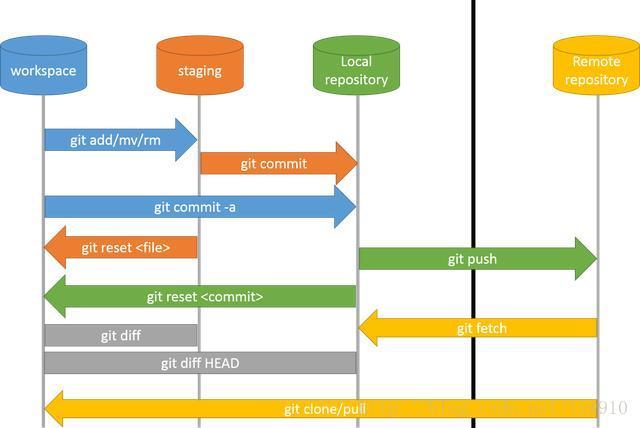
# git核心概念

## git总体流程图



* git从远程仓库clone项目后，默认为将远程仓库的名称称为：origin
* git默认分支可以关联多个远程仓库

# git基本操作

## git配置

# 列举所有配置

$ git config -l

# 为命令配置别名

$ git config --global alias.co checkout

$ git config --global alias.ci commit

$ git config --global alias.st status

$ git config --global alias.br branch

# 设置提交代码时的用户信息

$ git config [--global] user.name "[name]"

$ git config [--global] user.email "[email address]"

## 初始化

### 从git远程仓库克隆完成的项目

git clone 版本库

### 当前目录新建版本库

创建本地仓库

git init

本地仓库关联远程git仓库

git remote add origin git@远程仓库地址/learngit.git

## 增删文件

### 添加当前目录的所有文件到暂存区

$ git add .

### 添加指定文件到暂存区

$ git add <file1> <file2> ...

### 添加指定目录到暂存区，包括其子目录

$ git add <dir>

### 删除工作区文件，并且将这次删除放入暂存区

$ git rm [file1] [file2] ...

### 停止追踪指定文件，但该文件会保留在工作区

$ git rm --cached [file] #文件会保留在本地磁盘中

git rm -f [file] #文件会从本地磁盘删除（慎用）

### 改名文件，并且将这个改名放入暂存区

$ git mv [file-original] [file-renamed]

## 分支

### # 列出所有本地分支

$ git branch

输出内容中，当前分支会被标注\*

### # 列出所有本地分支和远程分支

$ git branch -a

### # 新建一个分支，但依然停留在当前分支

$ git branch [branch-name]

### # 新建一个分支，并切换到该分支

$ git checkout -b [new\_branch] [remote-branch]

以上命令等价于：

git branch new\_branch

git checkout new\_branch

### # 切换到指定分支，并更新工作区

$ git checkout [branch-name]

### # 合并指定分支到当前分支

$ git merge [branch]

### # 选择一个 commit，合并进当前分支

$ git cherry-pick [commit]

### # 删除本地分支，-D 参数强制删除分支

$ git branch -d [branch-name]

### # 删除远程分支

$ git push [remote] :[remote-branch]

### 本地远程仓库创建分支

#创建分支b1并切换至分支b1

git checkout -b b1

#将b1分支推送到远程

git push origin b1

# 关联本地分支和远程分支

git branch --set-upstream-to=origin/b1

## 查询

### 查看远程仓库信息

git remote –v

### 查看日志

* 查看版本提交日志，第一行为当前分支的版本

git log --pretty=oneline

* 查看执行的前后版本命令

Git reflog

### 查看分支合并图

git log --graph



### 查看多个版本之前的提交的文件列表

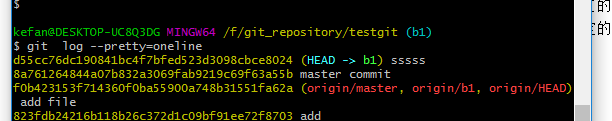
git diff-tree -r --name-only commitid1 commitid2

## 版本回退

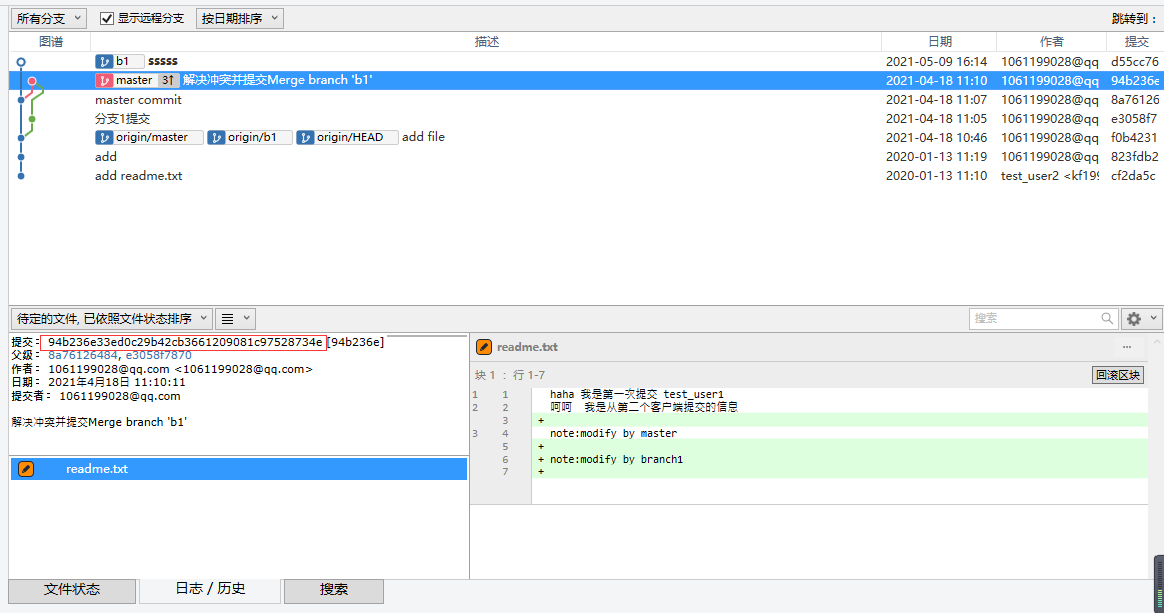
Git提供版本回退功能，假设你在开发新需求，因为某种原因，需要回退到以前分支的某个版本下继续修改，git中每次的提交称为一个commit.根据commit id可以回退到指定的版本。

1. 使用git log查看版本的提交历史

git log --pretty=oneline,从输出中可以看出，目前分支的最新版本备注为:ssss，8a761264844a07b832a3069fab9219c69f63a55b为前一个版本，在下列步骤中会从ssss回滚到master commit版本，再从master commit前进到ssss版本。



也可以使用source tree图形界面查看



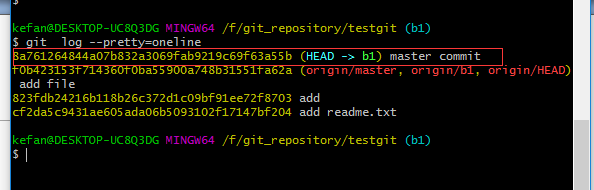
注意comit id，使用commit id回滚指定版本.

1. 使用git reset –hard回滚到指定的版本

git reset --hard 8a761264844a07b832a3069fab9219c69f63a55b

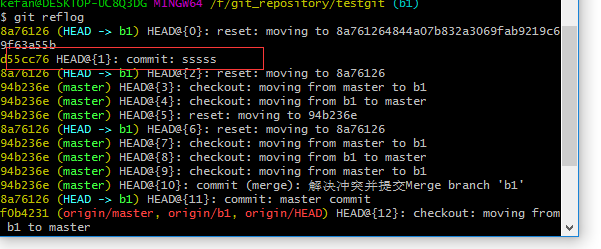
查看当前的所在的版本:

git log --pretty=oneline

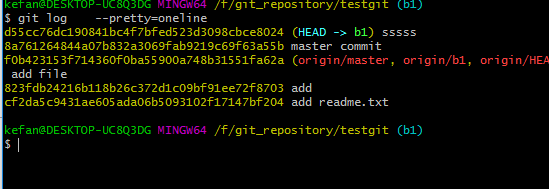


1. 使用git reflog可以查看最近的修改历史，前进版本信息

git reflog



git reset --hard d55cc76



# 分支管理

当我们从版本库克隆一个项目时，本地会默认创建一个master分支并且和远程仓库的master分支关联，远程仓库的关联可以成为:origin

## 创建合并分支

* 创建分支b1并切换到分支b1下

git checkout -b b1 #创建分支

git branch #查看分支，\*代表目前工作空间的分支

* 创建分支然后手动切换分支

git branch b1 #只创建分支不切换

git checkout b1 #切换至分支b1

* 查看和删除分支

git branch #查看分支

git branch –d <<分支名称>>

假如存在如下场景：

默认分支在master下，此时切换至b1分支，在b1分支下修改文件，添加至暂存区。并提交至本地仓库。然后切换回master分支，此时master分支看不到b1分支刚才修改的变动，需要使用git merge b1合并分支。

## 解决冲突

团队协作开发中，很多场景下git是可以自动合并的，但涉及到修改同一个文件时，git可能无法自动合并，需要手动合并后，再次提交。

假设存在如下场景：

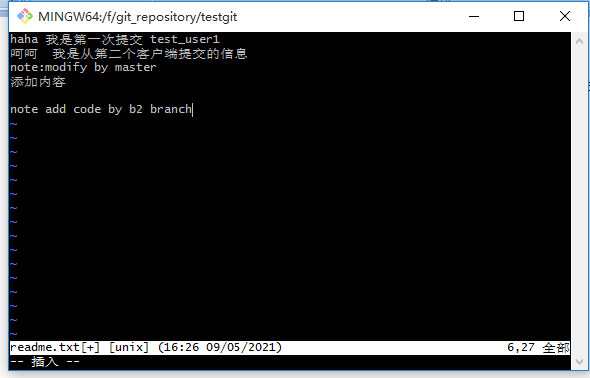
当前在分支b1下开发，需要创建一个新的分支b2，在b2上修复某一个bug，修复完成后，需要将修复的代码合并到b1,并在b1上继续往下开发。

1. 创建b2分支并切换到b2分支下

git branch b2

git checkout b2

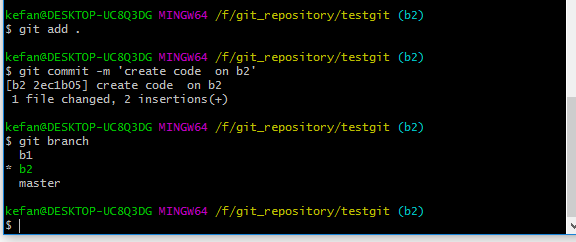
1. 在b2下修改文件



1. 保存并提交

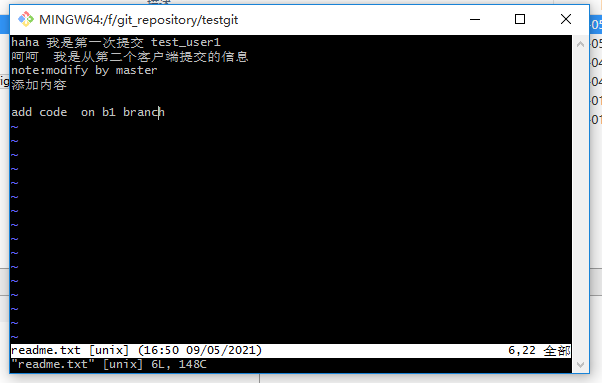
git add .

git commit -m 'create code on b2'



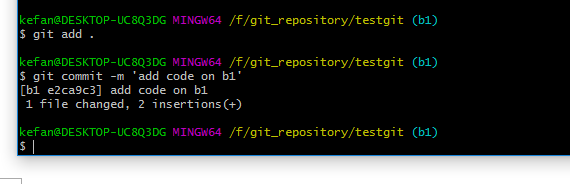
1. 切换到 b1分支下继续开发

git checkout b1



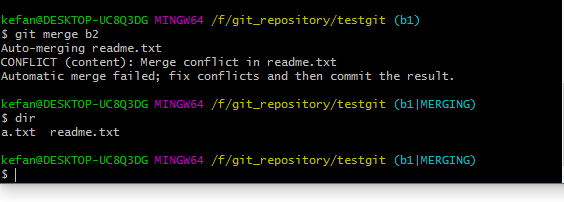
1. 在b1下保存并提交

git commit -m 'add code on b1'

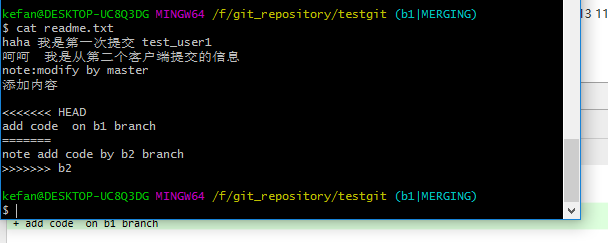


1. 将b2分支合并到b1分支

git merge b2



1. 使用git status查看冲突的文件



1. 修改文件后再次提交

git add .

git commit -m 'merge b2'



## 分支管理策略

## 团队协作

# 标签管理