的版本库目录内

2初始化版本库

git init

3添加文件到版本库，将文件从工作区添加到暂存区中

echo "git repo2" >> test.txt在当前仓库中创建test.txt文件并追加git repo2内容

git add test2.txt [文件名]

4提交，将文件从暂存区提交到版本区

git commit -m "提交的描述信息"

5 git status 查看仓库状态

6.查看本地分支：git branch

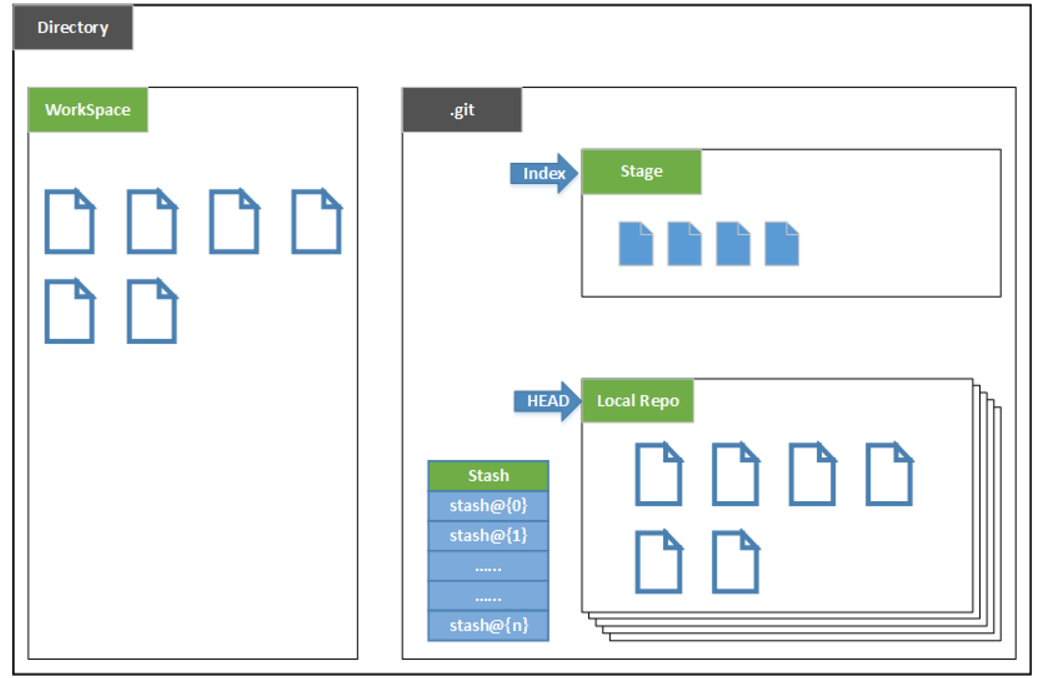
查看远程分支:git branch –r

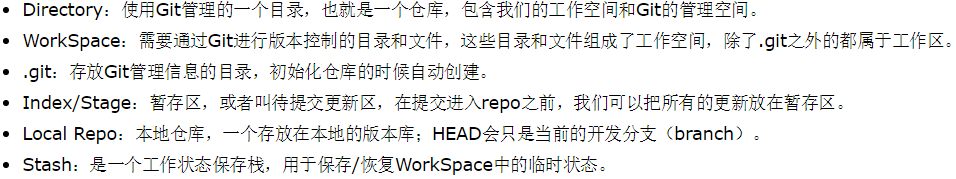
查看本地和远程的分支:git branch -a, \* 代表当前工作目录所代表的分支

7查看仓库状态

- git status

查看远程仓库的地址：



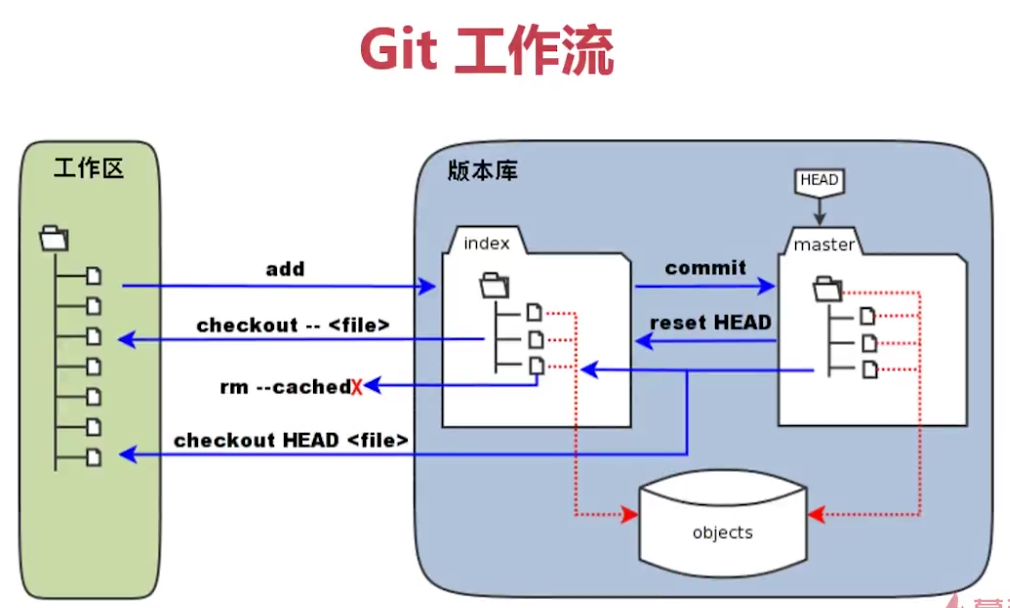




新版本的 Git 不再使用 branches 目录，description 文件仅供 GitWeb 程序使用，所以不用关心这些内容。config 文件包含了项目特有的配置选项，info 目录保存了一份不希望在 .gitignore 文件中管理的忽略模式 (ignored patterns) 的全局可执行文件。hooks 目录保存了客户端或服务端钩子脚本。

* objects 目录存储所有数据内容
* refs 目录存储指向数据 (分支) 的提交对象的指针,里面即有stash栈指针以及tag等
* HEAD 文件指向当前分支
* index 文件保存了暂存区域信息

git 工作流



暂存区（index区或者stage区）

工作流：工作区---->暂存区—>版本库

1 暂存区到工作区回滚： 将文件修改后添加到暂存区，要是回滚到以前的状态的话，在暂存区中丢去即可，然后回滚到修改前的操作

git reset HEAD bash\_demo.txt

* 这一步执行git status 会有 modify的提示

git checkout -- bash\_demo.txt(丢弃添加暂存区之前修改的内容)

2.将已经提交的内容回滚到第一次提交的状态

git log 查看提交的信息，并将第一次提交的版本号拷贝下来

git reset -- hard 1e78564b5a5d2e5ea3007dea34c63500b0c4c963[第二次提交的版本号]

3.将本地文件清除干净

git rm bash\_demo.txt

还需要将暂存区和仓库清空

git commit -m "delete bash\_demo.txt"

**git连接 github**

1.在github上生成ssh key

2.在终端中写入以下命令

3.ssh-keygen -t rsa -C "1558414400@qq.com"

4.在用户目录下有一个.ssh文件，执行命令后生成的ssh key在里面。

5.生成两个文件id\_rsa id\_rsa.pub

6.使用id\_rsa.pub的内容作为ssh key，添加到github中

7.然后使用ssh -T git@github.com 确认是否连接github通过。



使用github

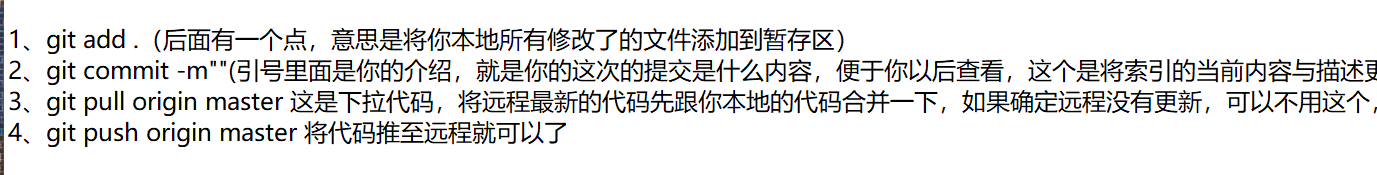
* 在github上新建一个仓库
* 将本地仓库和远程的github仓库进行关联起来

git remote add origin [git@github.com:shaoxiaohua/hua\_demo.git](mailto:git@github.com:shaoxiaohua/hua_demo.git)

* 在本地的仓库中，在本地仓库中添加了新的文件之后或更改了文件的内容之后，
* 使用 git add xxx[文件名]
* git commit –m “xxxx”
* git pull origin marster
* git push origin master：refs/for/master： 将本地的master分支推送到远程主机origin上的对应master分支， origin 是远程主机名，

git push -u origin master 如果当前分支与多个主机存在追踪关系，则可以使用 -u 参数指定一个默认主机，这样后面就可以不加任何参数使用git push，不带任何参数的git push，默认只推送当前分支。

## 简单的代码提交流程:



（1）git status //查看工作区代码相对于暂存区的差别，

（2）git add . // 将当前目录下修改的所有代码从工作区添加到暂存区 . 代表当前目录

（3）git commit -m “commit-message” //将暂存区的代码提交到本地版本库

（4）git push -u origin master // 将本地版本库推送到远程服务器，origin是远程主机，master表示是远程服务器上的master分支，分支名是可以修改的。

注:加了参数-u后，以后即可直接用git push 代替git push origin master

## 将项目clone 到本地:

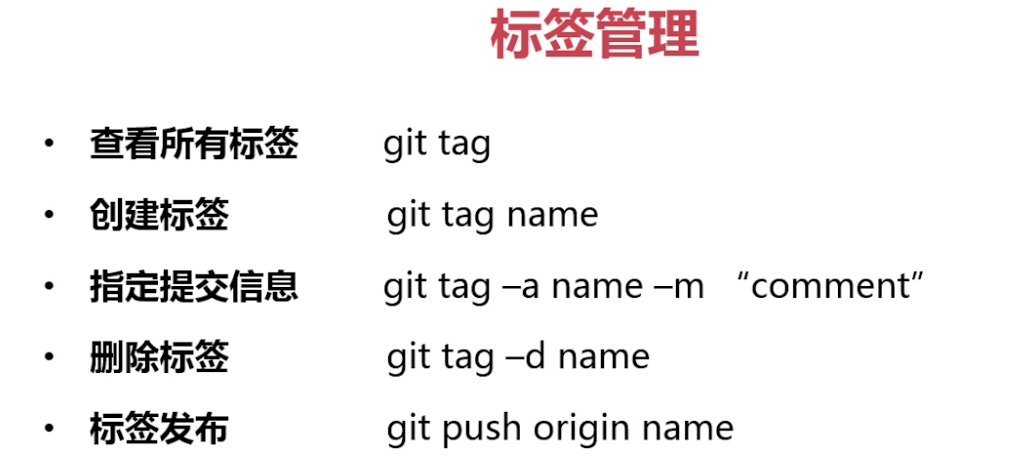
找一个不是git仓库的文件夹，然后使用clone [github上仓库的地址]

如：git clone [git@github.com:shaoxiaohua/remote\_repo.git](mailto:git@github.com:shaoxiaohua/remote_repo.git)

注意:在使用clone命令将项目clone到本地时，本地分支就已经和远程分支建立好关联了。

下次有更改的文件的时候，直接使用git push就可以了。

## 标签管理



获取标签

git tag

创建标签[本地]: git tag xxx[标签名]

git tag v1.0.1

git add .

git commit –m “”

将代码提交一次到远程仓库之后，并将创建好的标签也push到远程仓库

git push origin v1.0.1

当再一次更改了项目进行提交完成后，再创建新的标签并将其提交到远程仓库中。

git tag v2.0.1

git push origin v2.0.1

当项目进行回滚的时候可以回滚到上一个标签所对应的状态。

## 分支管理

git status 可以查看当前你所在的分支

创建新的分支：

git branch xxx[分支名]

查看当前所有的分支

$ git branch

feature\_x

\* master

\* 代表你在当前的的分支下

切换分支

git checkout feature\_x[切换到feature\_x分支上]

在新的分支上可以创建与主分支上相同的文件，并进行更改等操作。也可以执行一下命令：

git add .

git commit –m “xxx”

之后如果需要将其他分支的代码合并到master分支上。

1. 先切换到master分支上,使用git checkout master[从feature-x分支上切换到master分支上]
2. 合并分支，git merge feature\_x[合并分支]
3. 查看相应的文件，就会发现分支的代码合并到了主分支上。
4. 如果不需要其他的分支可以使用如下命令进行删除

git branch –d feature\_x[分支名]

## 总结

初始化:git init --🡪 git add --🡪git commit

远程仓库 git remote add(将本地仓库和远程仓库进行关联) --🡪 git pull--🡪git push-🡪git clone

分支管理 git branch --🡪git checkout 🡪 git merge

标签管理git tag🡪git push