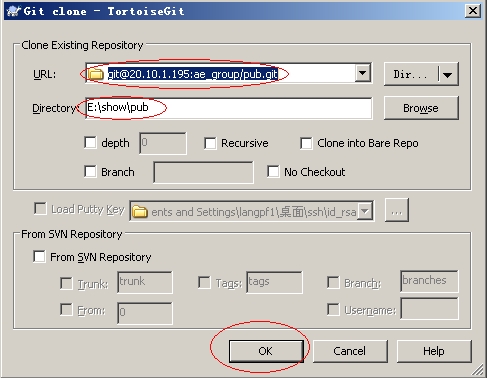
# Git代码管理操作性介绍

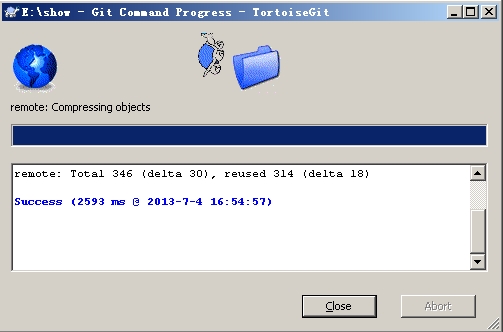
进行以下操作前，软件安装配置需完成，具体参考《Git操作手册》

## 1.仓库的clone

将远程pub仓库clone到本地E:/show/pub目录下 ：

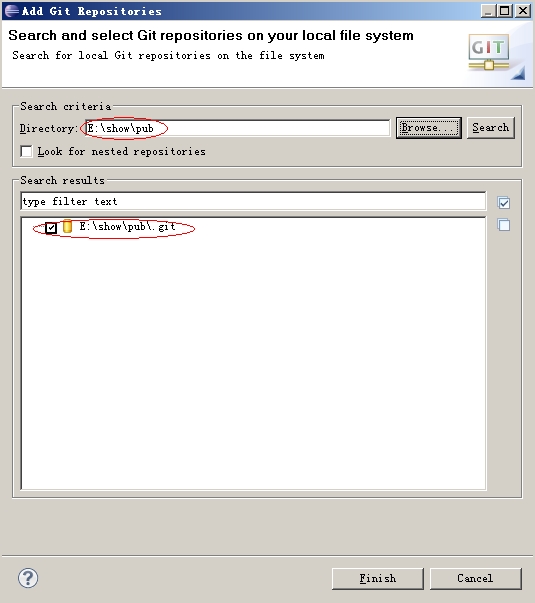
右键🡪Git clone…





## 2.将pub仓库添加到eclipse中

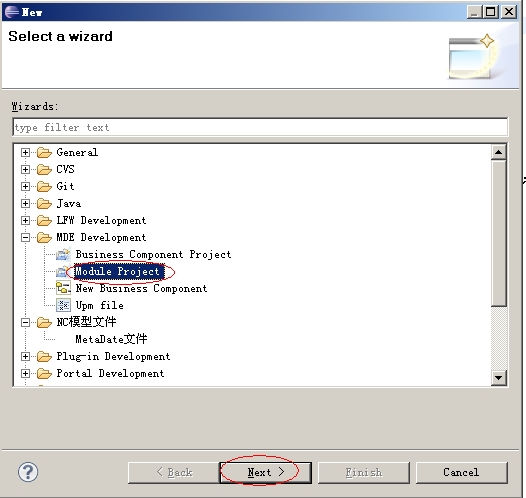
打开Eclipse，Windows🡪Show view🡪Git Repositores，点击Git Repositores面板下的CatchB006.jpg按钮：

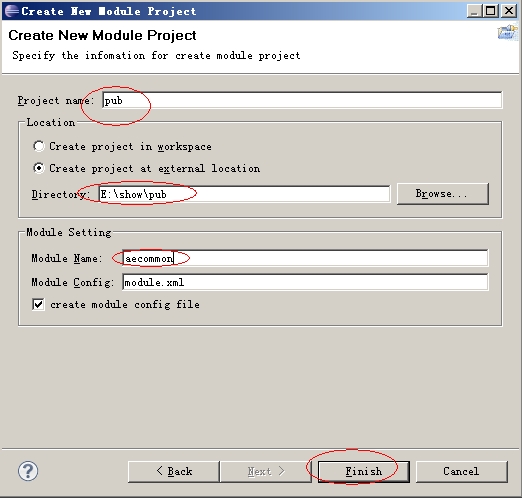


点击Finish结束。

## 3.新建project

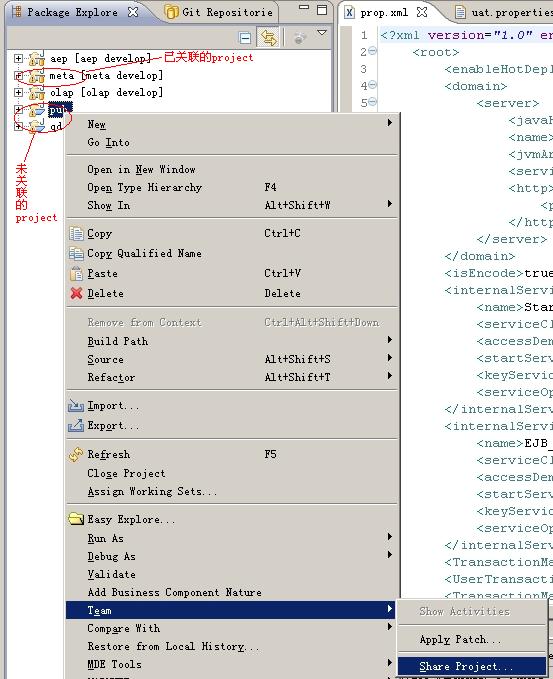
在Eclipse中，选择Windows🡪Show view🡪Package Explore，在Package Explore下选择NEW（独立组件不适合这么建，独立组件建立project参照《Git操作手册》）：



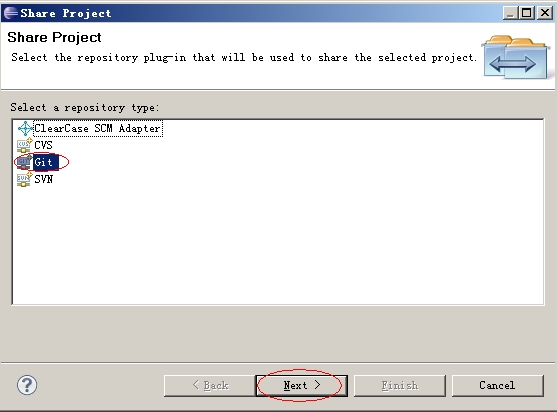


## 4.将新建的project与仓库相关联

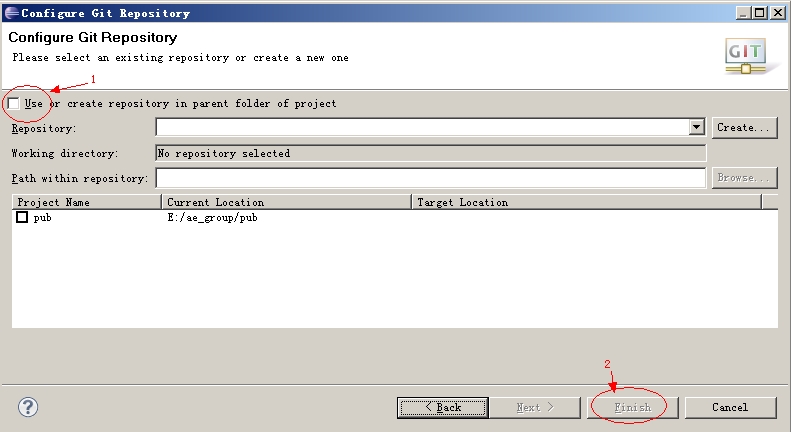
### 1>在未关联的project上右键🡪Team🡪Share Project…



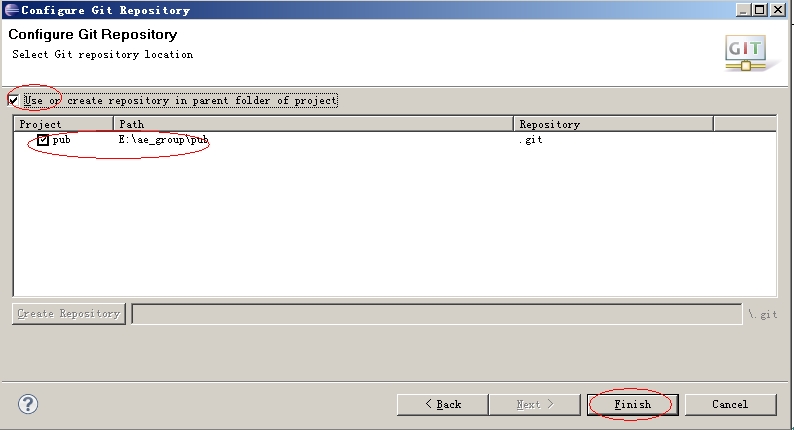
### 2>如下图，点击下一步



### 3>接下来，按下图操作

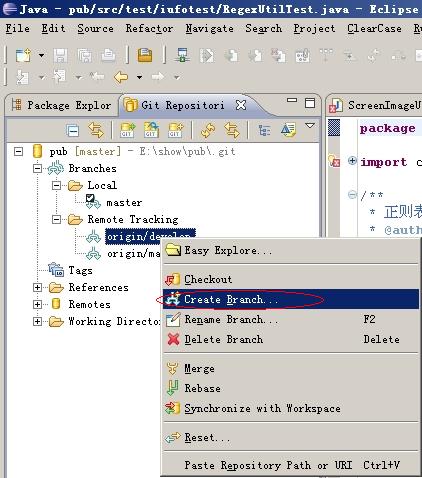


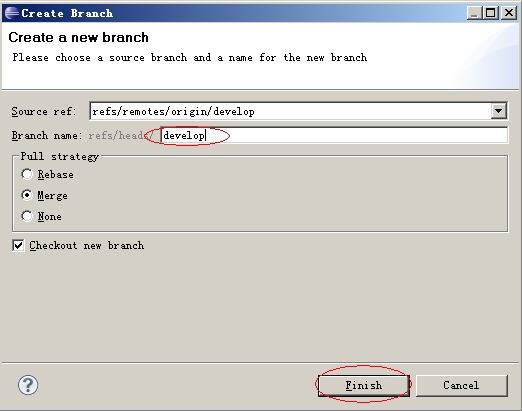
上图多选框选1后即变为下图的形式，点击finish即可：



## 5.新建develop分支

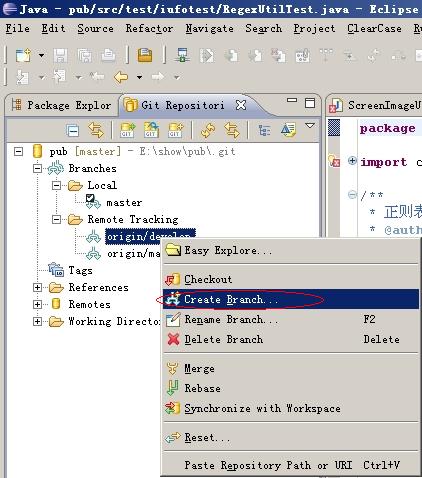
在Git Repositores面板下：

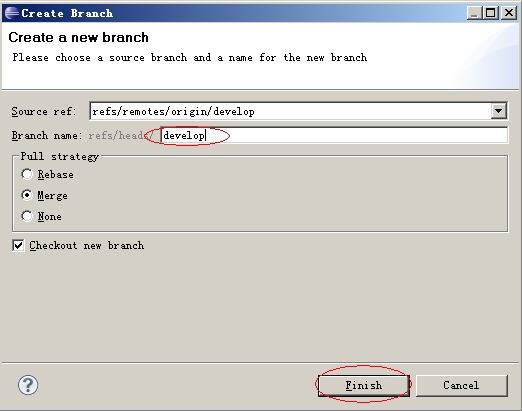




## 6.checkout到develop分支并删除master分支

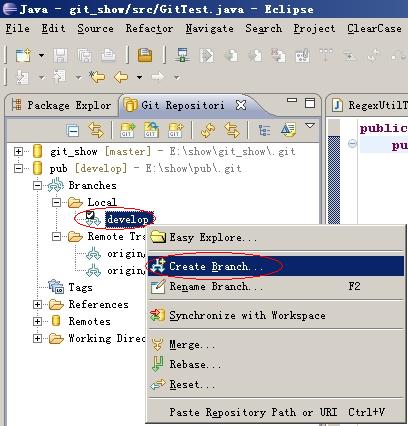
在Git Repositores面板下：

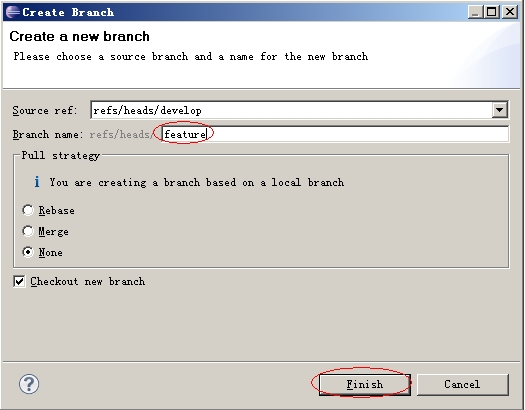




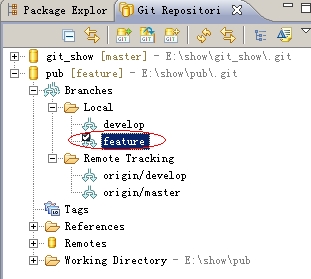
## 7.基于develop分支建立特色分支

在Git Repositores面板下：





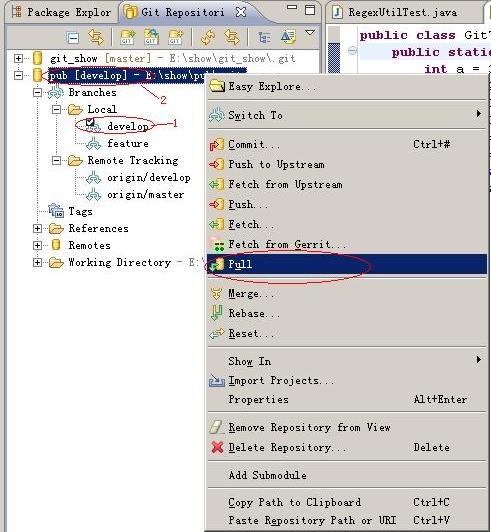
接下来切换 (Checkout) 到feature分支进行代码开发，切换方法：双击feature:



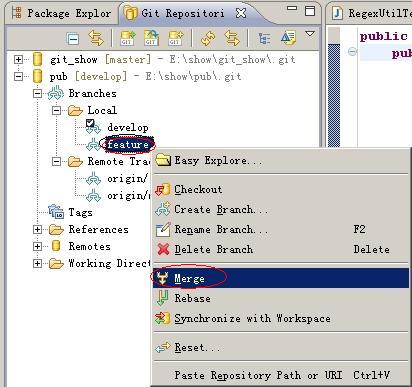
## 8.将新代码推送到服务器的步骤

假设当前在feature分支上进行了代码修改或增加了新的配置文件，并已经commit到了feature分支上。

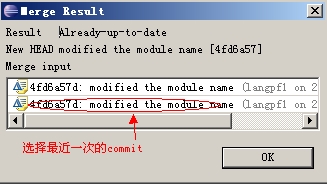
首先，checkout到develop分支上，在pub仓库上右键，选择pull：



成功pull下来后，本地develop分支即与远程develop已经同步，此时在feature分支上右键🡪Merge:

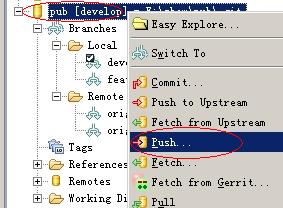


弹出对话框，选择最近的一次的commit，点击OK:

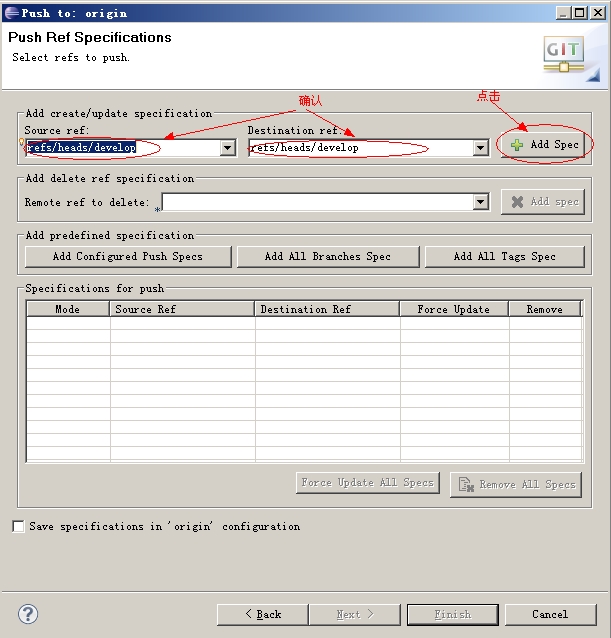


如有冲突，按照《Git操作手册》上的方法解决冲突并commit.

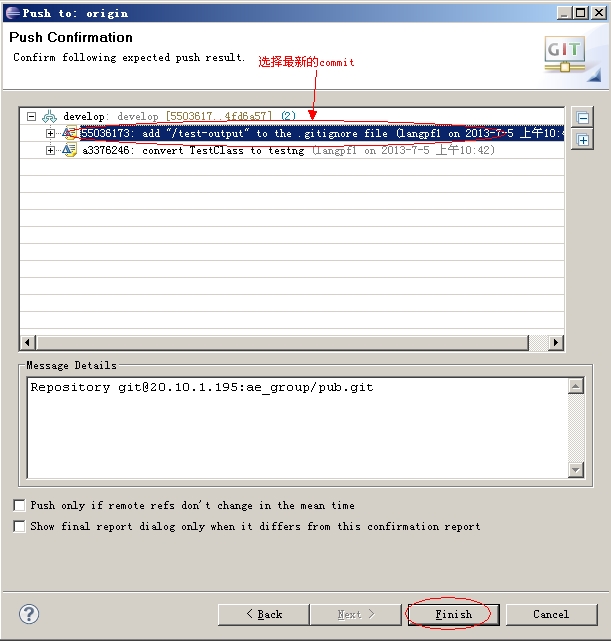
接下来在仓库上右键选择Push,Push到远程服务器上的develop分支上:



弹出”Push to Another Repository”对话框，点击Next到下一步:



点击Next到下一步:



接下来会弹出新的对话框，直接选择最新的commit并点击OK就行了

ps:有时候经过了这些操作仍然会遇到push不成功的现象，这时候很可能是远程仓库的develop分支有别人新提交的代码，只需要再重新将develop分支pull一下，接着再进行push操作就可以了。

## 9.代码的回退

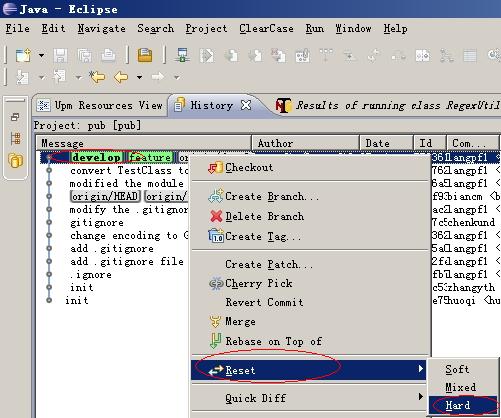
有时候会遇到这种情况：

自己在特色分支上修改了代码，或者误删了一些代码，但还没有进行commit操作，或者已经进行了commit操作，这时候想再恢复回原来的状态，就需要进行回退操作：

### 1> 如果没有进行commit操作

可以在工程上右键🡪Team🡪Show in History，打开历史面板：

然后在最新的commit上右键🡪Reset🡪Hard



### 2> 如果进行了commit操作

则是在第二个最近的commit节点上进行reset操作，恢复到上一次commit的场景。

### 3> 关于reset的3个选项的意义：

逆转与恢复：git reset

命令形式： git reset [--mixed | --soft | --hard] [<commit-ish>]

--mixed

仅是重置索引的位置，而不改变你的工作树中的任何东西(即，文件中的所有变化都会被保留，也不标记他们为待提交状态)，并且提示什么内容还没有被更新了。这个是默认的选项。

--soft

既不触动索引的位置，也不改变工作树中的任何内容，我们只是要求这些内容成为一份好的内容( 之后才成为真正的提交内容)。这个选项使你可以将已经提交的东西重新逆转至“已更新但未提交 (Updated but not Check in)”的状态。就像已经执行过 git-update-index 命令，但是还没有执行git-commit 命令一样。

--hard

将工作树中的内容和头索引都切换至指定的版本位置中，也就是说自<commit-ish>之后的所有的跟踪内容和工作树中的内容都会全部丢失。因此，这个选项要慎用，除非你已经非常确定你的确不想再看到那些东西了。

---------------------------------------------------------------------------------------------------------------------------------

以上内容，如有不正确的地方，欢迎指正

郎鹏飞