# 什么是SVN

SVN全名Subversion，即版本控制系统。SVN与CVS一样，是一个跨平台的软件，支持大多数常见的操作系统。

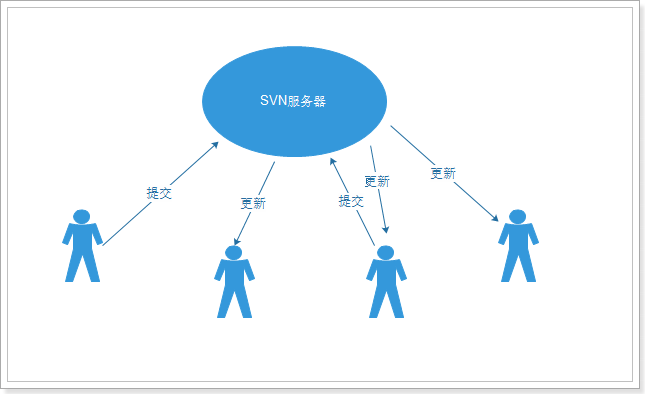
作为一个开源的版本控制系统,Subversion管理着随时间改变的数据。这些数据放置在一个中央资料档案库(repository)中。这个档案库很像一个普通的文件服务器,不过它会记住每一次文件的变动。

这样你就可以把档案恢复到旧的版本,或是浏览文件的变动历史。Subversion是一个通用的系统,可用来管理任何类型的文件,其中包括了程序源码。

Git – 版本控制系统。

SVN是集中式，Git是分布式。

## SVN工作原理



## 我们使用SVN能做什么？

1. 多人共享同一的资源，并且可以对资源实现修改和更新；
2. 记录资源的每一次变更，以及记录更改该资源的人，并且可以恢复到之前的任何一个修改点；

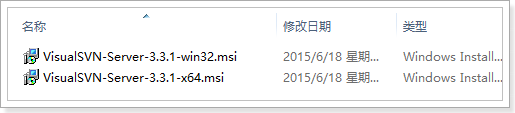
# 安装

SVN的安装分为2部分，第一部分是服务端安装，第二部分的客户端安装。

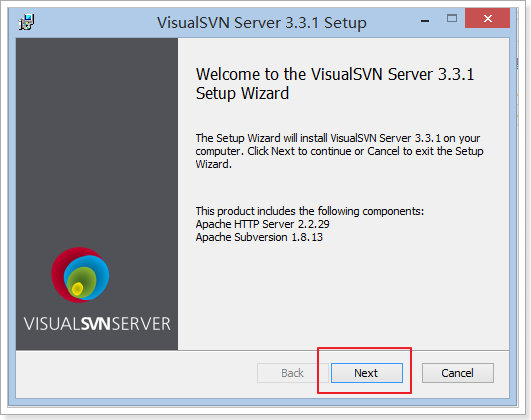
注意：一般情况下，我们只需要安装客户端即可，因为在企业中服务端是否运维来维护的，我们只需要开通一个账号即可。

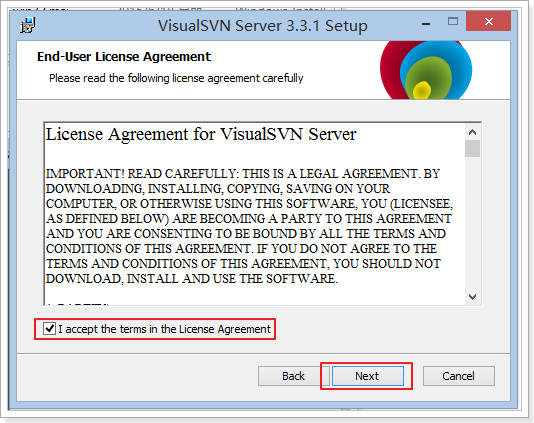
常用的客户端又分为2种，第一种是安装在操作系统中的客户端，另外一种是Eclipse的插件。

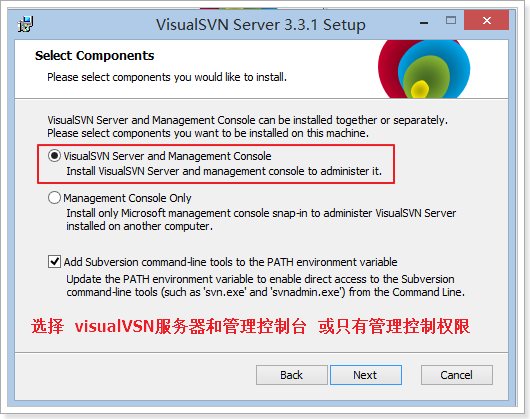
## 服务端安装（windows）



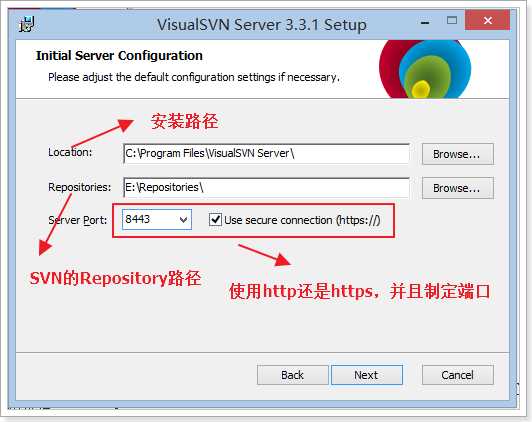
选择相应的文件安装。

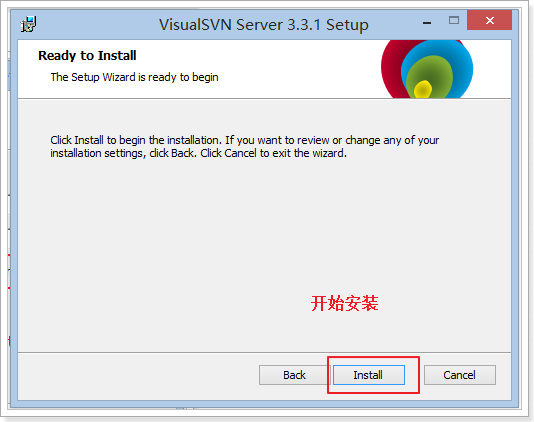


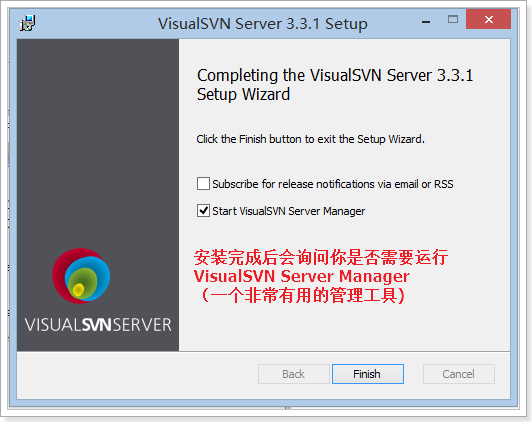




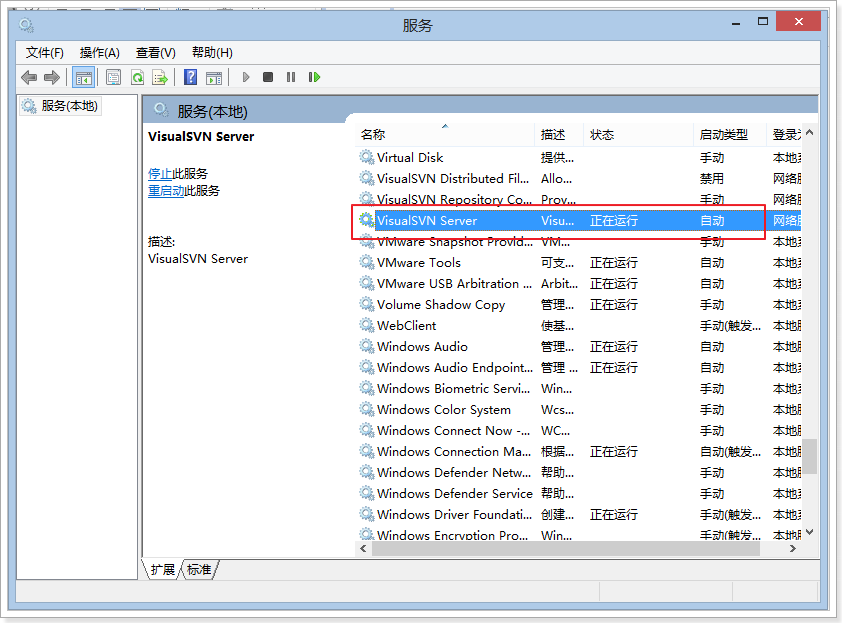




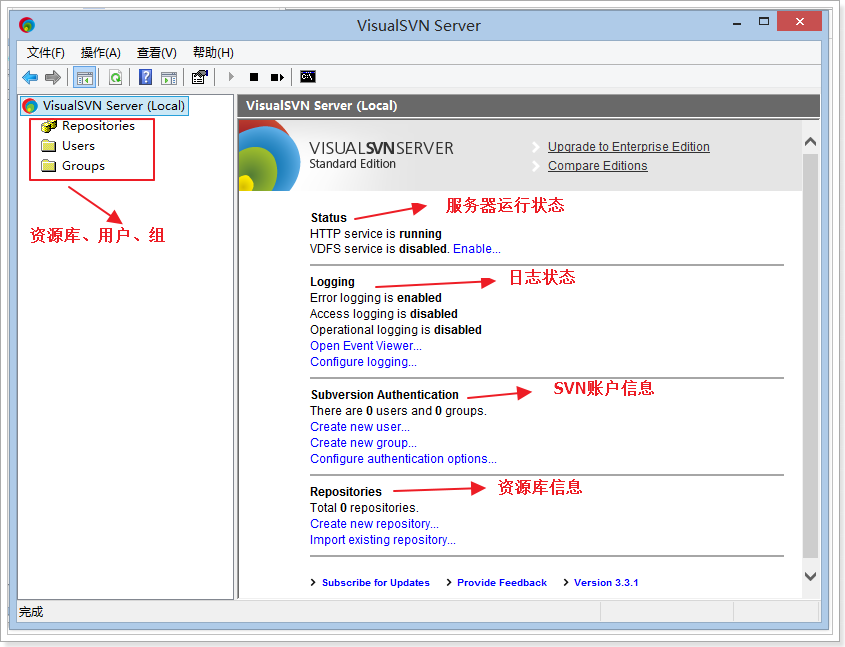




安装完成后会在系统服务中有SVN服务项，并且默认是开机启动。

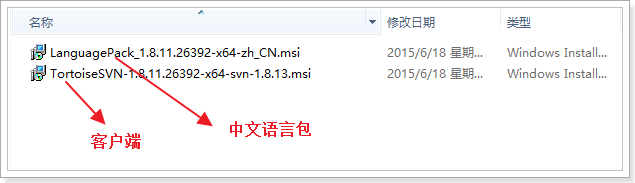


打开管理工具：



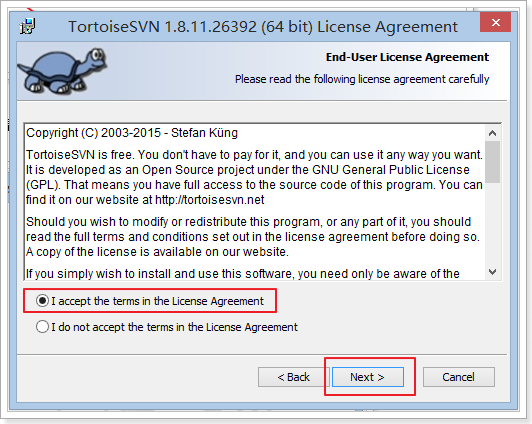
至此，服务端安装完成。

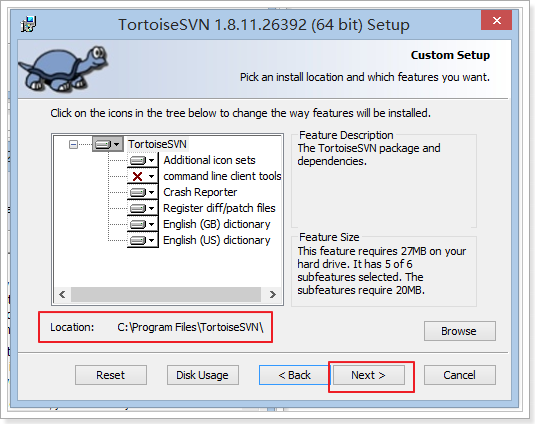
## 客户端安装

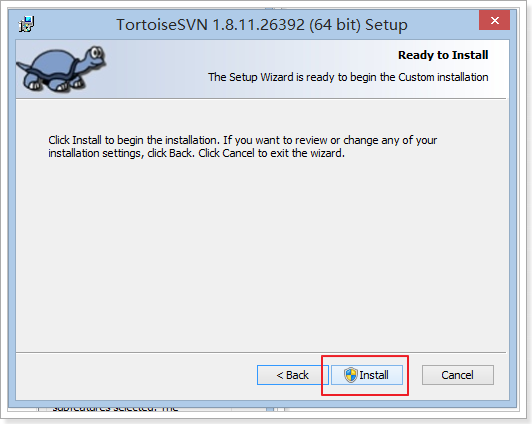


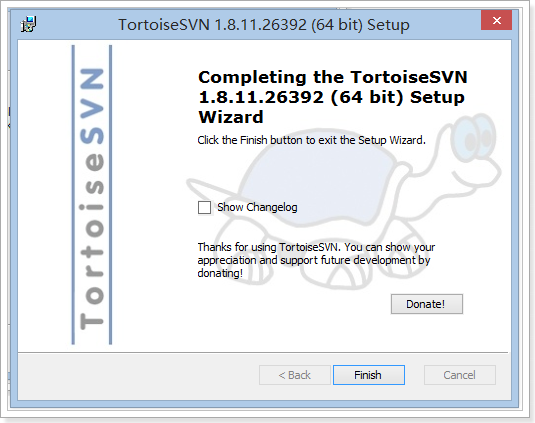
先安装客户端，再安装语言包。



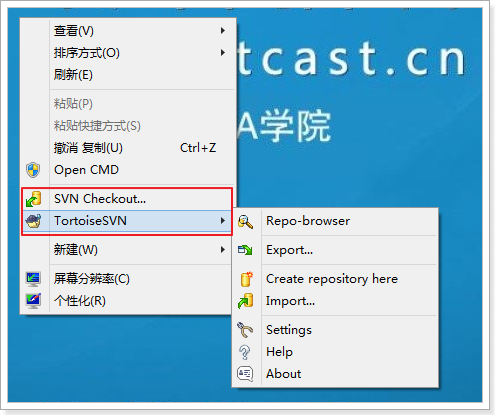




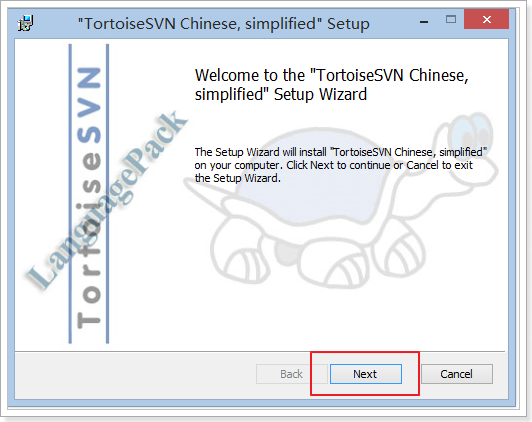


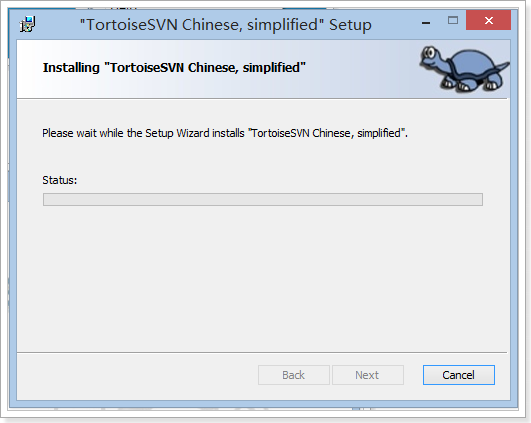


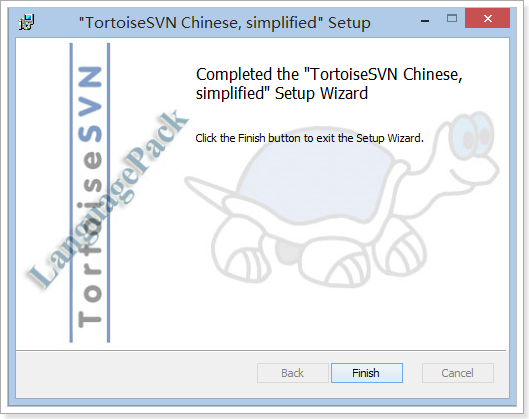
点击右键查看是否有SVN选项。



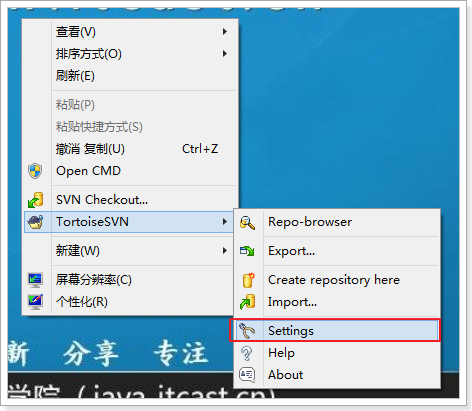
接下来安装语言包：

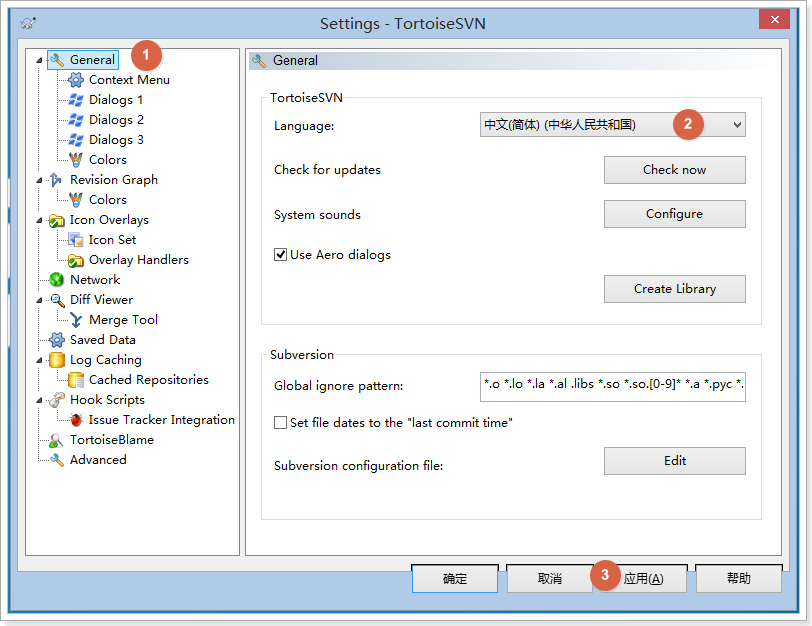




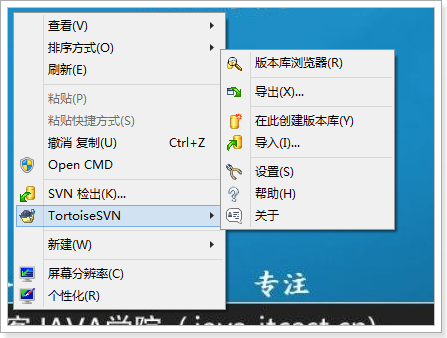


选择中文：





中文生效：



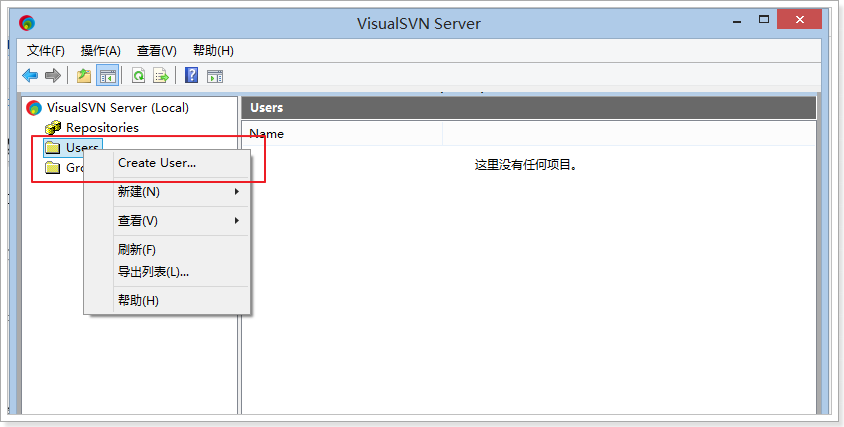
此处客户端安装完成！

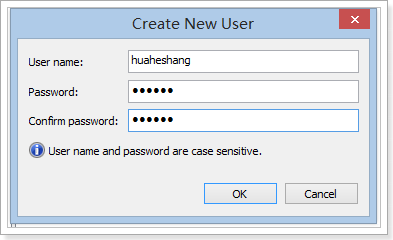
## Eclipse插件安装



# 创建资源库和用户

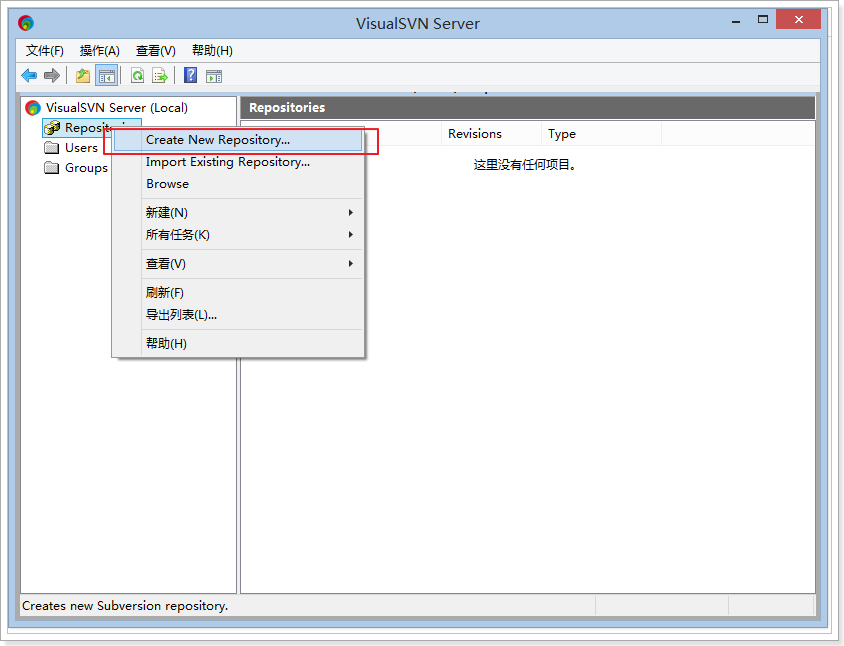
## 创建用户

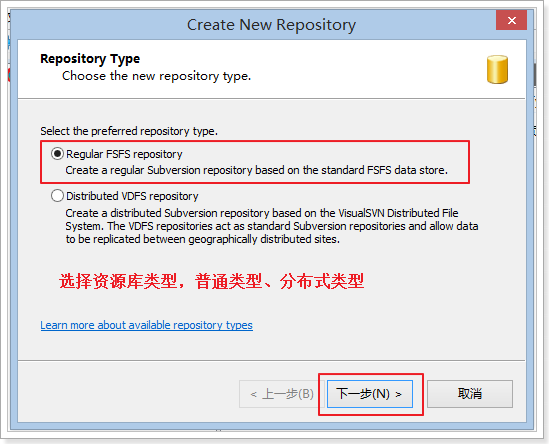


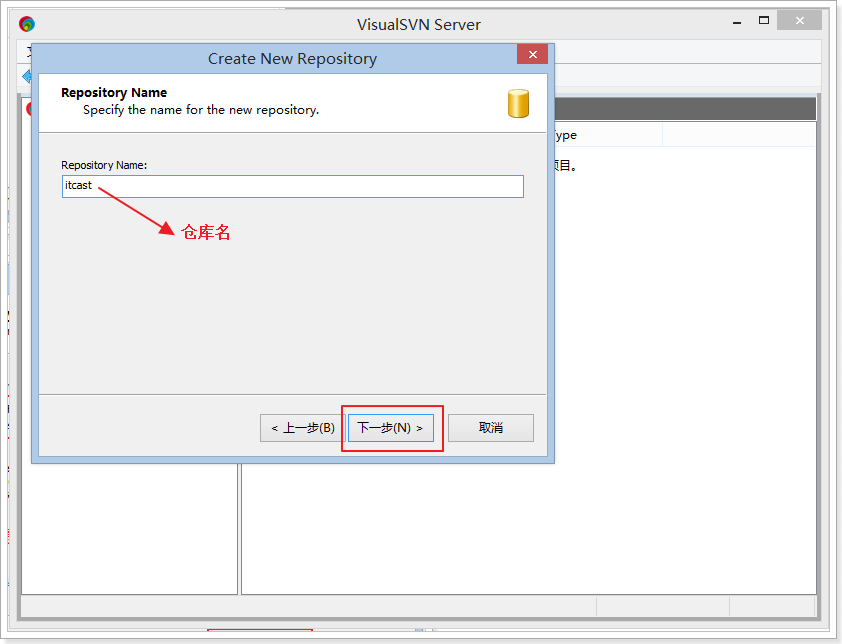


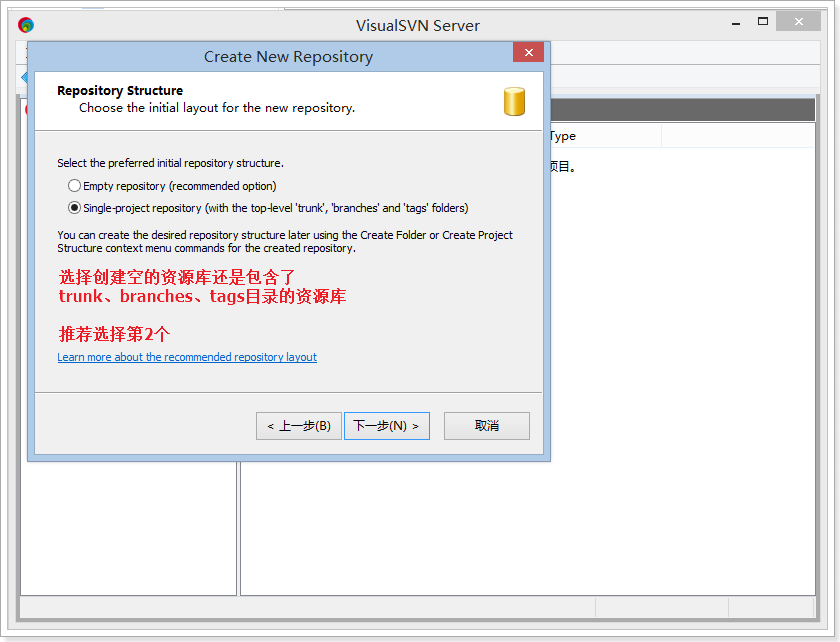
## 创建资源库

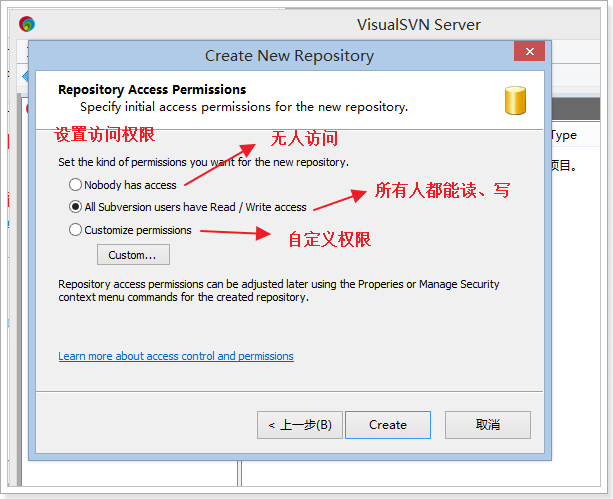
资源库：用户提交文件到SVN，存放文件的仓库。



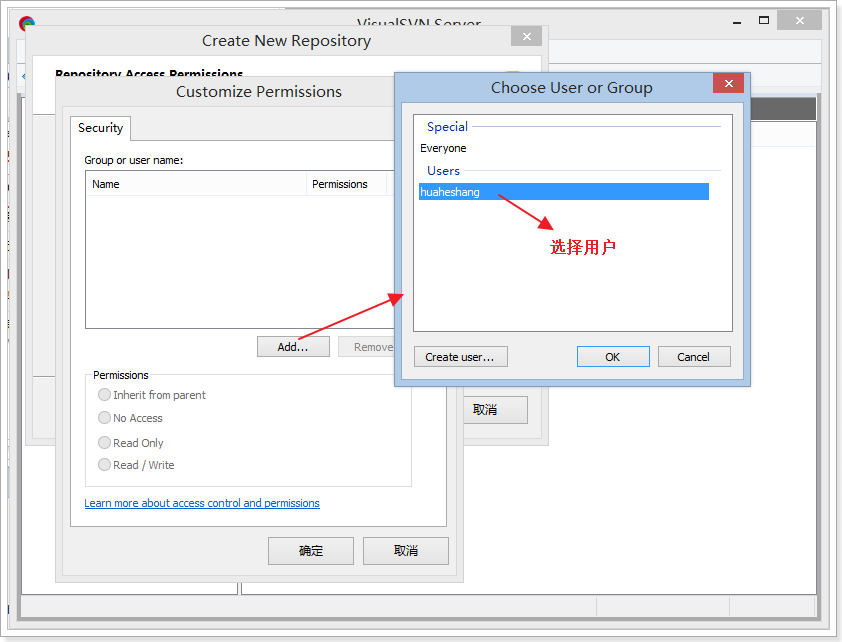


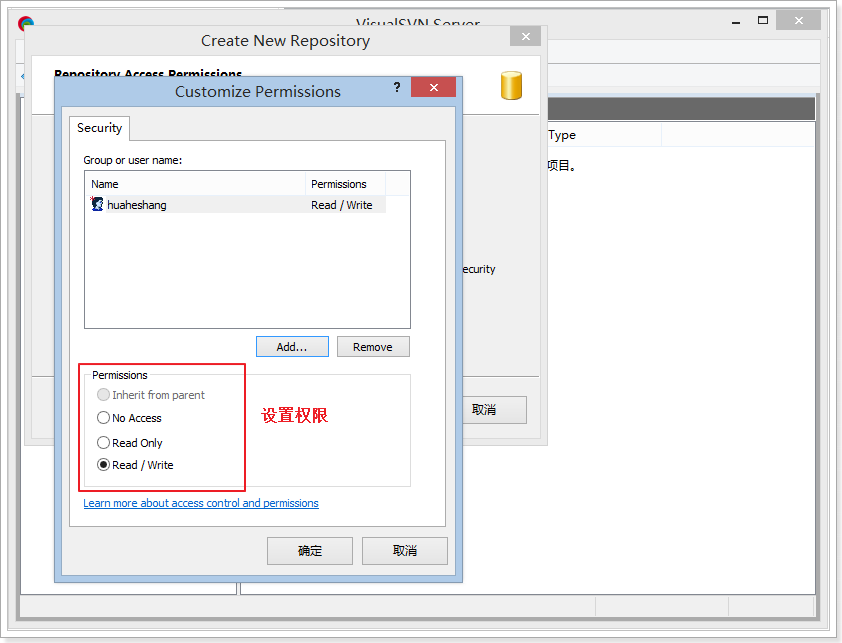


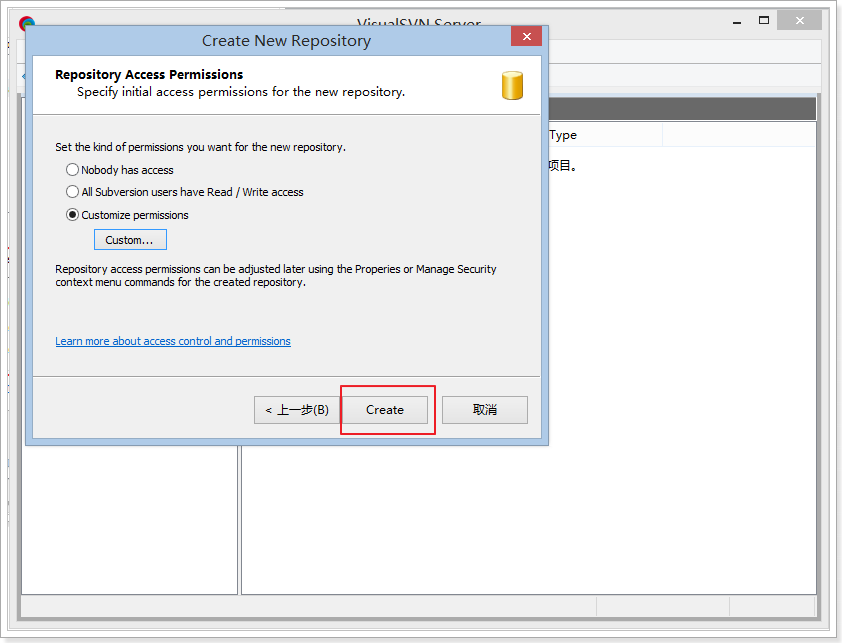


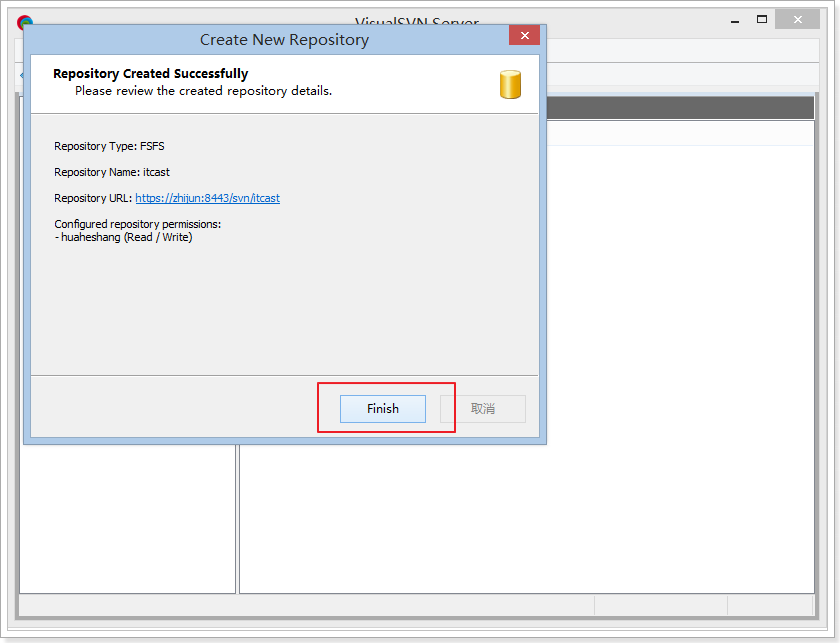


我们选择自定义权限：

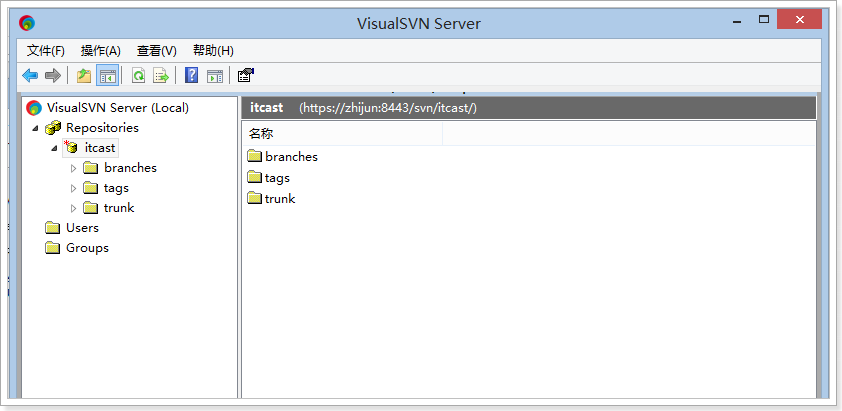






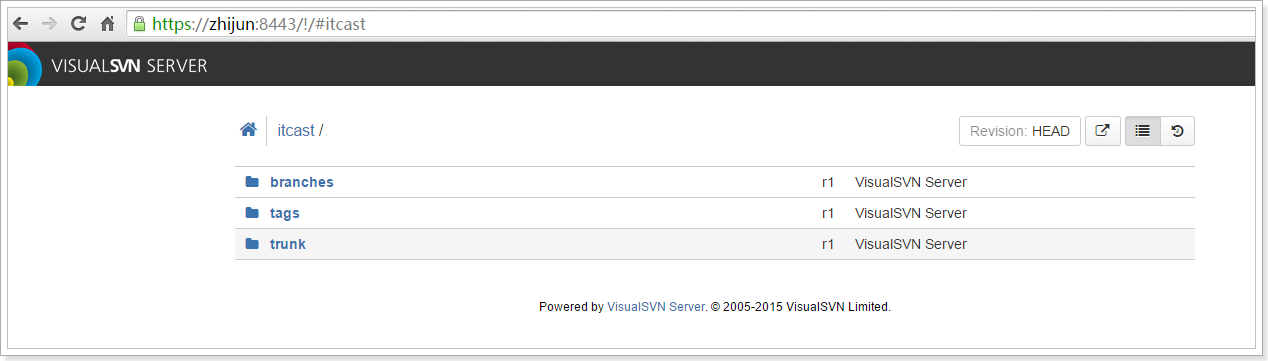


创建完成：



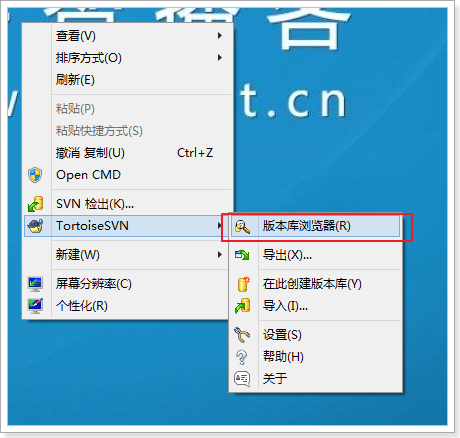
测试：



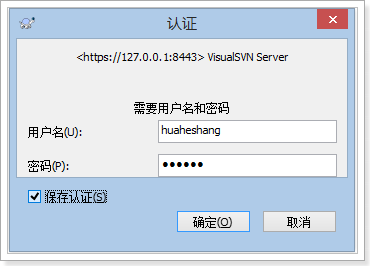


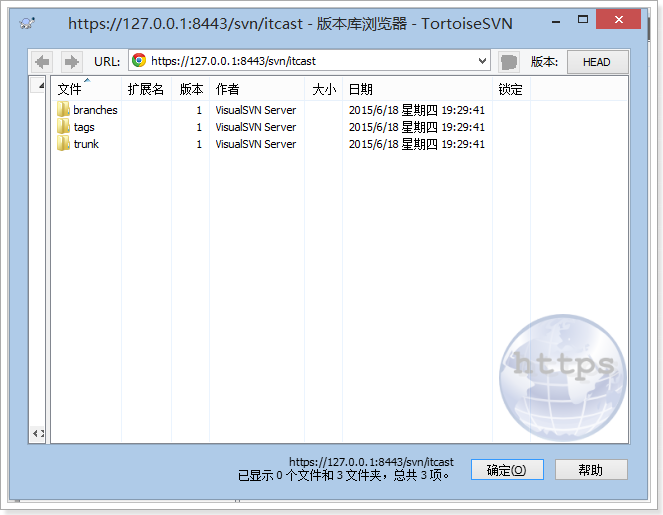
资源库创建完成。

## 使用客户端访问

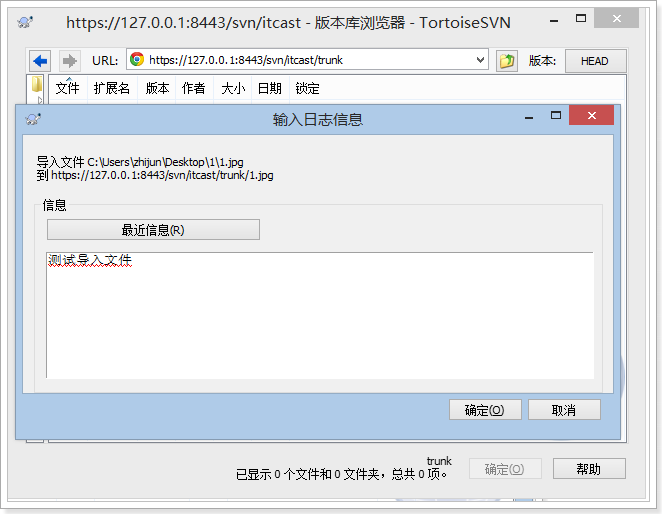


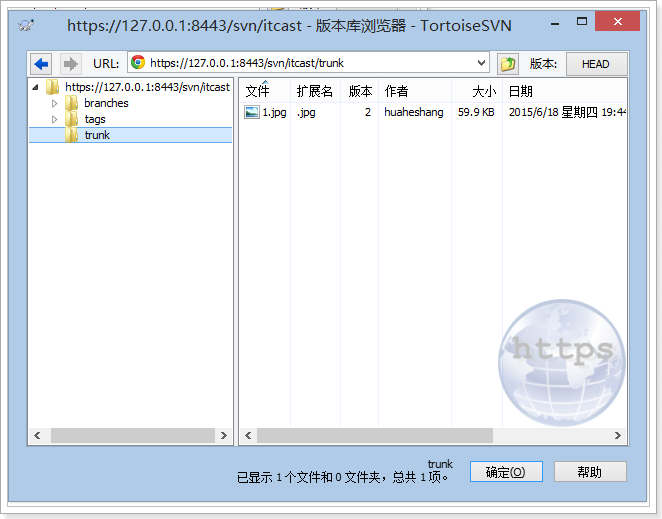






测试导入文件：





搞定。

# SVN协议

访问SVN资源库的协议通常有三种：

1. http
   1. 客户端和浏览器都可以访问。
2. https
   1. 客户端和浏览器都可以访问，同时比http更安全。
3. svn
   1. 只能通过客户端访问。

# 资源仓库

SVN资源仓库中有三个非常重要的目录，trunk（主干）、branches（分支）、tags（标签），其作用是：

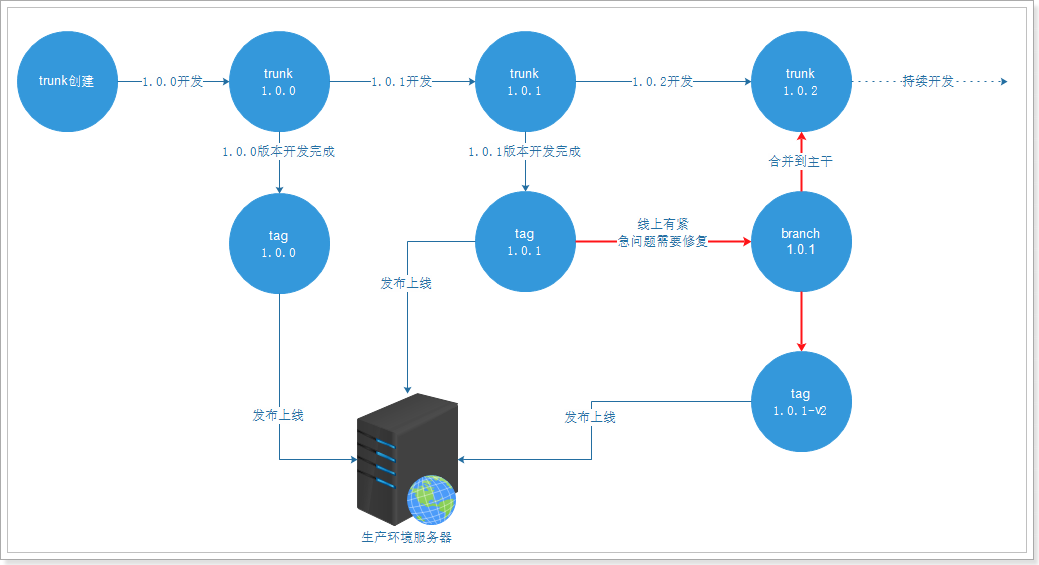
这三个目录，都是可以存放文件的，只是在功能方面有一些区别（只是约定）：

trunk：一般的项目都是基于主干开发的。

branches：分支，一般用于有某些开发功能时，需要基于主干开分支，开发完成后要合并到主干。

tags：标签，项目开发完成后，发布上线时，需要将主干的代码打标签到tags中，一般情况下，tags的内容是只读的。

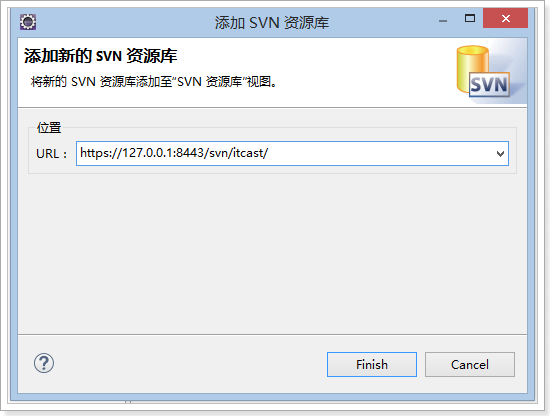
这三个目录在实际项目开发过程中的应用：

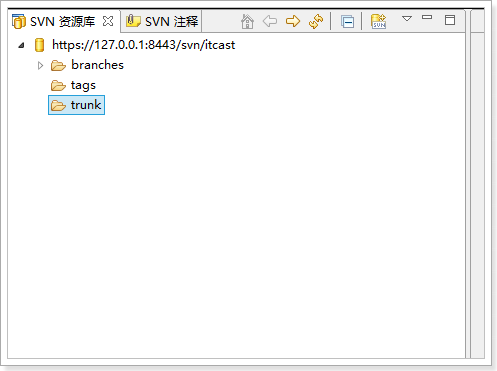


# SVN项目实战

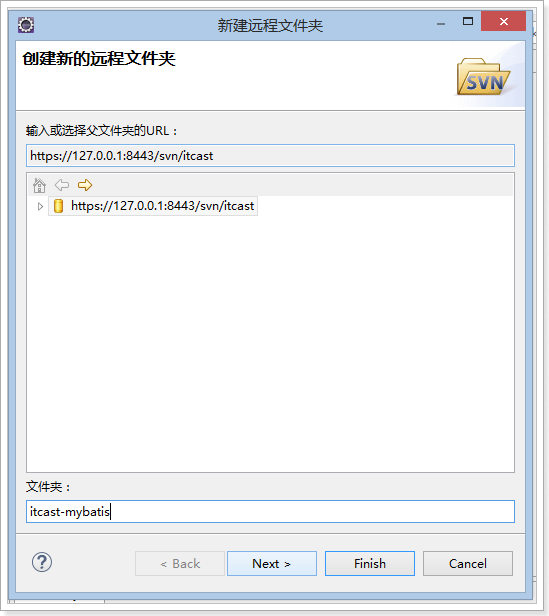
在企业开发过程中，更为常用的是通过Eclipse的插件来管理文件版本。

## 在Eclipse中创建SVN连接

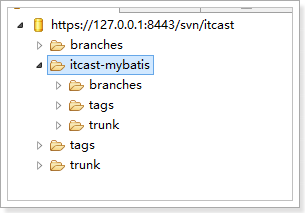


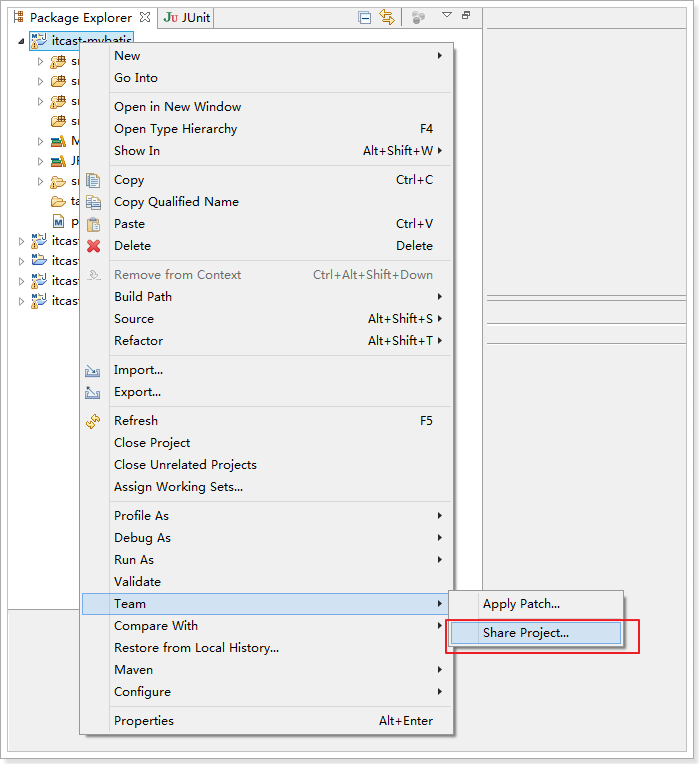


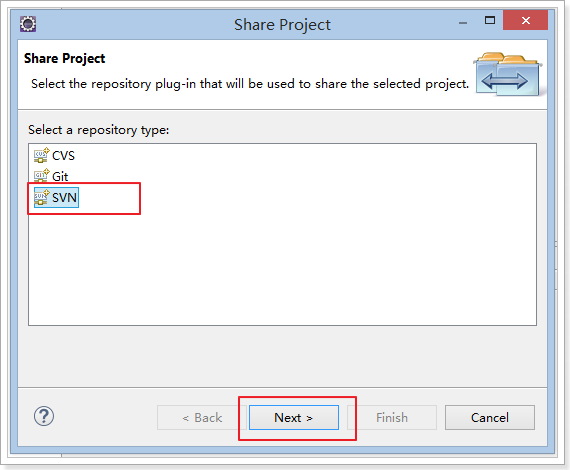
## 将项目发布到SVN的trunk

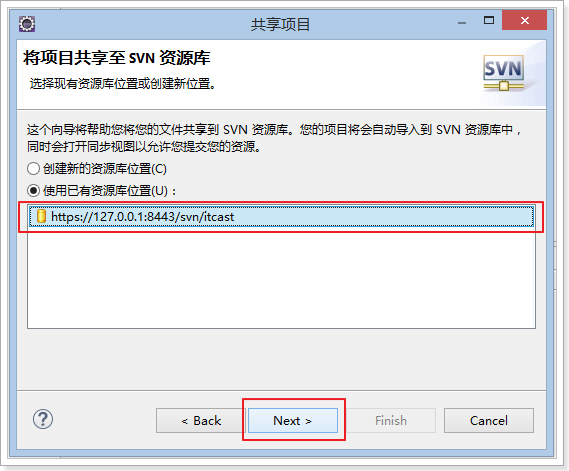


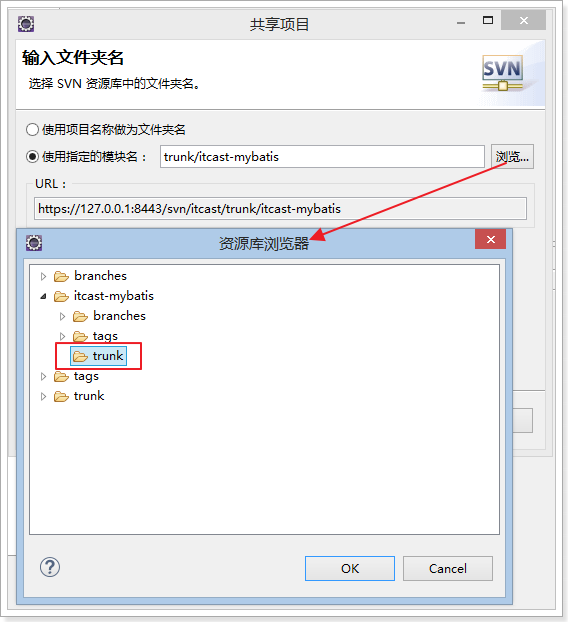
分别创建trunk、branches、tags：

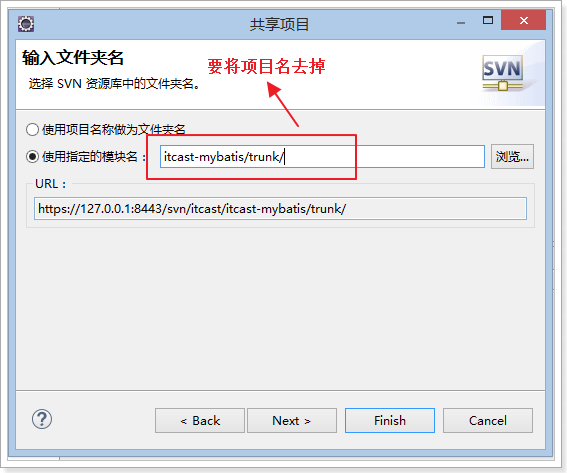


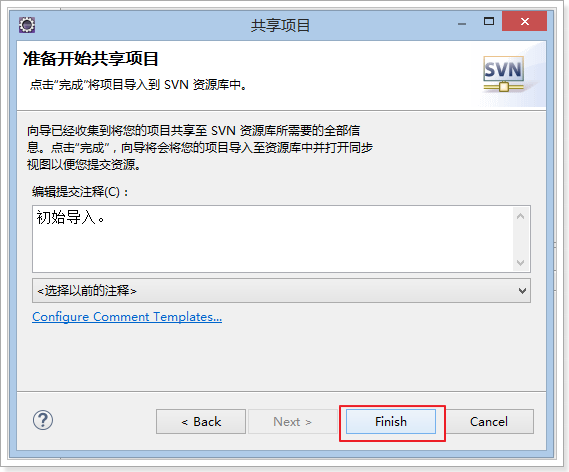


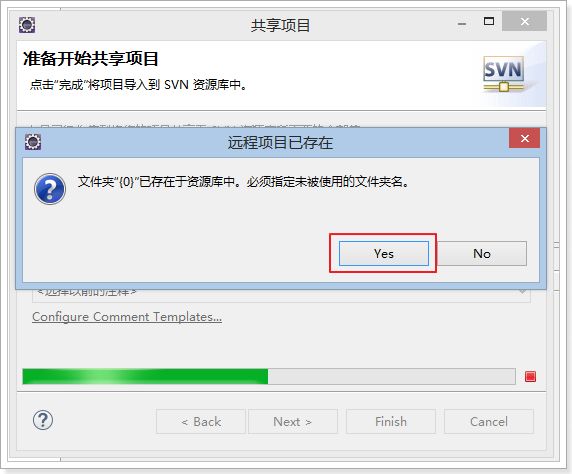




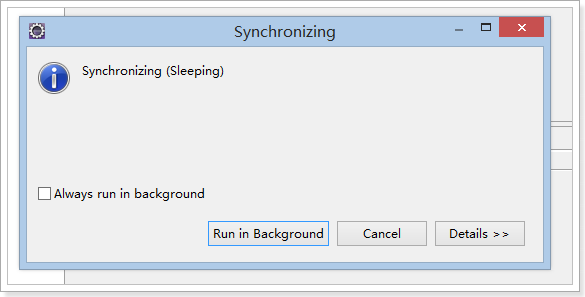


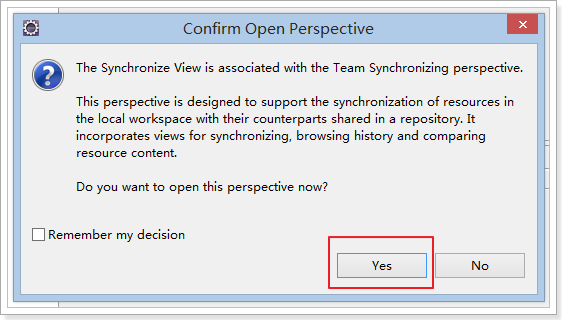


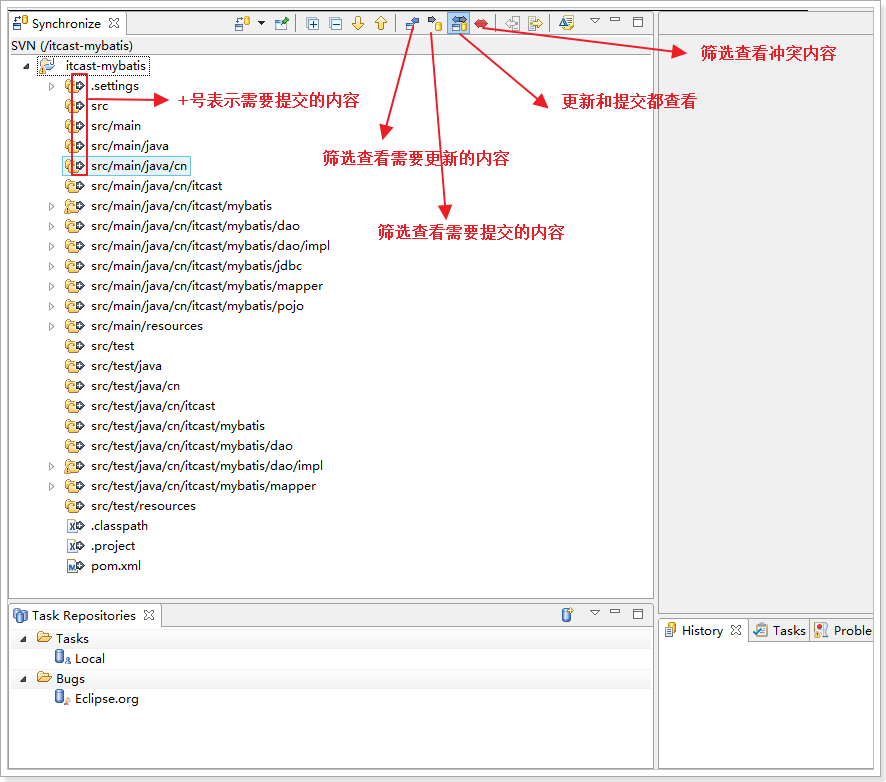




与资源库进行同步（比较不同）：

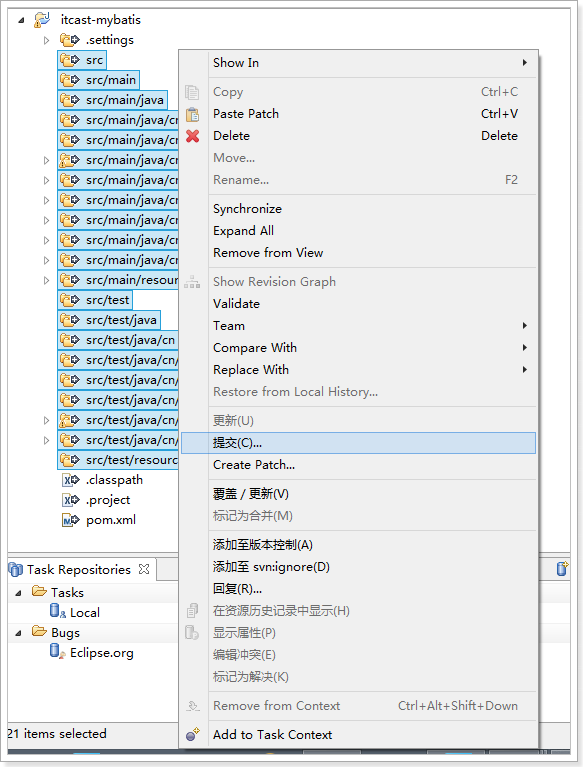


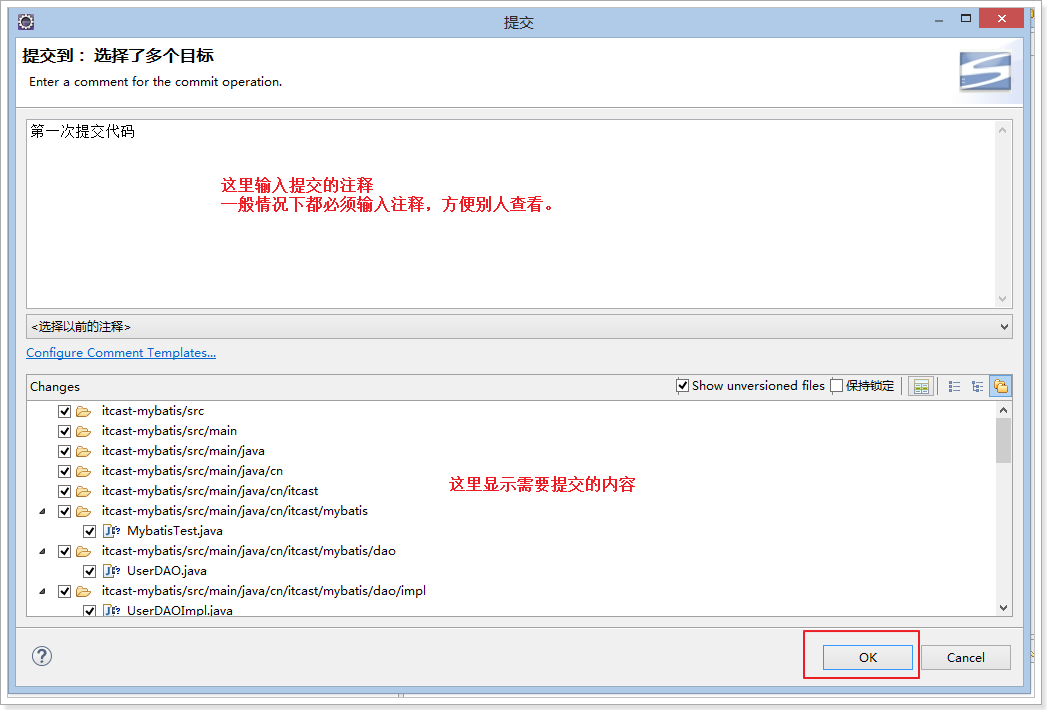




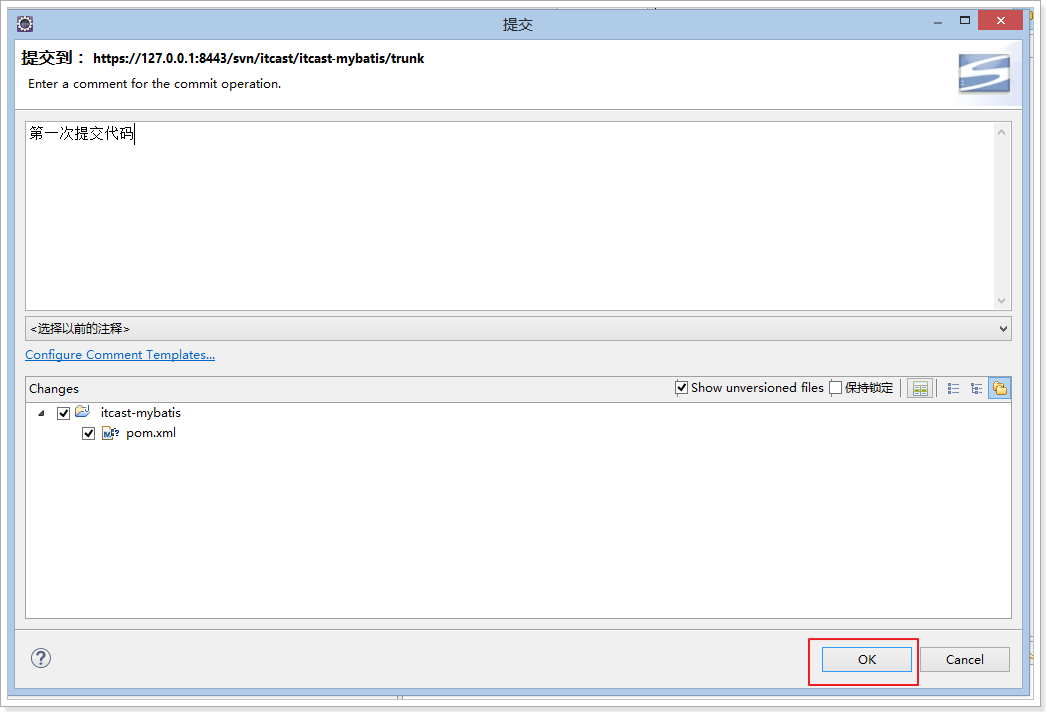
这么多内容都是应该提交到SVN的吗？

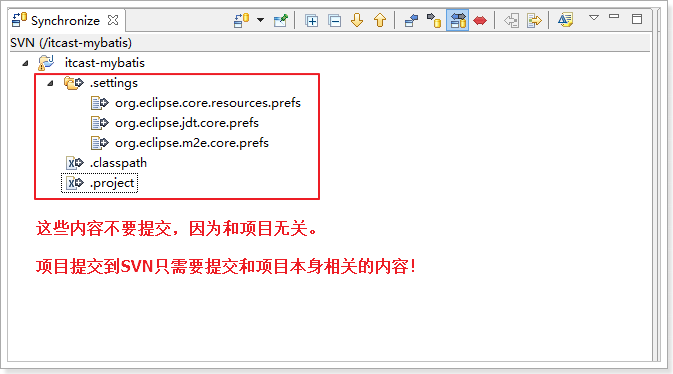
不是的，对于Maven项目而言，只提交**src**和**pom.xml**即可。





提交pom.xml

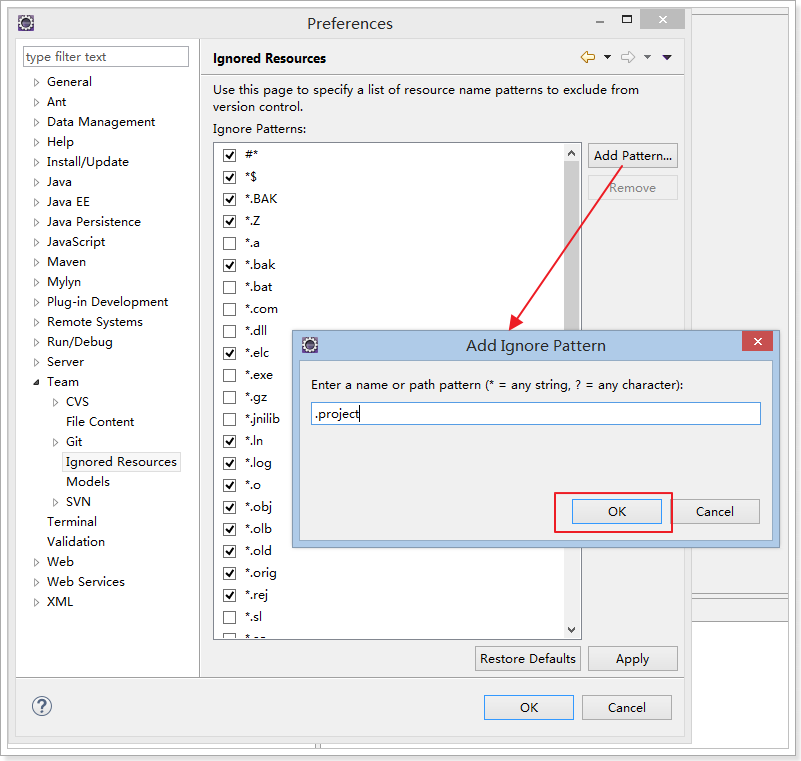


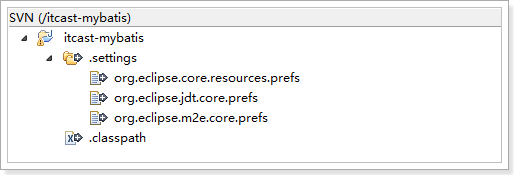


每次执行与资源库同步时都会有以上内容，会影响每次提交的选择，所以可以选择将这些目录文件忽略掉。

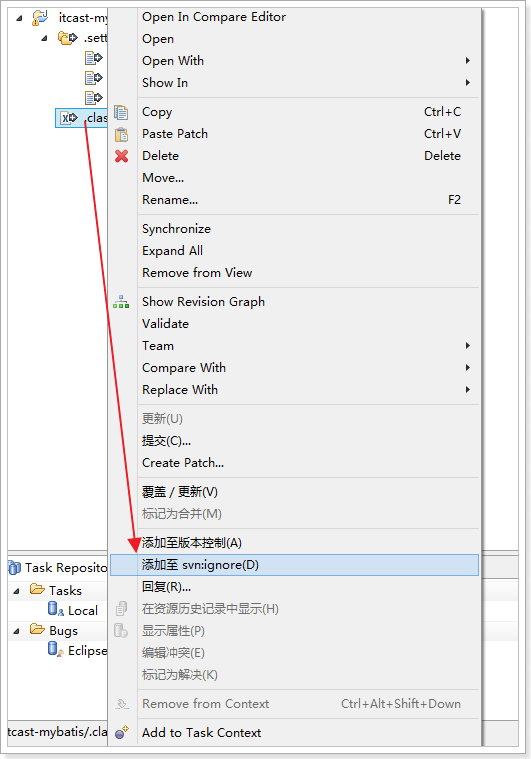
## 忽略指定的资源

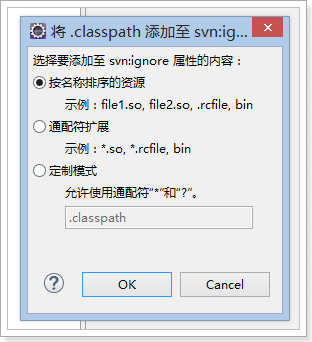
### 全局指定

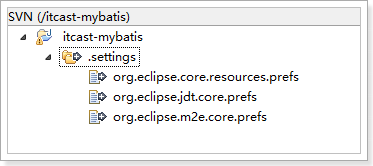




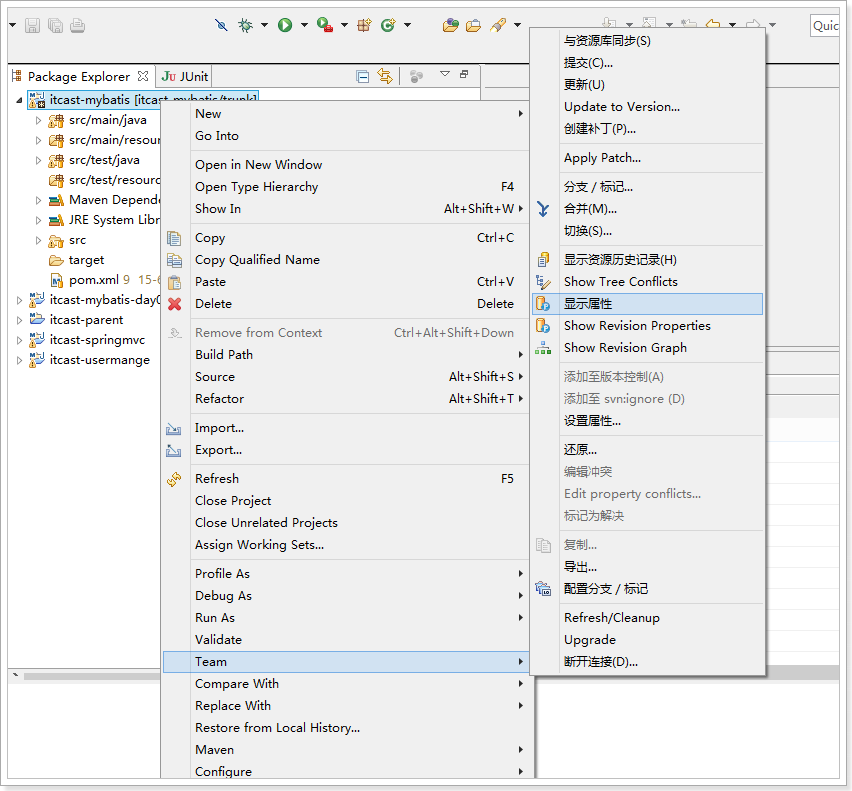
### 单个指定

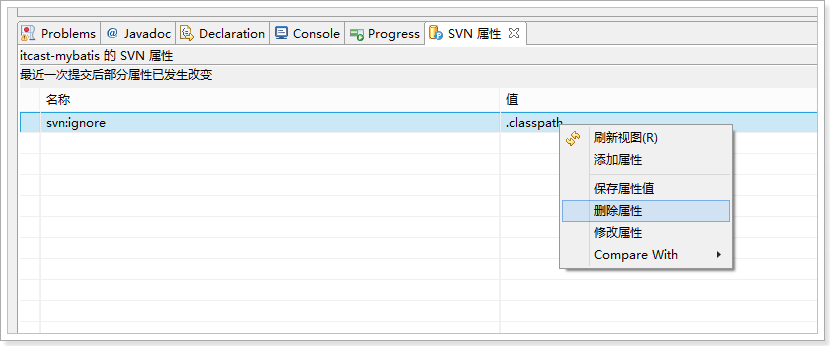






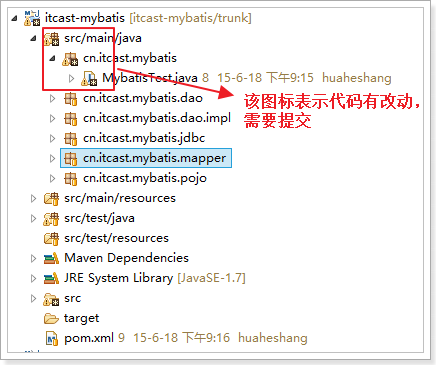
删除忽略：

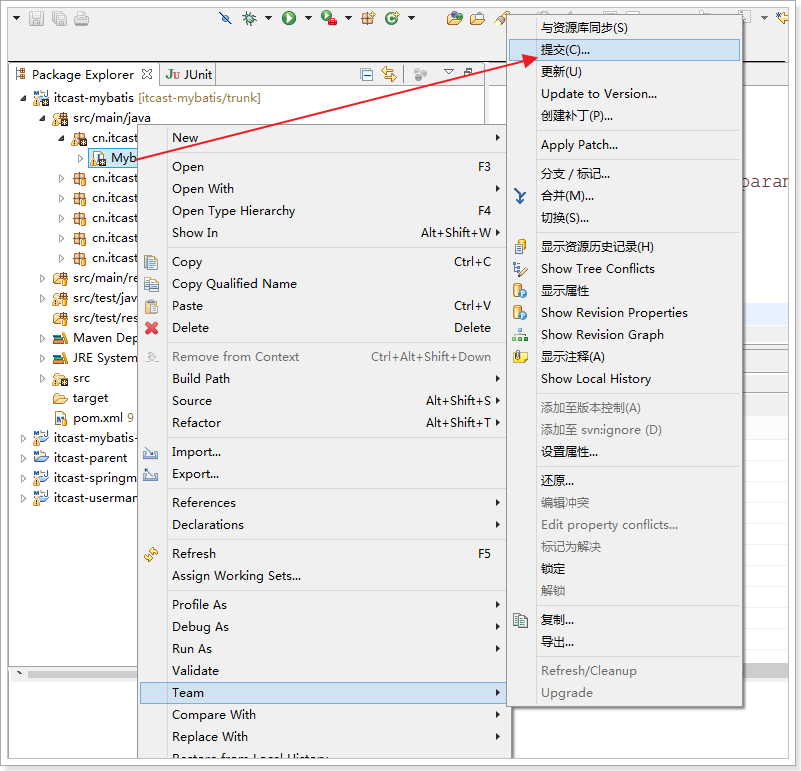




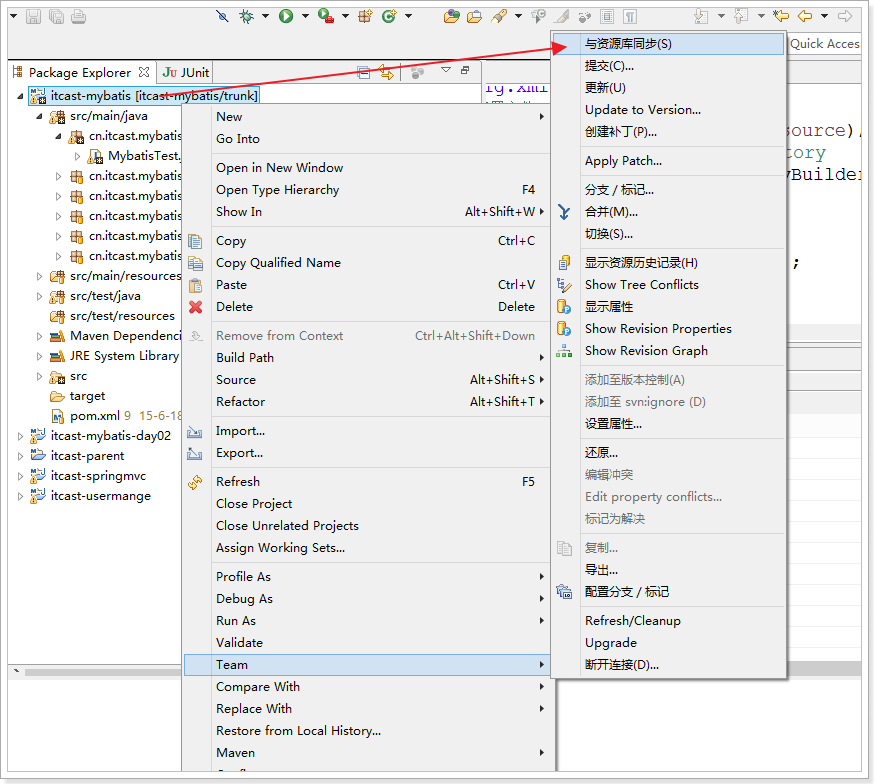
## 提交代码

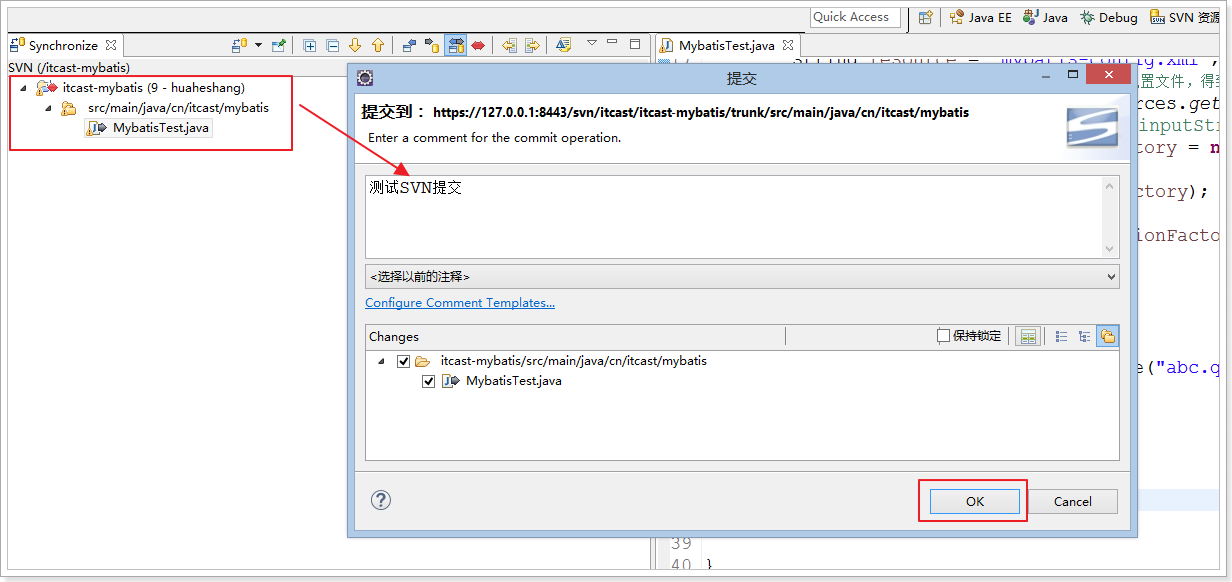
当我们在本地将代码修改后需要提交到SVN仓库，以便别人可以获取到最新的代码。





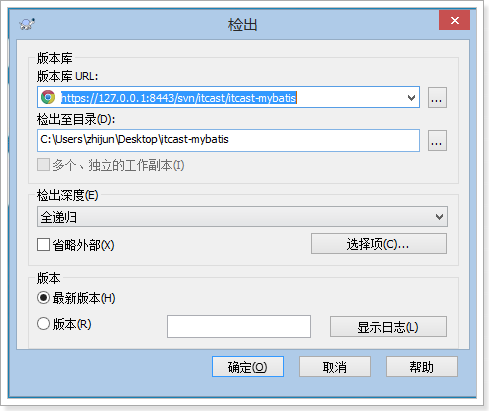
**注意：不建议直接提交，因为该文件可能会被其他人修改，从而造成冲突，推荐在提交（更新）之前先执行与资源库同步。**

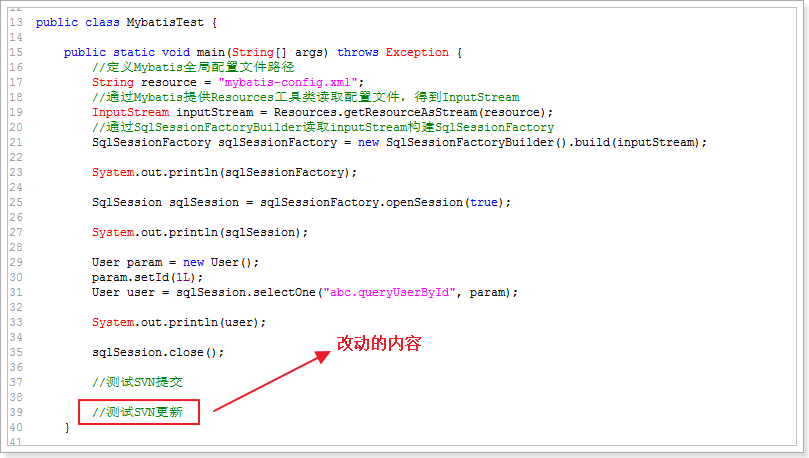


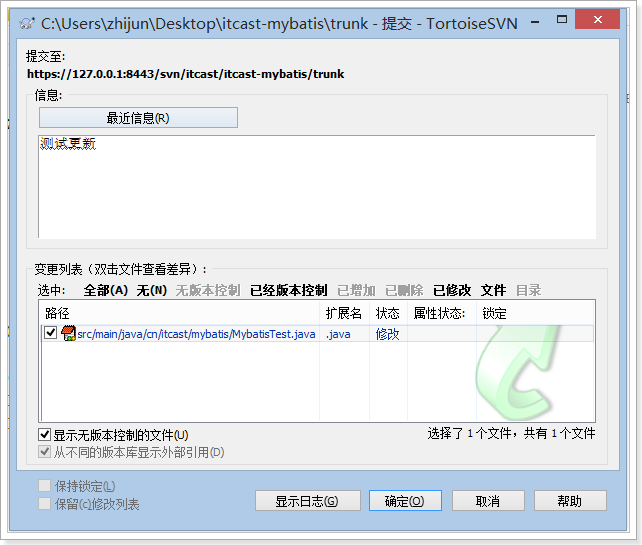


## 更新代码

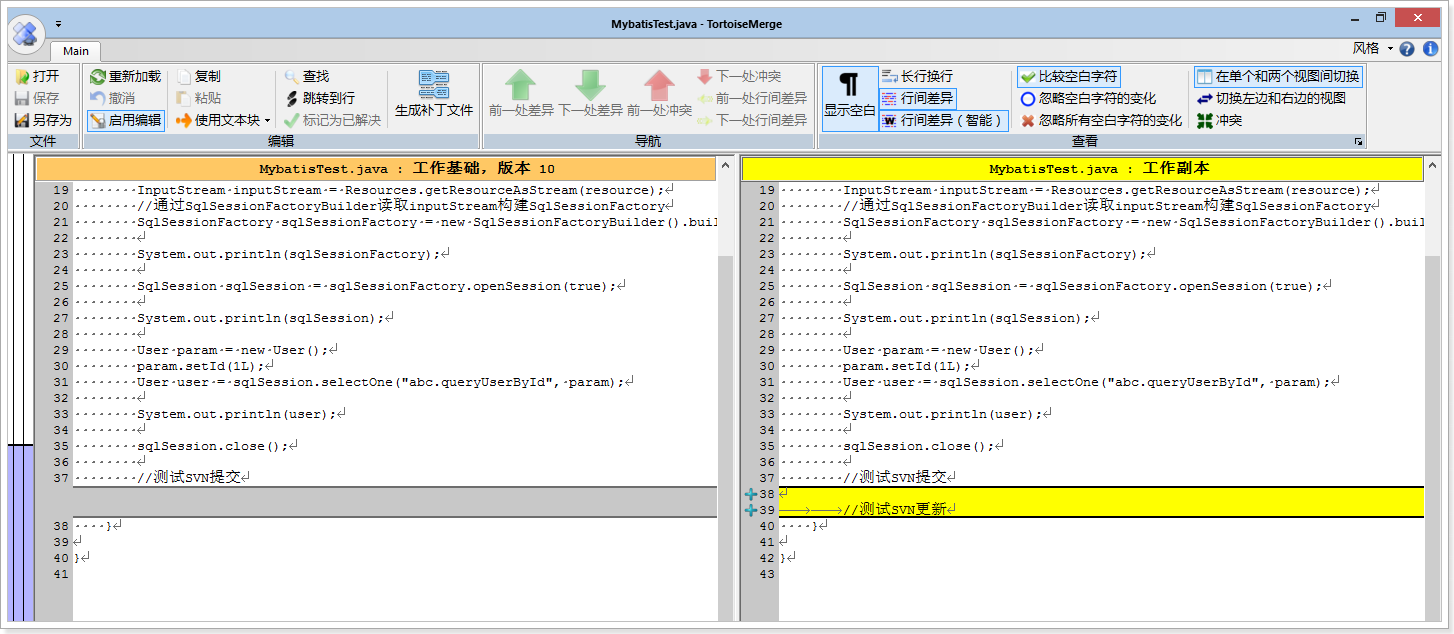
为了演示效果，在桌面将itcast-mybatis项目检出（check out），并且修改其中的文件完成提交。



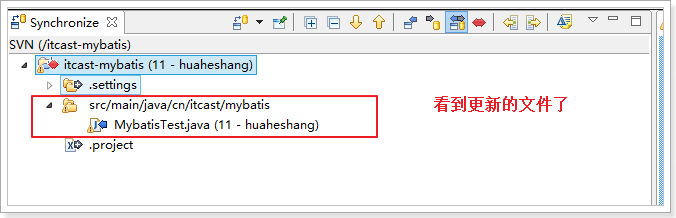


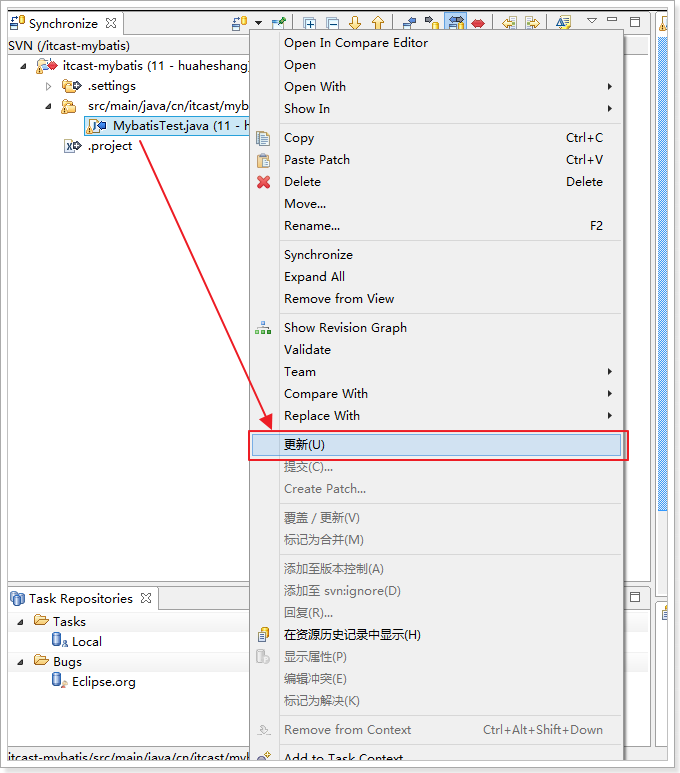


可以双击文件查看差异：



然后，在Eclipse中将项目与资源库同步：







## 冲突解决（难点亦是重点）

什么是冲突？

冲突就是在同一个版本基础之上，多个人对该文件修改了修改，其中一个人将文件提交到SVN，这时，该文件已经是新的版本，但是，其他人的本地还是旧的版本，

这时，其他人并不知道该文件已经有了新的版本，执行提交操作，这时就产生了冲突。

解决冲突的核心思想：为了避免冲突，要在最新的版本之上修改（也就是说修改之前先更新），再提交。

如果我更新了之后，在编写代码的同时别人将该文件再次更新（我不可能时时刻刻都查看更新），这时直接提交会造成冲突，正确的做法是：提交之前将该文件先执行与资源库同步操作，先将冲突解决掉再提交代码。

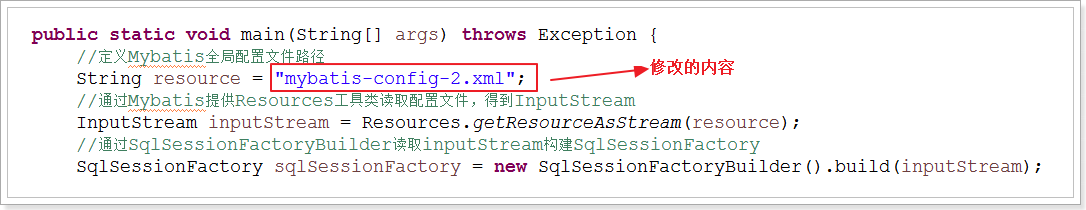
接下来就需要讨论下个话题了，如何解决冲突？

首先要先明确，解决冲突是不能通过工具自动完成的，必须人工完成，当然了，可以借助工具辅助完成。

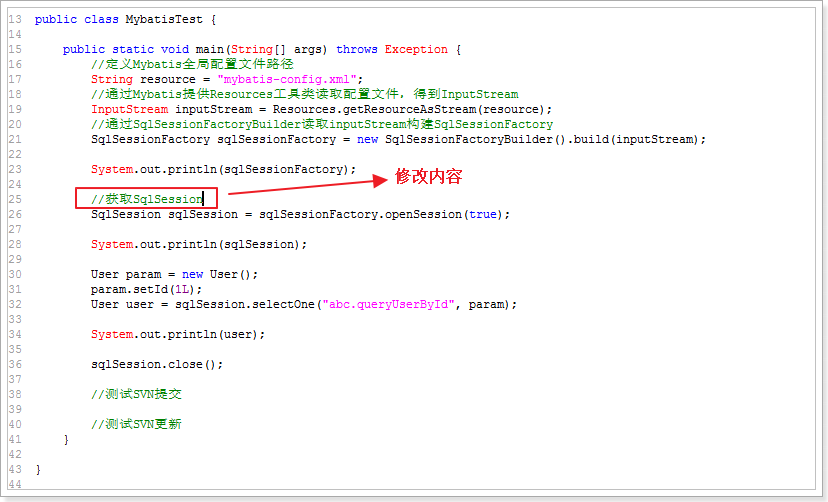
下面，演示冲突的解决过程：

### 制造冲突

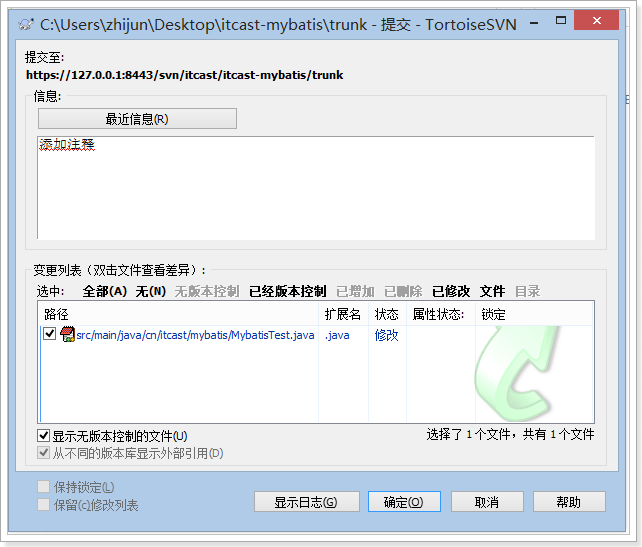
在Eclipse中将文件内容修改，用于模拟用户1修改文件：

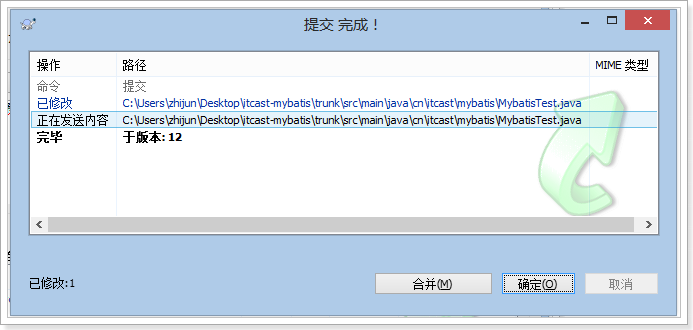


然后，在桌面中的目录中修改该文件，用于模拟用户2修改文件：



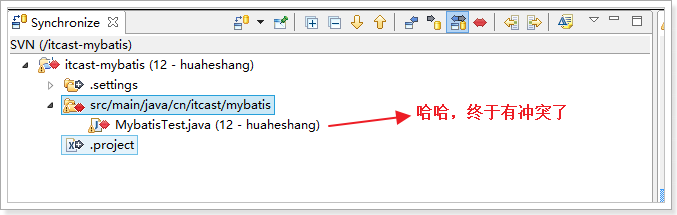
这时，先将桌面中的文件提交：



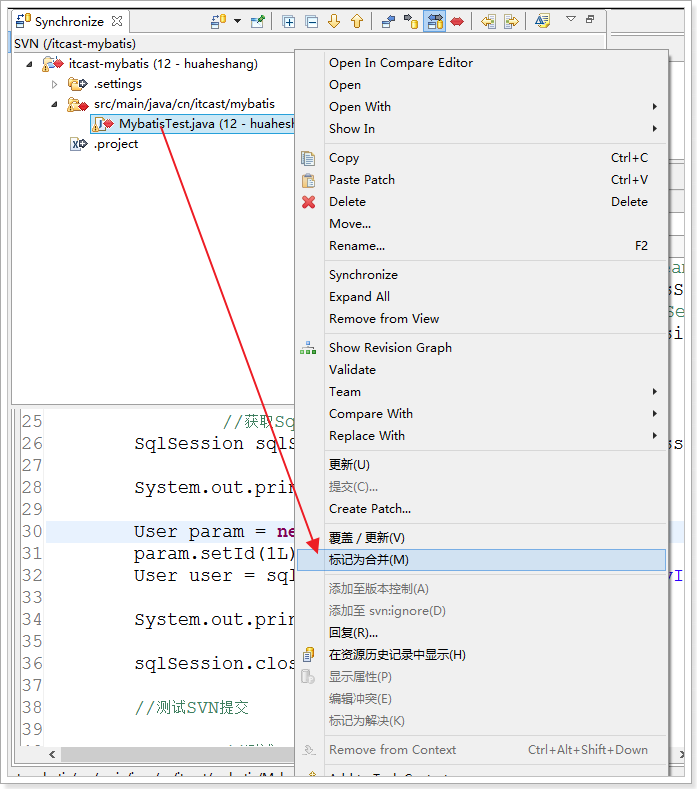
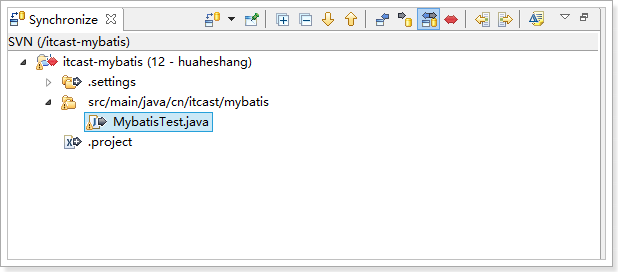


提交成功。

在Eclipse中将项目与资源库同步：



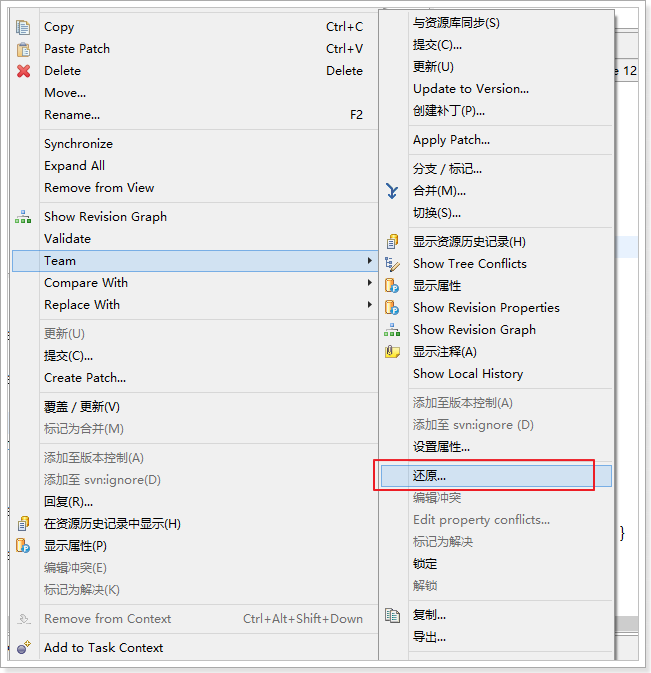
### 解决冲突（关键内容来了）

1. 双击打开该文件，查看冲突的内容  
   
2. 将远程更新的内容写到本地  
   
3. 将该文件标记为合并（注意，一定是已经处理完冲突了才能标记，要不然会将服务端的文件覆盖掉）  
   
4. 现在，就可以大胆的提交了  
   

## 冲突的另一种解决方案

有些时候可能会是这种情况：

服务端文件的内容被大量的修改，如果按照上面的方法一个个解决，非常的麻烦，这时你可以尝试以下的解决方案：

1. 备份本地的文件
2. 将该文件执行 还原 操作，并且再执行 更新 操作（也就说，放弃自己的修改，更新到最新的版本）  
   
3. 将备份文件中修改的内容，拷贝（不是全部啊，只是自己修改的部分内容）回该文件，再执行提交就OK了。

总结：这种解决方案的核心思想是，放弃自己的修改，把本地文件更新到最新版本，在最新版本基础之上修改，并且提交。