**GitHub for Windows使用教程**

1）**介绍**

1.1 什么是**GitHub for Windows**

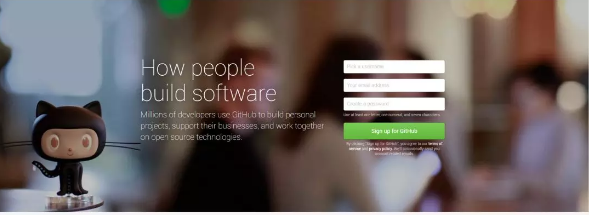
利用Git进行版本控制，用来管理和存放代码

和大家使用的小乌龟一样但是更简单易懂便于操作

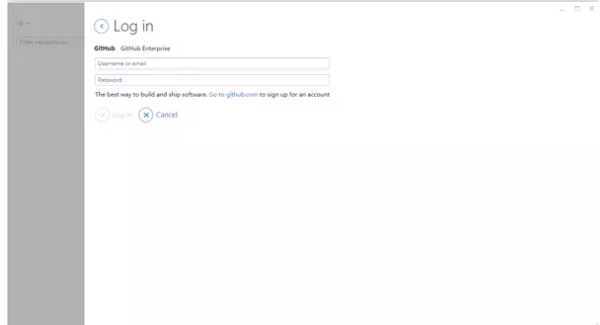
1.2 下载**GitHub for Windows (**就叫章鱼猫吧**)**

**https://desktop.github.com/**

1.3 注册github账号（https://github.com/）



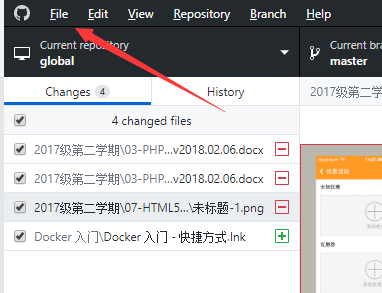
1.4打开章鱼猫 登录你的github账号



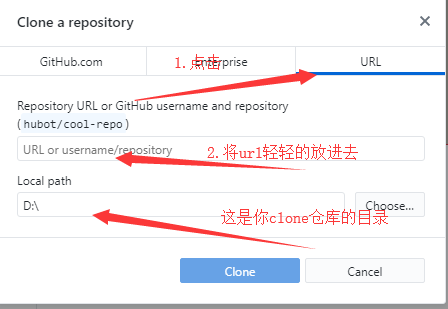
2）**简单的使用（**最常用**）**

2.1 ：如何把global拿到章鱼猫里？

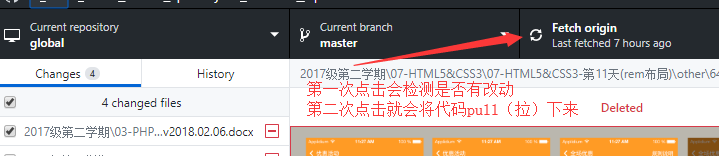
2.1.1：首先我们要拿到global的url 然后点击file



2.1.2：点击Clone repository(克隆库)

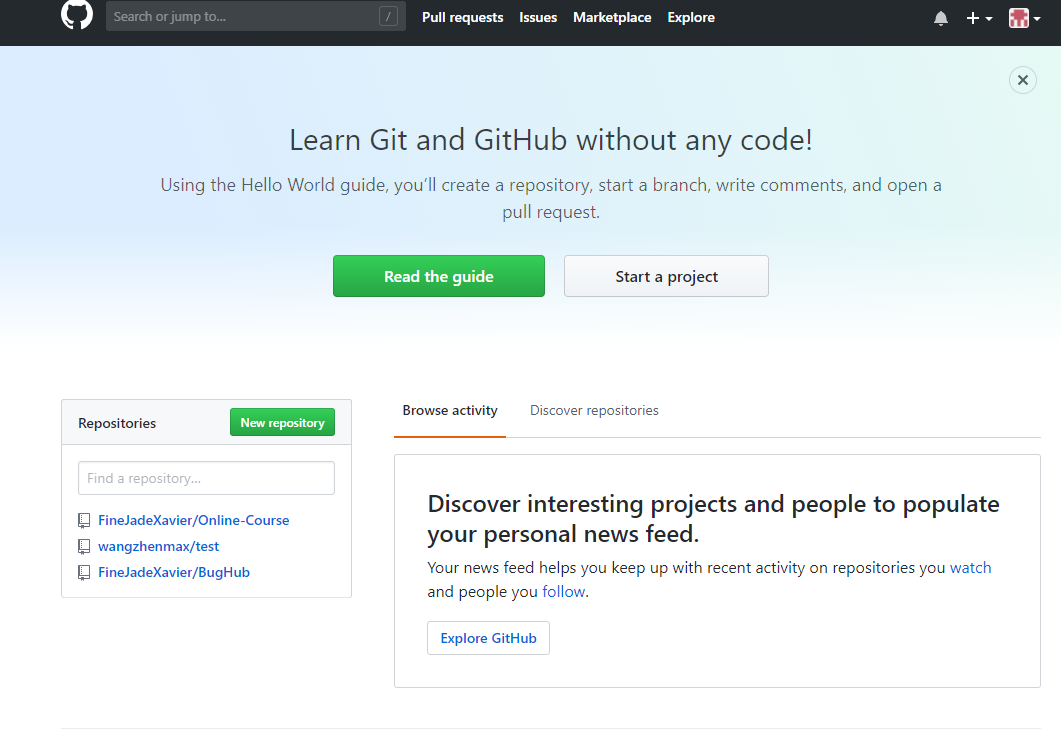


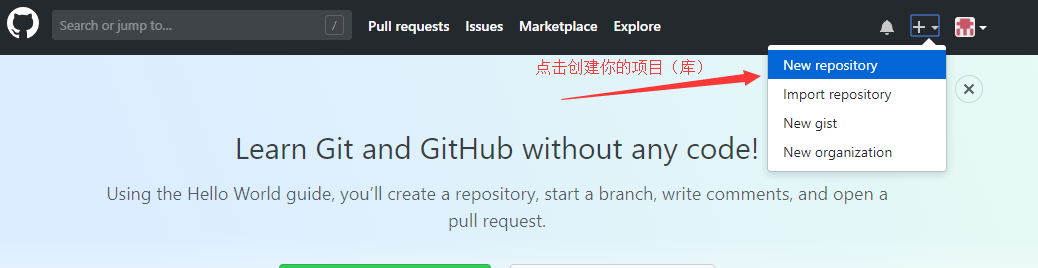
2.1.3然后他就开始克隆global了

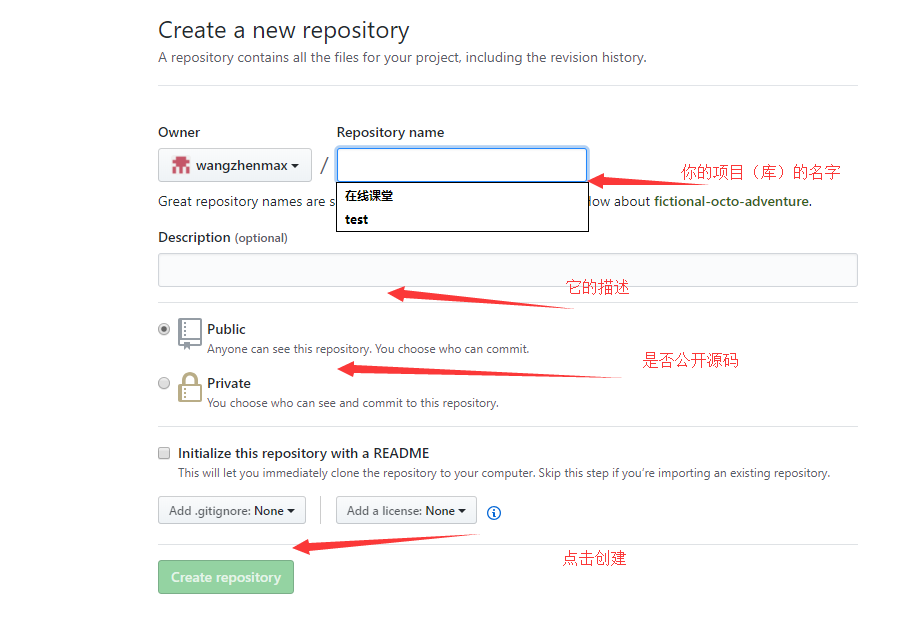
2.1.4 pull代码（需要点击两次）

**3）如何管理我们自己的代码**

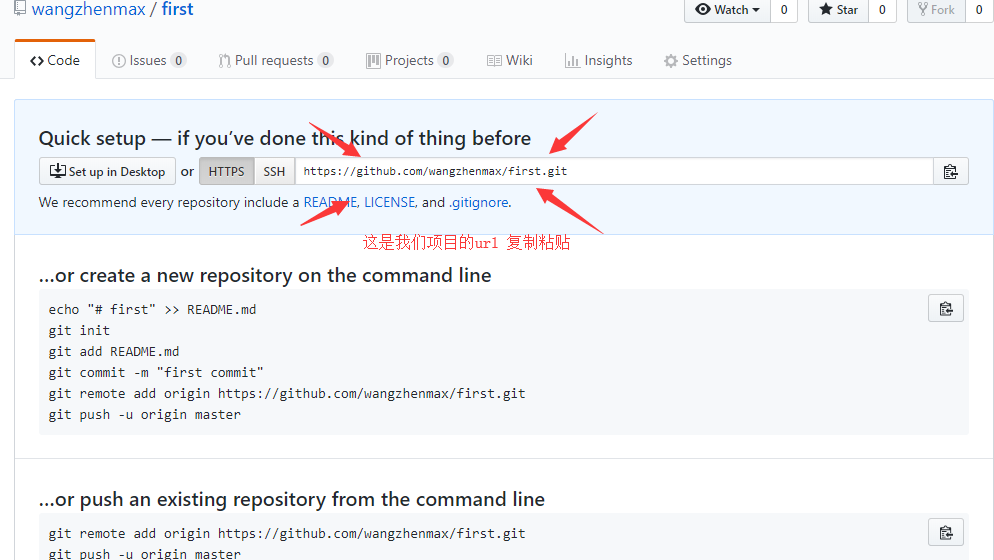
3.1**：**从远程仓库导入到本地 首先你要在你的github创建你的项目，由于部分同学不会使用github我就从创建项目开始讲起，会用的同学忍耐下，

3.2 进入github.com(登录注册) 登录以后我们的界面是这样的

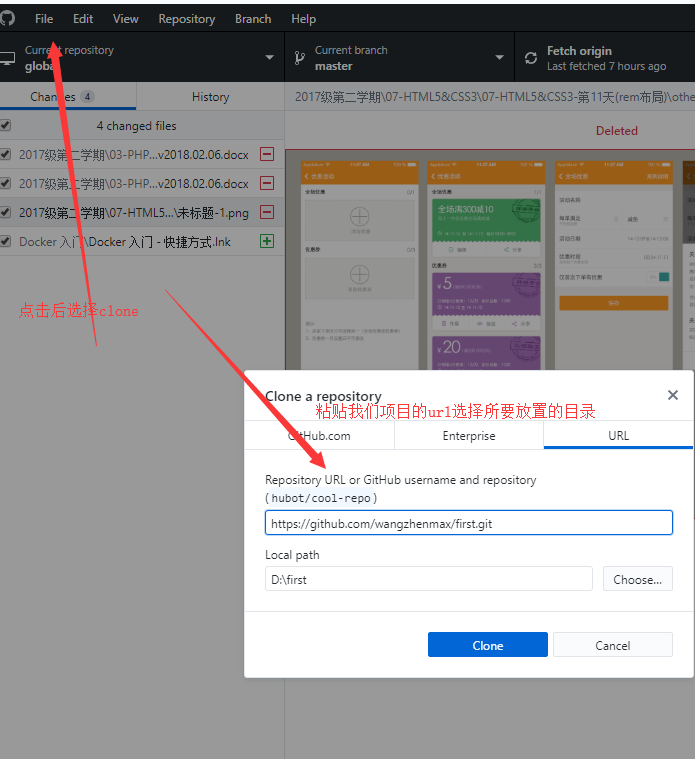
3.3：现在我们创建我们的项目

第二步

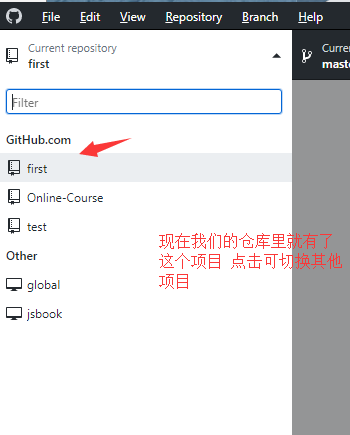
第三步



第四步 这里我们的步骤和clone global的步骤一样 将我们项目的url 克隆到本地



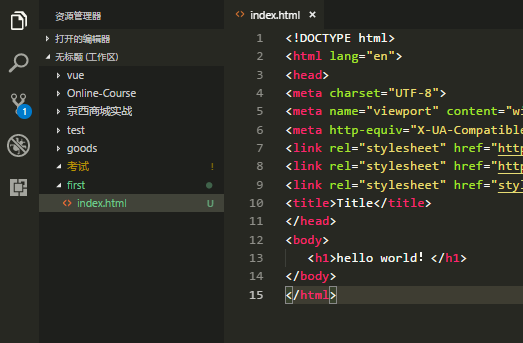
Clone后



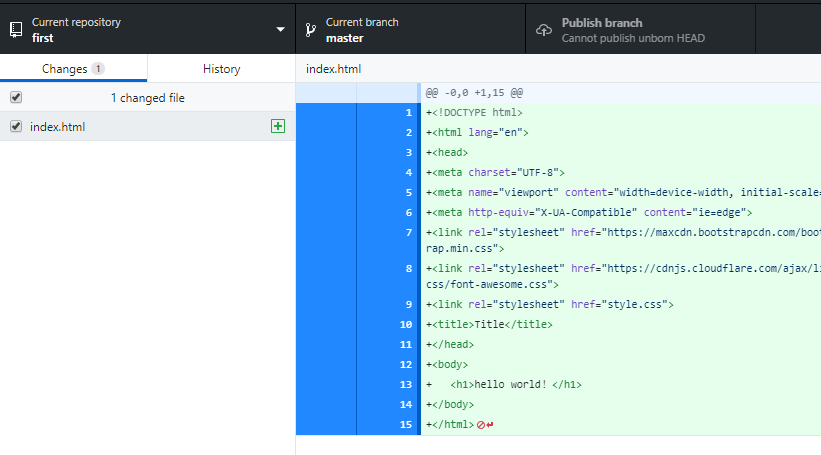
第五步：去我们的项目（克隆时选择的目录） 这里就称之为本地工作区 在这里我们可以**直接编写代码 （跟vscode使用更好）** 很多同学在写项目时是把项目pull下来以后复制一份放到我们用wamp配置的目录下进行编写 编写完以后又将代码拿回到本地工作区（本地仓库）进行提交 这样做的第一麻烦，第二耗时，第三容易出现代码冲突（比如覆盖代码）现在我们使用章鱼猫后将会让你的在管理代码上操作既简单又安全

**4）接下来就是重点了前面都是基本操作（模拟一个项目）**

4.1：首先我们来到我们的vscode来编写代码看看章鱼猫有何反应

我们在vscode新建了一个页面index 并编写了代码

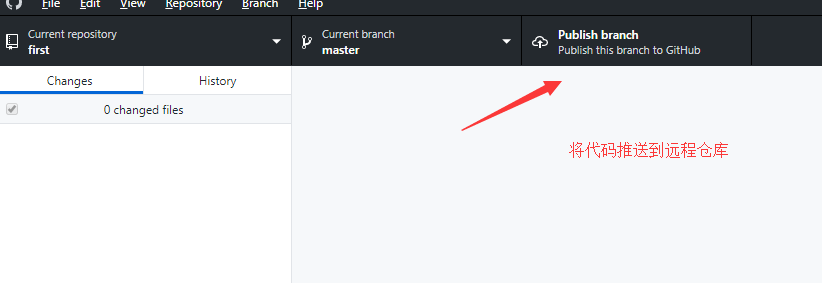
这个时候我们来到章鱼猫发现它有了变化

它提示我们change（改变）本地工作区有了改动以及改动的详情 这个时候我们来把我们的代码直接提交到我们的远程仓库

点击commit以后它会将代码放到缓存区 这个时候我们需要先pull一下代码（有可能其他成员修改了代码）然后再将我们的代码推到远程仓库去。这个软件也做了限制就是如果你修改了本地工作区 那么你每次pull代码的时候他会检测到是否修改 如果修改了会提示你需要commit 才能pull代码

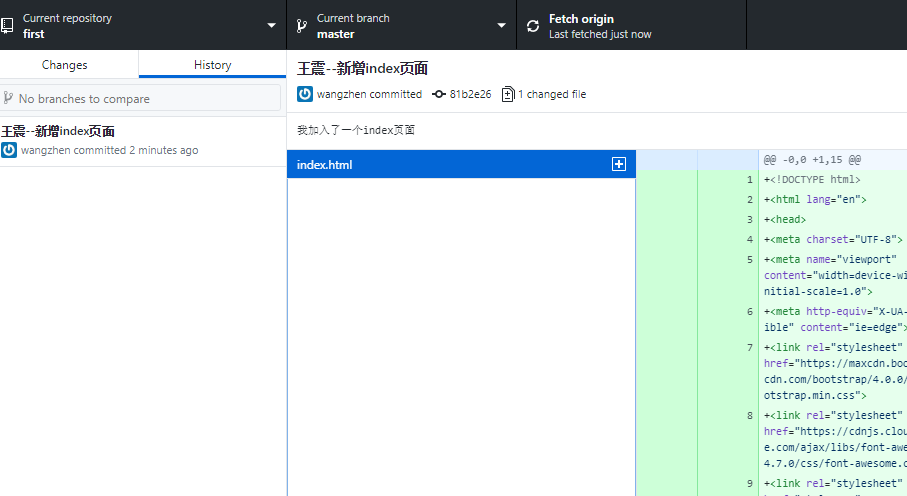
我们是一个团队开发 所以每次提交代码最好将你代码的改动详情写上便于其他项目成员理解

4.2;当我们点击了commit to master（提交代码到主支）后我们需要点击这里



这时他会将缓存区的代码推到你的远程仓库

4.3：这个时候我们点击History

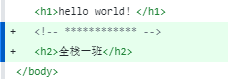


在这里我们可以看到我们的项目成员提交的历史代码 可以看到多久前提交 新增（绿色的+）的页面以及新增的代码（绿色包裹起来的）这很方便我们阅读

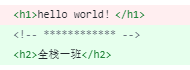
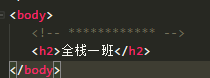
**重点讲解下这几种颜色的含义，接下里会提到**

4.4：现在我们来修改我们本地工作区的代码

新增的代码都会是绿色包裹起来



删除掉的代码会是红色包裹起来 现在我们删除原有的hello world



现在我们的index页面做了改动会发现



这个位置有一个黄色的小块意思就是有改动

如果他是绿色的



说明这是新增的页面

如果是红色的-说明这个页面被删除掉了



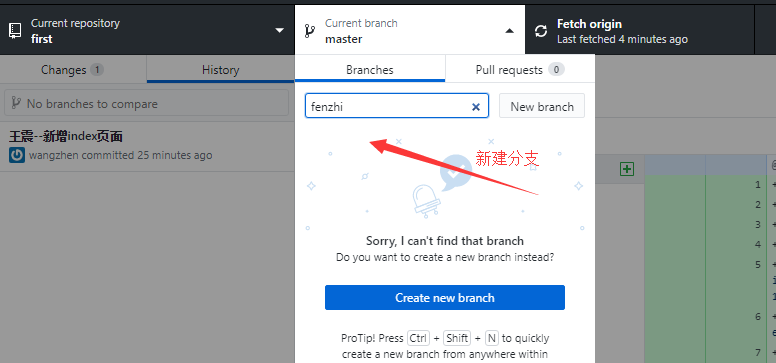
现在我们就能知道我们的这个项目那个页面的那行代码是谁做了改动

**5)接下来是难点 关于主支和分支**

5.1：现在我们模拟一个场景 现在 A 和B 会用到我们的同一个model 我们要如何让这个model 既有A的代码也有B的代码呢？ 如果我们都在master分支上进行改动 那么会出现一个问题 代码冲突 原因就是A编写了这个model后B又去编写 那么就会出现代码被覆盖的问题 如果我们要完美解决这个问题 那我们就需要使用到分支了

5.2 在这里建议大家以后写代码都在分支上写 养成一个好的习惯避免出现不可挽回的情况 接下来开始讲主支和分支的概念

5.3：首先我们需要创建分支 其实可以理解为将我们的代码复制一份到分支里

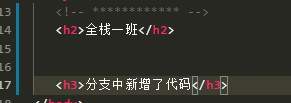


点击创建分支后会自动切换到分支中

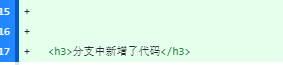


现在我们来到我们的vscode **其实现在我们的vscode里面的代码是分支里面的代码 你做任何修改都不会要想主支（master）的代码** 我们来试试

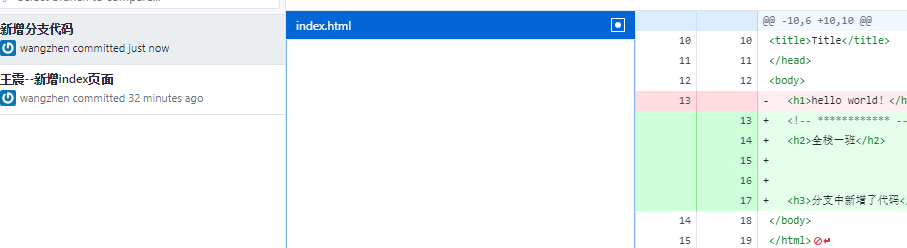
现在我们是在分支中修改代码



修改完后我们可以看到changes有了改动

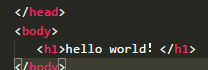


现在我们commit提交以下分支的代码



我们看到本次commit的记录以及修改情况

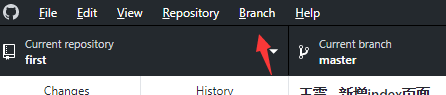
我们再看看主支 如何查看呢？请观察你vscode的代码 当你切换到master分支后再进入vscode观察代码你会发现

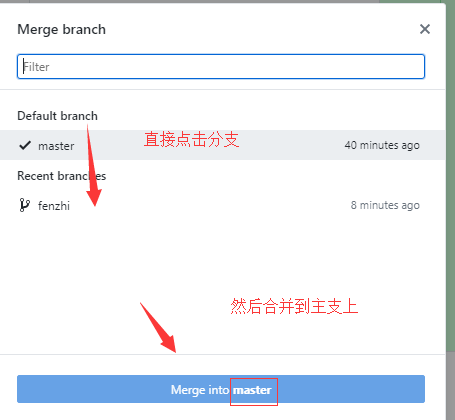
 

你会发现你vscode的代码还是原来的代码并没有改动

所以我们可以从这里知道 分支和主支是互不干扰的两份代码

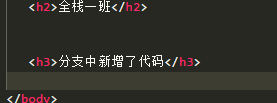
5.4：现在我们要将我们分支的代码合并到主支上面该如何操作

点击这个选项 选择marge into current branch（合并分支）



就是将分支的代码合并到主支上 也可以理解为追加分支里新的代码进去 （然后pull一下）

现在我们去vscode看看我们主支的代码



分支里的代码合并到了主支里 但如果其他成员没有使用分支他可能会覆盖掉你的代码 但你却不会

所以建议都在分支里编写代码后再合并

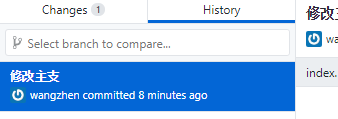
5.5 分支其实还可以有另一个用途就是将你的代码复制一份 这样你如果做了什么不可挽回的错误时能找回我们的代码 那么我们如何让分支的代码也保持最新的呢 其实我们只需要在做了大的改动前将我们的**主支的代码合并到分支中** （主支代码覆盖分支）这样分支的代码就和主支相同了 然后再提交主支的代码 如果出现了错误 还能到我们的分支拿回我们的代码

如何将主支代码合并到分支 只需要切换到分支然后操作步骤同上 选择主支然后合并即可这里就不演示了

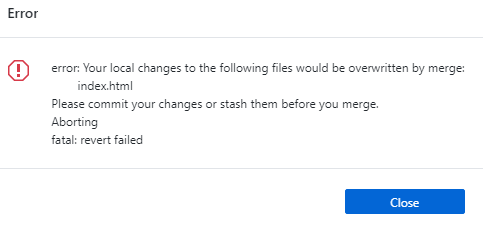
就和回滚一样 现在讲一下

**5.6：回滚**

在章鱼猫里回滚很简单，



只需要来到提交记录里 右键选择最近的一次提交然后remove就可以了（如果你改动了代码没有commit会出现这样的错误）

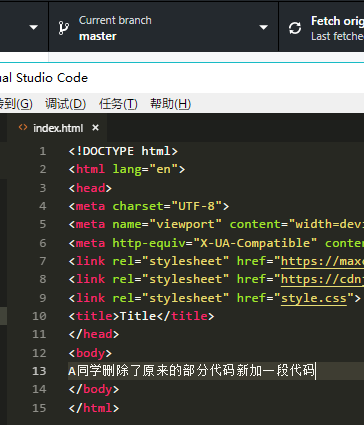


意思是你本地工作区发生了更改 需要提交才能回滚 所以在回滚前就尽量不要修改代码了 个人觉得还是提交前把代码放分支好用点

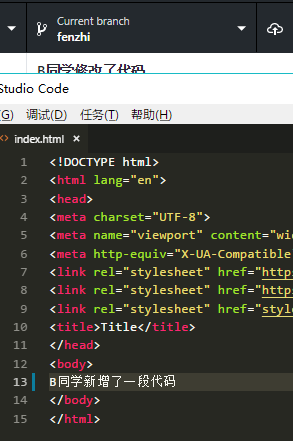
**6）最后讲一下代码冲突的解决方法**（配合vscode）

现在模拟最常出现的一个场景

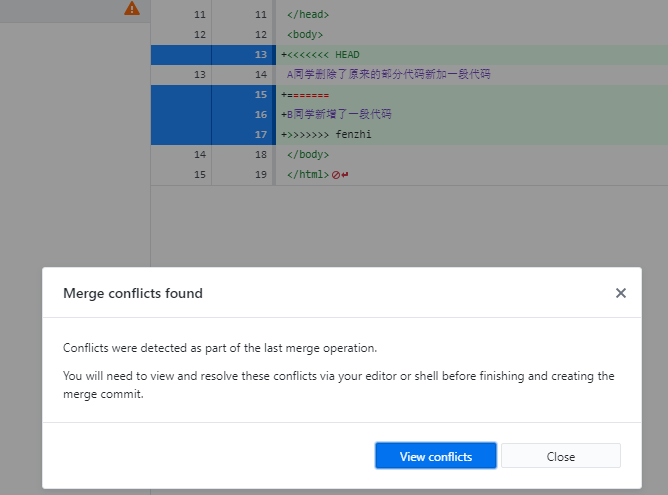
A成员在主支将index中的代码删除了然后新增了一段代码 后并提交了代码



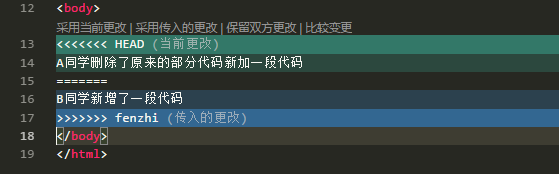
然后B同学在分支里也新增了一段代码



这个时候合并会出现什么情况？



这个时候就提示代码冲突了 但是别慌 我们现在来到vscode



你会看到这样的代码

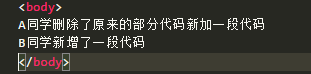
第一个选项就是保留之前的代码 作废分支的代码

第二个选项就是保留分支的代码 作废主支的代码

第三个选项就是保留双方的代码 代码都会留下（基本都是选择这个）

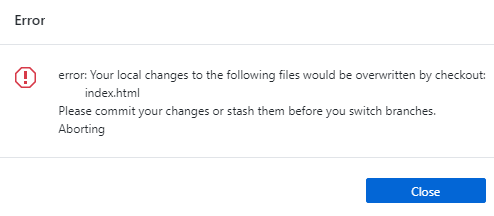
第四个就是进行对比

现在我们点击保留双方更改



就解决了代码冲突的问题

常见的错误



弹出这个错误的原因是你修改了本地工作区的代码但是没有commit 又去做了其他操作 比如切换分支 或者pull代码的时候就被提示你 并把你做了修改的页面都一一列举出来

尽量完善一个功能以后就将代码提交到远程仓库里

这些都是常用的操作 还有其他的需要自己去探索….