# Git分支

比如当你正在专注开发一个功能时，突然有一个紧急的BUG需要你来修复，这个时候我们当然是希望在能够保存当前任务进度，再去修改这个BUG，等这个BUG修复完成后再继续我们的任务。又假如，你在写一篇论文。你已经写出了第一个版本，并提交审稿。后来，你获取了新的数据，你正在把你的数据添加到论文中。在这个过程中，审稿人让你改变你的论文的格式。**（然后，你说，滚！）**很明显，你不能把你正在更改的版本修改格式后提交给他们。你只想在以前版本基础之上对格式进行修改之后发给他们。

这种情况下，就得用到分支的思想。



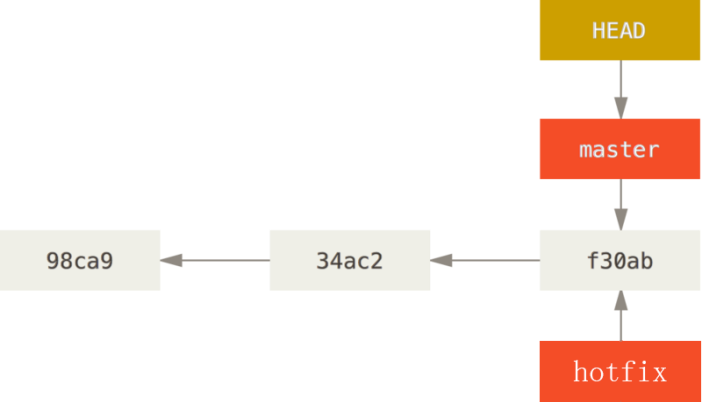
当我们在初始化仓库，第1次提交（commit）时，git会默认帮我们创建了一个master的分支，并且有指针（HEAD）指到了末端。

指针（HEAD）用来标明当前处于哪个分支的哪个版本，如上图指的处于master分支的最后1个版本。

## 创建分支

**git branch 分支名**

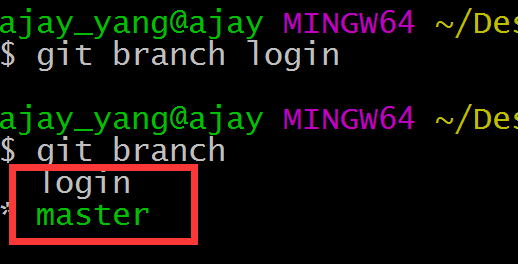
新的分支会在当前分支原有历史版本的结点上进行创建。



新建的子分支会继承父分支的所有提交历史。

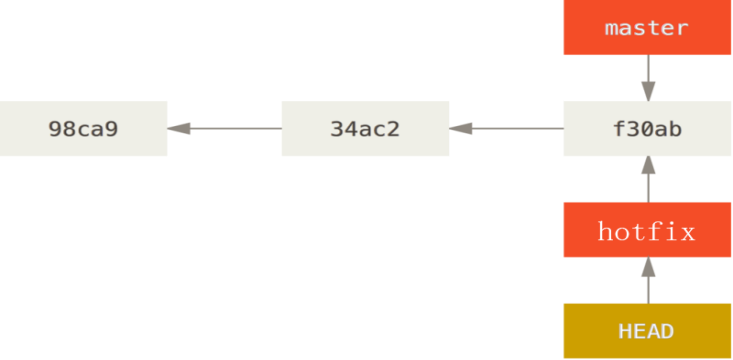
## 查看分支

**git branch**



## 切换分支

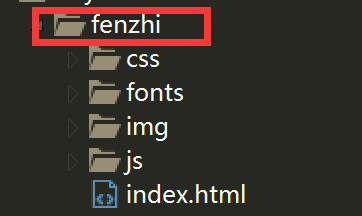
**git checkout 分支名**

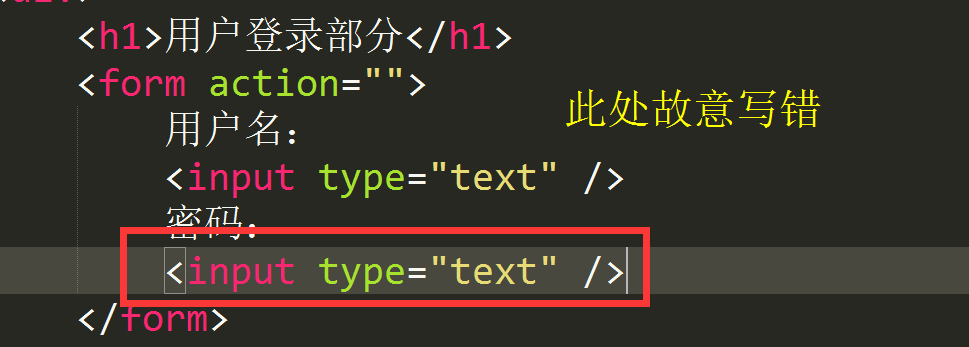


此时的HEAD指针会指向**分支名**的末端。

## 具体演示

### 新建一个项目，并编辑，提交





执行：

**git init**

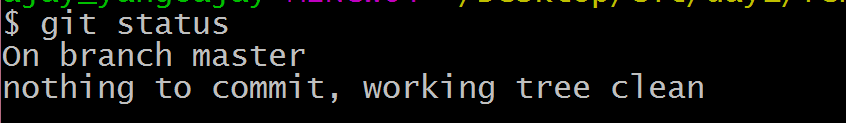
**git status**

**git add .**

**git commit –m ‘完成用户登录部分’**

## 

再次 **git status ,**没有可提交的内容了。



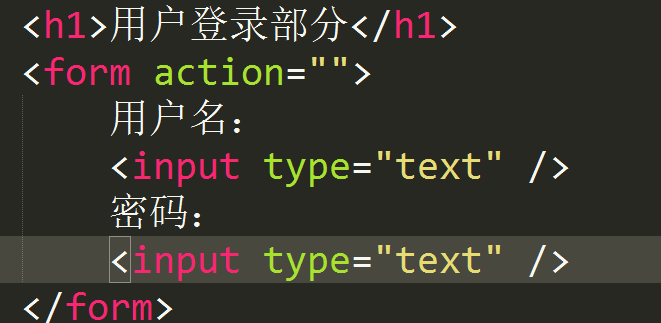
### 新项目需求，需要添加购物车功能（新建分支）

**git branch shop**

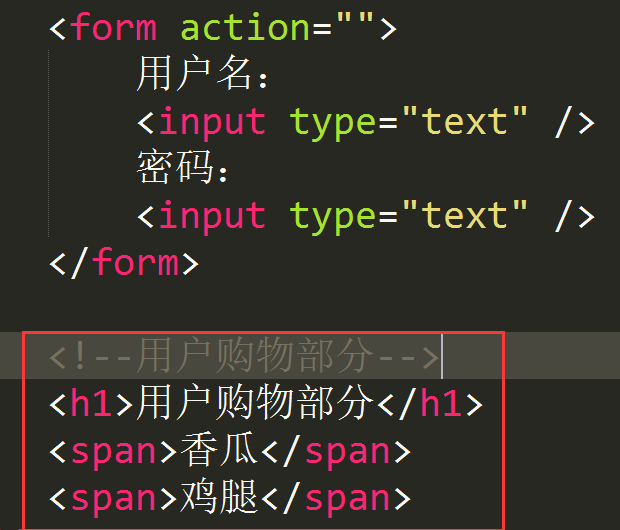
**git branch**

**git checkout shop**

**此时，shop 分支还保存 master 分支的内容，但工作区域已经发生变化。**



继续进行项目。。。。。。。。



### 此时，测试人员告知，登录部分有bug,必须马上修复，怎么办？

**切换分支**

**切换之前一定要先保存好本分支还没有提交的内容。**

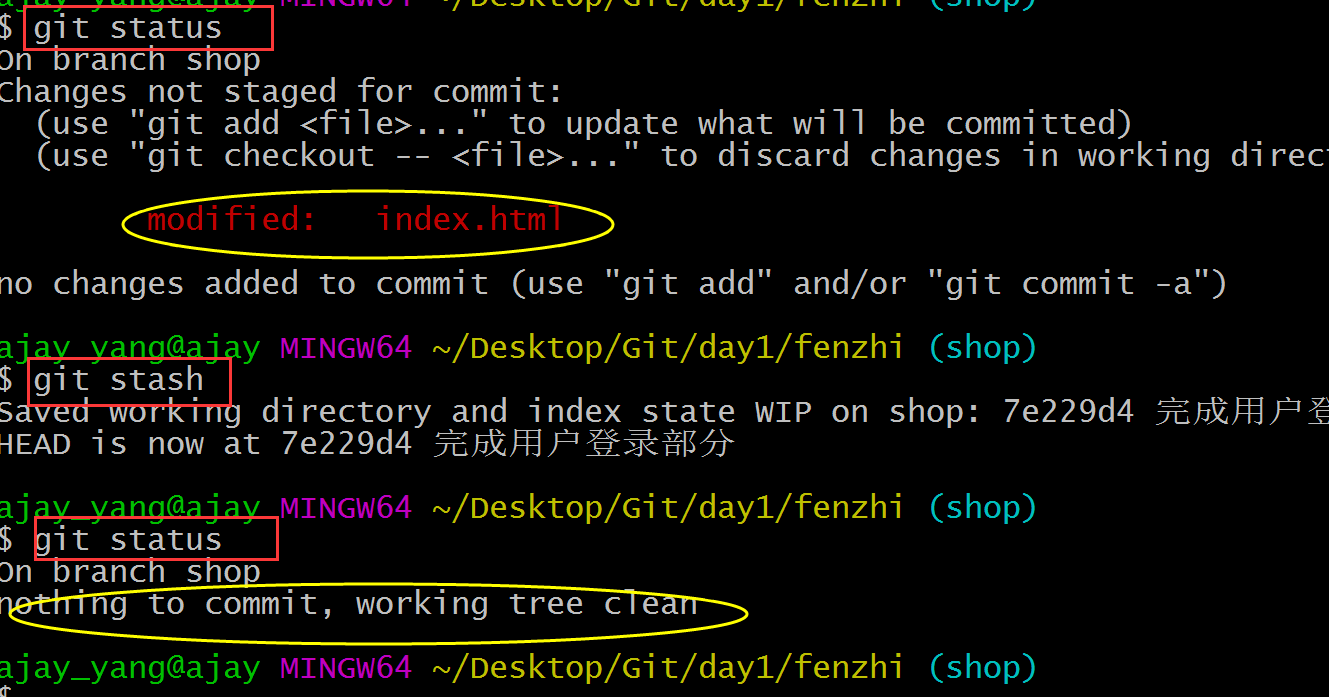
**git status**

**git stash**

**git status**

git stash 保存当前的工作内容，方式去处理bug的过程中本分支的数据丢失。

保存完后，工作去没有内容了。



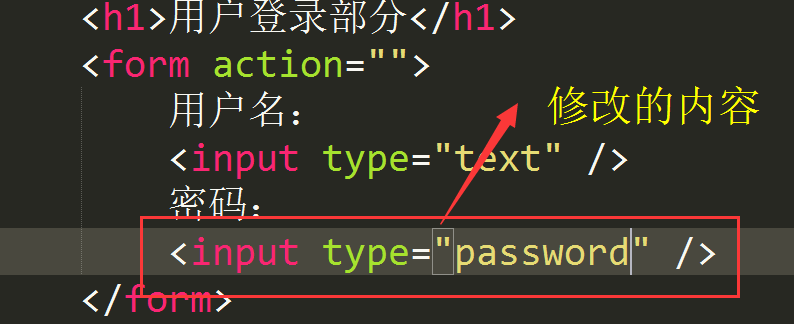
### 切换分支

切换分支，并新建修复bug 的分支**bug001**

**git checkout master**

**git branch bug001**

**git checkout bug001**



此时，工作区的内容不再包含 **shop** 分支

**git status**

**git add .**

**git commit –m ‘修复登录bug001’**

**git log //---查看修改状态**

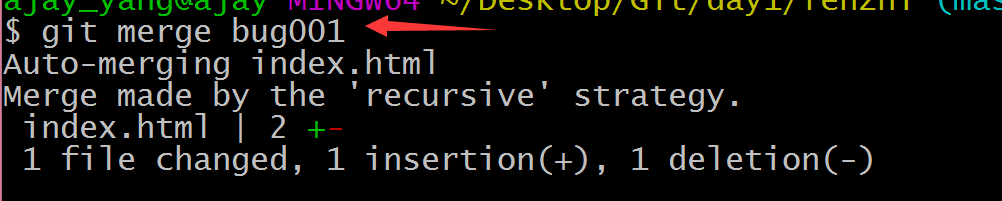
## 

### bug001分支合并(融合)到master分支，并删除

**git checkout master**

**git merge bug001**

**git branch –d bug001 //---删除bug001分支**



### 回到shop分支继续开发

**git checkout shop**

**git stash list**

**git stash apply stash@{0}**

**git add .**

**git commit –m ‘购物车功能完成’**

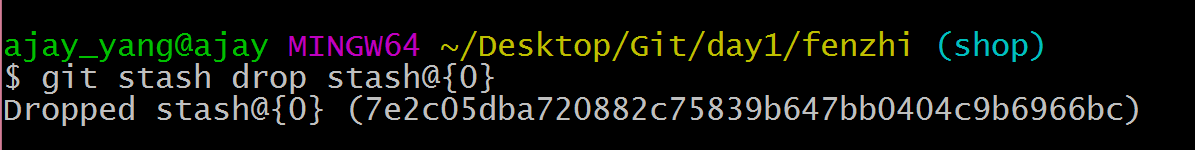
## 

**git log**

## 

如果stash中的内容不需要了，就可以删掉

**git stash drop stash@{0}**



### 项目开发结束，合并分支（融合）

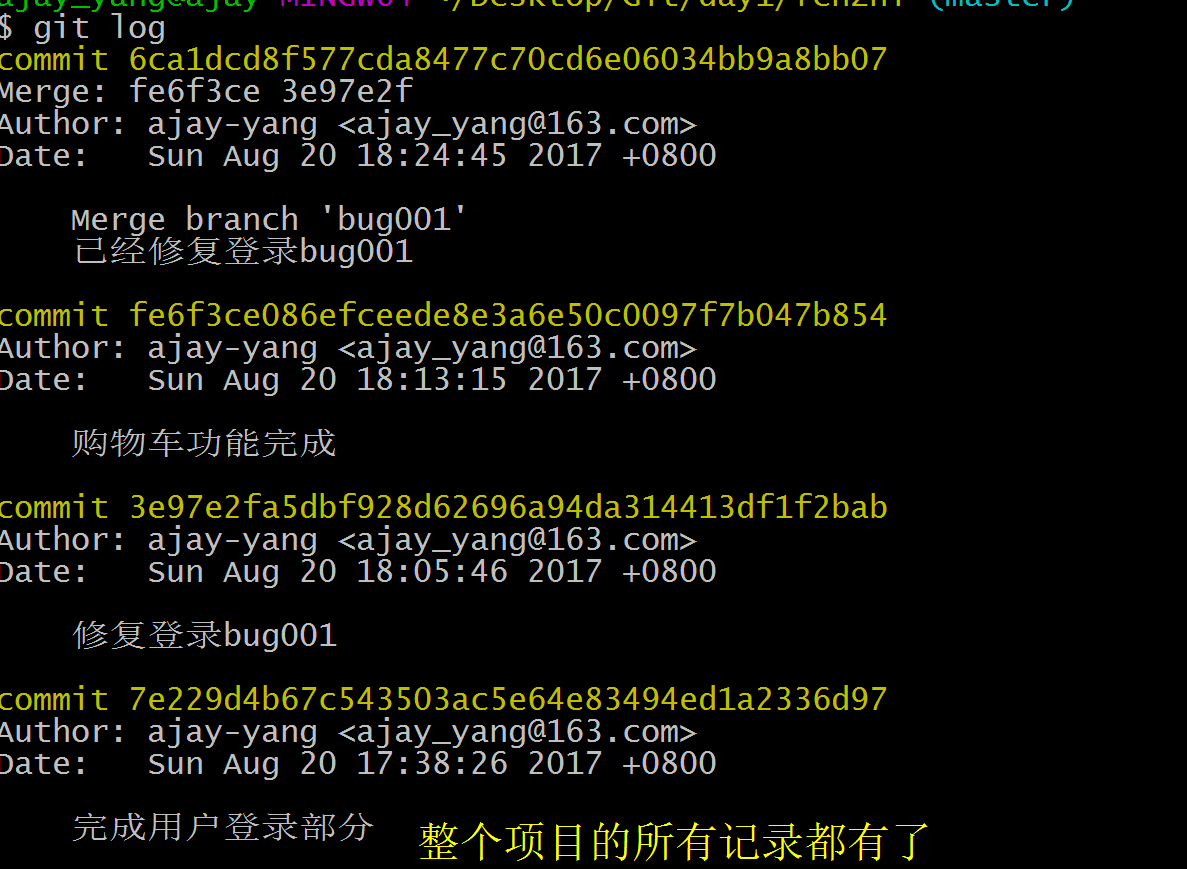
项目已经结束，需要在**master**分支上上线

**git checkout master**

**git merge shop //---合并shop分支的内容**

**git branch –d shop //---删除shop分支**

**git log**



## 还有冲突的合并

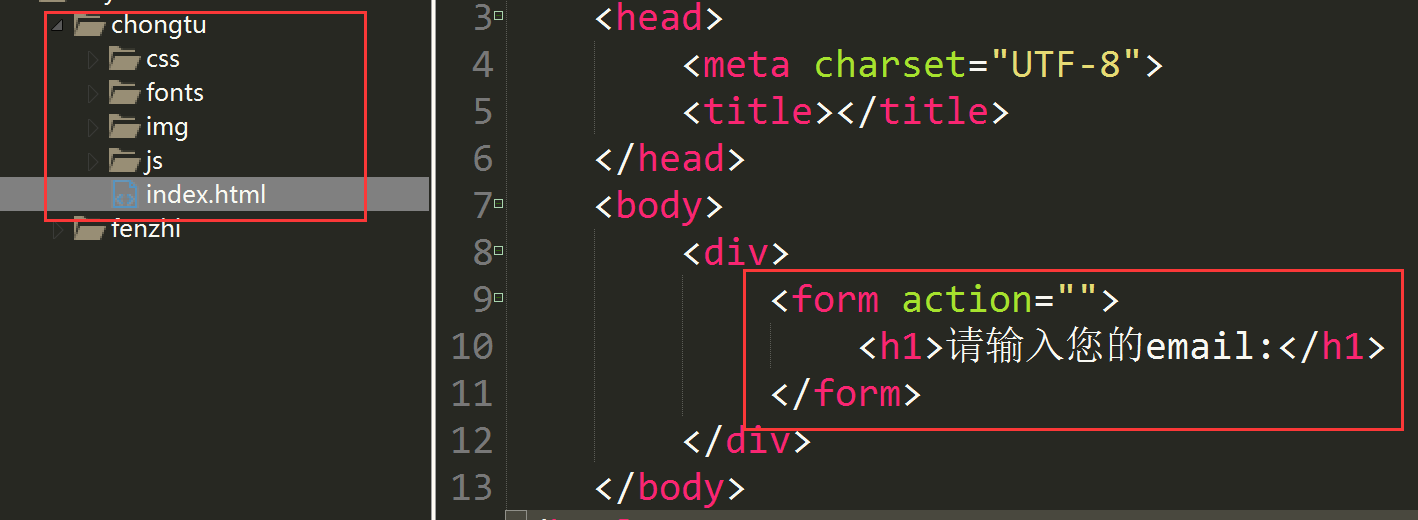
### 新建项目，编辑并提交（master分支）

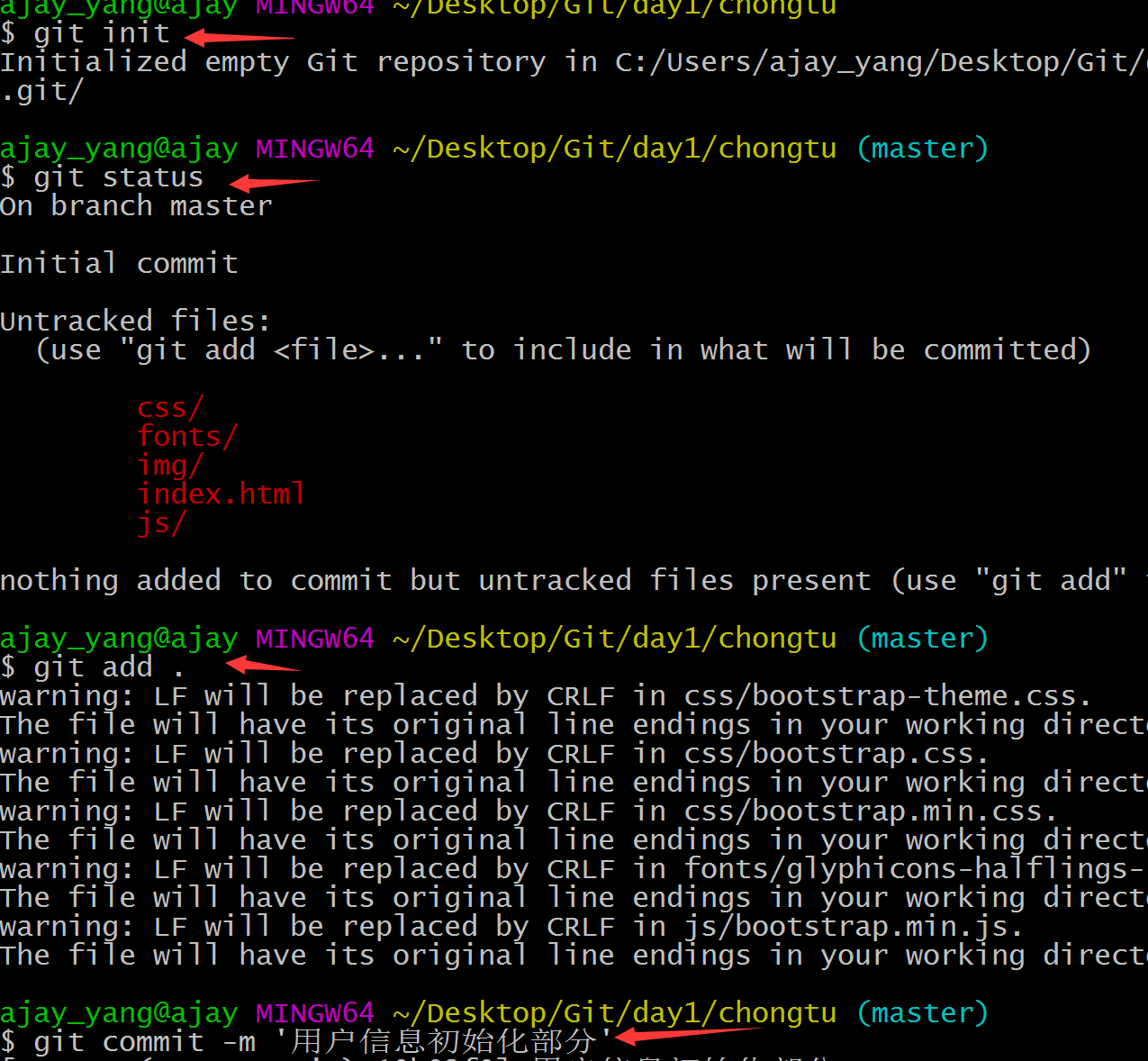
**git init**

**git status**

**git add .**

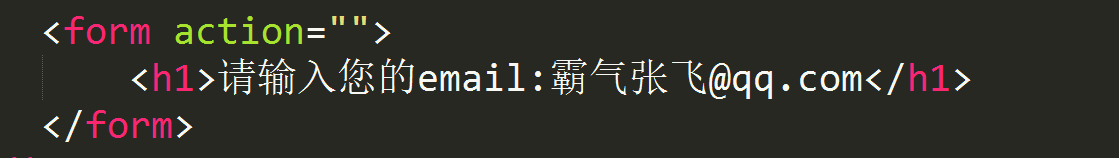
**git commit –m ‘用户信息初始化部分’**





**需求：master分支有张飞开发、flag分支由关羽开发。**

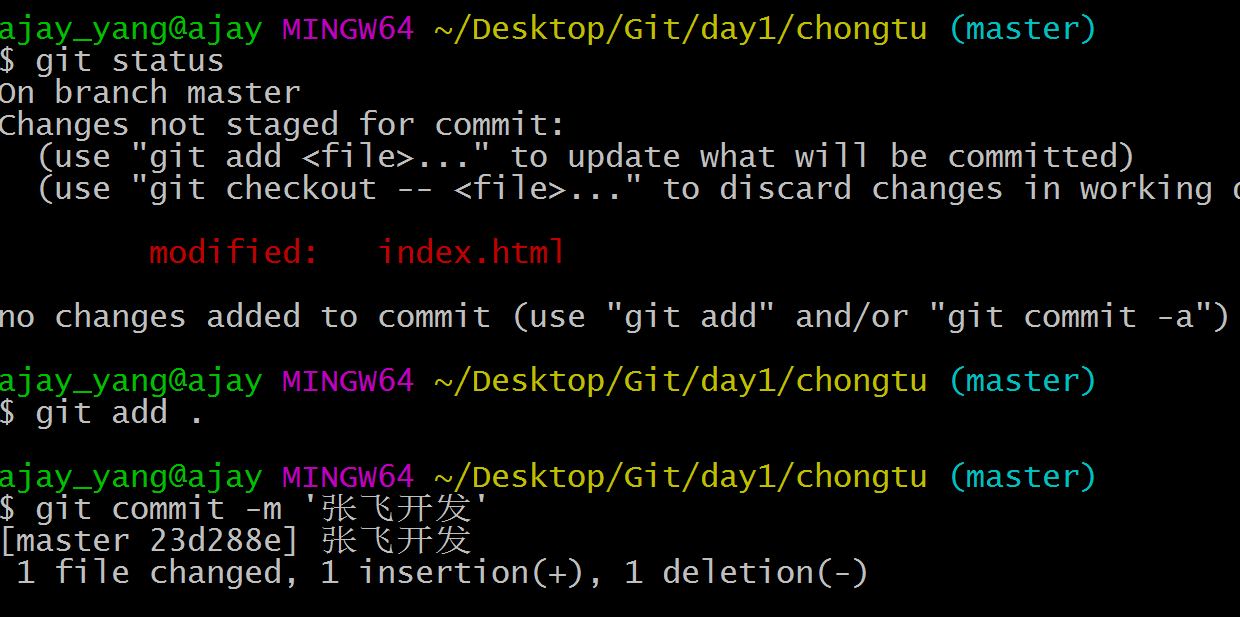
### 张飞开发



**git status**

**git add .**

**git commit –m ‘张飞开发’**



### 关羽开发（新建分支flag）

**git branch flag**

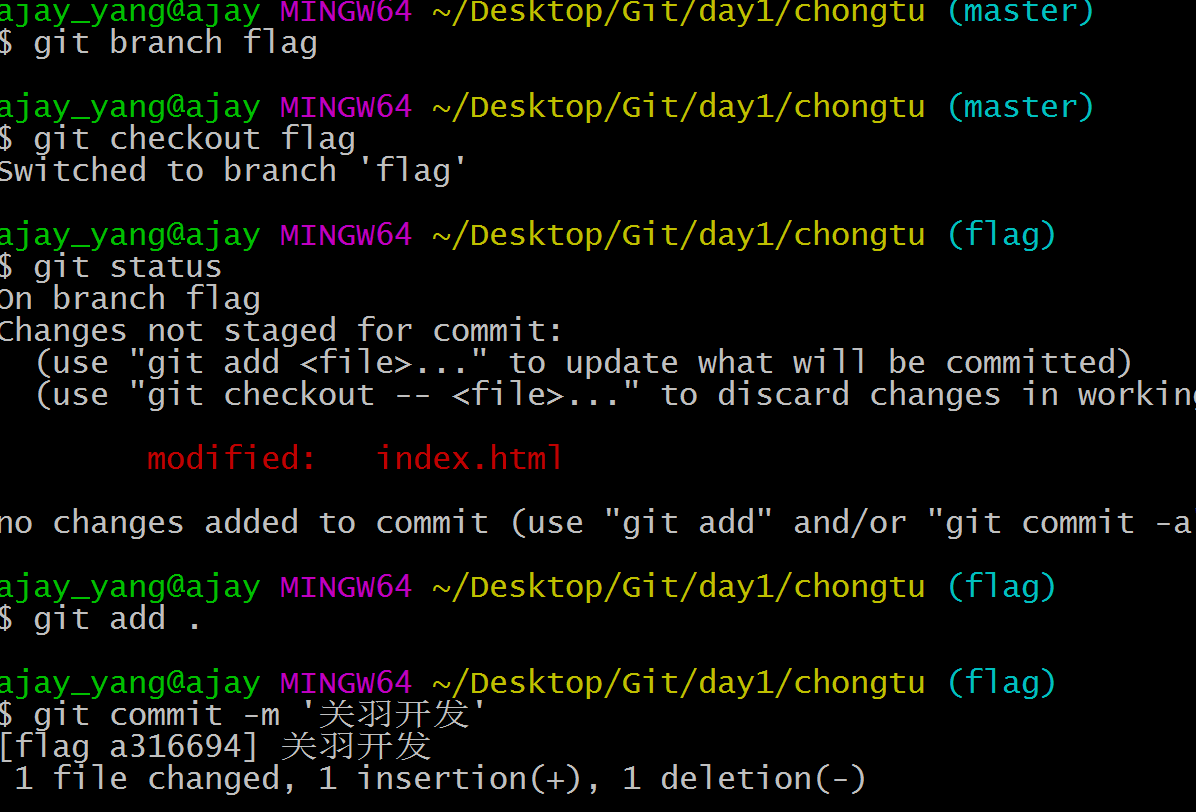
**git checkout falg**



**git status**

**git add .**

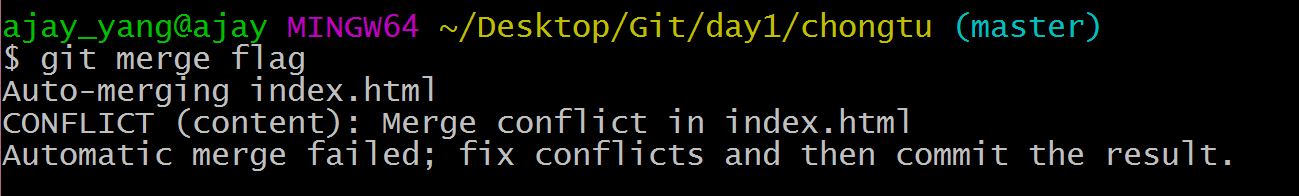
**git commit –m ‘关羽开发’**



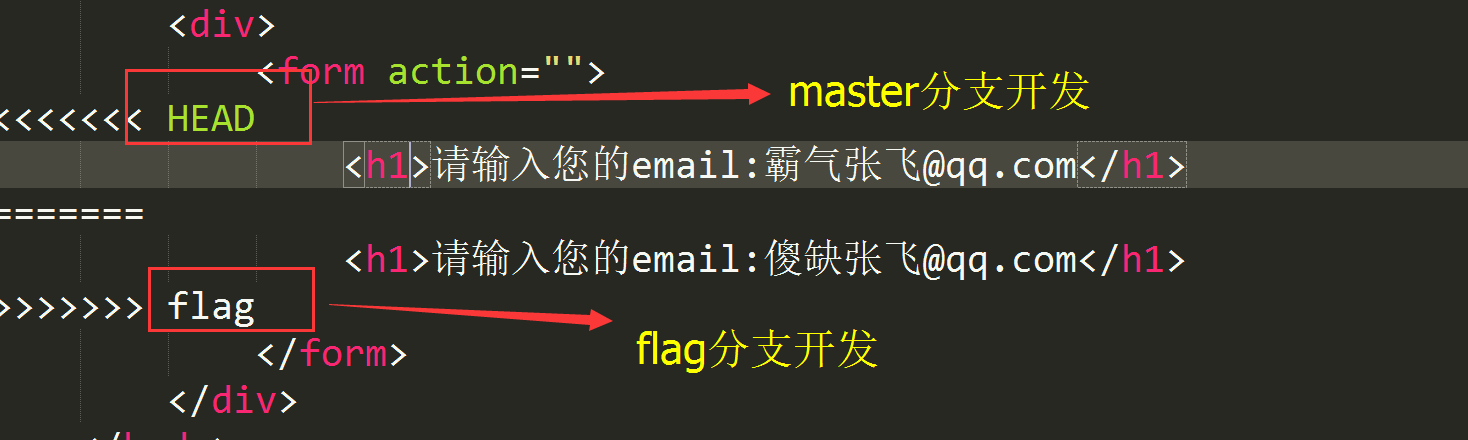
### 项目结束，刘备要求合并到master分支上线。

**git checkout master**

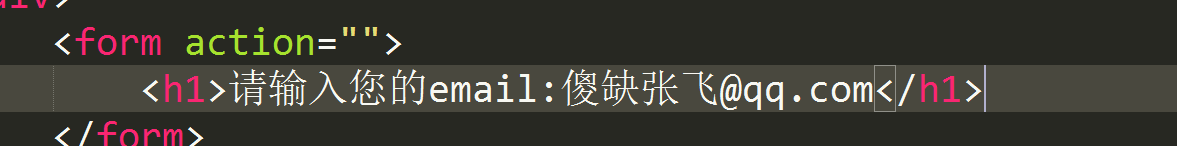
**git merge flag**



此时的文件：



### 根据需求进行修改（没用的删除即可）



**git add .**

**git commit –m ‘意见统一，已经同意张飞是傻缺’**

**git branch –d flag //---删除flag分支**

**git log //---查看开发记录**

