## git介绍及文档说明

图文说明工作原型



Remote 远程仓库

Repository 本地仓库

Index 暂存区

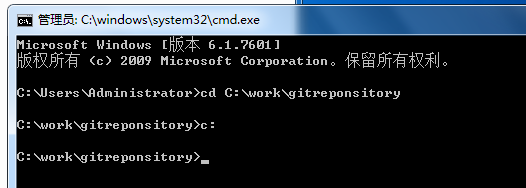
Workspace 工作区，也就是eclipse等编辑代码的地方

这里，暂时只介绍windows环境下从远程仓库拿文件，及各种本地操作，再提交到远程仓库，也不介绍安装git

## 初始化一个目录为本地仓库并拉取代码

1.1切换到目的目录，想作为本地仓库的目录

方式一，

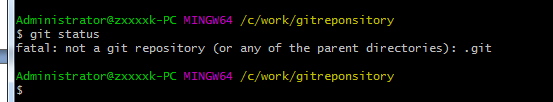


如果是其他盘符下的目录，在切换命令之后，需要追加 盘符的跟目录 d: 这样

方式二,打开目的目录,成功安装git后 在目标目录内空白处点击鼠标右键，会有如下命令

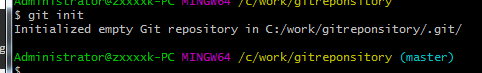


1.2 查看状态 git status

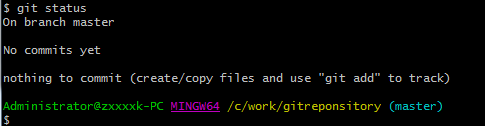


此时提示还不是一个git仓库

1.3初始化本地仓库 git init

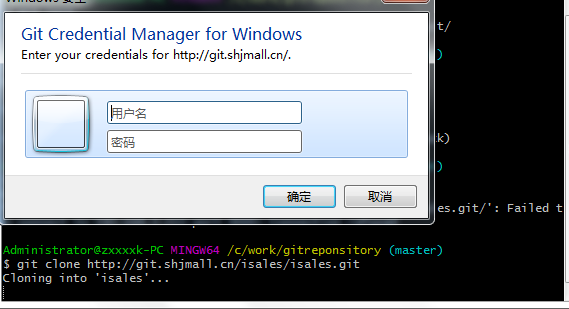


再次查看状态



可以看到已经提示在master分支上，还没有需要提交的内容，master是默认分支

1.4接着尝试拉取远程代码到本地仓库,看原型图，可以看到需要使用fetch 或者clone命令

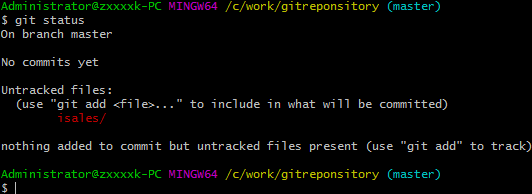


运行命令后，提示需要输入账号密码，这是在远程仓库端配置的，一般是创建远程仓库的人员提供

假如输入错了账号密码，就需要在windows控制面板里面的凭据管理先删除错误的凭证.



再次使用git status 查看发现，多了一个目录，也就是我们成功的复制了远程代码过来到本地仓库 master默认分支

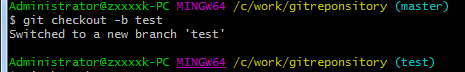


## 2、创建工作区作为开发分支

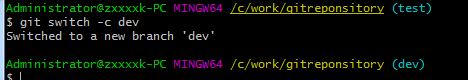
工作区就是一个分支，master是本地仓库里面默认的分支。目录来看工作区和本地仓库可以是同一目录

**2.1创建并切换到分支 test**

Git checkout –b test

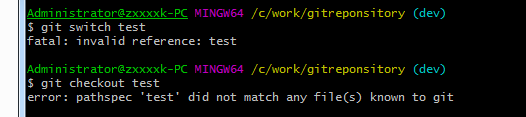


Git switch –c dev



两种命令都是创建分支并切换过去

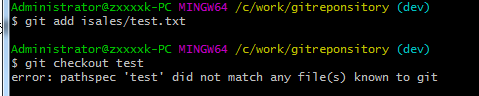
在dev下新建一个文件test.txt,然后直接切换分支



切换都是失败的，因为dev分支我们没有提交到暂存区或者提交到仓库

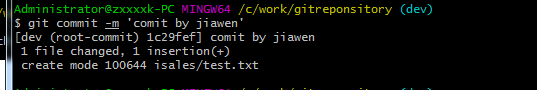
先尝试添加到暂存区

Git add file 将文件夹或文件提交到暂存区



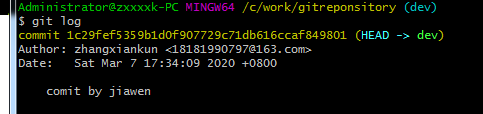
可以看到再切换还是不行，那再操作提交试试

Git commit –m ‘描述’ 提交暂存区修改到当前分支



查看提交日志

Git log

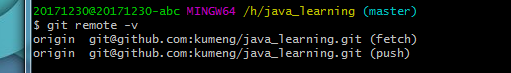


**git add & git commit**  
看到这里估计很多人会有疑问，我想要提交直接进行 commit 不就行了么，为什么先要再 add 一次呢？首先 git add 是先把改动添加到一个「暂存区」，你可以理解成是一个缓存区域，临时保存你的改动，而 git commit 才是最后真正的提交。这样做的好处就是防止误提交，当然也有办法把这两步合并成一步，不过后面再介绍，建议新手先按部就班的一步步来。

使用SSH 远程连接本地仓库和远程仓库,具体步骤，略，自行查找

**查看远程仓库应用**

git remote -v



怎么来的呢，

git remote add origin 从github页面复制你项目的SSh地址过来 git@github,com:xxxxx.git

**推送到远程端**

git push origin master

**对比本地仓库(master分支)和远程仓库(master分支)间的差异**

git diff --stat master origin/master

### retore 还原代码

git restore --source origin/master --staged --worktree .

### 关联远程分支

1. # 建立追踪关系，在现有分支与指定的远程分支之间

$ git branch --set-upstream [branch] [remote-branch]

### 基于指定分支新建一个新的目录并指定分支 git worktree add <新路径> -b <新分支名> <指定分支名> 将基于指定分支，新建一个 worktree 目录，新的分支名是指定的名称。

git worktree add ../../gitrun/bid -b dev dev\_l

Git worktree remove 目录名

### merge合并代码

git merge --no-ff 在每次合并都会产生一个新的合并记录; git merge 的话只有解决冲突的时候才会产生一个新的合并记录

比如，dev 合并到 master,需要如下两条命令

Git checkout master

git merge --no-ff dev

### reset三种模式区别和使用场景

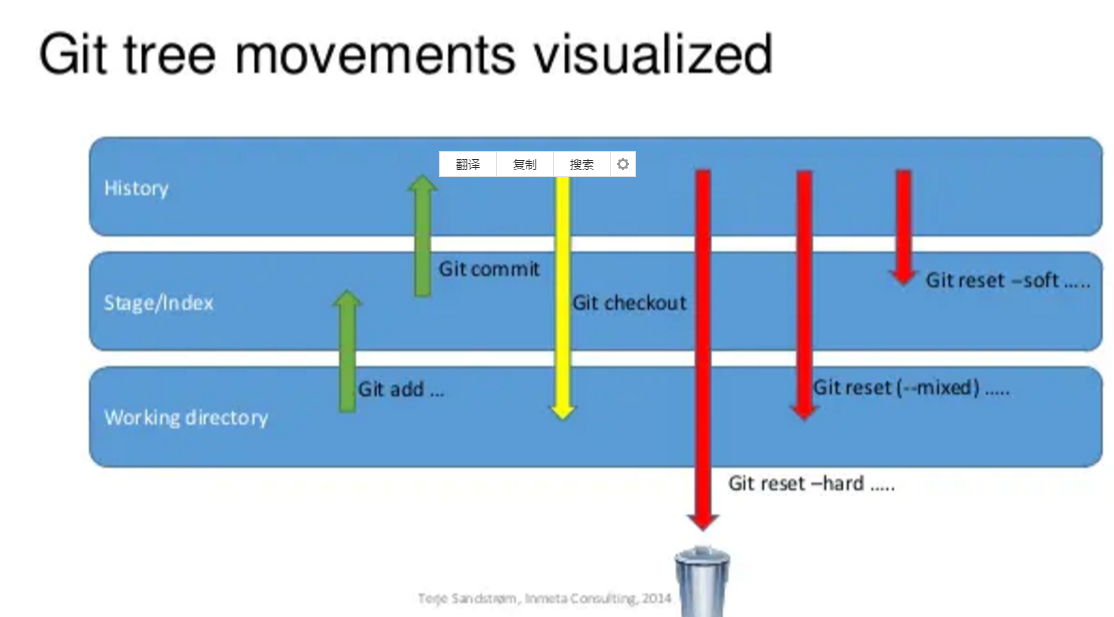
git reset它有三种模式，soft,mixed,hard

例子: git reset e3d6820cd7aee6a183c474e0972e0a5f7ed3b507

reset --hard 模式会撤回 Repository,Index,Working Tree中本次提交的全部修改，也就是完全回滚

reset -soft 只撤回Repository , 其他两个位置不变

reset -mixed 默认的模式，撤回Repository,Index ，不变的是Working Tree



先简述一下文件从工作地道仓库的条件流程:



1刚开始 working tree 、 index 与 repository(HEAD)里面的內容都是一致的



2,当工作地文件变化，即Working Tree的內容就会跟 index 及 repository(HEAD)的不一致



3，执行 add后如上图，只有repository(HEAD)的不一致



4,执行commit后，才会一致

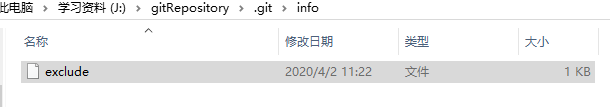
## Git RESET强制覆盖本地，同步远程分支

**git强制覆盖**：

    git reset --hard origin/master

## Git配置忽略文件

1，本地仓库的设置位置



只影响本地仓库

2，本地全局设置，针对本地所有仓库

设置忽略文件路径

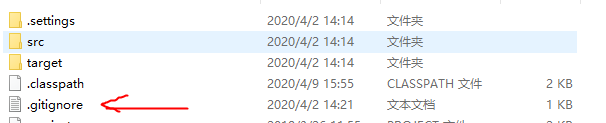
git config --global core.excludesfile ~/.gitignore

查看设置的文件路径

git config --global --get core.excludesfile



3，项目中也会有.gitignore文件设置单个项目的忽略



## Git配置忽略规则

#是注释符号

最后加(/)表示忽视目录

(!)表示取反

(\*)表示匹配任意个字符

[abc]表示匹配一个a或者b或c

(?)匹配任意一个字符

[0-9]匹配0到9的数字

示例:

/target/ 表示target目录

.idea

\*.iml

Out

/\*\*/build