IntelliJ IDEA 之 Git 使用

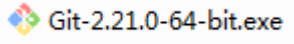
原创 一线大码 已于 2022-05-24 14:46:51 修改 961 收藏 14 版权
分类专栏： 软件工具 文章标签： git idea

文章目录

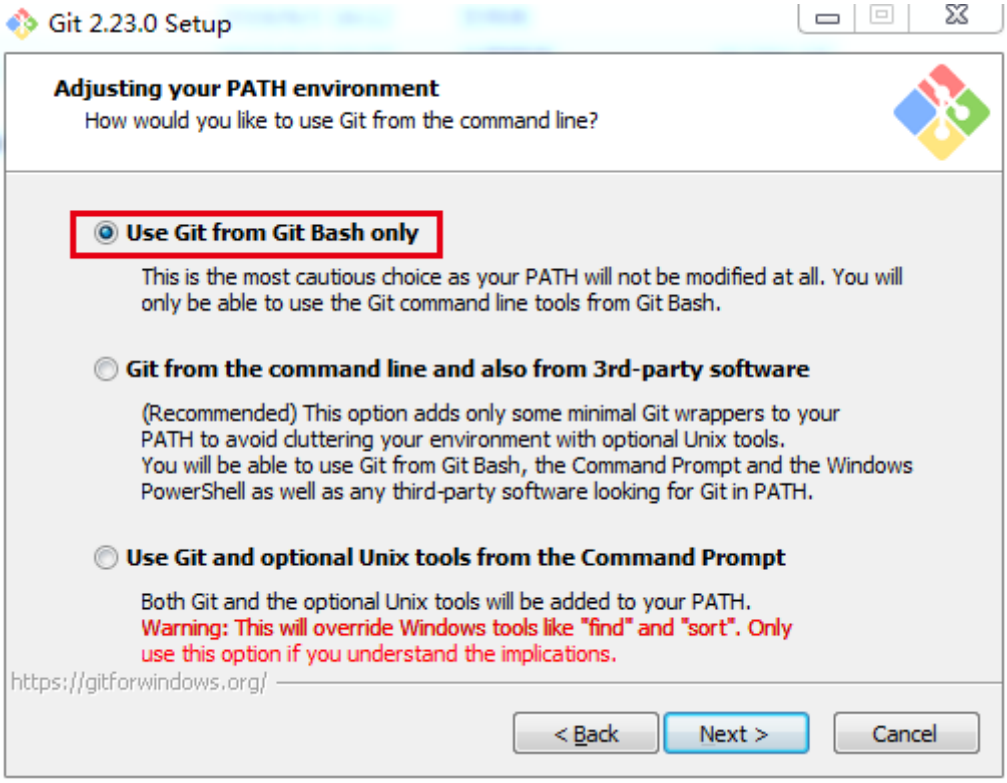
---

1. 本地安装 Git

官方下载地址
不过这个地址一般下不动，我们可以选择在腾讯软件中心下载，速度很快。
腾讯软件中心下载地址



接着就是安装了，一直点击 `Next` 即可，使用默认安装。这个页面选择 `git` 使用的 命令行 ，建议使用第一个 `git` 自带的即可。

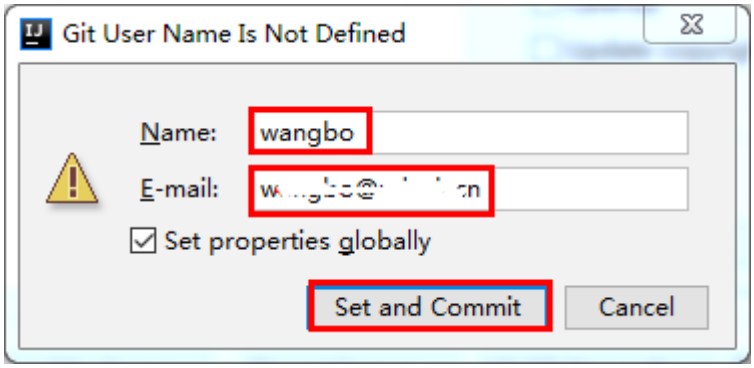


接着需要设置 `git` 的用户名和邮箱，这个用户名会显示在提交历史中。打开 `git-base.exe` ，这里需要执行两个命令，一般直接设置全局的即可：

```
--修改当前项目用户名和密码：
git config  user.name  "username"
git config  user.email  "email"
--修改完毕查看一下：
git config user.name
git config user.email
```

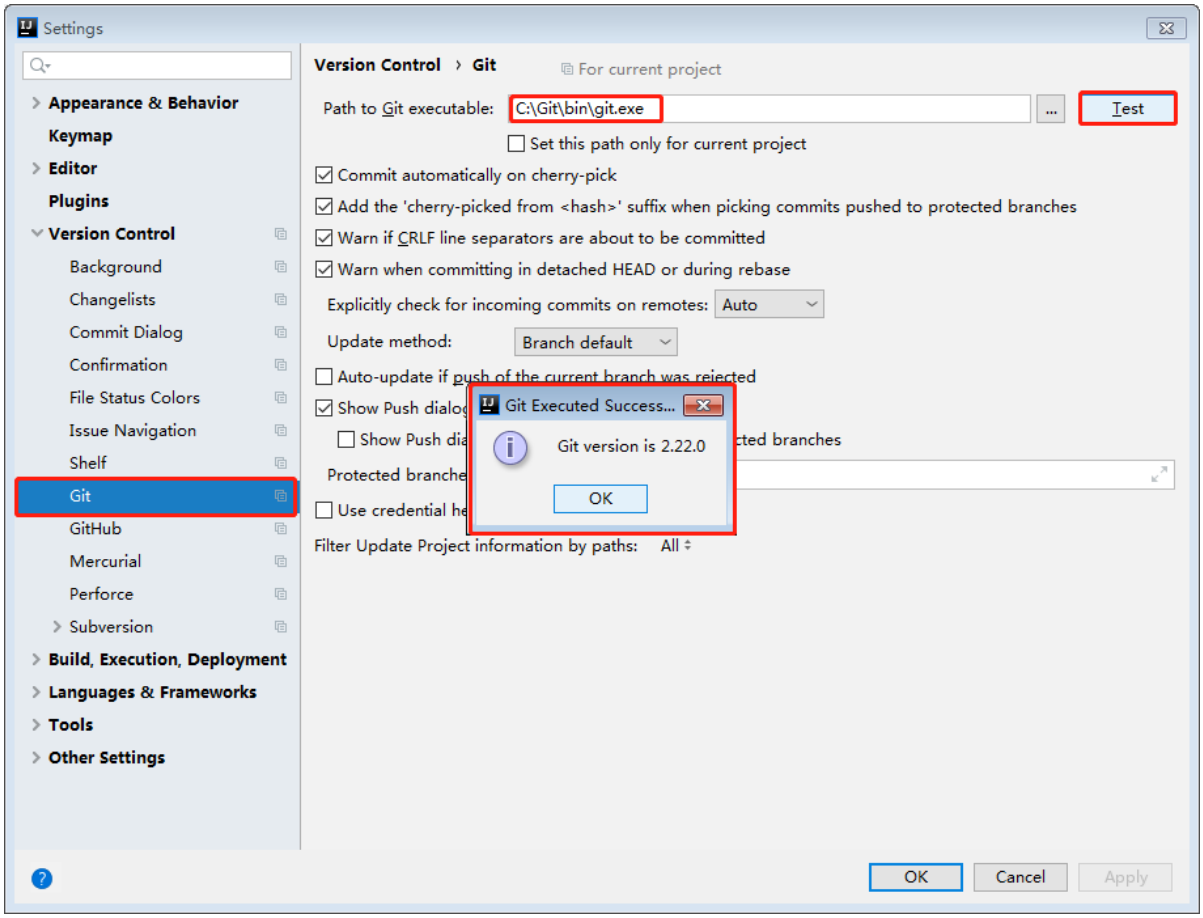
```
--修改全局用户名和密码：
git config --global user.name "Your Name Here"
git config --global user.email "your_email@example.com"
--修改完毕查看一下：
git config --global user.name
git config --global user.email
```

如果这里没进行设置，会在你第一次提交的时候提示你设置：



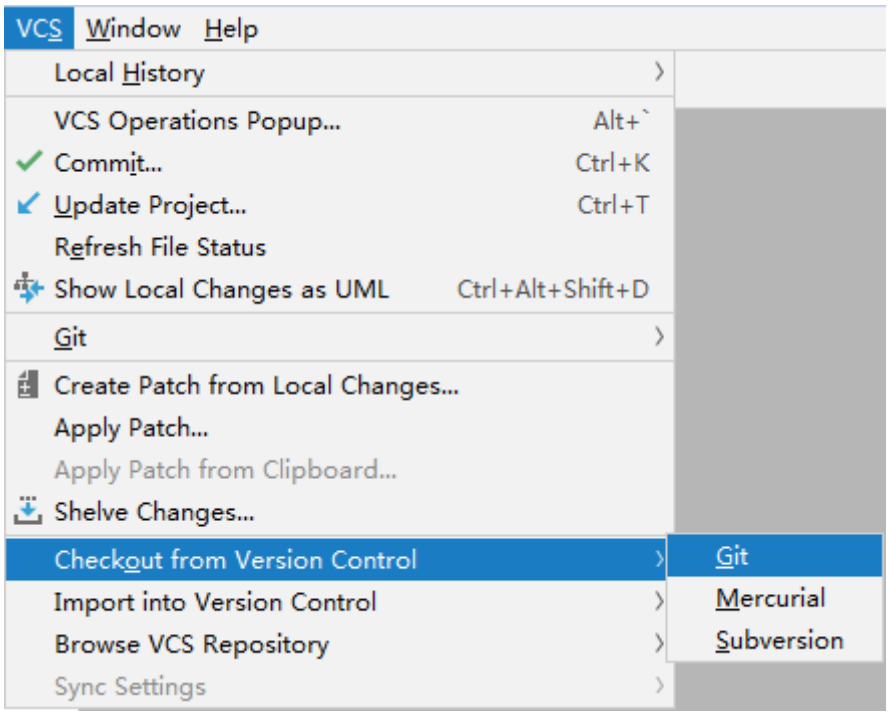
2. IDEA 配置 Git

使用的 IDEA 版本为 2019.2 ，首先需要安装 Git 软件，然后在 Settings 选项中配置 Git 安装路径，点击 Test 进行测试。

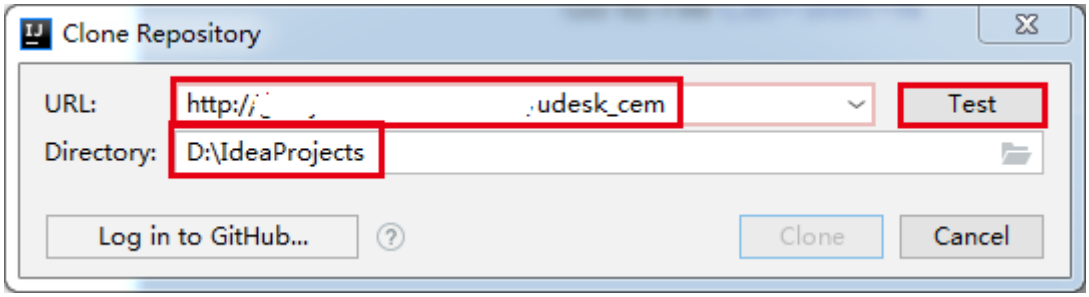


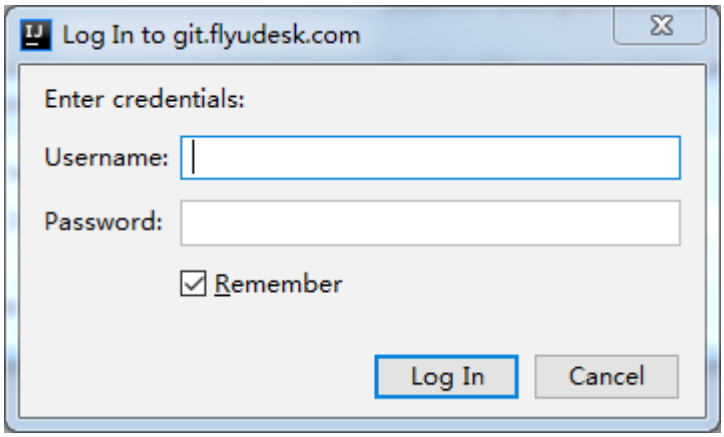
3. 首次检出项目

首次检出需要在 IDEA 顶部工具栏选择 VCS--->Checkout from Version Control--->Git 。

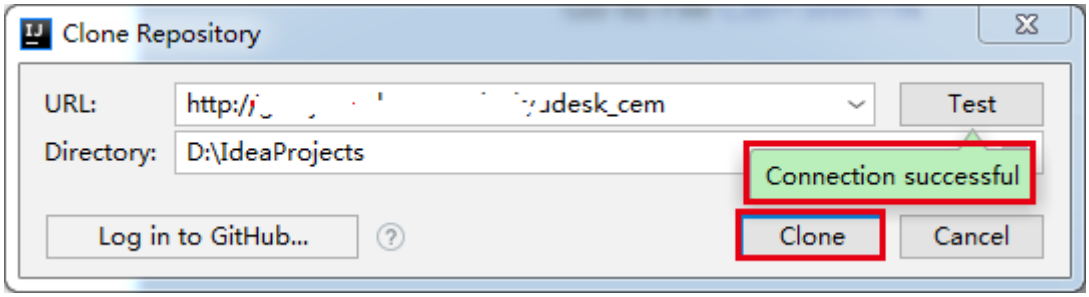


然后在 URL 填写上公司 GitLab 中目标项目远端地址，目录 Directory 注意填写到具体的项目名（图片上的是错的，应该是 D:\IdeaProjects\udesk_cem ）点击 Test ，接下来会提示输入用户名和密码。





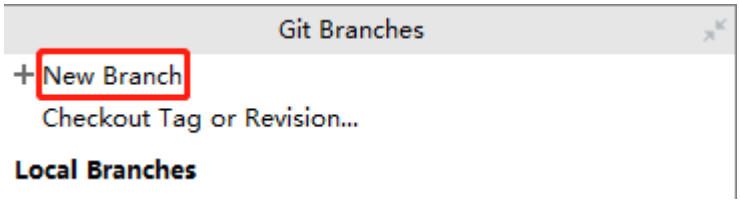
填写好用户名和密码后，如果正确，会提示连接成功，接着点击克隆，将远端项目检出到本地。

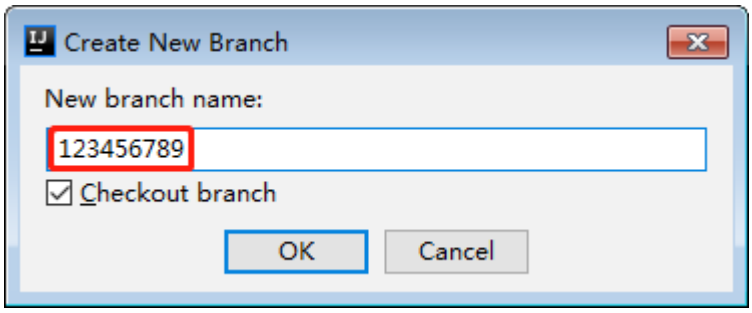


这是首次检出的过程，之后的每次检出就不需要这么麻烦了，直接在 IDEA 右下角的分支列表进行操作即可。

4. 项目分支创建

可以在右下角的分支列表上方选择 +New Breach 新建分支，新建的分支是基于当前所在分支的代码的。所以在新建分支之前可以先切换到想要作为基础版本代码的分支，再进行创建分支操作。

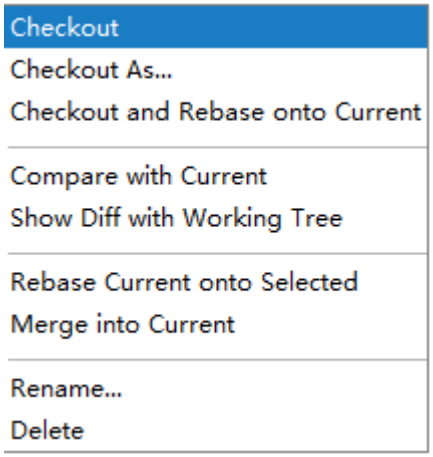




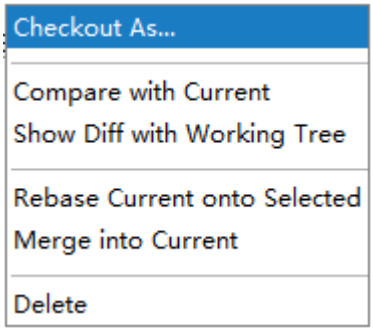
5. 项目分支切换

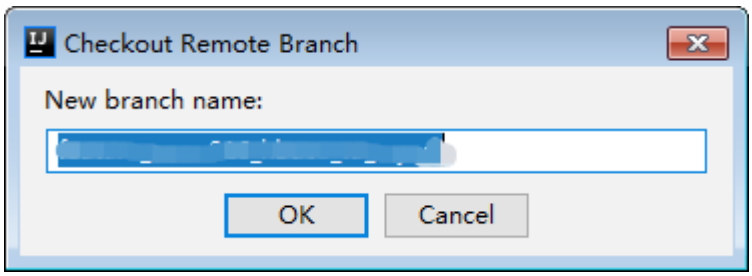
可以直接在右下角的分支列表中选择需要切换的分支，直接鼠标左键点击，选择想要切换的分支。切换完毕会在右下角显示当前所处的分支名称。

（1）本地分支 `Local Branches` 的检出切换选择 `Checkout` 。



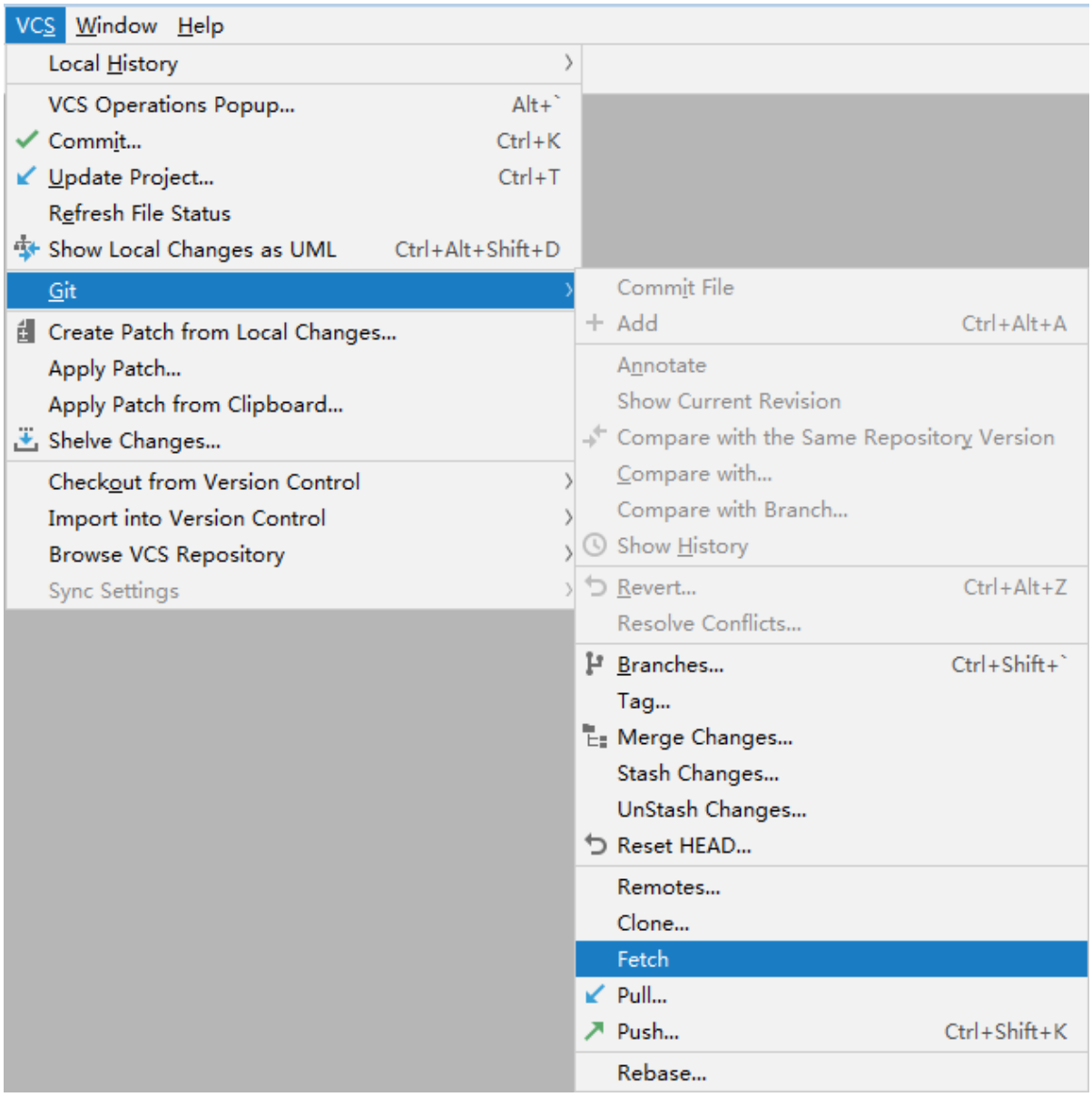
（2）远端分支 `Remote Branches` 的检出选择 `Checkout As...` ，新的分支名称不需要修改，默认和远端分支名称一致。点击 `OK` 即可。





6. 项目刷新分支

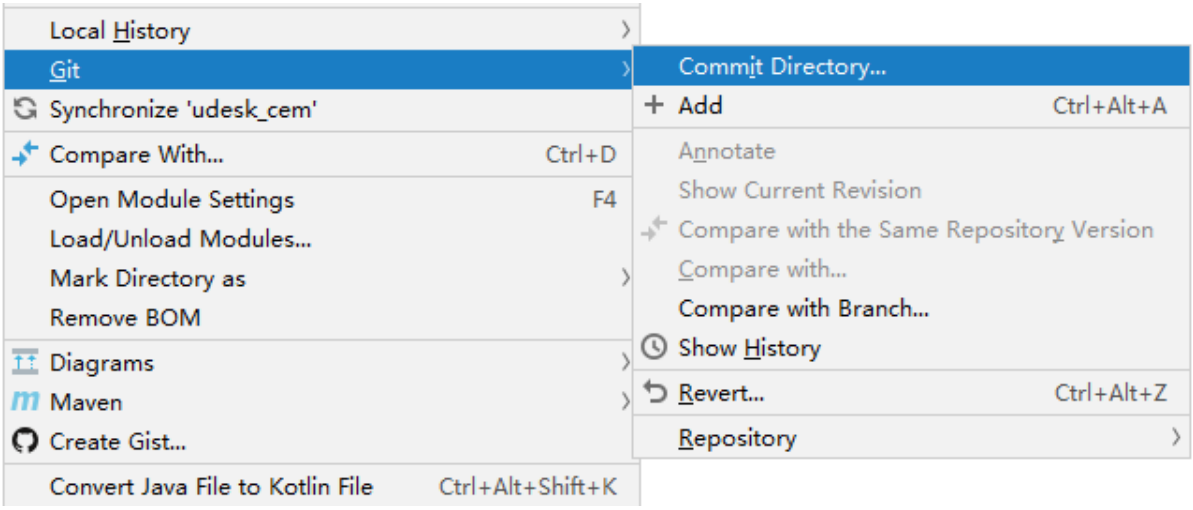
有时候右下角的分支列表里没有想要检出的分支，也就是说你本地显示的远端分支列表和 GitLab 远端分支列表不同步，这时只需要点击顶部工具栏的 VCS--->Git--->Fetch（取来） 进行重新拉取即可实现右下角远端分支列表的刷新。



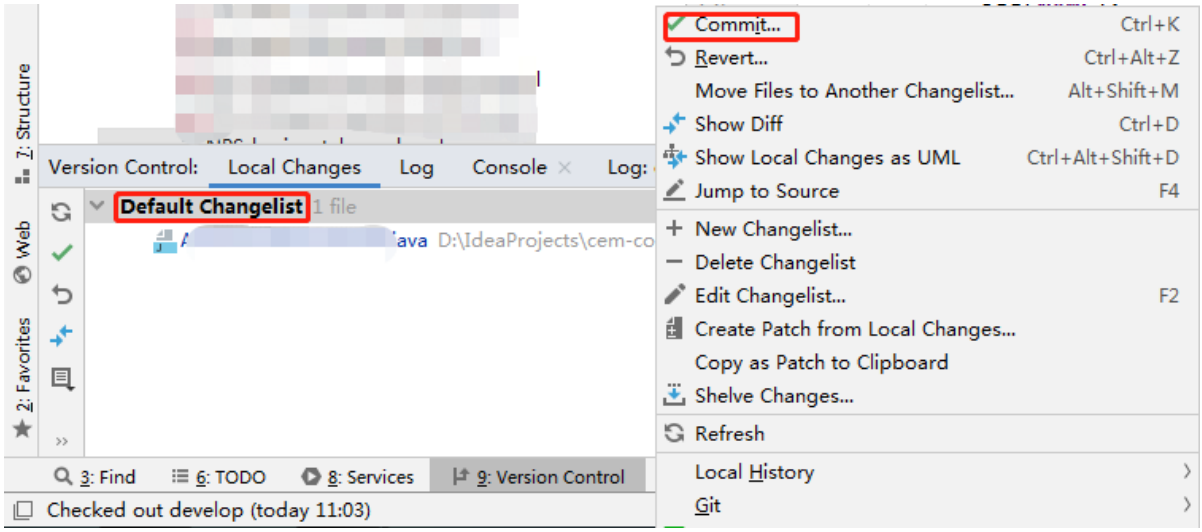
7. 分支代码提交

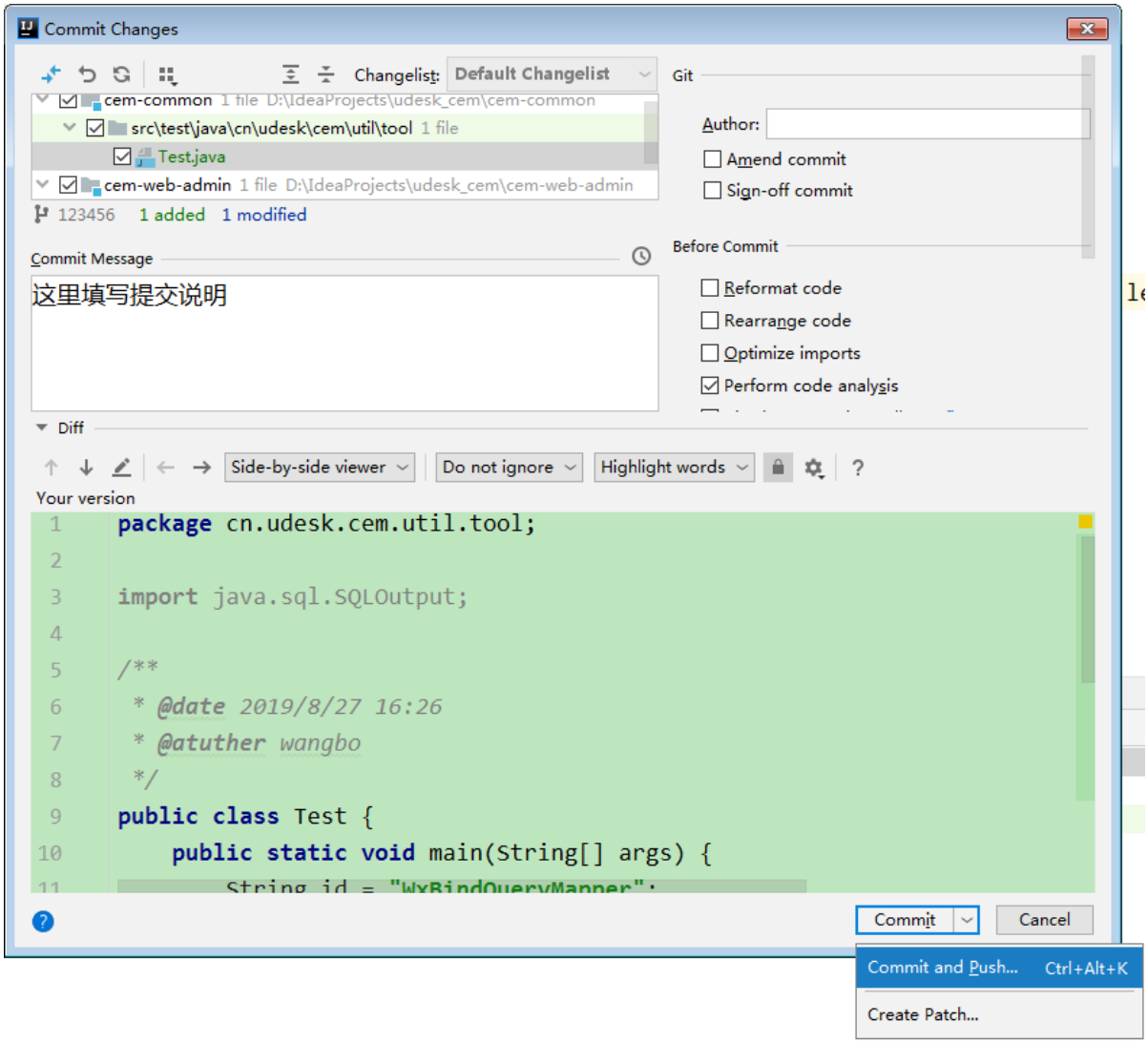
在开发分支上开发完毕后，可进行代码提交，先提交（Commit）到开发分支的本地仓库，再推送（Push）到开发分支的远端仓库。

首先进行 Commit 操作，在项目上 右键--->Git--->Commit--->Directory... ，可以在弹出界面查看本次提交的文件，可以添加提交信息，还可以在右下角选择提交选项。默认是 Commit ，也就是只提交到开发分支的本地仓库，还有一个选项是 Commit And Push... 表示会将 Commit 操作和 Push 操作连起来，在 Commit 完毕后会接着提示你进行 Push 操作，也就是推送到开发分支的远端仓库。

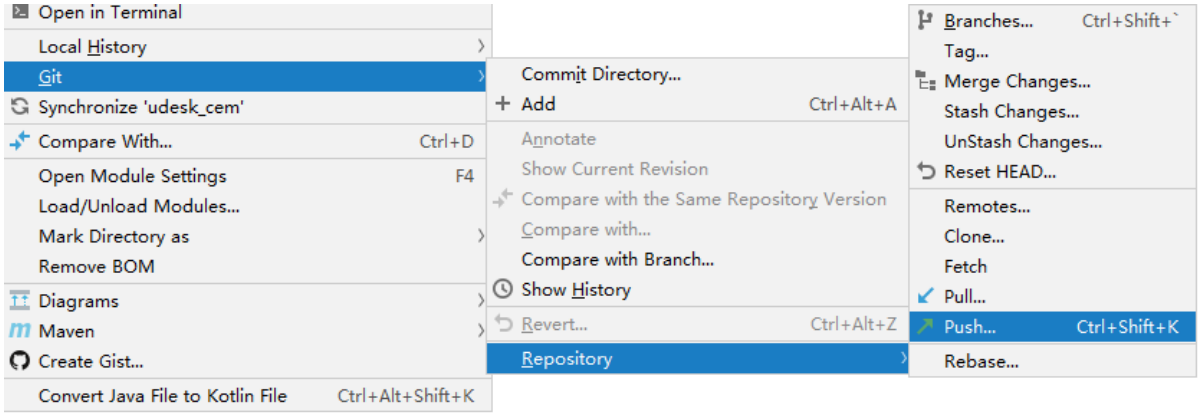


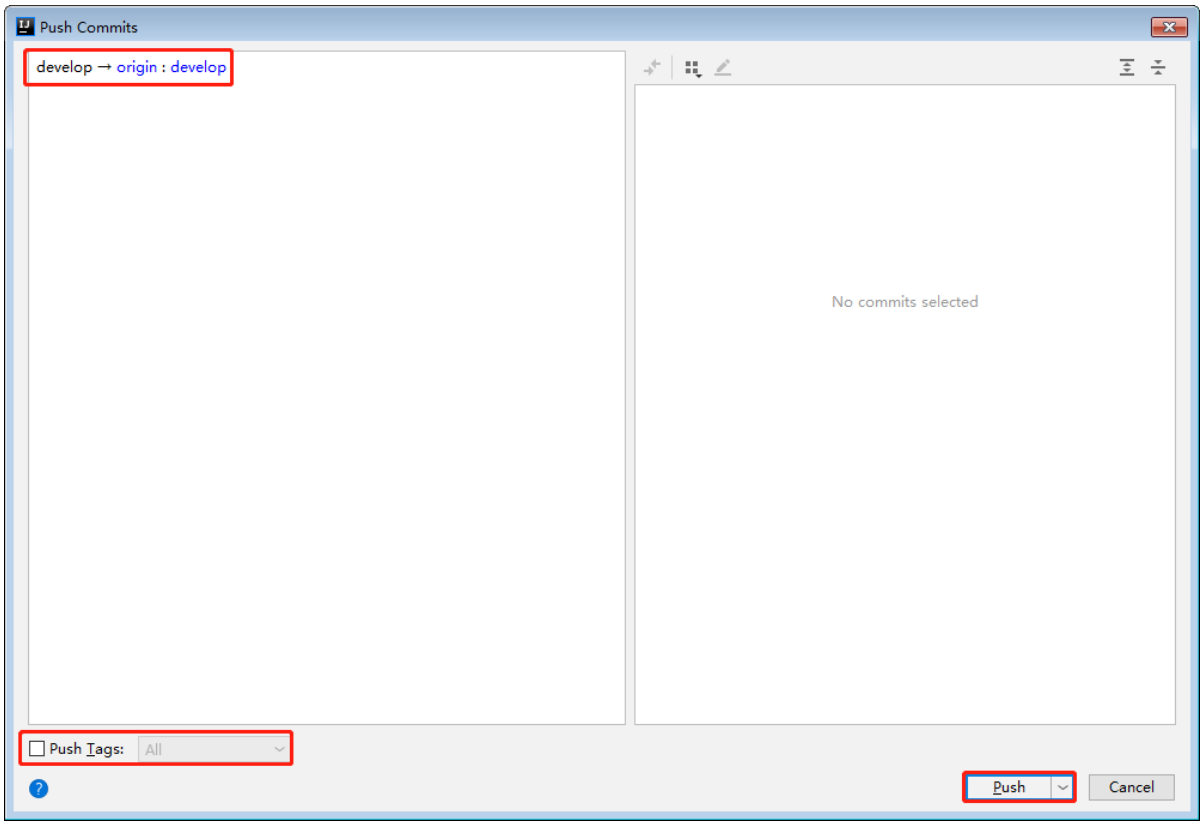
也可以在 IDEA 下方选项卡 Version Control 的 Local Changes 菜单下选中要提交的文件进行 Commit， 也可以右键 Default Changelist， 选择 Commit)





接下来进行 Push 操作，可以选择直接在项目上 右键--->Git--->Repository--->Push 。注意不要勾选弹出框左下角的 Push Tags ，否则会推送到所有远端分支。注意检查弹出框的左上角的远端分支名称是否正确。

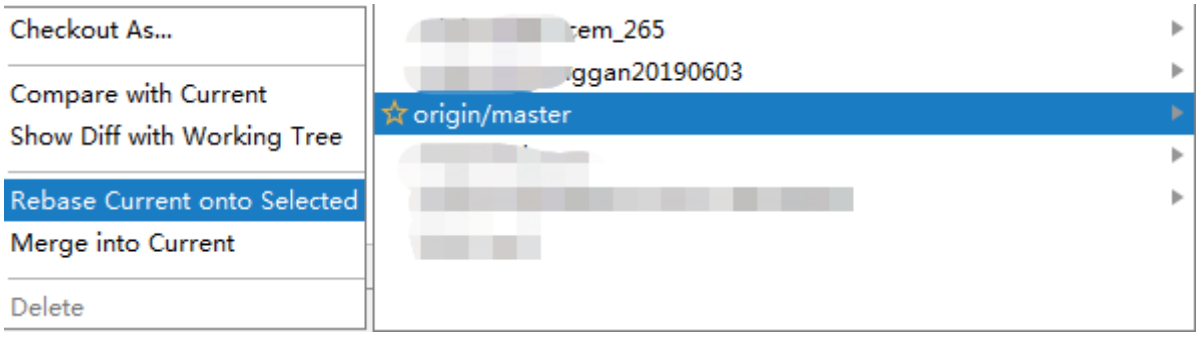




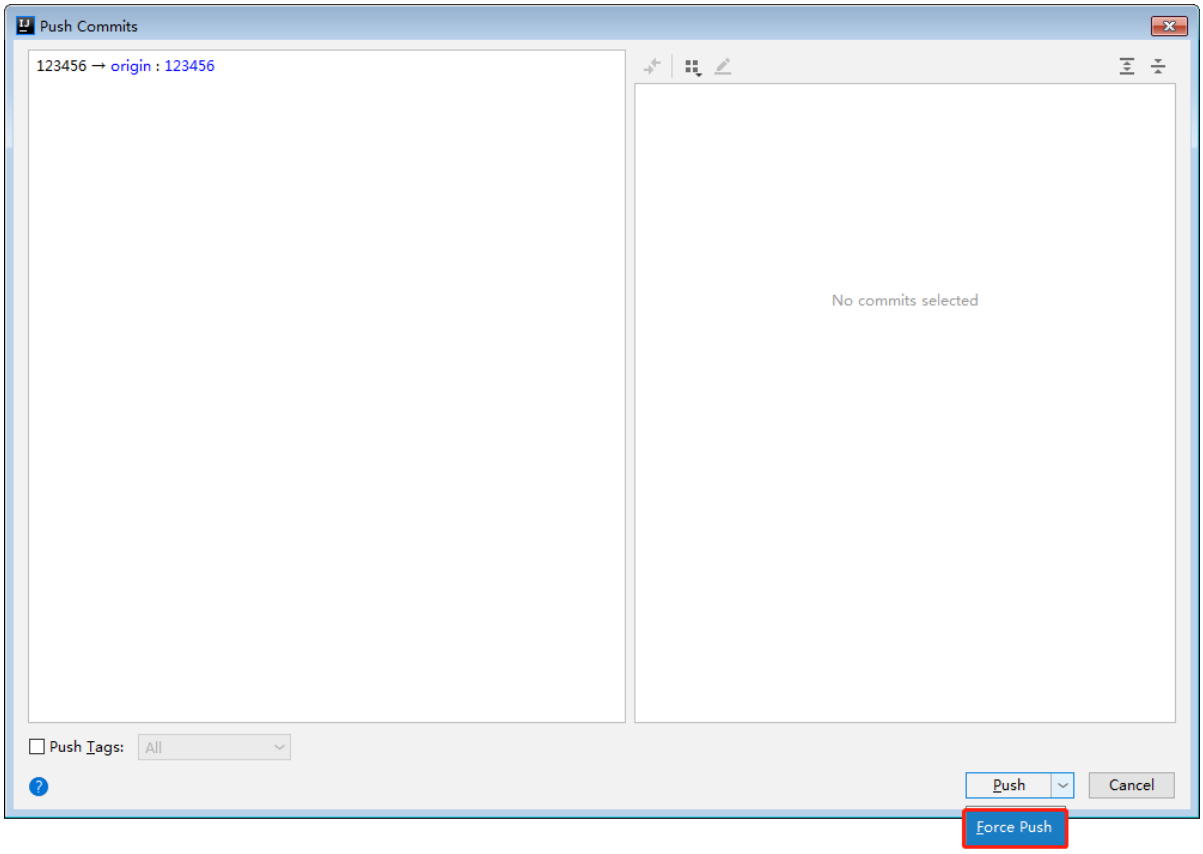
8. 开发分支合并主分支代码

本地分支为开发分支。将主分支的代码合并到开发分支上。
在开发分支上开发完成后，需要进行测试。但是在开发期间，可能其他同事已经在主分支上提交了代码，这时就需要 `rebase` 主分支的提交内容到开发分支上。

（1）首先 `rebase` 主分支远端仓库代码到开发分支的本地仓库。注意需要在右下角的分支列表中选择远端分支的主分支仓库进行 `rebase` ，这样是为了 `rebase` 到最新的主分支内容。 `Rebase Current Onto Selected` 是将当前重设为选定的意思。



（2）如果有代码冲突，会有弹窗提示进行代码合并和冲突处理。处理完后进行再次提交，最后需要进行强推到远端，对开发分支远端仓库的提交记录进行覆盖。（这个注意一定要强推，否则提交记录会出现分叉）。

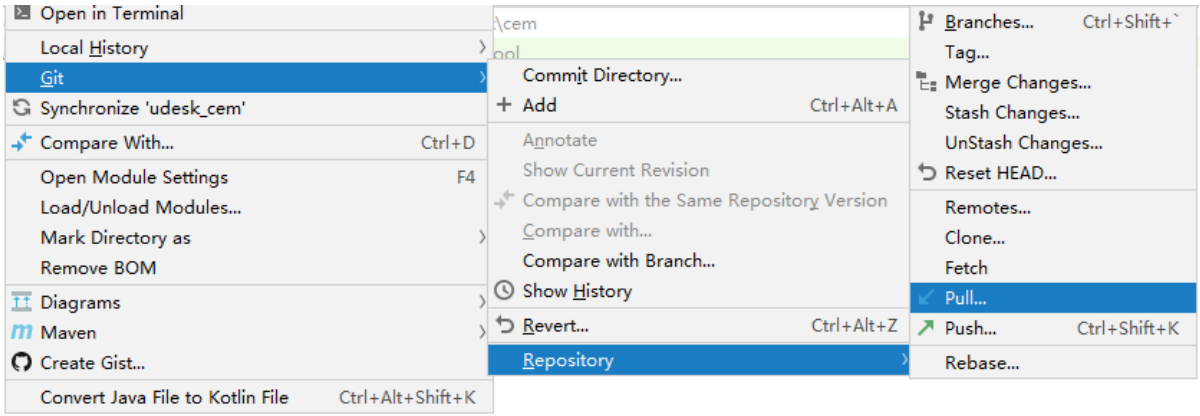


9. 主分支合并开发分支代码

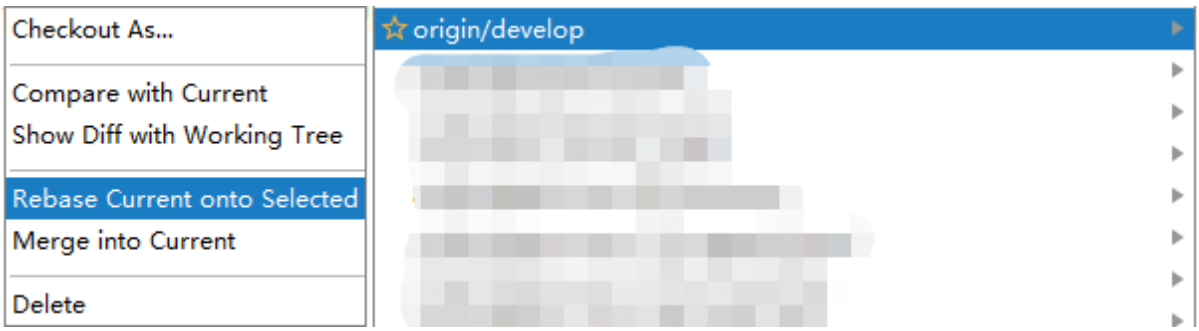
本地分支为主分支。将开发分支的代码合并到主分支上，最后使用主分支进行上线。

（1）首先切换本地分支为主分支，同步主分支远程仓库的最新代码到本地。

项目右键--->Git--->Repository--->Pull



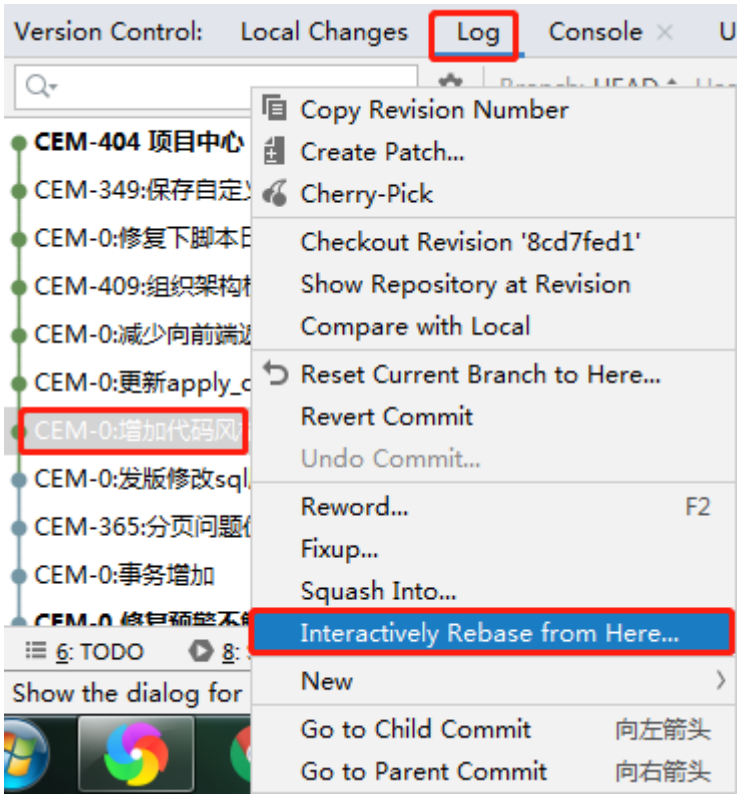
（2）接下来 rebase 开发分支远端仓库代码到主分支本地仓库。



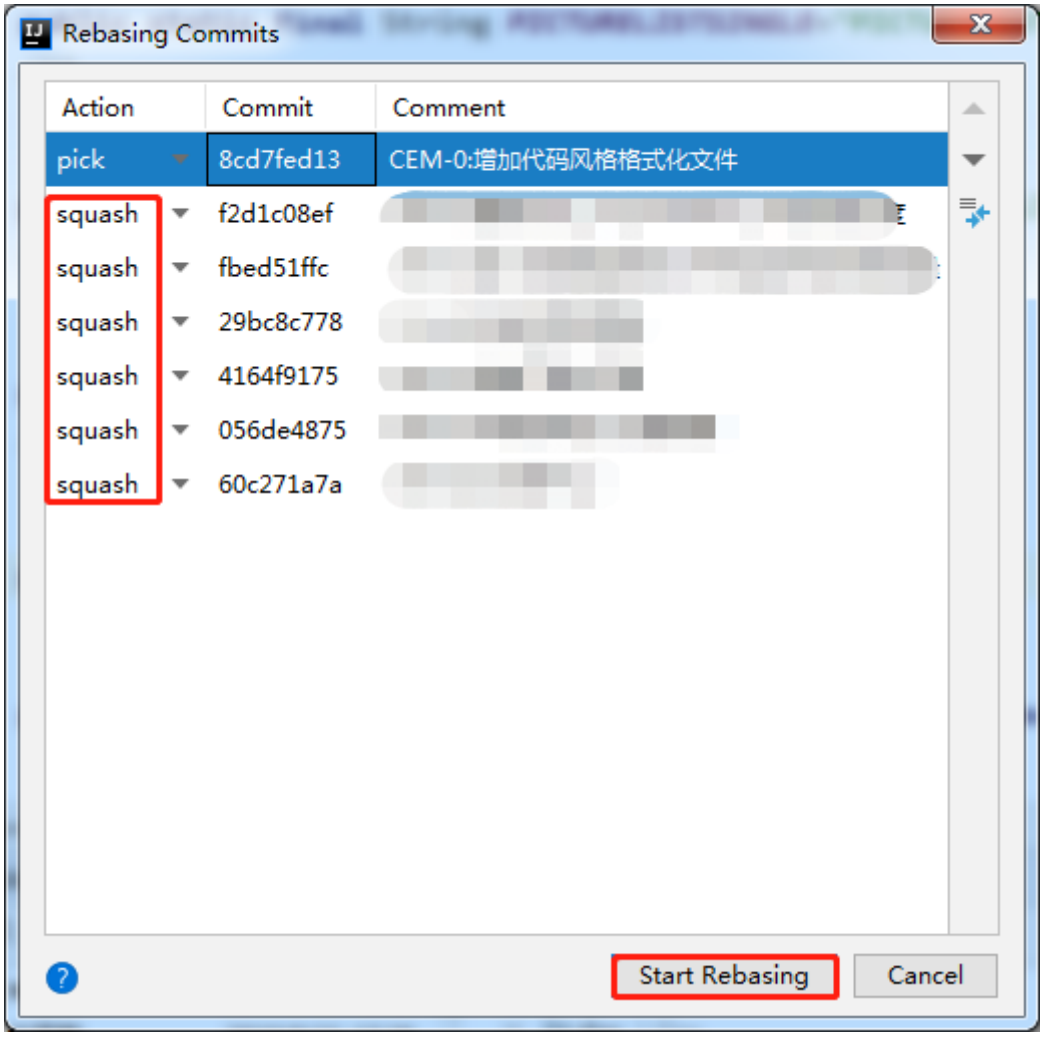
（3）进行 `push` 操作，将主分支的本地仓库代码推送到主分支的远端仓库。（这里不需要使用强推）

10. 代码提交记录进行合并操作

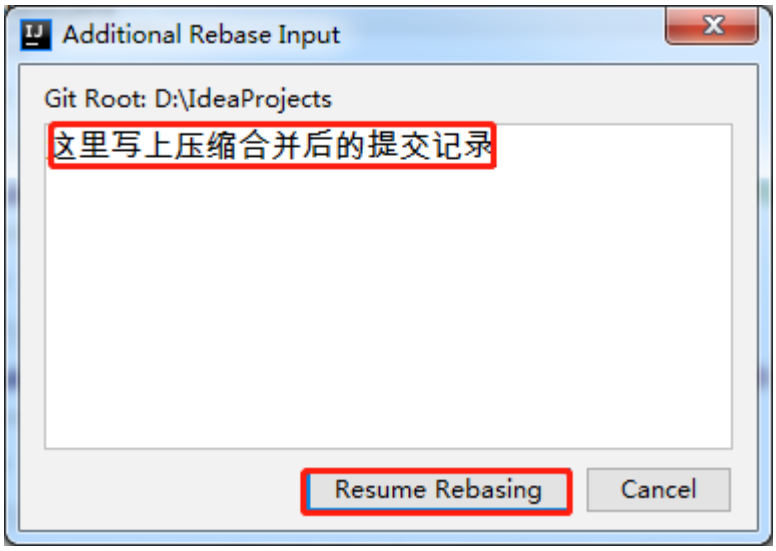
选中需要合并的记录，右键选择 `Interactively Rebase from Here` （交互式地从这里重新建立基地）



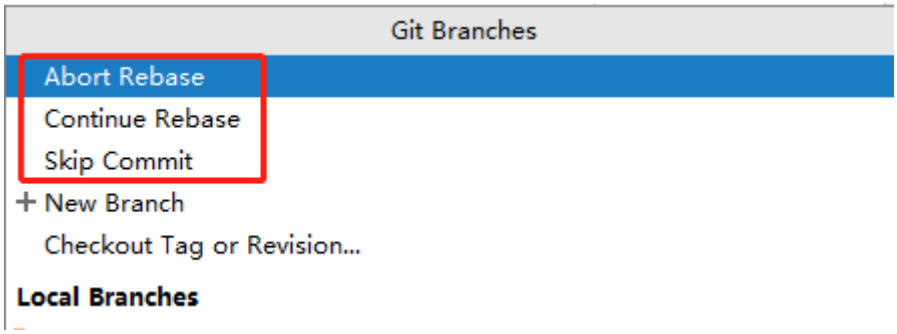
除了第一个选中的开始节点选择 `pick` 外，其他节点选择 `squash` （压缩）。



接下来写上压缩后的提交记录描述。点击 `Resume Rebasing` （重新开始），最后把压缩完的提交记录 `force push` 到远端即可。



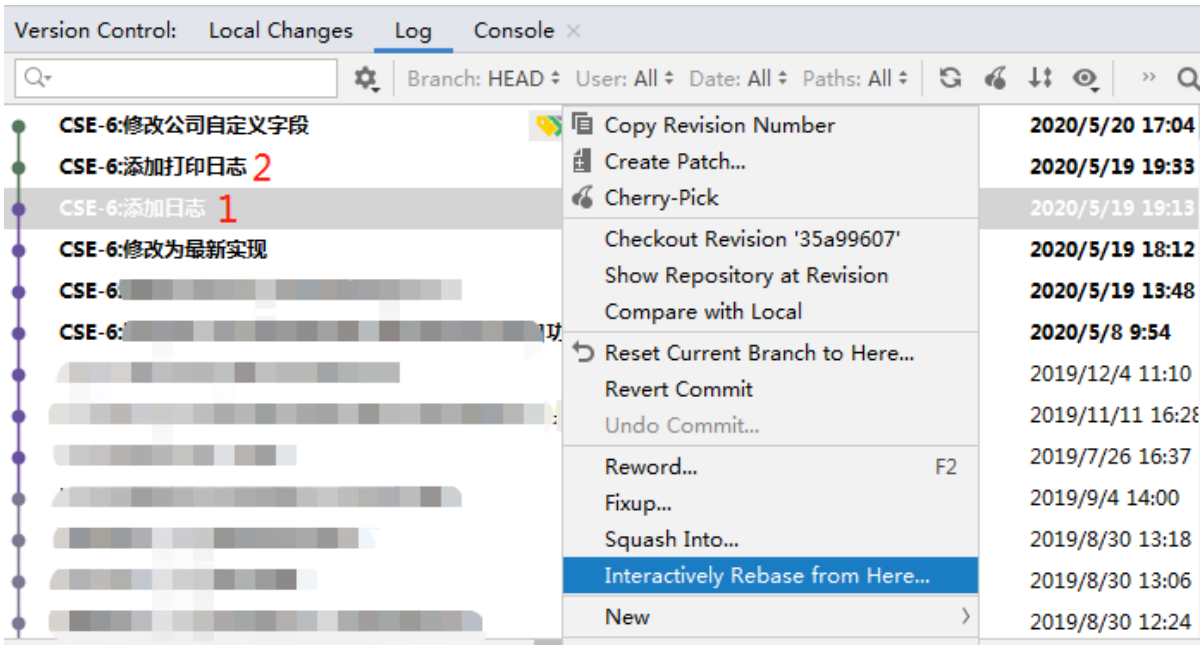
有时合并的过程中放弃了，会显示一直处于 `Rebase` 中，这时可以直接点击右下角分支列表中的 `Abort Rebase` ， 中止 `Rebase` 操作。



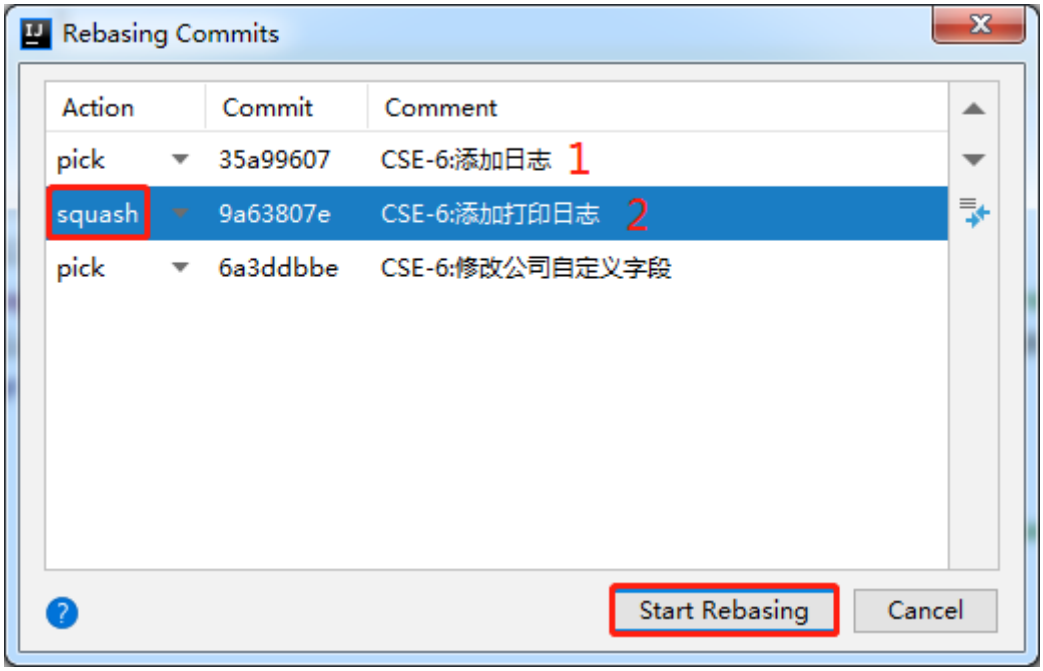
操作过程示例：

目标： 对多次提交记录进行合并。

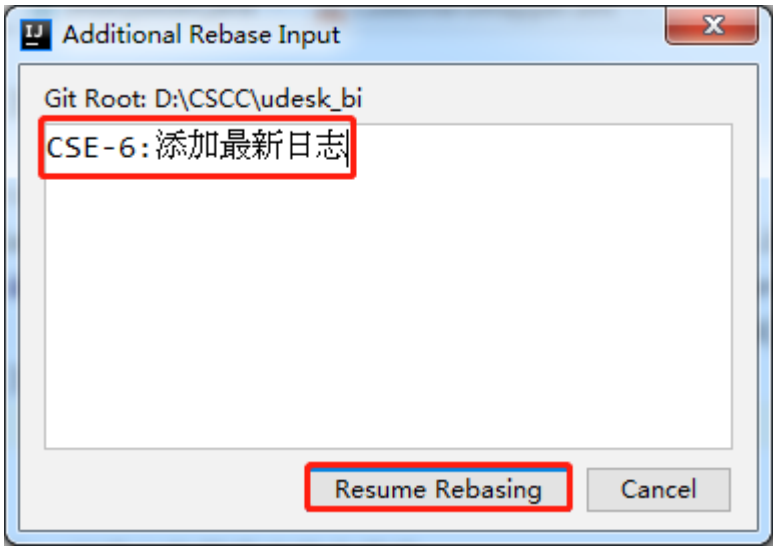
（1）比如我需要把2号提交合并到1号提交中，也就是把这两合并为一条提交记录。直接在1号提交上右键，选择 `Interactively Rebase from Here` （交互式地从这里重新建立基地）



（2）接着会弹出 `Rebasing Commits` 窗口，将2号提交选为 `squash` （压缩），点击 `Start Rebasing` 。



（3）接这会他弹出 `Additional Rebase Input` （重新输入描述信息）窗口，这里可以填写上压缩合并后提交记录的描述信息。点击 `Resume Rebasing` （重新开始）



（4）接下来如果你压缩的是远端的提交记录，则需要进行 `force push` （强推）到远端；如果压缩的只是本地的提交记录，则直接正常 `push` 即可。

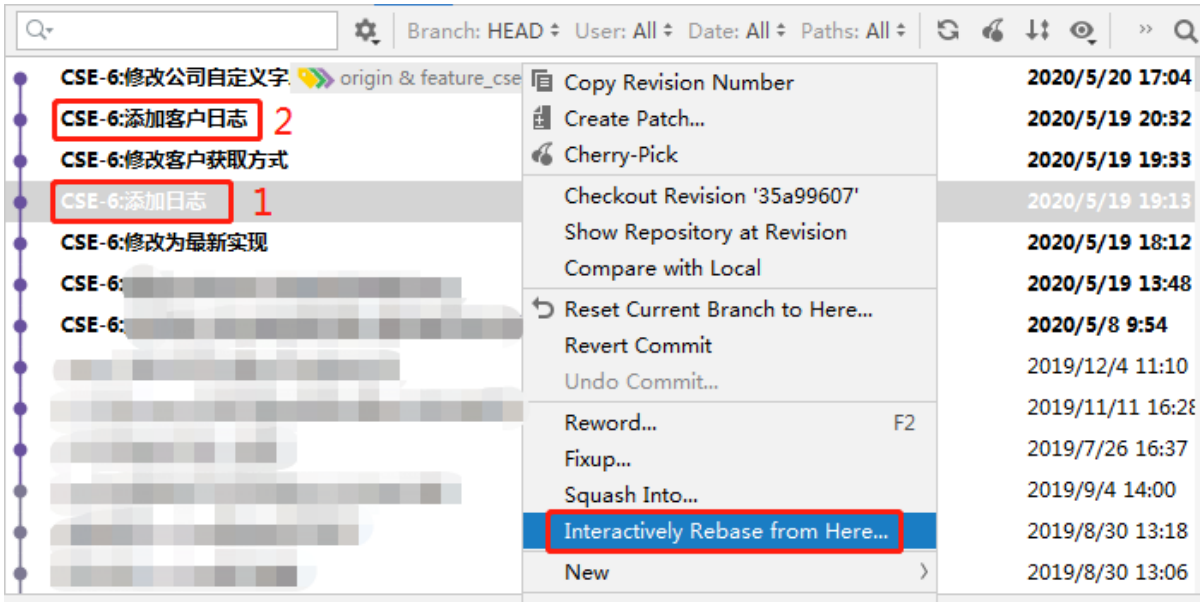
注意，Action 有以下几种选项：

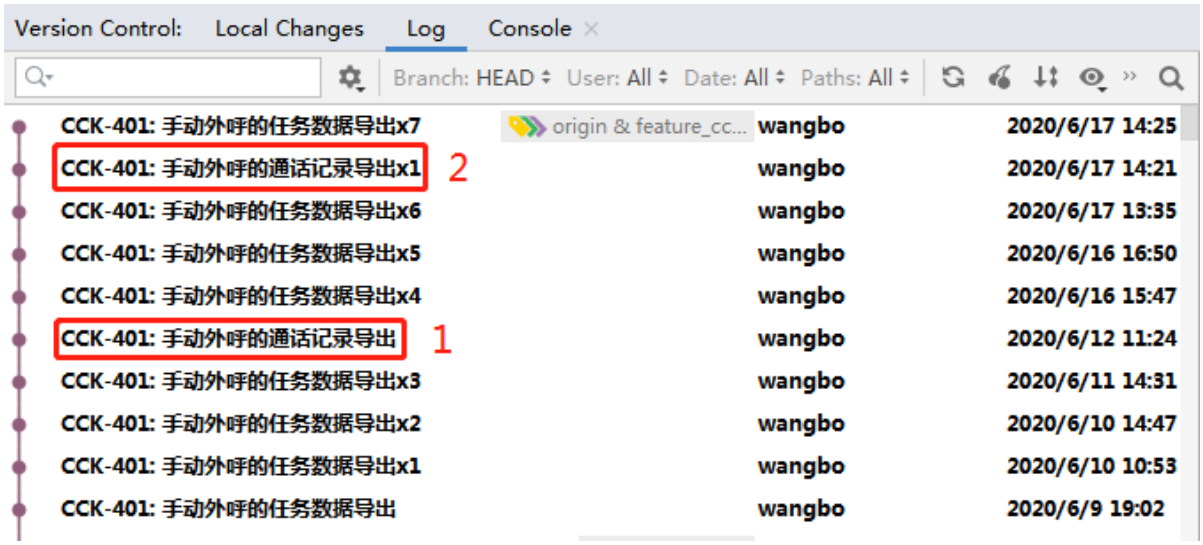
pick（选择） / edit（编辑） / skip（略过） / squash（压缩） / reword（重述） / fixup（修正）

- pick 就是 cherry-pick。
- reword 就是在 cherry-pick 的同时你可以编辑 commit message，它会在执行的时候跳出一个界面让你编辑信息，当你退出的时候，会继续执行命令
- edit 麻烦点，cherry-pick 同时 ，会停止，让你编辑信息，完了后，你要用 git rebase --continue 命令继续执行，相对上面来说有点麻烦，感觉没必要啊。
- squash，合并此条记录到前一个记录中，并把 commit message 也合并进去 。

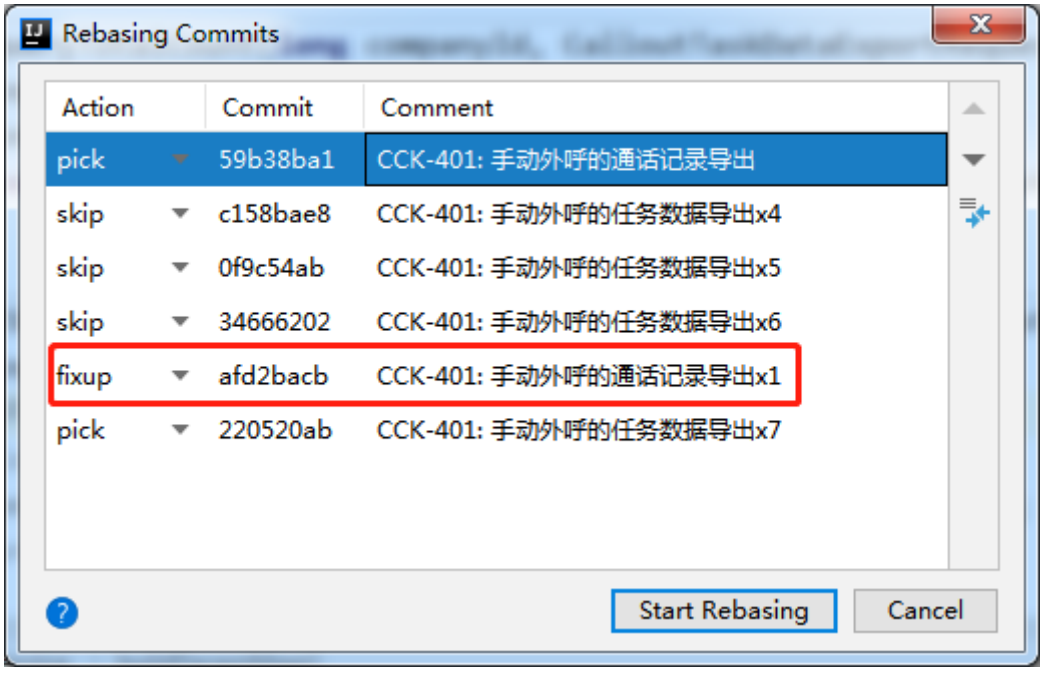
- fixup ， 合并此条记录到前一个记录中，但是忽略此条 commit message。

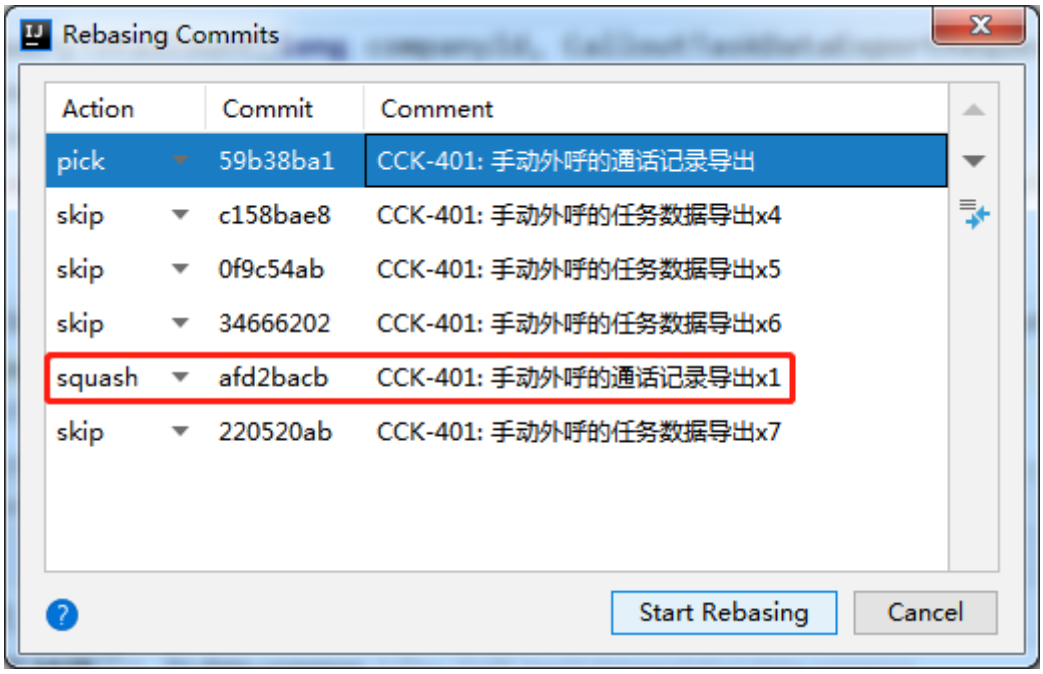
这种操作需要合并的提交是连续的，一般用于把同一个开发任务的多次连续提交合并为一次提交。我试了下面这种不连续的提交记录合并操作，这样操作的话会把2号提交合并到它上面的那个提交中（也就是图中的“CSE-6:修改客户获取方式”），并没有合并到1号提交中。





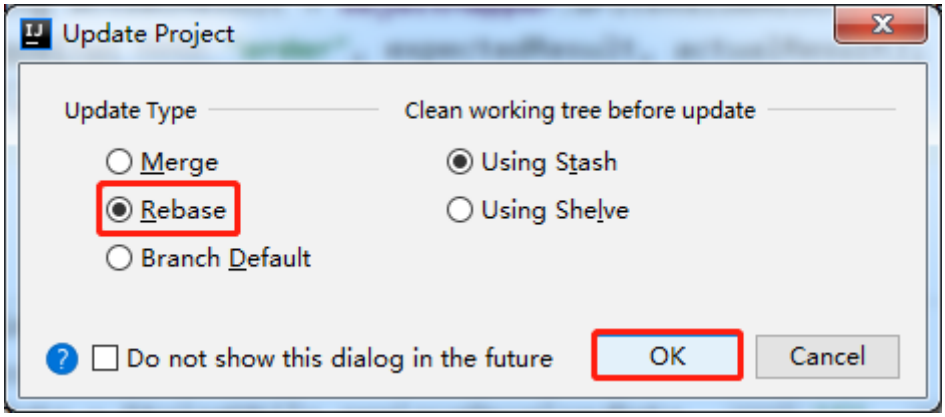
把2号提交记录合并到1号提交记录：



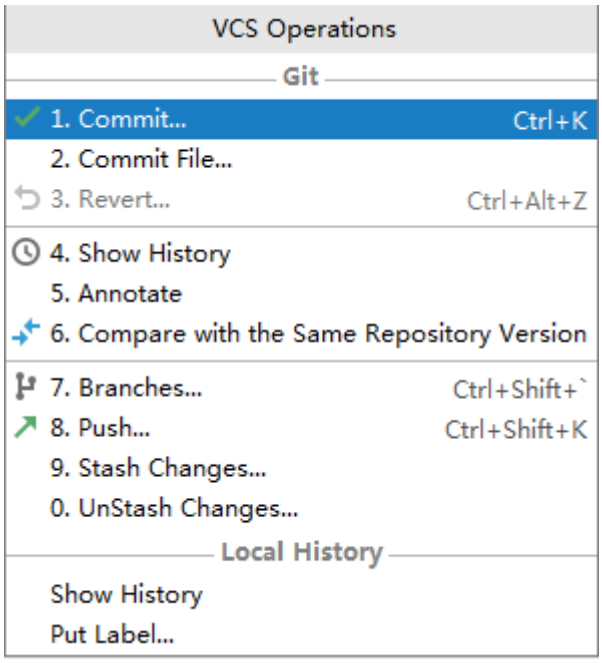


11. 常用操作的快捷方式

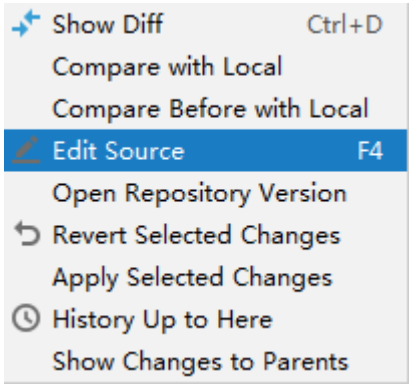
如果你使用的是 Git 管理项目，每次更新远端仓库代码都需要 选中项目名--->Git--->Repository--->Pull... ，操作比较麻烦，可以直接使用**Ctrl+T**快捷键，该快捷键可直接从 Git 更新项目，免去了多次点击选择 Pull 的操作。注意弹出框选择 Rebase 。同样，**Ctrl+K**能直接进行项目提交，免去了多次选择进行提交的操作。



使用 **Ctrl+`** 可直接显示版本控制中的常用操作菜单。可看到一些常用的快捷键。



使用 **F4** 可以直接打开提交记录中的文件，进行编辑。

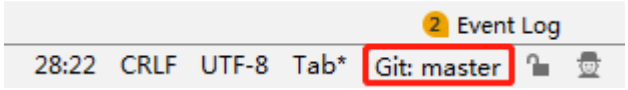


12. 开发分支部分合并到主分支

使用到**Cherry-Pick**。

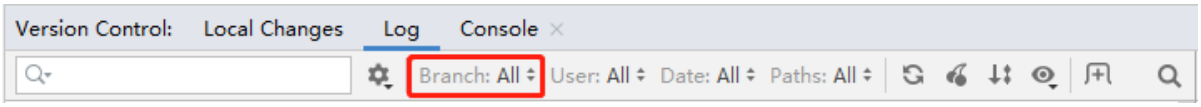
目标：将 `develop` 分支的**部分**提交记录合并到 `master` 分支。

（1）首先切换到需要进行代码合并操作的分支，比如我们是 `master` 分支。



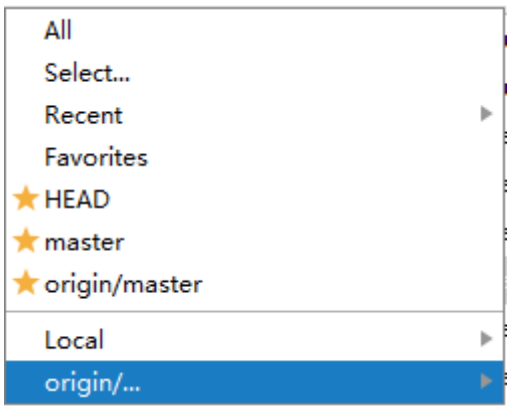
此时的 `Version Control` 中的 `Log` 下的 `Branch` 选择的是 `All` ，表示显示的是所有分支

的提交记录，应该可以看到好多的分叉记录。

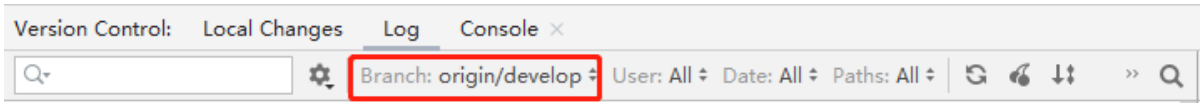


（2）将 Log 下的 Branch 切换到所要合并的代码所在的分支。比如我们是 develop 分支。

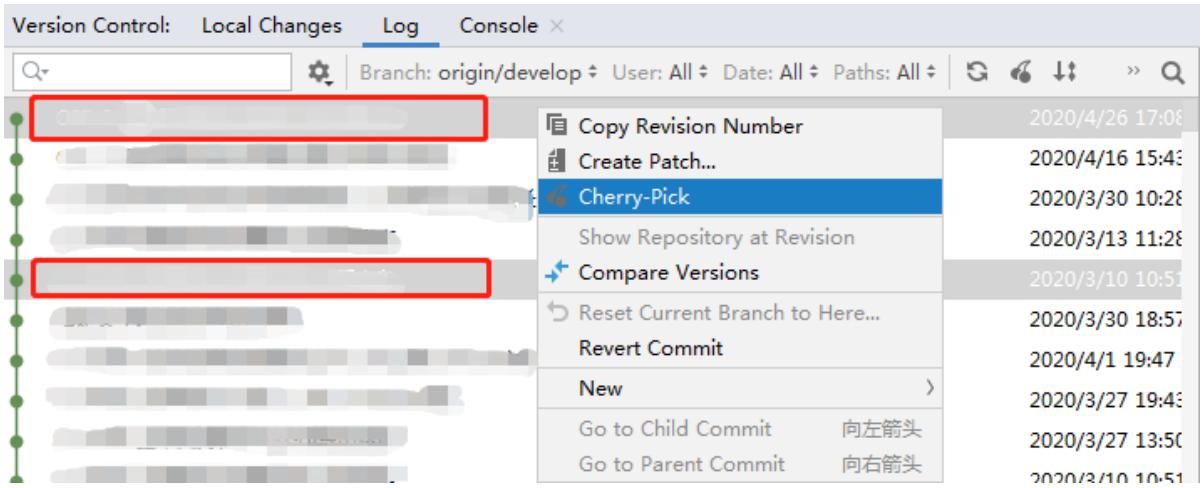
鼠标左键点击 Branch ，点击 origin/... ，会出现远端所有分支的列表，在列表中选择目标分支即可。



这里选中 develop 分支，此时 Log 下显示的就是 develop 分支的所有提交记录了。



（3）接着就可以直接按住 Ctrl ，选择需要合并的提交记录了，选中后右键，选择 Cherry-Pick 即可。后续就是正常的处理冲突和提交流程了。

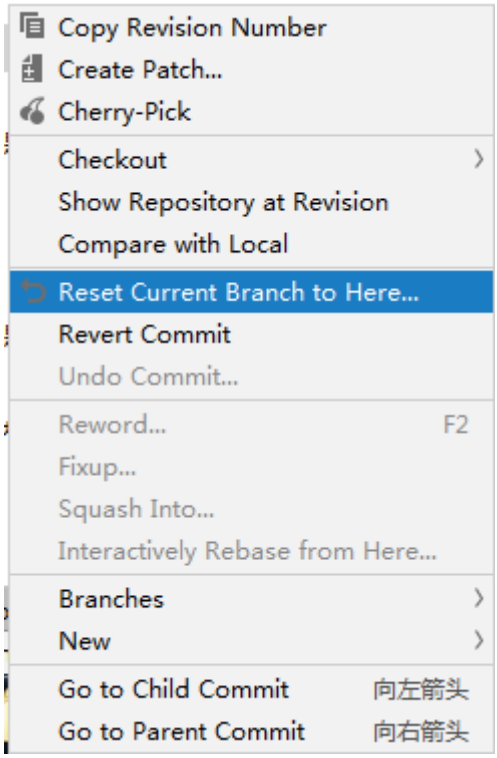


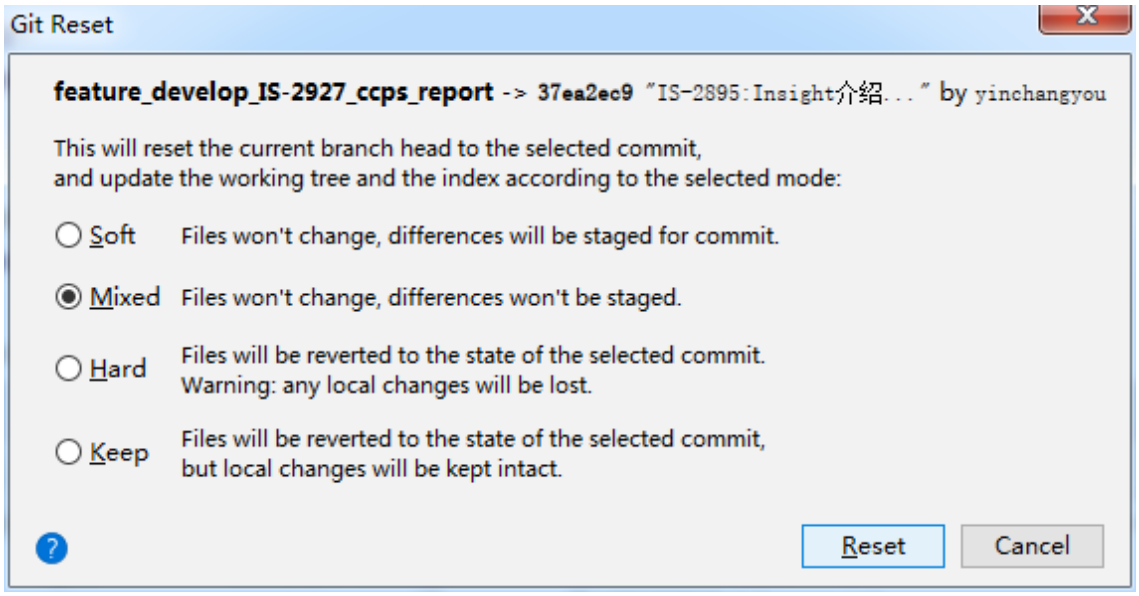
13. 主分支被强推的解决办法

需要使用到**Reset Current Branch to Here…**。

如果有人强推了主分支，此时我们本地的就无法提交，会产生冲突。

- 这时如果本地没有其他修改，可以直接删除本地代码，重新检出。
- 还可以询问下同事强推的记录开始的地方，我们可以做一个回退。也就是在提交记录中右键具体的提交记录，选择“将当前分支重置到这里”来实现。之后再进行一次 Push 即可。





翻译一下这个界面：

Thie will reset the current branch head to the selected commit, and update the working tree and the index according to the selected mode.

将当前的分支头重置为所选的提交，并根据所选模式更新工作树和索引。

Soft: File will not change, differences will be staged for commit.

文件将不会更改，差异将暂存以提交。

Mixed: File will not change, differences will not be staged.

文件将不会改变，差异将不会暂存。

Hard: File will be reverted to the state of the selected commit. Waring: any local changes will be lost.

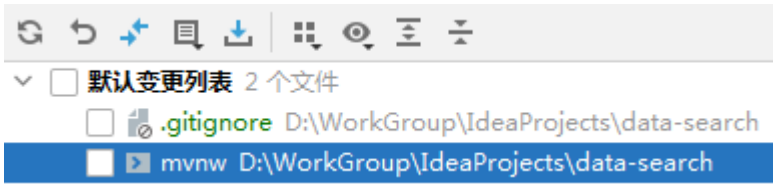
文件将恢复到所选提交的状态。警告：本地的所有改变都将丢失。

`Keep: File will be reverted to the state of the selected commit. but local changes will be kept intact.`

文件将恢复到所选提交的状态。本地的改变将会被保留。

14. 移除被添加到版本管理的文件

场景：有文件手误操作添加到了版本管理，如下图中的 mvnw 文件

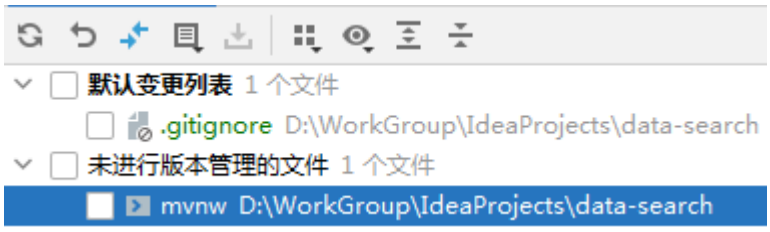


现在想要将该文件从被管理文件中删除，图形界面没有找到操作方法，只能通过命令行来操作了：

```
D:\WorkGroup\IdeaProjects\data-search>D:\WorkGroup\Git\bin\git rm --
rm 'mvnw'

D:\WorkGroup\IdeaProjects\data-search>
```

可以发现，mvnw 文件已经在**未进行版本管理的文件**列表中了。



无关内容：对于没有 `.git` 目录的文件夹下，会有以下提示，直接使用 `git init` 就可以了。

```
D:\WorkGroup\Git\bin>git rm --cache -r .mvn
fatal: not a git repository (or any of the parent directories): .git
D:\WorkGroup\Git\bin>git init
Initialized empty Git repository in D:/WorkGroup/Git/.git/

D:\WorkGroup\Git\bin>
```

15. IntelliJ IDEA 使用 Git 慢的问题

最近公司给开发机断网了，就是说不能连接外网了，然后我就发现我的 IDEA 中执行 Git 相关的操作的时候特别的慢，很是抓狂。断网前不是这样的，执行很快的。下面是 @遥星 的文章，解决了我的问题，这里转载下，做个记录。

我是按照下面方法修改的，直接改了 IDEA 安装目录下的 bin 目录下的那两个文件的后缀。

起因

在公司电脑上使用 IntelliJ IDEA 的时候发现操作 Git 特别的慢，status、fetch、pull、checkout、commit 等基础操作都执行的特别慢，下方的 Task 进度条一直处于等待中，等待差不多 10 秒多的时候才开始执行进度。最难以忍受的是，在 Settings 里面检查 Git 的版本操作都需要接近 20 多秒的时间，才能返回结果。

结论

花了几天时间跟踪 IDEA 的执行日志以及翻看 IDEA Git 插件的源码，偶然发现 IDEA 在执行 Git 命令的时候，其实是调用了一个 exe 程序来执行命令的，问题就出在这个 exe 程序上面，猜测应该是公司给初始化的系统本身有什么域设置不合理导致在我这个系统版本执行的特别慢，就像是队列里面排队等候一样，最终试着把这个 exe 改了后缀名，问题就解决了。

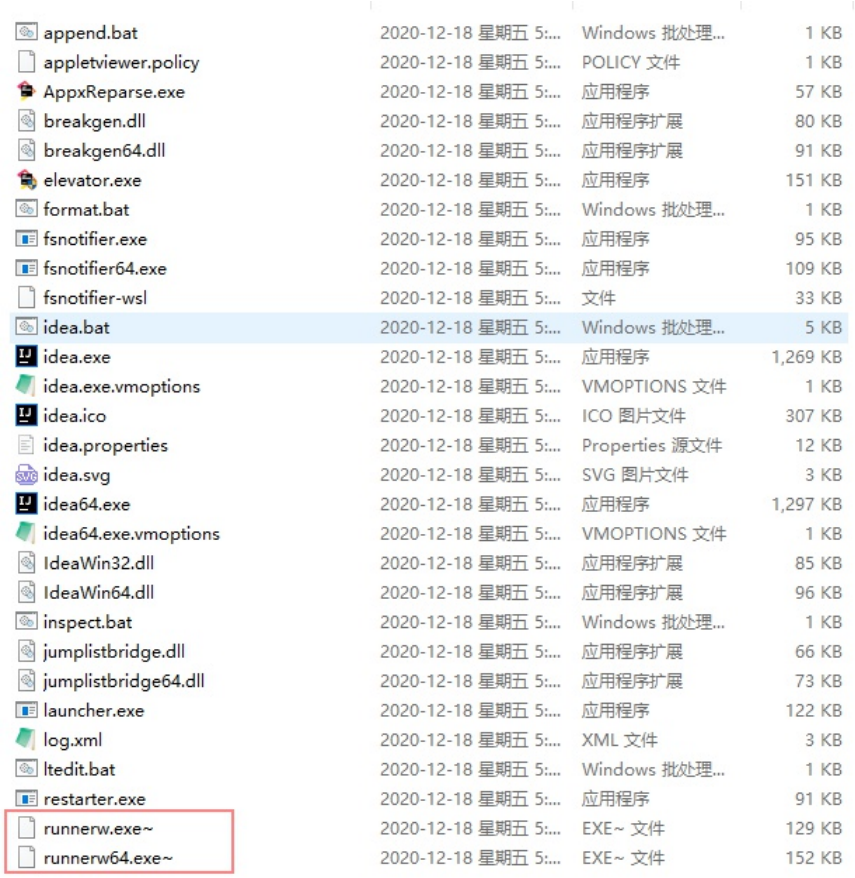
解决方法

昨天公司看到有小伙伴在大群里问运维同事能不能处理一下这个问题，我突然意识到居然不止我一个人遇到了这个问题，所以分享到知乎上，供遇到这个问题的朋友们参考。

处理方案：只需要删除掉 JetBrains 系 IDE 安装目录下的 runnerw.exe（64 位系统是 runnerw64.exe）或者直接更改后缀名，反正只要让 IDE 找不到这个 exe 就可以，然后这个问题就解决了。不需要重启 IDE ，即时生效，如果你使用的是 JetBrains ToolBox 来安装升级 IDE 的话，每次 IDE 更新，你都需要重新去屏蔽那两个文件；如果是官网下载安装包安装的，改一次就行了。

我的这两文件是在下面目录中：

```
D:\work\IntelliJ IDEA 2021.1.2\bin
```



关于 Toolbox

如果你是 Toolbox 安装的 IDE，要找到 IDE 的安装目录的话，你可以在开始菜单搜索 IDE 快捷方式右键"打开文件所在的位置"；也可以直接在 Toolbox 里找到你的 IDE，然后点击右侧的"三个点"更多菜单，里面选择"Settings"进入 IDE 设置，依次点击"Configuration"展开选项，找到"Install lacation"，可以直接复制安装路径，或者点击下方的"Show…"直接打开安装目录。

后续

已确认，卡顿和执行慢的根源是某些钩子程序，我这边的是"IP Guard"导致。

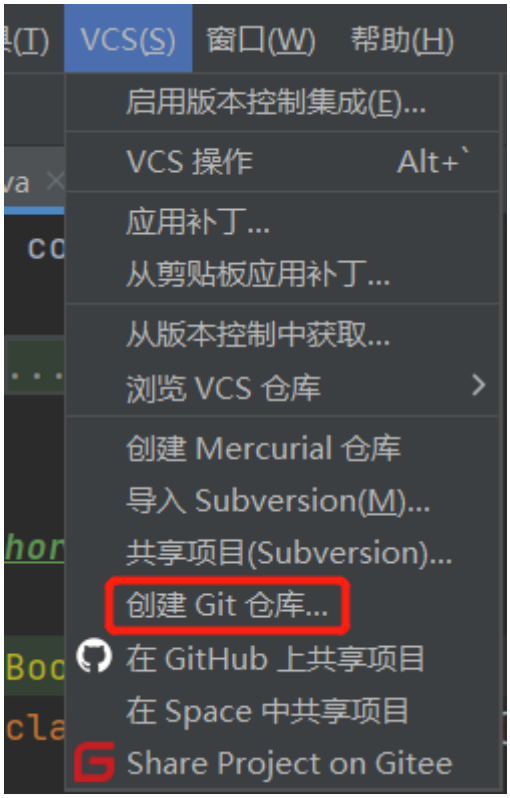
如果你是公司电脑，按照上面提到的方法处理就行，毕竟公司的没办法。

如果你是自己私人电脑，那就想办法删掉这个"IP Guard"吧，这个玩意是用来监控你的电脑的，有些公司是强制要求，然后通过域设置推送到你的计算机上的，那就不能删除。如果是私人电脑，百度一下怎么删除，按照文章，一般是通过 pe 进去，然后删掉那些文件就行了。

关于如何查看你的电脑是否也被"IP Guard"所支配，去看看你 C 盘系统目录就知道啦，具体路径的话，看看有没有"winrdlv3.exe"这个程序就好了，参考路径如下：

```
C:\Windows\System32\winrdlv3.exe
C:\Windows\SysWOW64\winrdlv3.exe
```

16. 本地新建项目添加 Git 管理



如果有远程仓库，提交的时候配置远端地址即可。