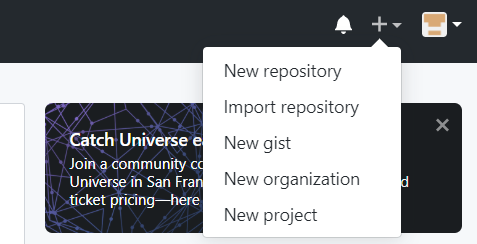
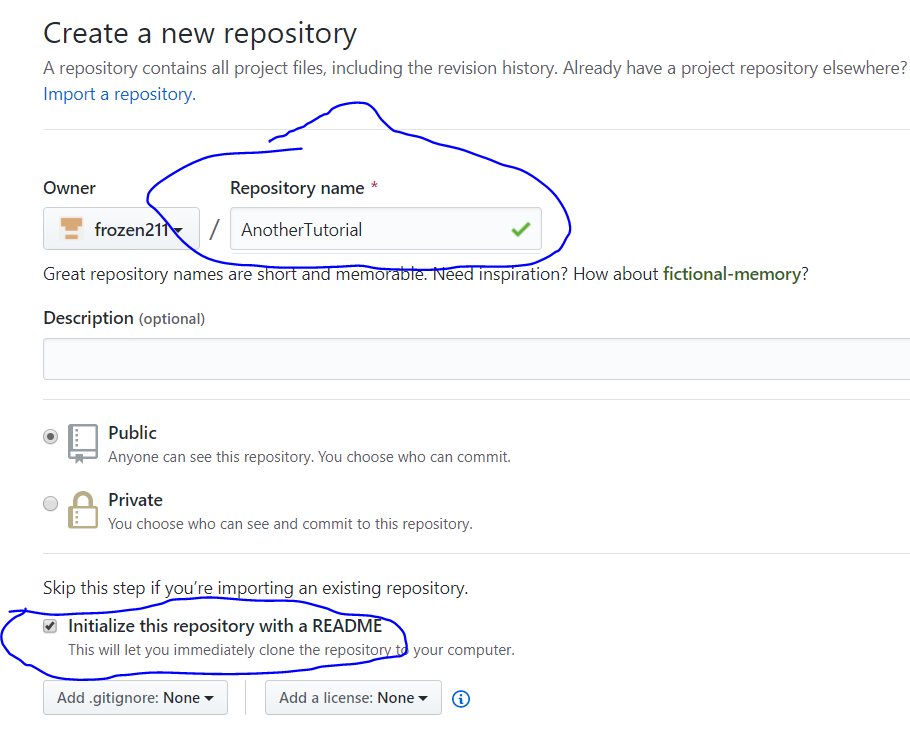
在开始看程序之前，先学习一下github的用法。

创建新repo





Windows的命令行里，cd还是照用，但ls对应的是dir。

Git指令：

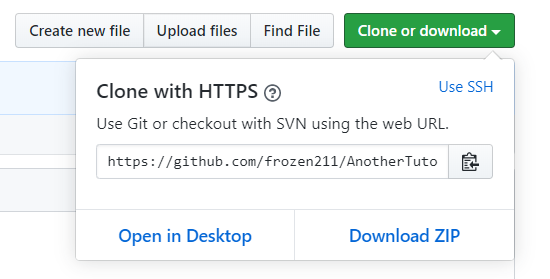
Pwd：和linux一样，显示当前路径。

Cd：和Linux一样。

Mkdir: 和linux一样。创建文件夹。

Ls：和linux一样。

在网页上，复制repo的网址。



我在yoga的D盘建立一个文件夹mygit，依靠git GUI快速将其设为一个repo。

Dell的E盘也有repo。

在git bash里输入

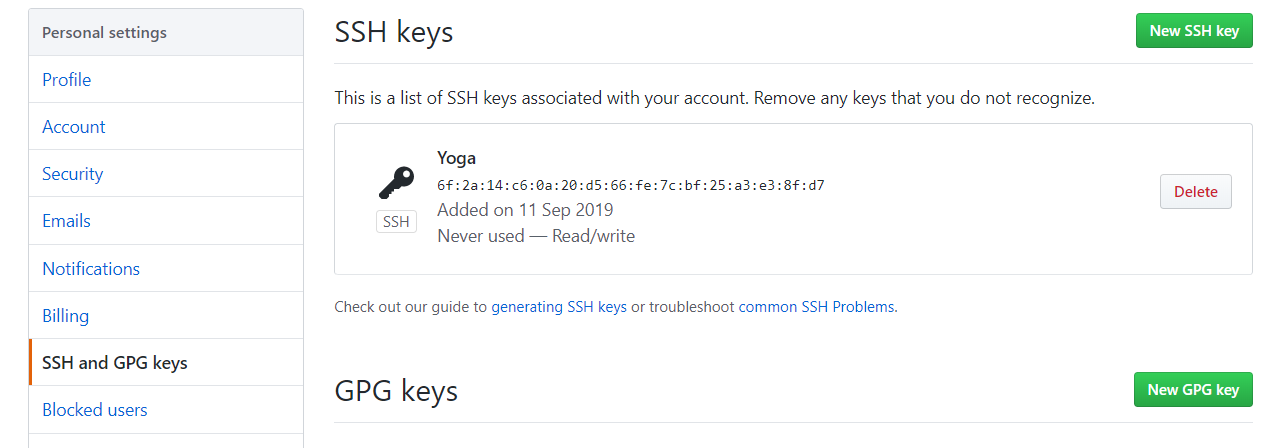
Git clone <https://github.com/frozen211/AnotherTutorial.git>

就把那个repo下载到本地了。

Mygit是本地仓库，可以在本地工作。

通过SSH的方式，可以和我网站上的repo同步。

我通过git GUI生成了ssh key，将rsa.pub里的内容复制。在网站上，我在settings里添加ssh keys。把rsa.pub里的内容（也就是公钥）添加进去。



把本地库的内容推送到远程，使用 git push命令，实际上是把当前分支master推送到远程。

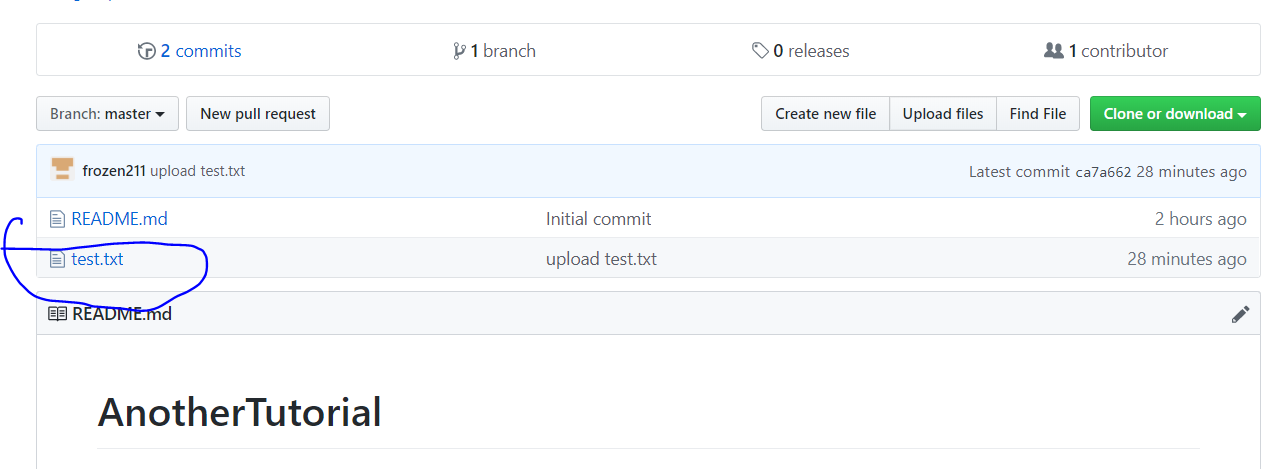
git push的一般形式为 git push <远程主机名> <本地分支名> <远程分支名> ，例如 git push origin master：refs/for/master ，即是将本地的master分支推送到远程主机origin上的对应master分支， origin 是远程主机名，第一个master是本地分支名，第二个master是远程分支名。

我在第一次Push的时候用的指令是：

git push -u origin master

使用 -u 参数指定一个默认主机，这样后面就可以不加任何参数使用git push

此时再看网页这边，我在本地建立的文件都上去了：

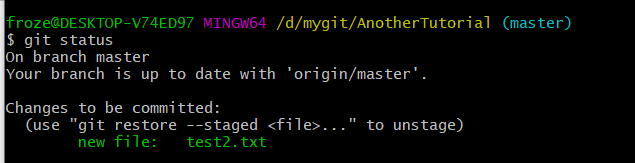


我又在本地建立了一个test2.txt。

这时在本地的git bash使用

Git status

我就发现有个新的文件需要上传：



使用

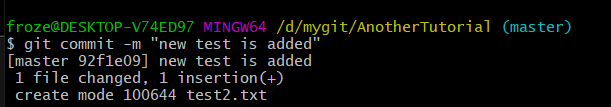
Git add test2.txt

将需要上传的文件放入暂存区。



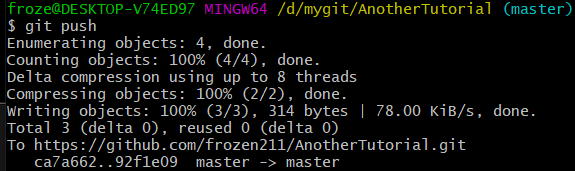
使用

Git commit -m “我要输入的message”

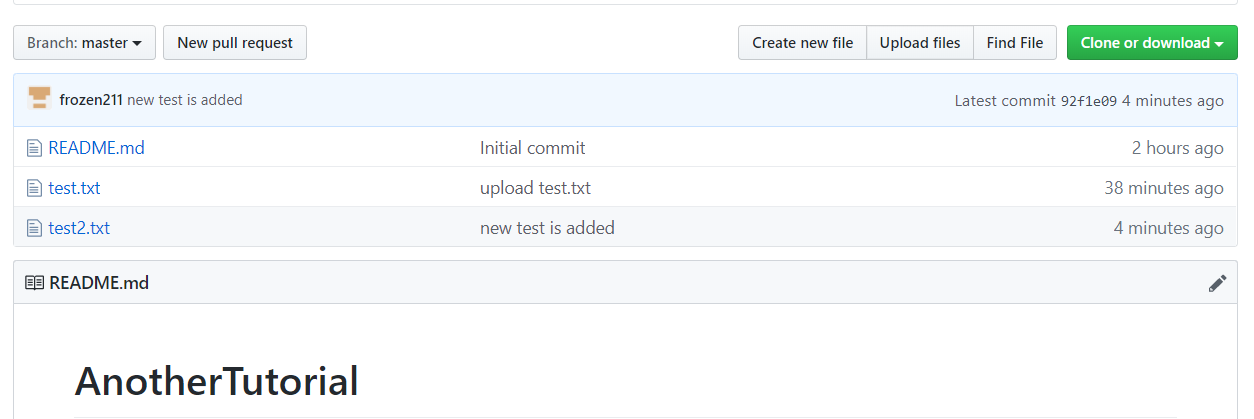


此时就在本地repo做好了更新，但还没同步到云端。

此时再用git push就可以完成同步了。

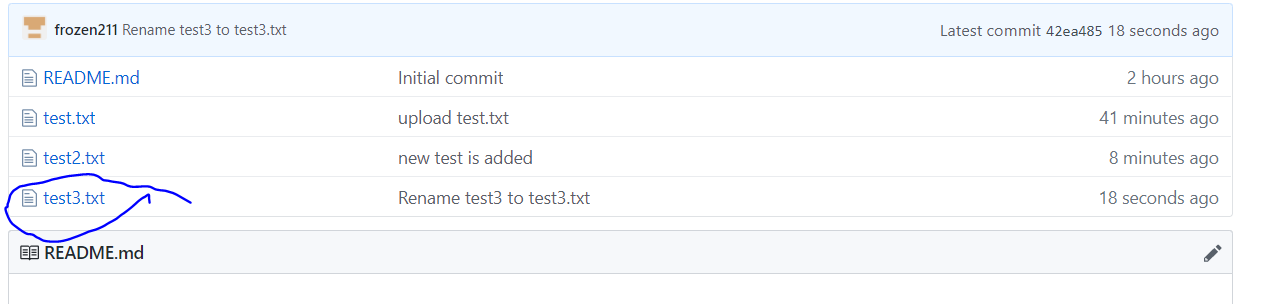


现在在网页上，test2也在了。



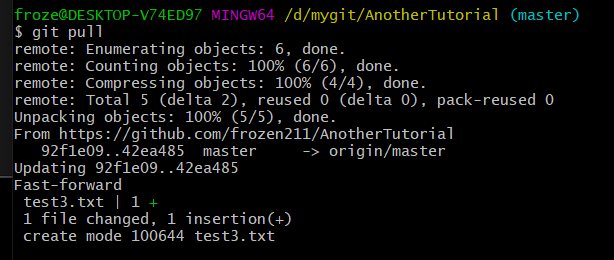
那么，如果我们要从云端同步到本地呢？

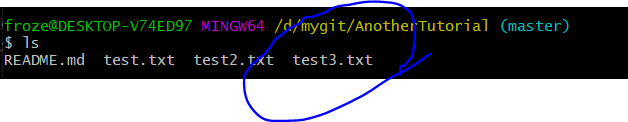
我在云端建立了一个新文档test3。



在本地使用

Git pull

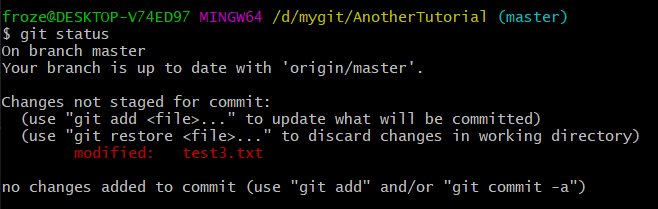




此时本地就自动获得test3了。

现在我们打开test3，对其进行修改。

修改后再检查状态就会发现，有个文件没同步。



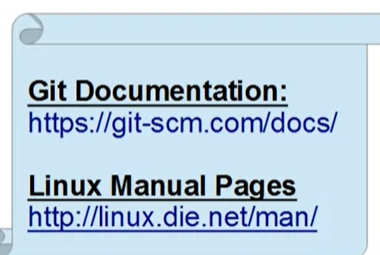
一样，用add和commit以及push来上传。

Git add -a可以将本目录一切都加进去。

就算是删除文件，也要add再commit再push云端才知道。

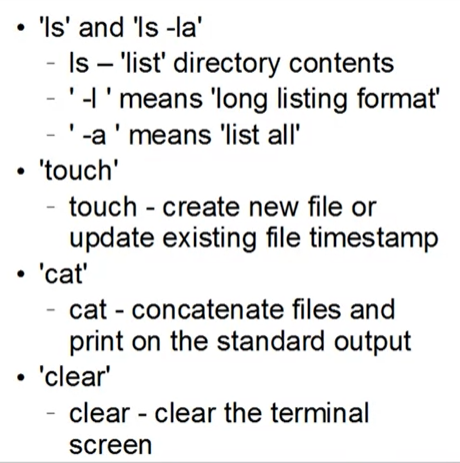
那么现在我就知道如何使用git来进行基本工作了。

下面是另一个gitlab教程，来自范德堡大学的Android development课。

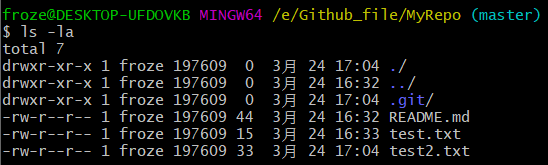


这里可以查到git的指令。

我们先讲一些常用指令。



Ls不用说了。

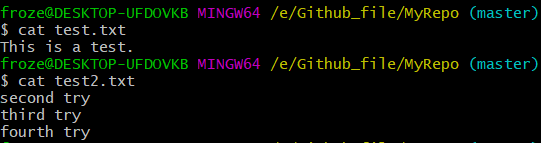


值得注意的是这个蓝色的.git，它是这个文件目录下的一个隐藏文件夹，用来存放各个版本的repo。

Touch是用来创建新文件的。

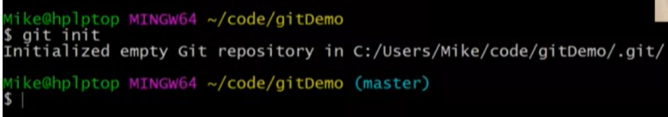


Cat用来输出文件里的内容。



Clear清屏。

使用git init可以创建一个master分支：



用touch创建一个README后，检查status会发现它没有同步。

用add添加：



之后用commit和push即可同步到云端。

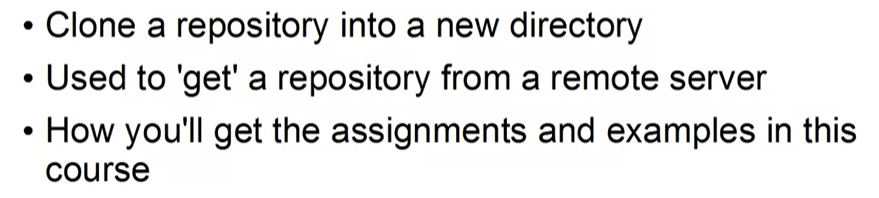
Git log可以看到过去修改记录。

如果停止在commit这一步，就只是本地仓库的各种操作。

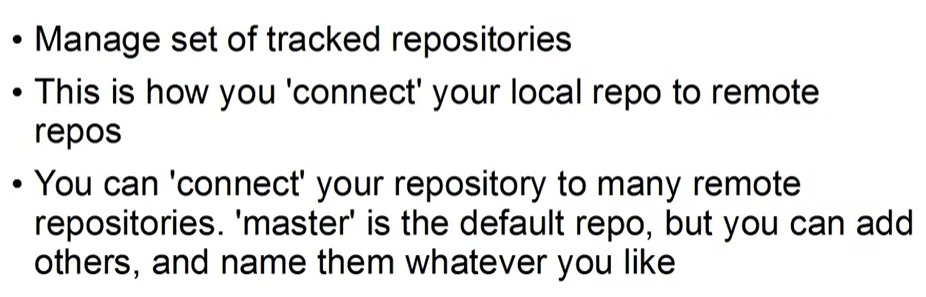
像clone, push, pull这样的操作，属于远程仓库的操作。

Gitlab.com可以免费获取私人repo。

Git clone



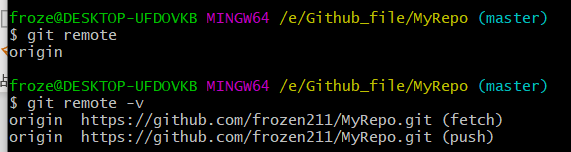
Git remote



1.git remote 不带参数，列出已经存在的远程分支

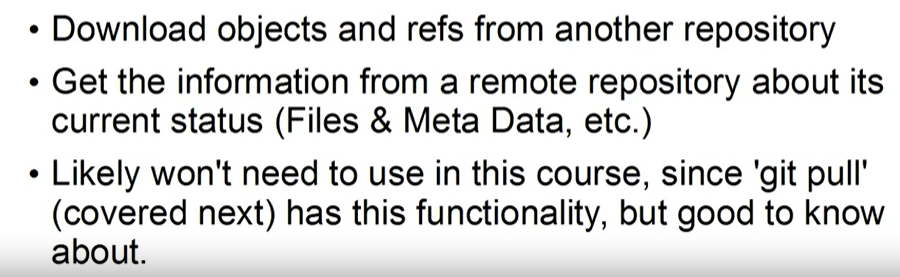
2.git remote -v | --verbose 列出详细信息，在每一个名字后面列出其远程url，此时， -v 选项(译注:此为 –verbose 的简写,取首字母),显示对应的克隆地址。

3.git remote add url 添加一个远程仓库



可见我这里存在的远程分支只有我自己的那个MyRepo。

Git fetch



一个类似git pull的指令。

1.git fetch 不带参数

将某个远程主机的更新，全部取回本地。默认情况下，git fetch取回所有分支的更新。

2.git fetch <远程主机名> <分支名>

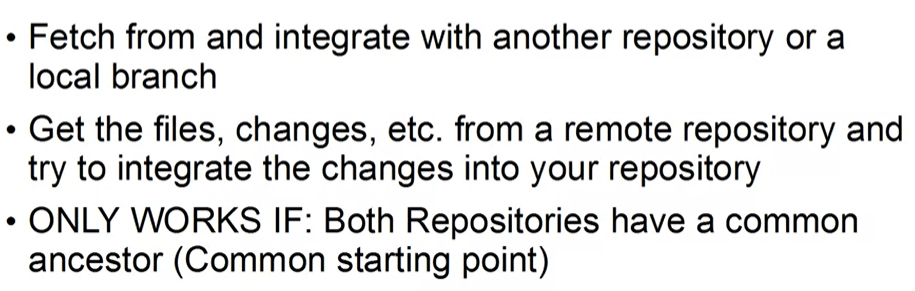
指定取回某个分支的更新，例如，取回origin主机的master分支。

git fetch origin master

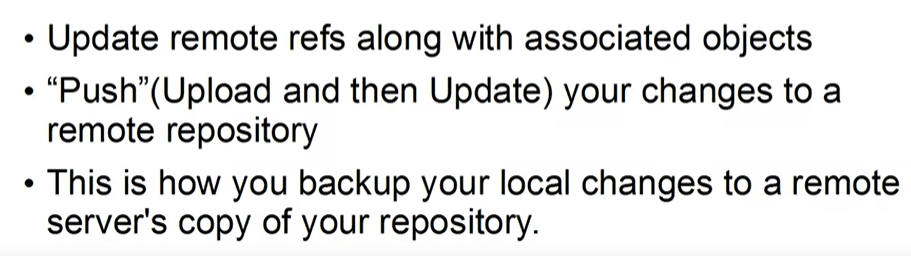
也可以不带分支名，此时将某主机的全部分支更新取回。

git fetch <远程主机名>

git pull



Git push



如果没有修改权限，就不能push，只能用另一个指令提供建议修改。

最常用的还是Push和pull。

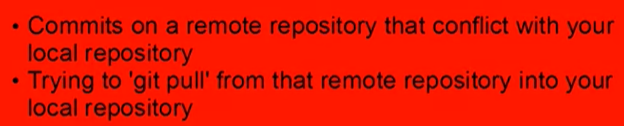
Git 的branch比较复杂，这里有网址：

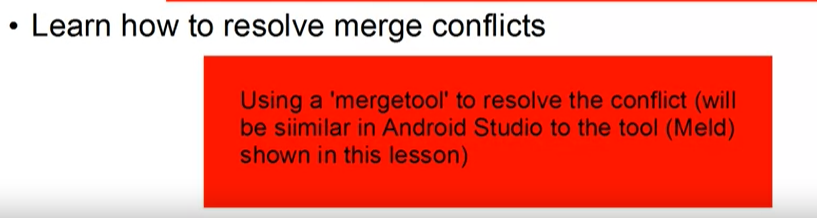
<https://www.atlassian.com/git/tutorials/using-branches>

<https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell>

<https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging>

merge conflict

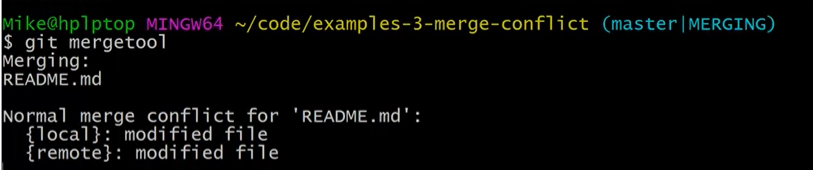




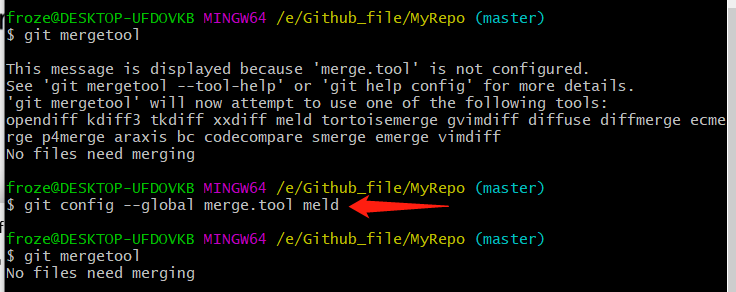
Merge conflict出现的情况是，我在本地对文件进行了修改，有人在云端对同一个文件进行了另一个修改。

此时本地和云端的仓库里的同一个文件有了不同的修改。如果我们试图pull，就会出现merge conflict。

这个时候直接在git里输入mergetool即可使用各类工具处理merge conflict。



注意在使用之前要对使用哪一种merge tool进行设置，要不然它就会随便选一种来用：

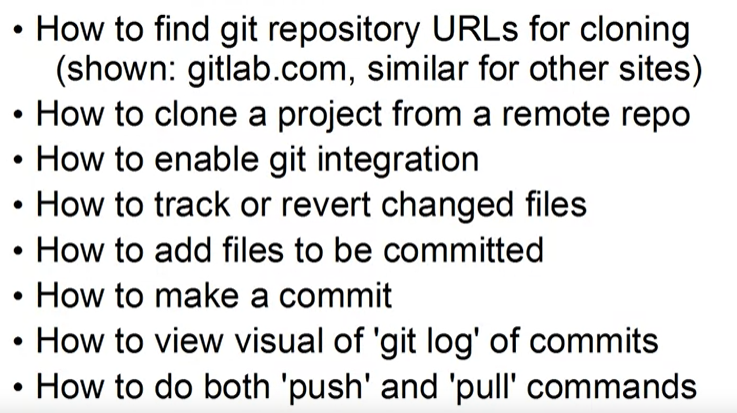


上面的图里我用config进行了配置，使用meld作为默认mergetool。

运行成功后会生成一个图形化的窗口让你处理conflict。

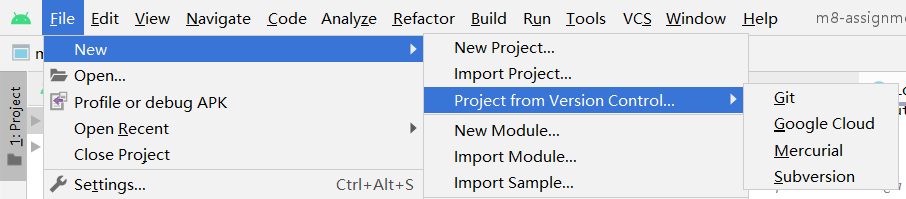
处理完以后保存commit即可。

最后，我们假装自己在和人合作项目，尝试以下操作：

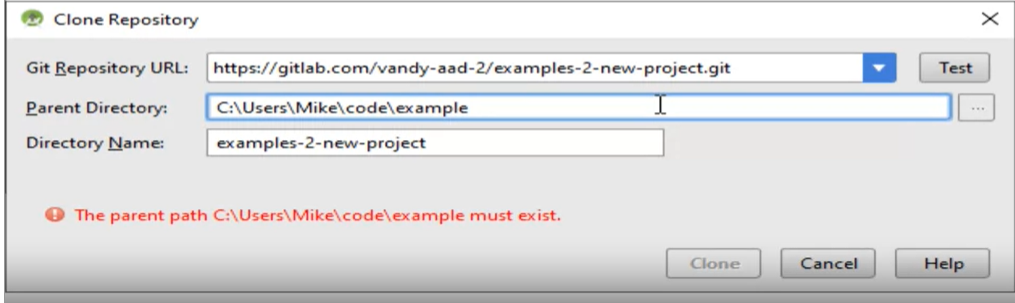


第一步，在gihub找到那个项目，复制它的HTTPS或者SSH链接。

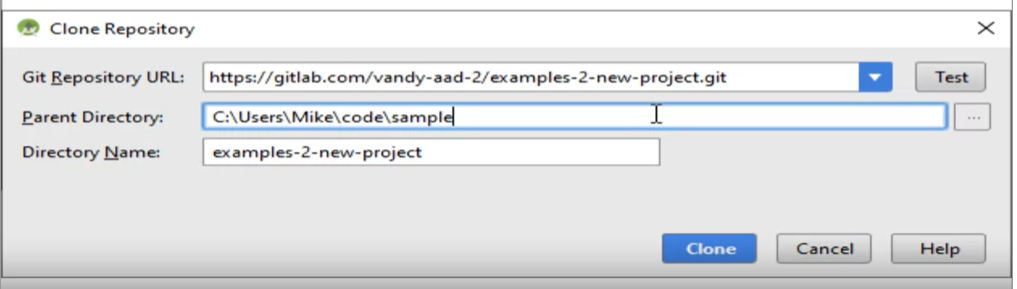
Android studio可以直接支持git：



选择git然后将链接复制进去：



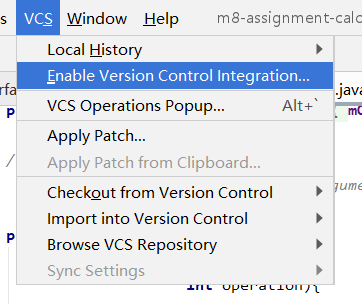
注意看如果设置的地址不存在的话会有问题，必须选择存在的地址。



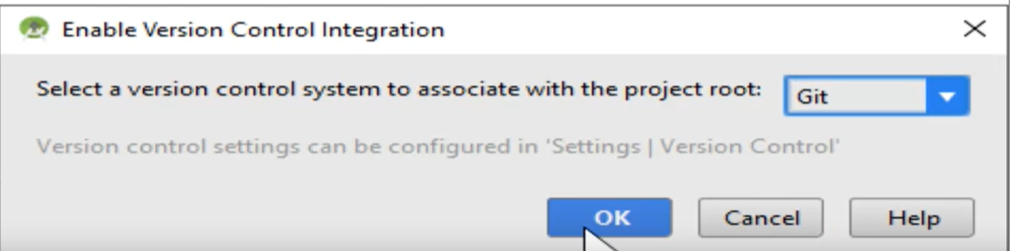
Test可以测试连接目标是否成功。

最后点clone即可将目标项目下载下来并在软件里打开。

此时只是下下来还没有激活version control。

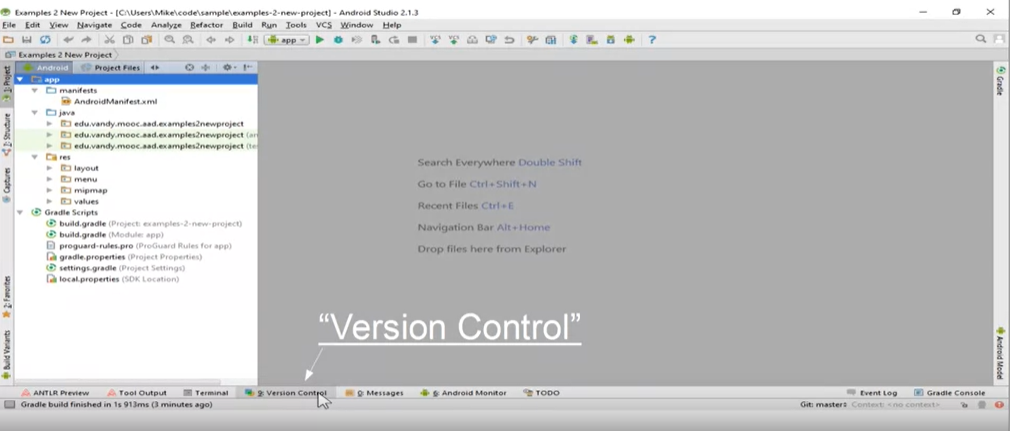


要在VCS这里使能。



选择git并点击OK。

此时就可以在界面上看到version control了：



这时，你点开就能看到本地改变的list，你在本地开发的任何改变都会显示在上面，同时你改过的文件的名字还会变成蓝色。

选中列表中的改变，右键菜单选git，即可进行add操作。

再在列表那里右键，选commit changes，在弹出的窗口里写好自己的名字和commit message，最后进行commit and push即可。

此时如果查看log就能看到push的记录。

如果在窗口只是选择了commit，那么可以在VCS那里选择git -> push进行push。pull也在那里。