## 本手册在上次准备文档的基础上添加了日常操作的基本步骤，同时之前准备工作没完成的仍可以按这里的步骤进行准备。

## 本文档由已有文档整理所得，用于Git代码迁移的准备工作。

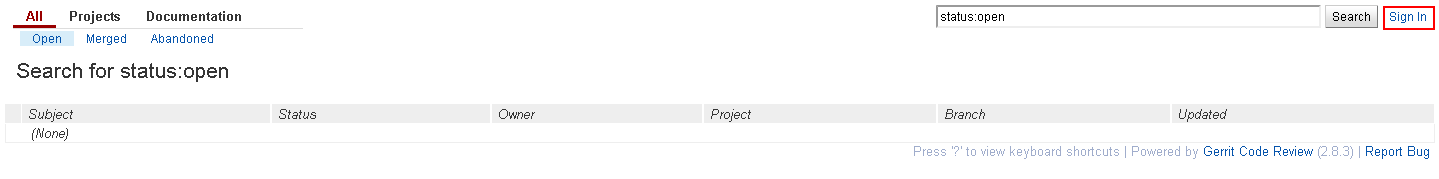
参考文档：[\\hrdev1\Share\NC持续集成](file:///\\hrdev1\Share\NC持续集成)

其他重要文档（含培训视频）：[\\nc-ci-ptgg-002\install\_backup](file:///\\nc-ci-ptgg-002\install_backup)

建议多学习以上文档中的东西。

## 1).Gerrit用户设置

<http://20.12.4.60:7070/gerrit/login/admin/groups/>

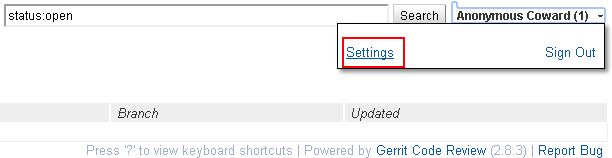


点击Sign in登录,进入下图:

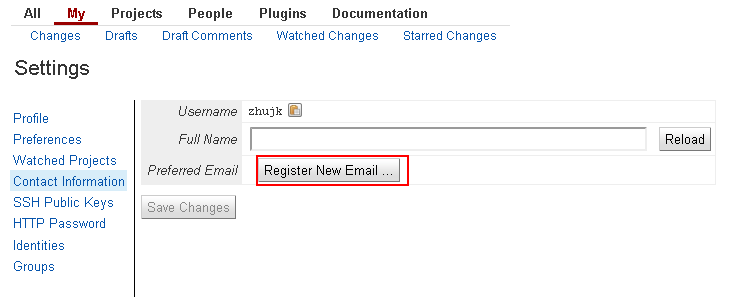


输入自己的**域账户**和**域密码**

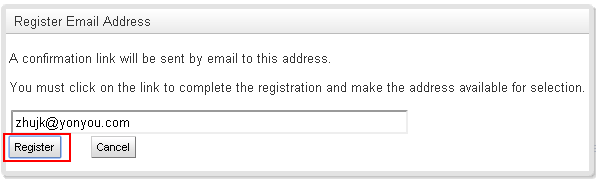
点击Sign In,进入下图:



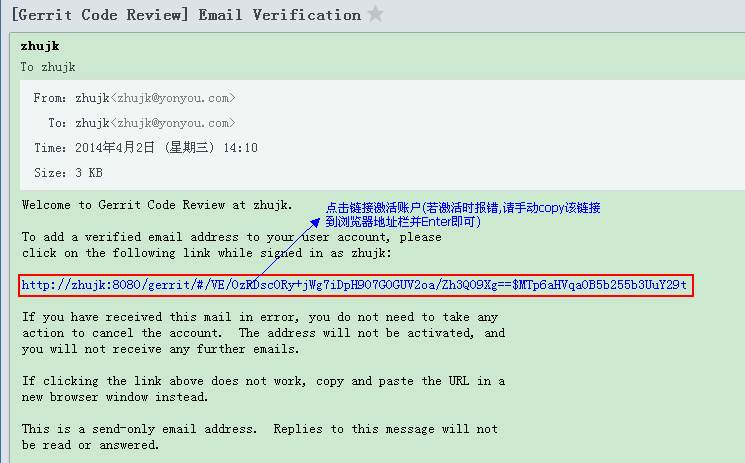
点击Settings:



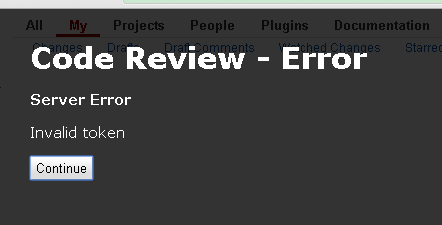
点击 Register New Email…:输入**个人公司邮箱**



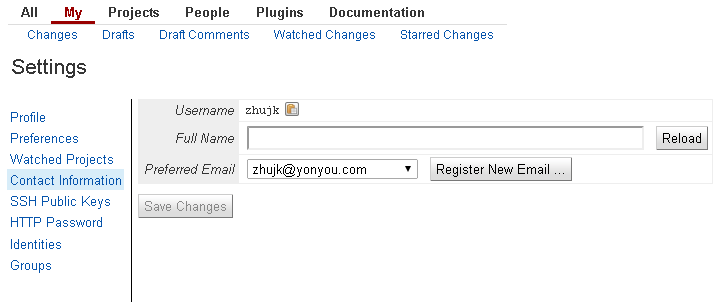
点击Register后,收取邮件如下:



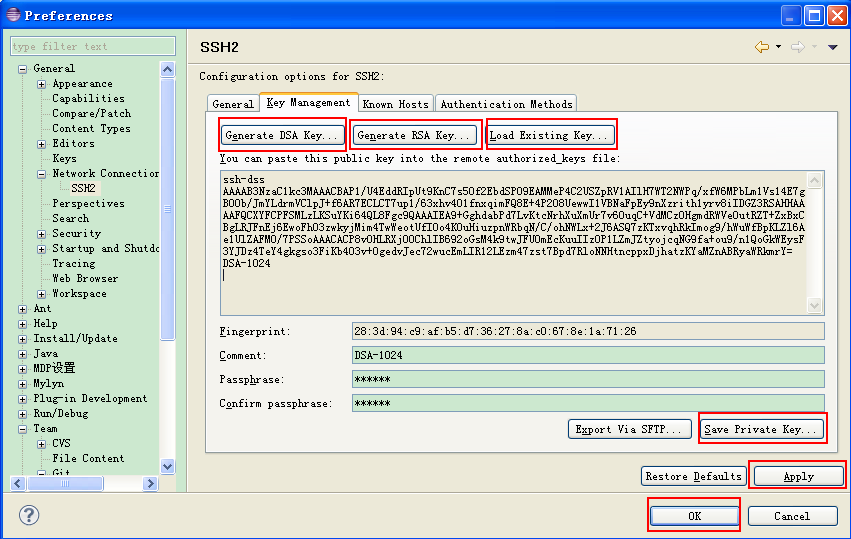
如图中,点击链接提示如下(有时出现):

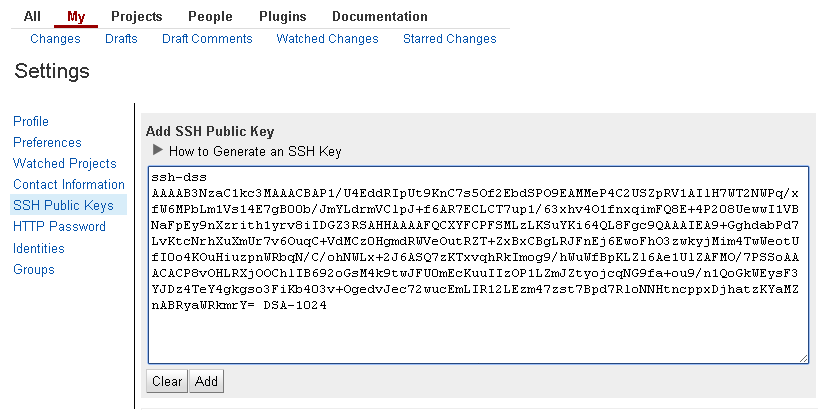


此时需要手动复制链接到浏览器地址栏,进入即可



上传公钥:





点击Add

## 2）本地配置

目前我们常用的Eclipse版本不含有相关插件，**我们强烈建议，请直接使用eclipse 3.8.2版本（**[\\nc-ci-ptgg-002\install\_backup\eclipse](file:///\\nc-ci-ptgg-002\install_backup\eclipse)直接解压即可**），**可以使用egit 3.3版本，基本所有高级的操作可以在egit中完成，另外如果需要使用命令完成git相关操作需要安装[\\hrdev1\Share\NC持续集成\7. 工具包\1 mSysGit](file:///\\hrdev1\Share\NC持续集成\7.%20工具包\1%20mSysGit)，以下步骤可以跳过就已经完成本地配置。

如果需要继续使用自己的版本，请按以下方法安装插件：

安装EGit插件：[\\hrdev1\Share\NC持续集成\7. 工具包\2 egit](file:///\\hrdev1\Share\NC持续集成\7.%20工具包\2%20egit)

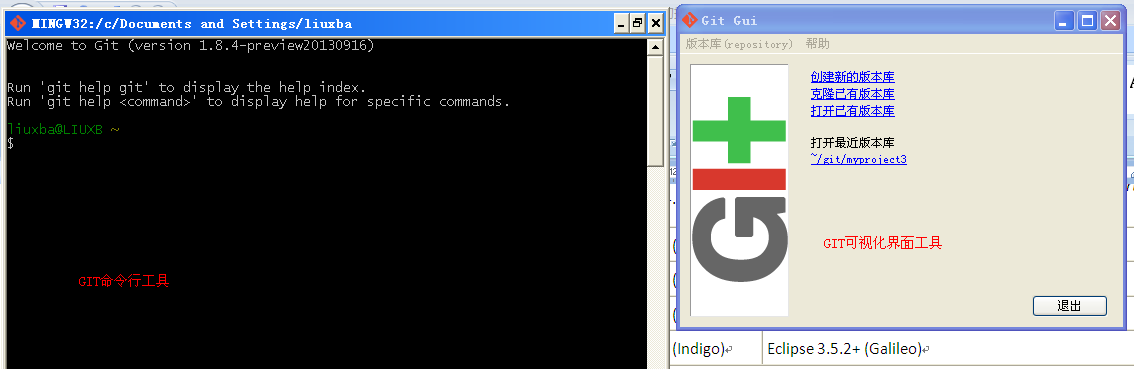
由于目前开发人员使用eclipse版本较低，暂时还不能全部升级，造成egit插件版本也较低，一般使用V2.1版本。Eclipse版本与egit版本关系如下：

|  |  |  |
| --- | --- | --- |
| **EGit** | **released for** | **Supported Version of Eclipse** |
| 3.1.0.201310021548-r | N/A | Eclipse 3.8.2/4.2.2 + (Juno) |
| 3.0.3.201309161630-r | 4.3.1 (Kepler SR1) | Eclipse 3.8.2/4.2.2 + (Juno) |
| 3.0.1.201307141637-r | 4.3.0 (Kepler) | Eclipse 3.8.2/4.2.2 + (Juno) |
| 2.3.1.201302201838-r | 3.8.2/4.2.2 (Juno SR2) | Eclipse 3.7.2+ (Indigo) (except *EGit Import Support* feature, which is optional) |
| 2.2.0.201212191850-r | 3.8.1/4.2.1 (Juno SR1) | Eclipse 3.7.2+ (Indigo) (except *EGit Import Support* feature, which is optional) |
| 2.1.0.201209190230-r | 3.8.1/4.2.1 (Juno SR1) | Eclipse 3.5.2+ (Galileo) (except *EGit Import Support* feature, which is optional) |
| 2.0.0.201206130900-r | 3.8/4.2 (Juno) | Eclipse 3.5.2+ (Galileo) (except *EGit Import Support* feature, which is optional) |
| 1.3.0.201202151440-r | 3.7.2 (Indigo SR2) | Eclipse 3.5.2+ (Galileo) |
| 1.2.0.201112221803-r | 3.7.2 (Indigo SR2) | Eclipse 3.5.2+ (Galileo) |
| 1.1.0.201109151100-r | 3.7.1 (Indigo SR1) | Eclipse 3.5.2+ (Galileo) |
| 1.0.0.201106090707-r | 3.7.0 (Indigo) | Eclipse 3.5.2+ (Galileo) |

对于V2.3以下的egit，有些重要的git功能在egit里不能完成，例如本地分支多次提交合并到develop分支上为一个提交的操作，需要借助命令行工具，因此要求开发人员除安装egit插件外，需要安装mSysGit客户端命令行工具。又有mSysGit工具是命令行工具，很多开发人员使用不太习惯，特别是在用eclipse开发时需要切换，影响开发效率。

Egit可视化环境操作虽然方便，但是很多GIT的高级功能不能使用，对于高级用户来说，需要安装MSysGit（在Windows下选择缺省安装就可以），可以在命令行方式下使用GIT高级功能。[\\hrdev1\Share\NC持续集成\7. 工具包\1 mSysGit](file:///\\hrdev1\Share\NC持续集成\7.%20工具包\1%20mSysGit)



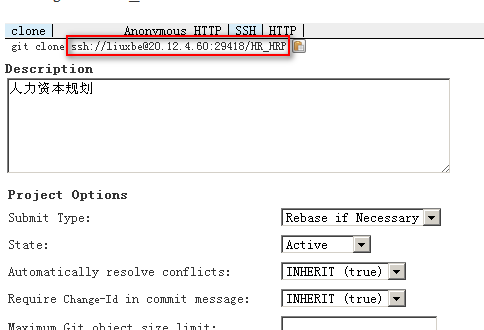


## 3）日常操作基本步骤

PS：进行以下操作前，请退出360相关软件和进程，否则会有不可预计的错误。

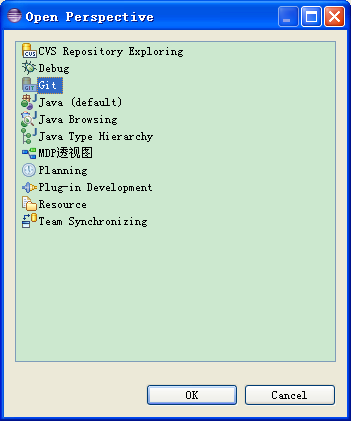
### （1）Clone项目到eclipse，即将服务器上的分支在本地建立映射

进入要Clone的项目,复制下图中红色框中内容

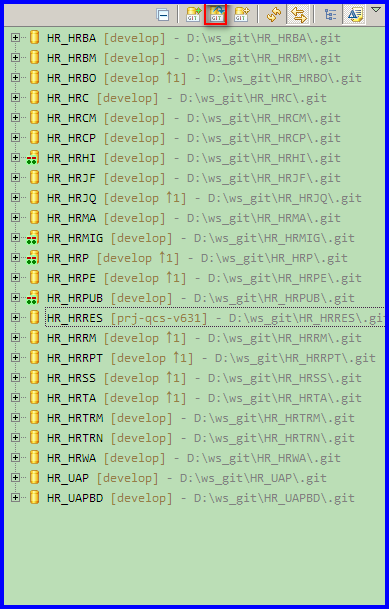


这里有两种方法clone到本地Eclipse，综合考虑，这里推荐一种方法，参考文档里的方法不推荐使用，

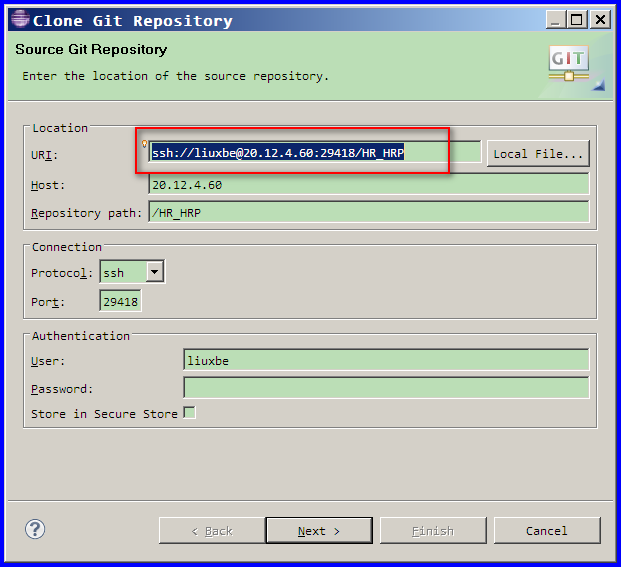
首先调出Git Perspective



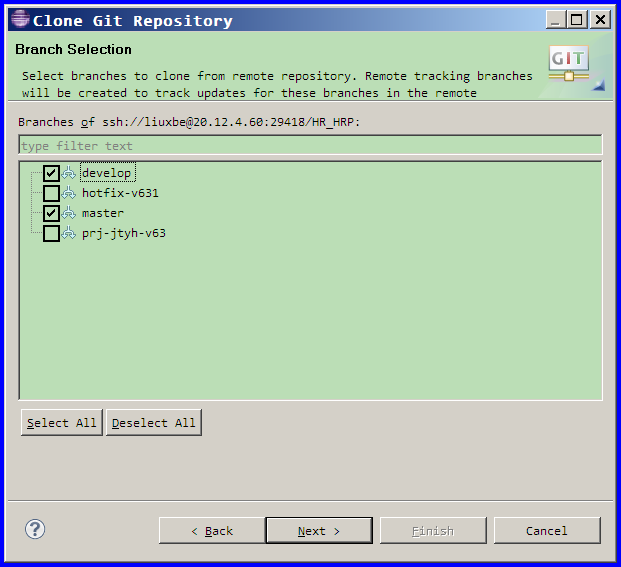
然后点下图中红色框内按钮：



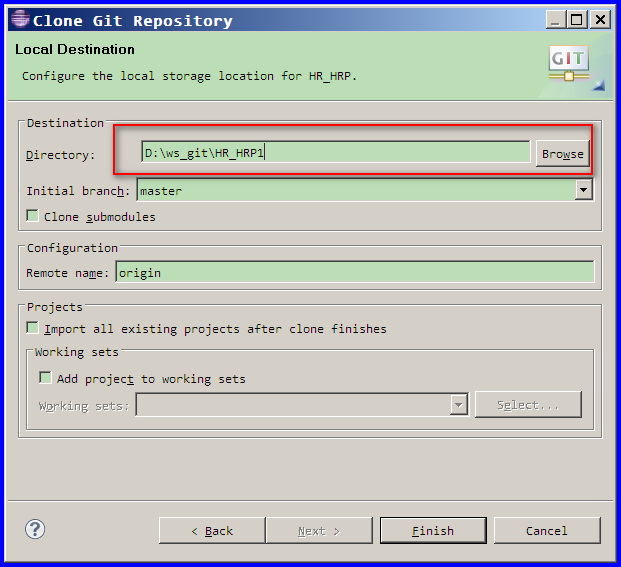
此时下图框中会自动填充之前复制的内容，如果没有则重新去网页中复制ssh内容粘贴到框中：



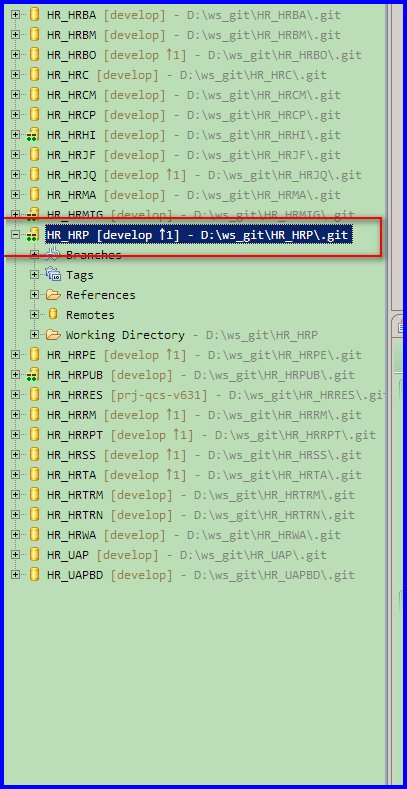
Next>进入



勾选所需要的分支，Next>

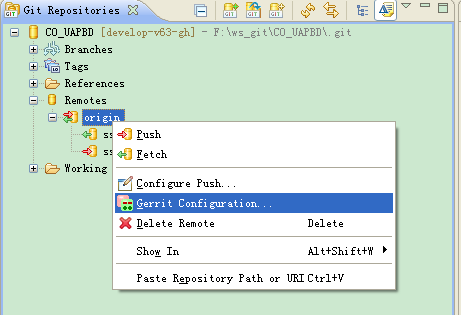
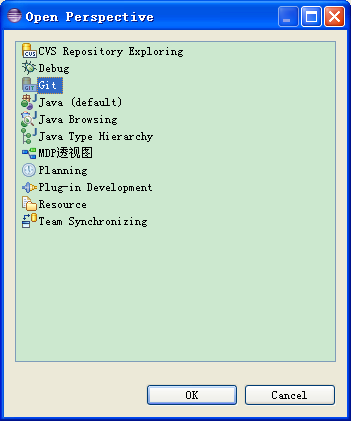


选择你要保存代码的路径，这个路径和使用cc时View类似，命名自取：例如GitView等等。Finish进入

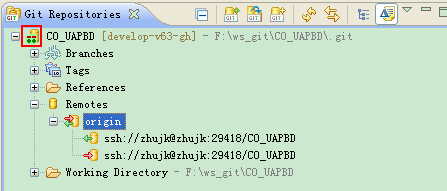
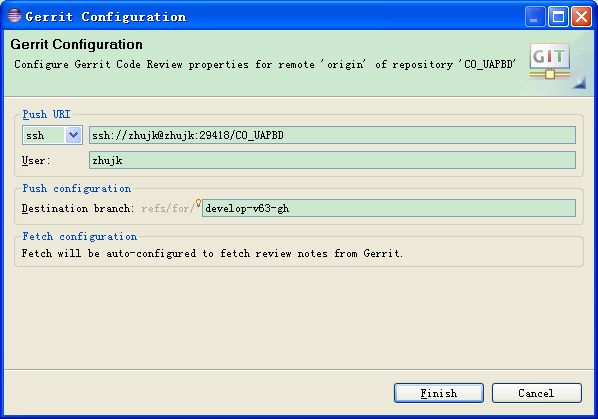


至此，将Git上的代码clone到本地的操作已经完成。

### （2）配置Gerrit

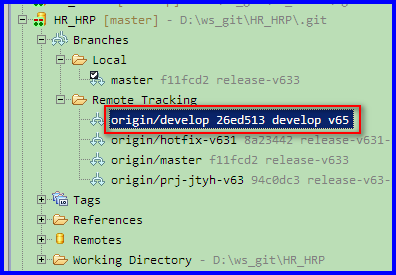
****

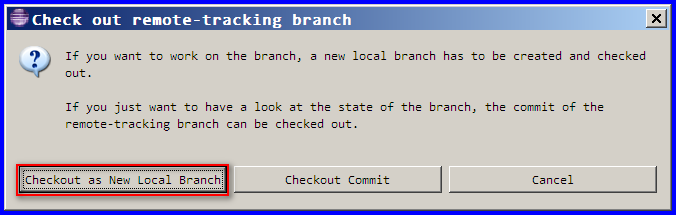
**下图中的Destination branch为需要修改代码的分支，通常应该是develop分支。**

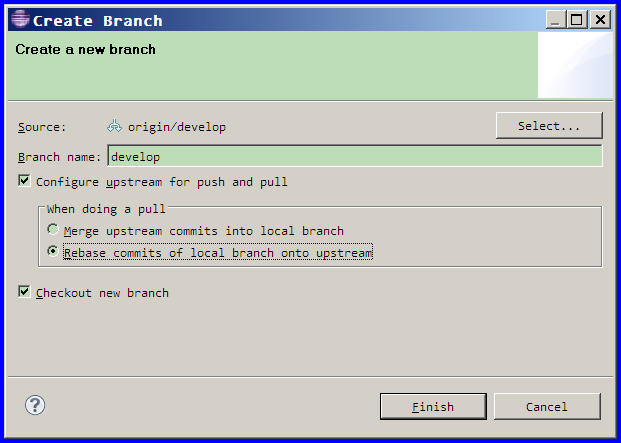
****

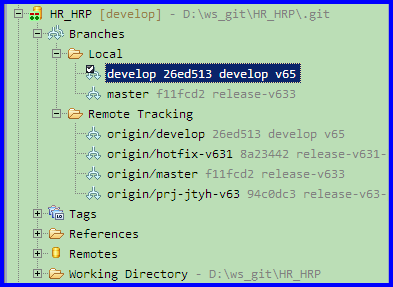
### （3）将本地的分支映射复制成本地分支

右击下图需要复制到本地的分支，这里以develop分支为例





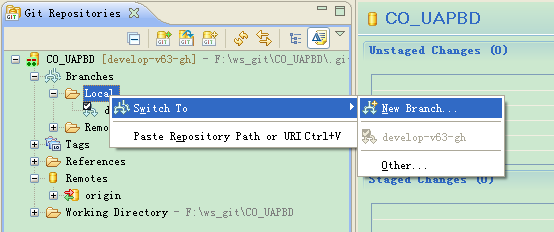


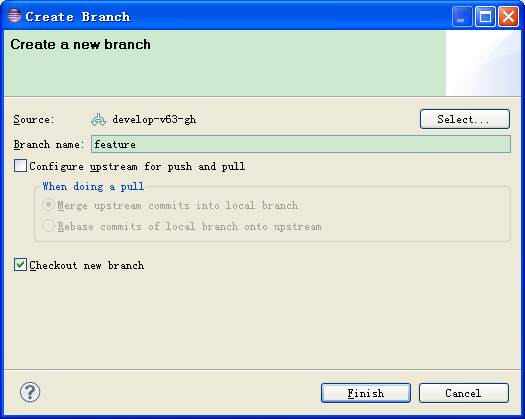


这样你有自己的develop分支啦。

### （4）创建本地分支进行开发

我们推荐按照一个相对独立的功能进行建分支，并且取与功能相关的名称。



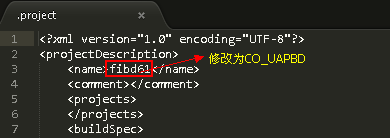




这样就有了自己专属的分支啦，接下来的修改代码就这这个分支下进行。

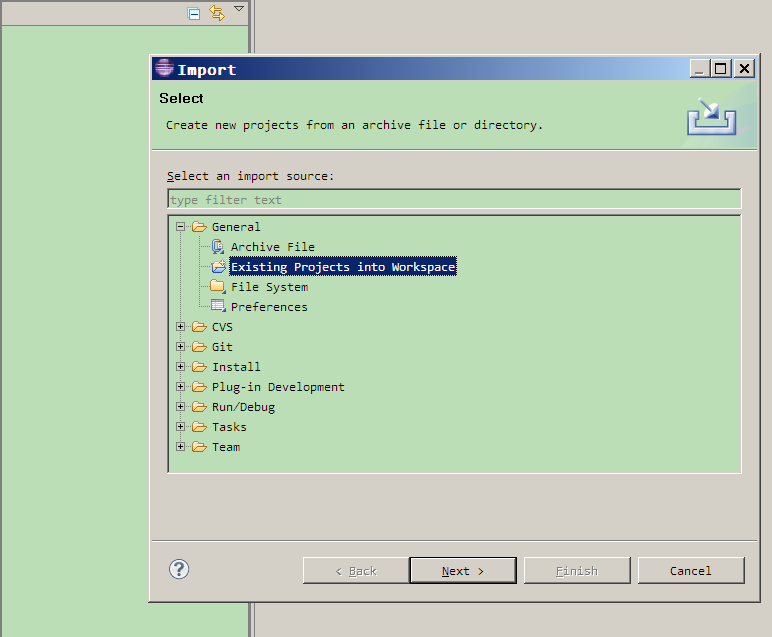
### （5）创建业务工程

这里有两种情况：a）之前使用cc时建过工程，希望继续使用之前的工程，则将原来目录中的.project、.classpath、.module\_prj三个文件复制到你的gitView中对应的模块文件夹下，同时用编辑器编辑.project文件



修改为当前模块的工程名，如HR\_HRPUB

然后在在Eclipse的java perspective中Import 已经存在的工程



选择模块目录，导入刚刚复制过来的工程就可以了，调整组件顺序等等操作和以往一致。

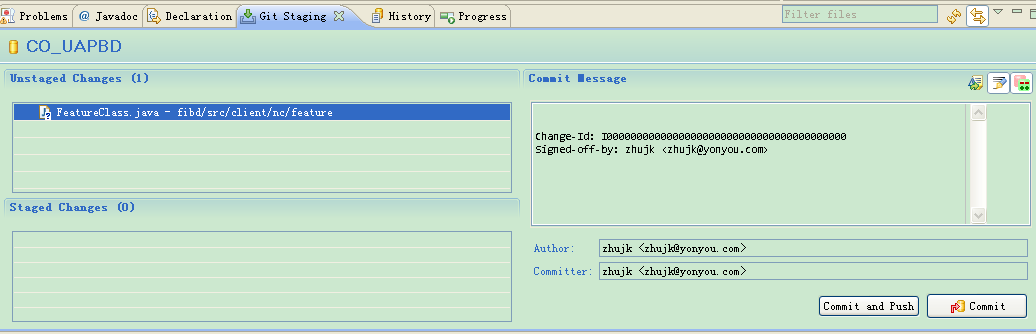
b）原来没建过相应的工程，或者希望重新建工程，则操作与原来一致，与之前区别是：原来我们新建“Module Project”，这里的Eclipse翻译成了中文“模块项目”。

### （6）修改代码

首先确认当前checkout的是你自己的分支“我的开发分支”，即在项目的名称后面紧跟着你自己的分支名称。

另外还要配置.gitignore文件，如果之前目录中没有这个文件，可以从之前邮件《git忽略的文件列表》中复制一份，或者在上一步结束后在下图Unstaged Changes中右击.project等等文件选ignore。

与之前用cc不同，此时在“我的开发分支”上开发，包括修改、增加、删除文件，直接修改保存就可以。完成修改后右击项目选择“Add to Index”、然后“Commit”，也可以在下图中做等价操作（推荐），将Unstaged Changes中的文件拖到Staged Changes中，然后在Commit Message中添加备注，备注最好要清晰表达所修改内容，然后务必添加签名和Change-Id。

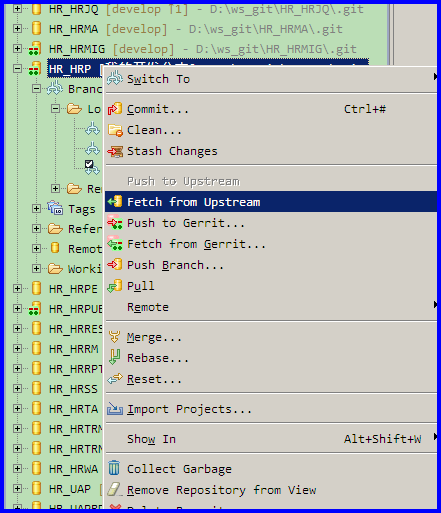


将上图中文件拖拽到下方,并添加注释(注意格式,注释从第一行开始,与Change-Id之间有一空行)，此时你修改的内容已经保存到“我的开发分支”了，但是还没有提交到服务器，这时候别人是看不到你的修改的。

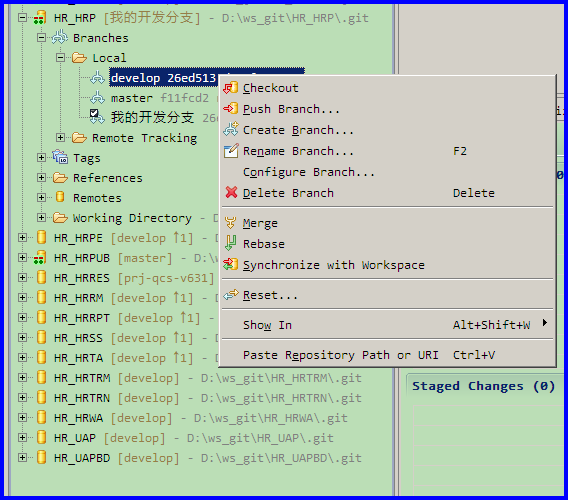
### （7）提交代码

切到Git Perspective，

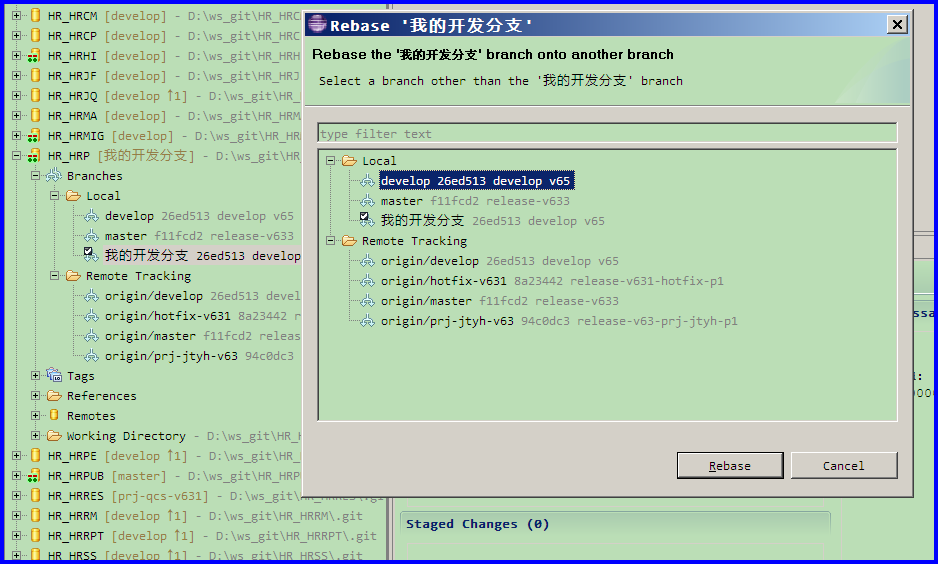
Fetch（更新romote上的分支，相当于之前的update）



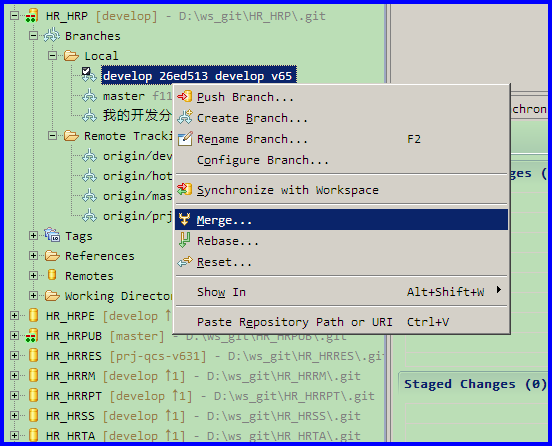
Rebase（将local上的develop分支与最新的remote上的develop同步，如果有修改则rebase时选择destination为remote上的develop分支）

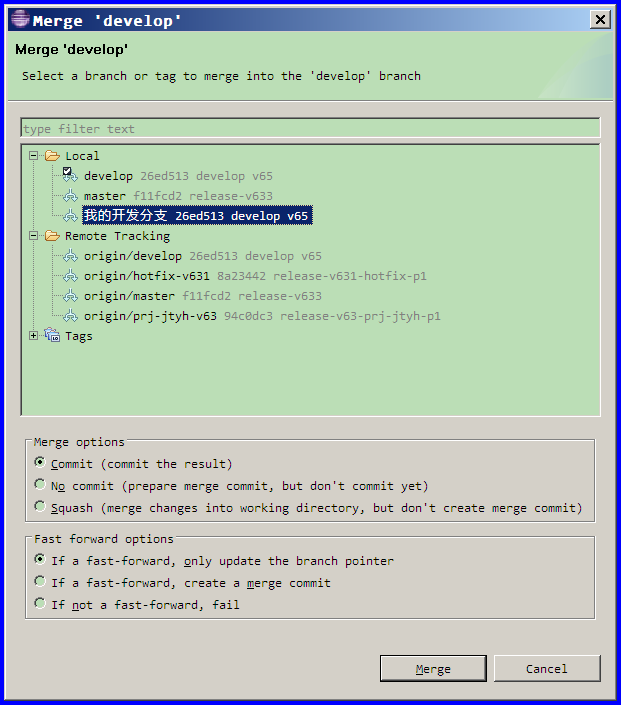


Rebase（更新“我的开发分支”的起点，此时Rebase的destination选择local的develop分支，这里可能会产生冲突需要解决）

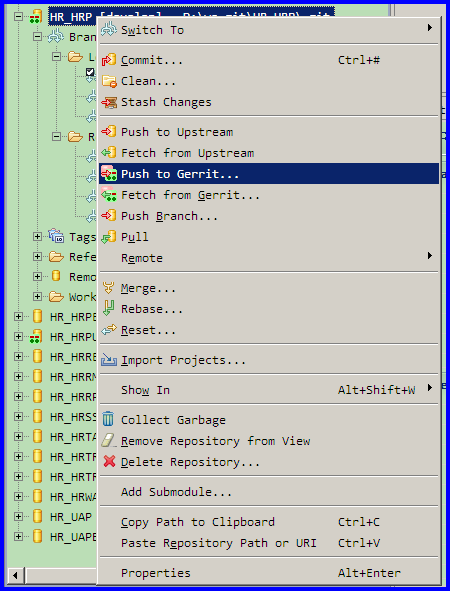


Merge（合并代码，首先checkout develop分支--双击Local下的develop）





Push to Gerrit（如果没有这项，说明没有配置Gerrit，按照上面的方法配置）



### （8）进入浏览器查看



等待相关人员Review并Submit后,Developer就可以执行Fetch操作,进而与本地的代码进行merge或者rebase等的操作了.