SVN 服务器端、客户端安装以及集成到 eclipse 的详细步骤

以下的每一步都是本人(steve_wang_victor)亲自操作后写下的。

需要使用的软件版本如下：
Eclipse3.3
svn-1.4.5-setup 服务器
TortoiseSVN-1.4.5.10425-win32-svn-1.4.5 客户端

一，安装服务器端 svn 和客户端 svn

1， 首先安装服务器端软件“svn-1.4.5-setup.exe”（附件可以下载），这个安装就很简单，下一步，下一步就行了。
（我这里安装到目录：D:\Program Files\Subversion）
2， 其次安装客户端软件“TortoiseSVN-1.4.5.10425-win32-svn-1.4.5”
这个安装也很简单，下一步就行了。客户端软件安装好以后，它会要求你重新启动电脑。重启一下。
现在距离成功前进一步了!!(*^__^*) ……

二，建立 svn 版本控制的服务目录

1， 这里就是把“D:\svn_service_root” 这个目录指定为 SVN 版本控制的服务目录（网上很多帖子把上面这个命令称作建立仓库是不对的）其实这个目录应该称作仓库所在的目录，假如我有三个仓库：repository_1,repository_2,repository_3 他们的位置应该是：
d:\svn_service_root\repository_1，
d:\svn_service_root\repository_2，
d:\svn_service_root\repository_3。
2，建立了一个仓库：有 2 中办法如下
2.1）命令行模式进入”C:\Program Files\Subversion\bin\“
再打入命令：svnadmin create d:\svn_service_root\repository_1，这样第一个仓库就建立好了。以后你的项目就可以
导入到这个仓库中。（其他仓库的建立类似）此时进入目录：桌面－－我的电脑－－本地磁盘 E:
－－svn_service_root－－ repository_1 。你会看到文件夹 conf,dav,db 等等

2.2）打开目录：我的电脑－－本地磁盘 E:－－svn_service_root。在这个目录下新建文件夹，取名“repository_1”，右击刚才新建的文件夹－－>TortoiseSVN→Create Repository Here.效果同第一种方法一样。

3，打开 svn 的服务

在命令行上转到 subversion 目录下输入(就是刚才安装 svn 服务器的目录)：

```
svnserve -d -r D:\svn_service_root
```

该命令解释：

注：

-d 参数效果同于--daemon

-r 参数效果同于--root

svnserve 将会在端口 3690 等待请求，

--daemon（两个短横线）选项告诉 svnserve 以守护进程方式运行，这样在手动终止之前不会退出。不要关闭命令行窗口，关闭窗口会把 svnserve 停止。

可直接创建.bat 文件来处理当做系统服务 如下内容

```
sc create svnserve binpath= "C:\Program Files\CollabNet\Subversion
Server\svnserve.exe --service -r e:\svn" depend= Tcpip start= auto
sc start svnserve
pause
```

--root 选项设置根位置来限制服务器的访问目录，从而增加安全性和节约输入 svnserve URL 的时间

如果不加 root 参数，服务 url 为：svn://localhost/svn/repos

而如果加上 root 参数,服务 url 为：svn://localhost/repos

此处的启动配置会影响服务 url，如果输入 url 错误，会导致访问的时候出现异常 。

为了验证 svnserve 正常工作，使用 TortoiseSVN -> Repo-browser 来查看版本库。在弹出的 URL 对话框中输入：

svn://localhost/repos

成功访问后可看到空的文件目录!

这一步是建立开启客户端访问服务器端仓库目录下指定的文件

执行完这步，需要测试以下：

右键→ TortoiseSVN -> Repo-browser 来查看版本库。在弹出的 URL 对话框中输入：

svn://localhost/repository_1

访问成功后，会看到一个空的文件目录。（因为现在你还没有放任何东西在里面）

4，配置用户和权限

用文本编辑器打开 d:\svn_service_root\repository_1\conf 目录，修改 svnserve.conf:

将：

```
# password-db = passwd
```

改为：

```
password-db = passwd
```

即去掉前面的 # 注释符，注意前面不能有空格。
然后修改同目录的 passwd 文件，增加一个帐号：

将：
```
[users]
# harry = harryssecret
# sally = sallyssecret
```
增加帐号：
```
[users]
#harry = harryssecret
#sally = sallyssecret
admin= admin
```

如此就可以用客户端 Tortoise 进行操作了

SVN 安装和服务开启全部结束!!! 距离在 eclipse 中使用已经完成一大半了！

关于 svn://localhost/repository_1 路径问题多说一句：
如果你的服务端装在本机就可以这样访问：右击－－SVN checkout .此时会打开一个对话框。
在 URL of repository 下面输入 svn://localhost/repository_1

如果是其他电脑访问这个仓库：
svn://10.1.246.68/repository_1，其中 10.1.246.68 是我的 IP 地址，到时候大家可以自己替换 。
其实这里的 svn://10.1.246.68 地址对应 D:\svn_service_root 目录
【原因：因为我在第三步中使用的命令： svnserve -d -r D:\svn_service_root ，把 D:\svn_service_root 目录安装成可以通过 SVN 协议来访问】

三，Eclipse 集成 svn 的使用

1，
之前安装好了 SVN 的客户端和服务器端，也配置了用户权限，现在看看如何在 eclipse 中集成 SVN 的插件 Subclipse. 其实官网上写的很清楚！这里只是稍微说说。

在 help—>software updates →find and install… →选择 search for new features to install –>下一步->new remote site..->name: subclipse url: http://subclipse.tigris.org/update_1.2.x -->选择一个版本-->下一步--->下一步----->install all->重启 eclipse
在 window ->open prespective ->看到”SVN 资源库研究”

这就已经集成好了。

2， 运用到所建立的工程项目中

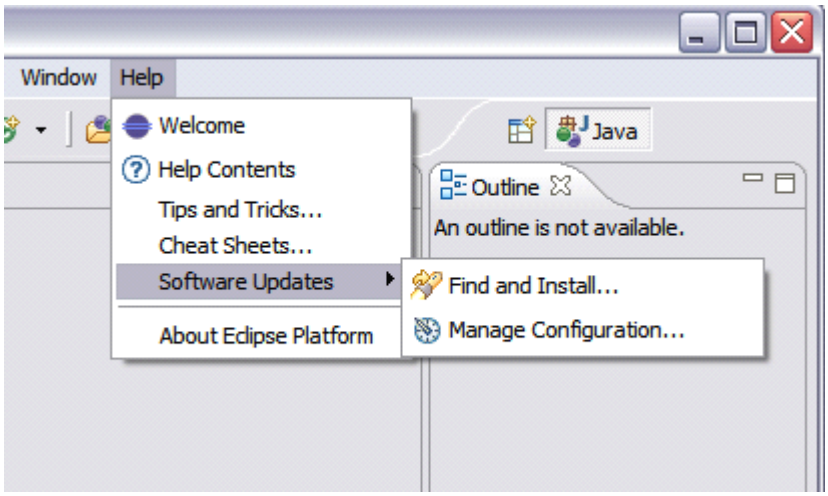
随便选一个工程，右键，选择“team”，选择 SVN，点击 next, 选择:使用已有资源库位置（svn://localhost/repository_1）,点击 next，选择项目名或者新建项目名（就是重新取一个名字而已），点击 next，编辑提交备注，点击 finish.

执行到这步，就已经提交给 svn 服务器中的 repository_1 仓库管理该工程的版本了。

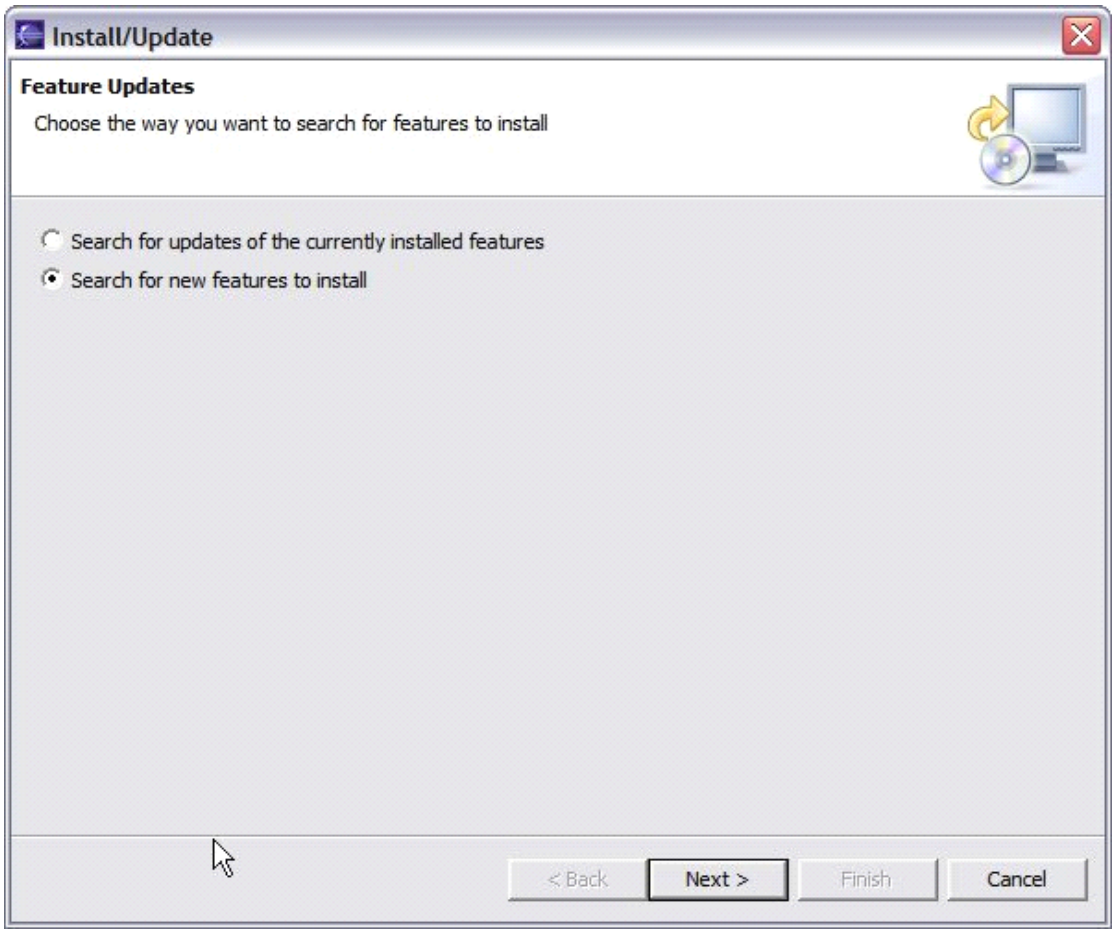
终于执行完了，累了吧，高兴下，好好休息下吧！嘿嘿,(*^__^*) ……

附件

以下是 svn 集成 eclipse 的图解：
以前提到过 SVN 版本控制器和客户端的安装和配置，这里再在说一下在编译器中怎么使用 SVN，使其既可以单独使用，也可以配合 SVN 客户端一起使用。由于编译器种类众多，插件种类也不同，这里只简单介绍下 Eclipse 的插件安装。我用的版本是 Eclipse3.2.2+MyEclipse_5.1.1，由于本身只有 CVS 而没有集成 SVN，所以要想使用 SVN 做版本控制只好装一个 Eclipse 插件 Subclipse，Subclipse 的详细安装过程在其官方网站 http://subversion.tigris.org 写的很清楚，还带配图，我就直接 copy 了：
一、Install Subclipse in Eclipse 3.x（安装）
Step 1:
Begin the installation from the Eclipse Help menu item.

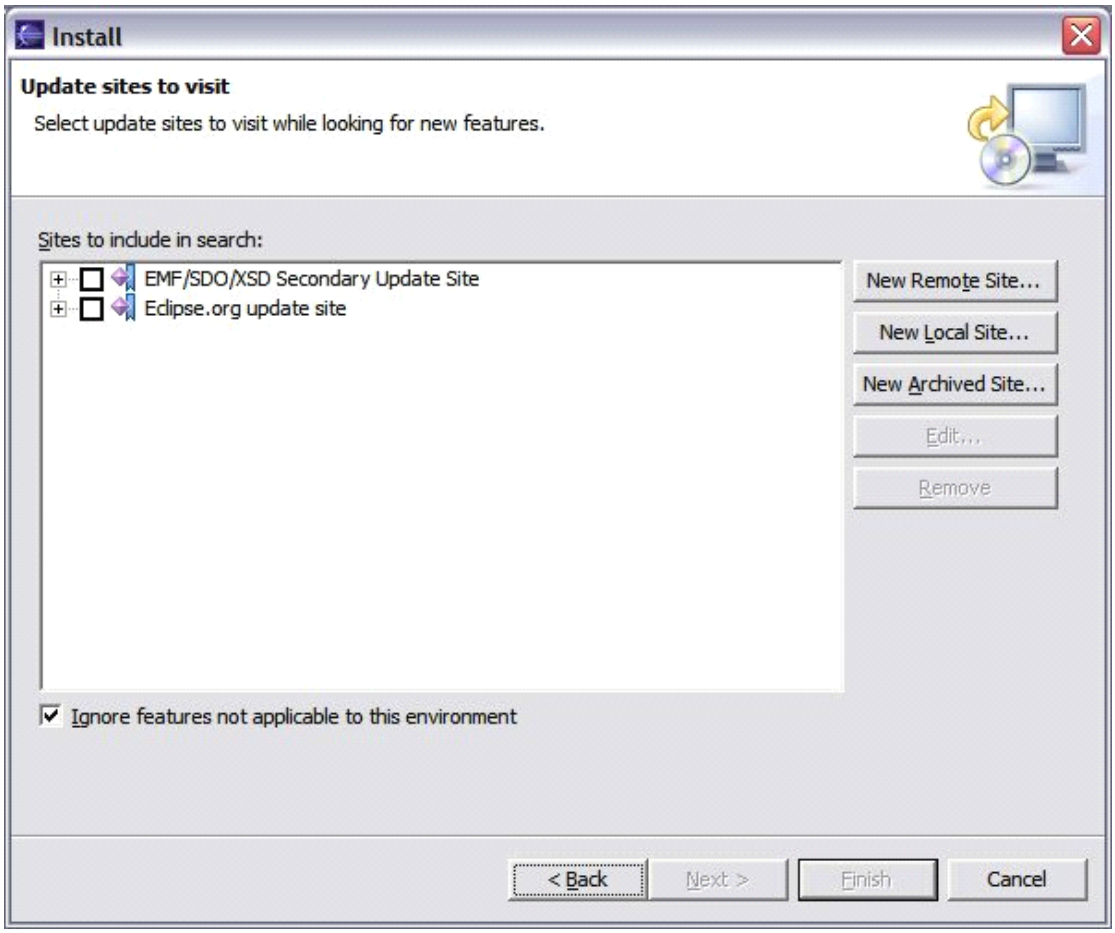


Step 2:
This screenshot show the screen as it initially comes up. In this case you will need to change the radio button to indicate that this is a new install.

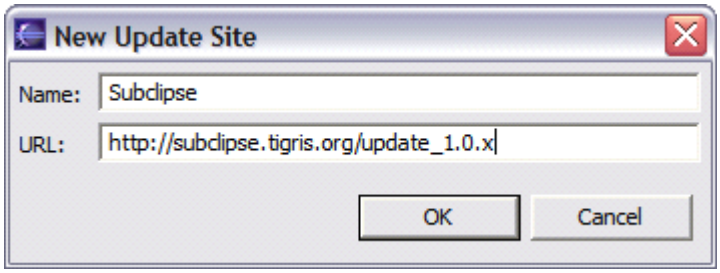


Step 3:

This screen will vary depending on the features you have installed already. You want to click on the New Remote Site button. If you are behind a proxy and the Eclipse install mechanism does not work, then you can download a zipped version of the update site and then click the New Local Site button instead.

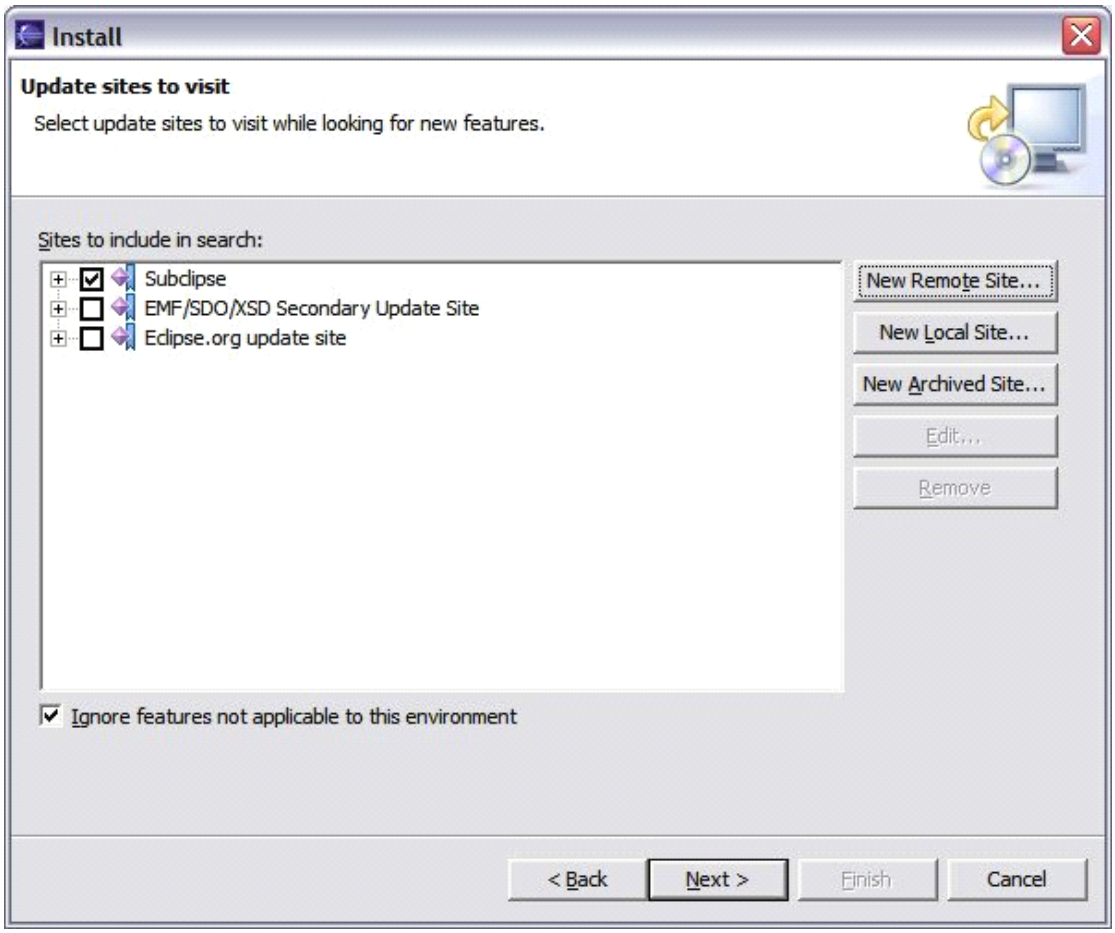


Step 4:
This screen is showing the New Remote Site dialog, filled in with the correct information to install Subclipse

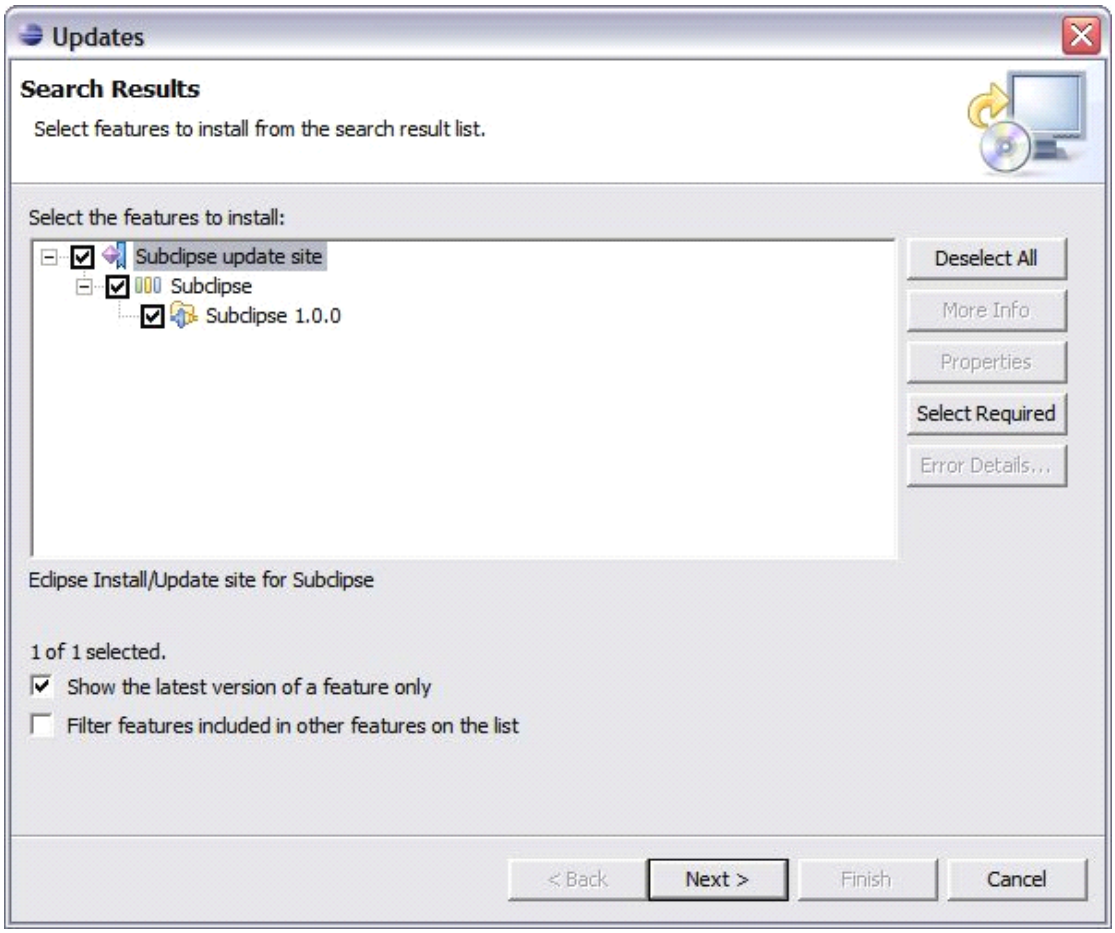


Name: Subclipse 1.2.x (Eclipse 3.2+)
URL: http://subclipse.tigris.org/update_1.2.x

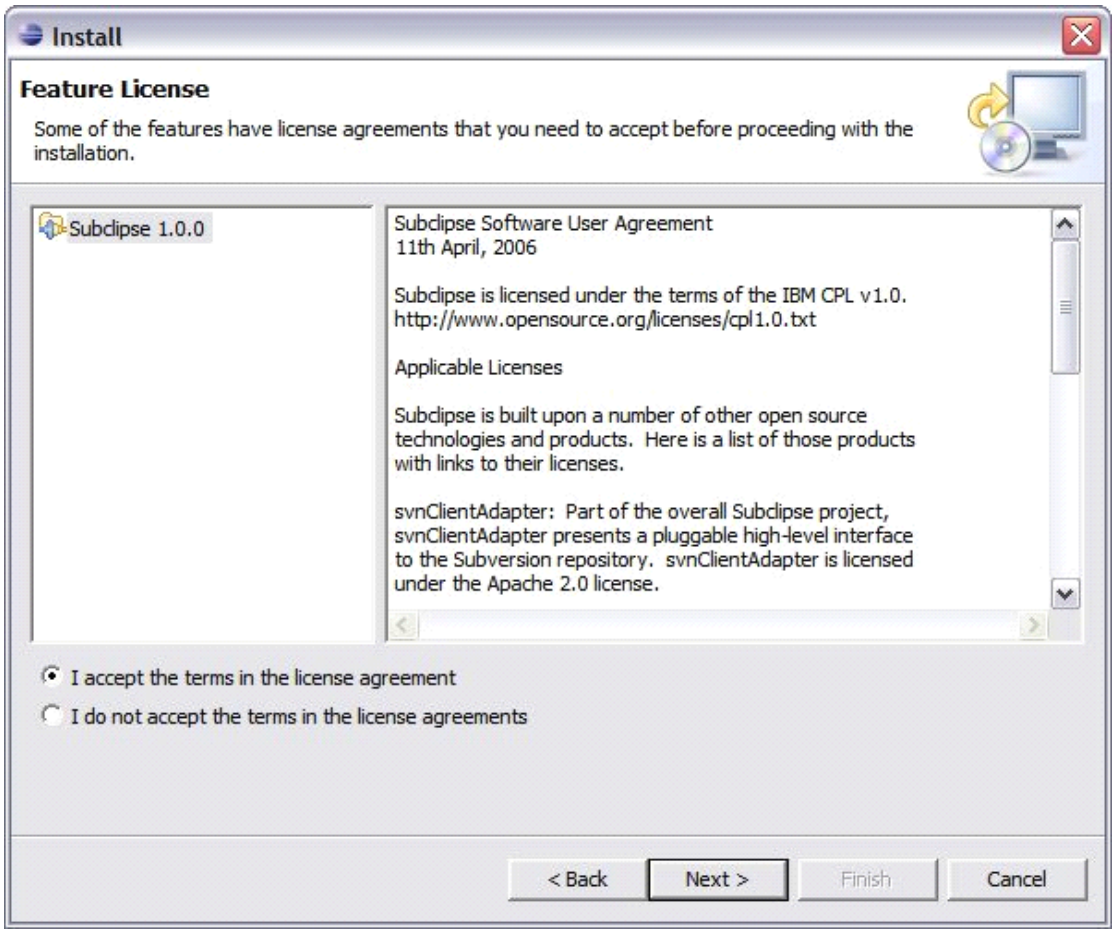
Name: Subclipse 1.0.x (Eclipse 3.0/3.1)
URL: http://subclipse.tigris.org/update_1.0.xStep 5:
When you first come back to this screen, the site you added will NOT be selected. Be sure to select it before clicking Next.



Step 6:
This next screen shows all of the features that are available to install.

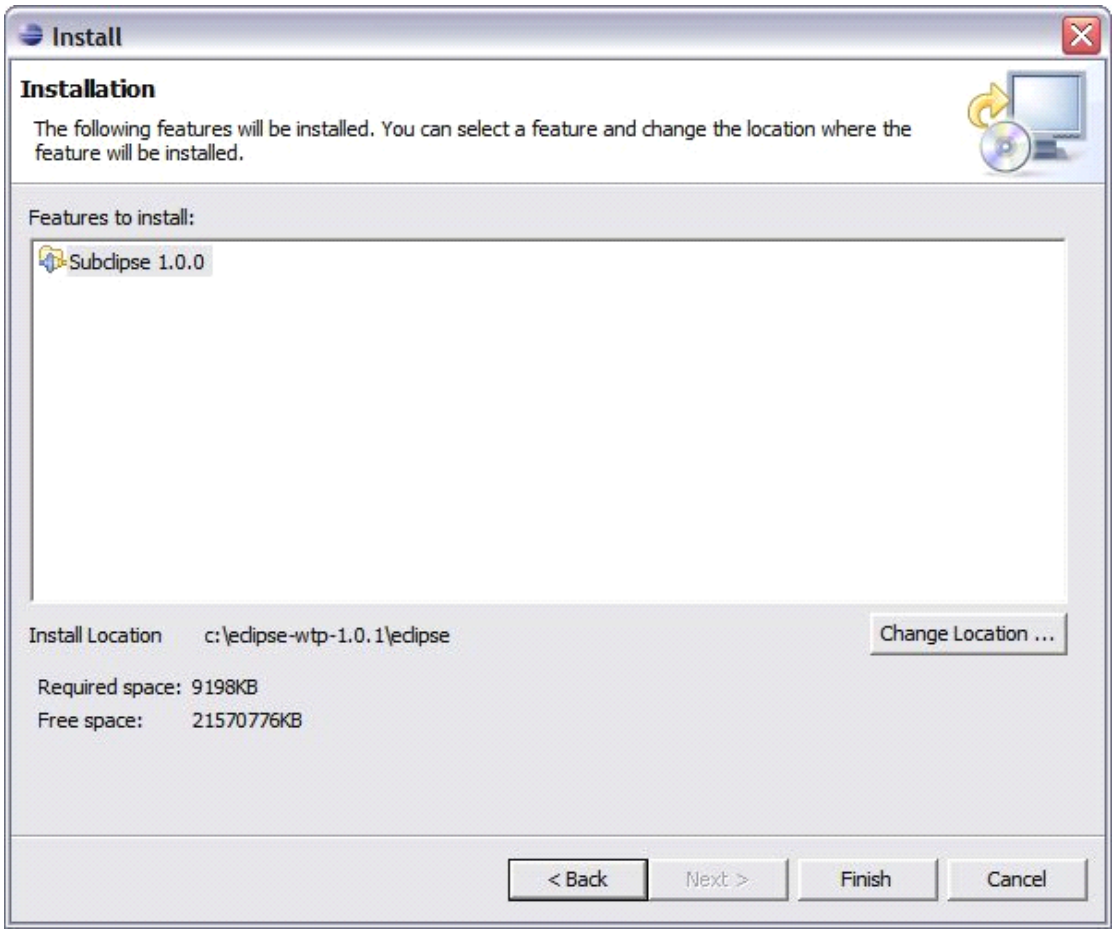


Step 7:
Click the button to accept the license agreement.



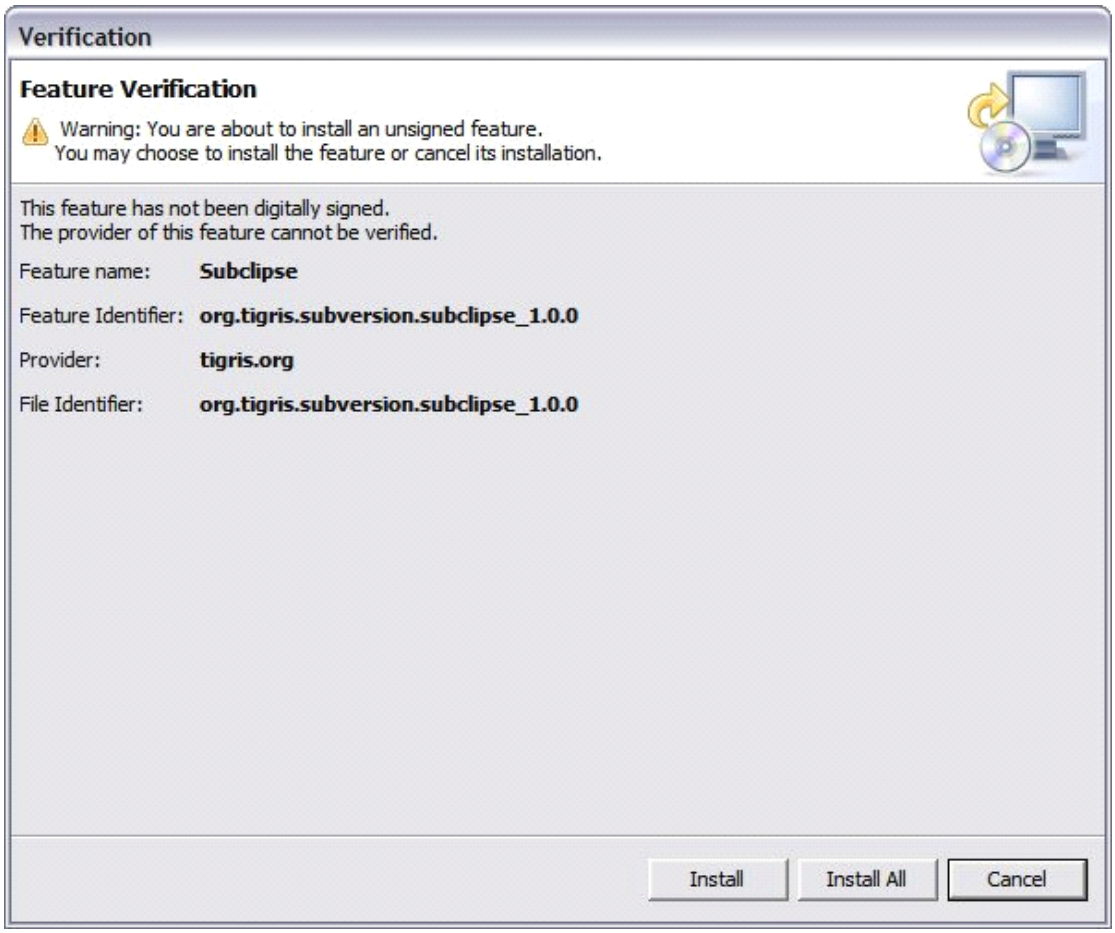
Step 8:

Confirm the install location

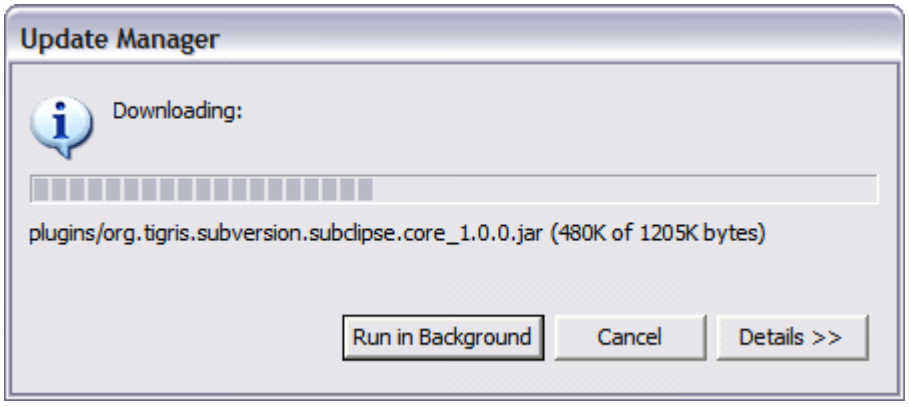


Step 9:

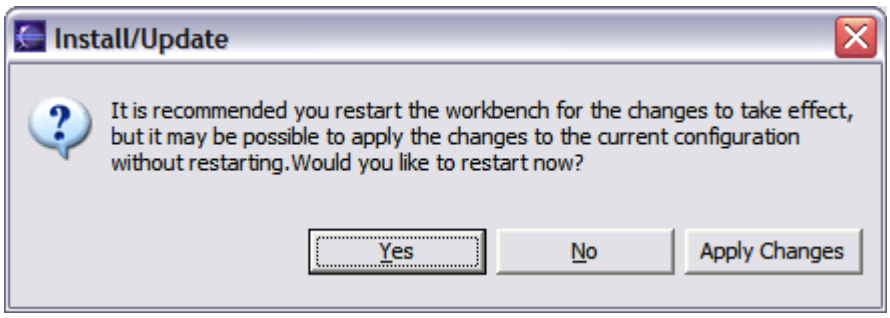
There is an Eclipse preference to turn off this next dialog. I have never seen a signed feature. Not even Eclipse.org nor IBM sign their features.



Step 10:
Just a screenshot of the in-process installation.

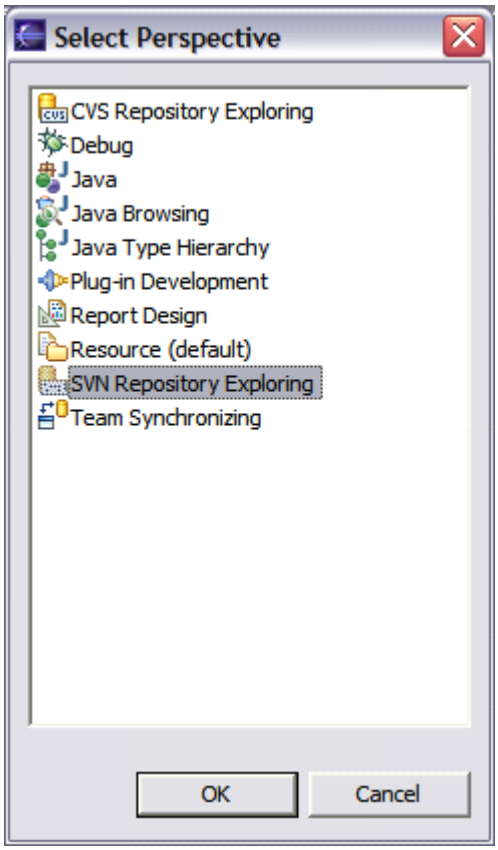


Step 11:
Eclipse needs to be restarted after installing Subclipse.



Step 12:

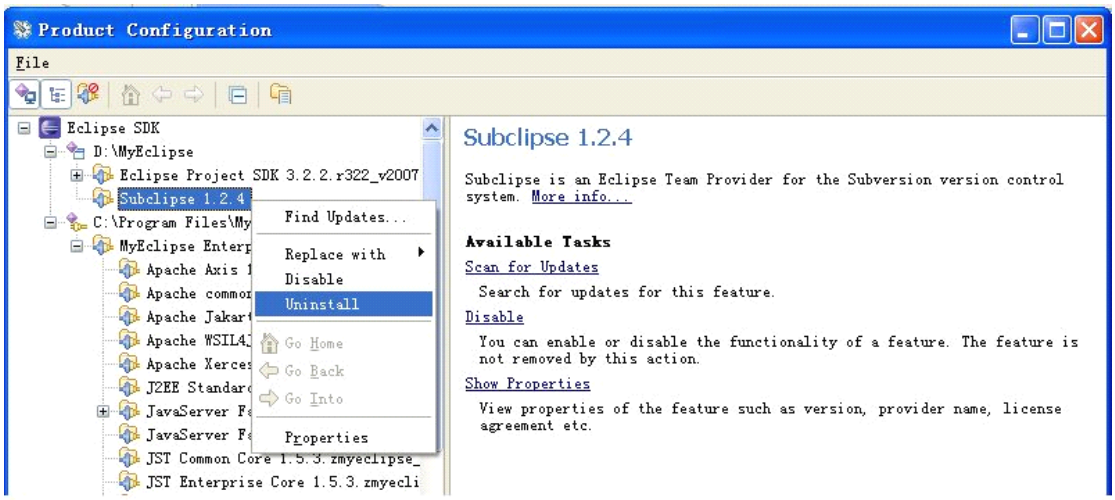
Finally, after restarting Eclipse, the first thing you will typically want to do is open the Subclipse Repository perspective where you can define your repositories. Be sure to also check the online help as well as the Subclipse preferences located under Team -> SVN.



OK

到此你的插件已经安装完毕了，说明一下，在 Step 5 中选择的是在线安装，也可以在 http://subversion.tigris.org/servlets/ProjectDocumentList?folderID=91 下载后进行本地安装，本地安装选择 New Local Sit...，然后找到解压出来的那个文件夹，后面的都一样了。

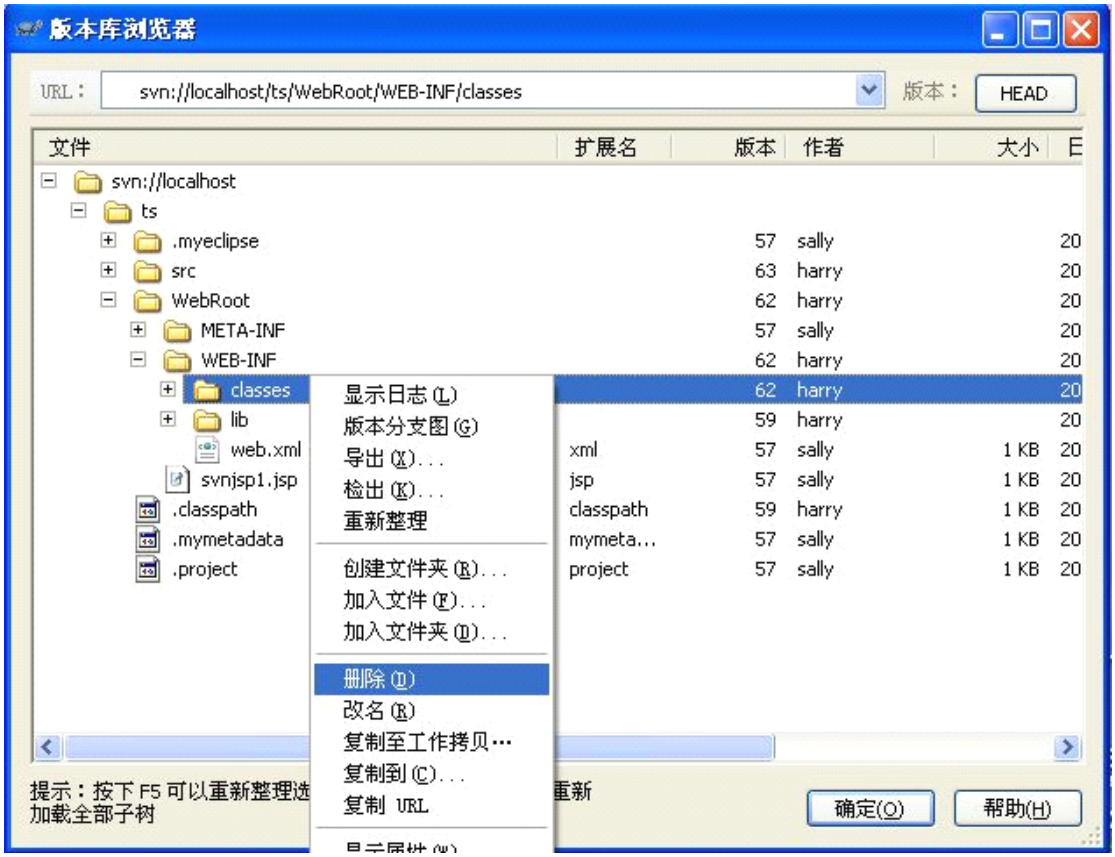
卸载的方法也很简单，也是点击 Help => Software Updates => Manage Configuration

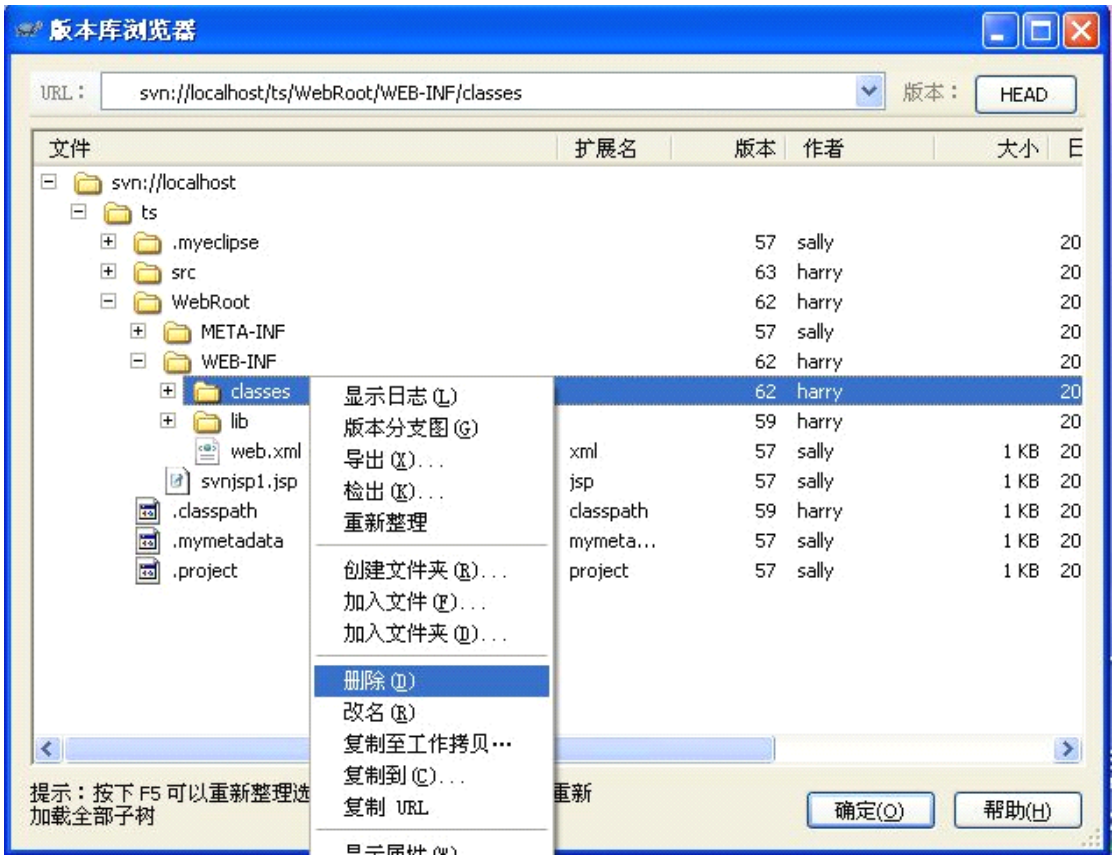


按上图操作就可以卸载了。

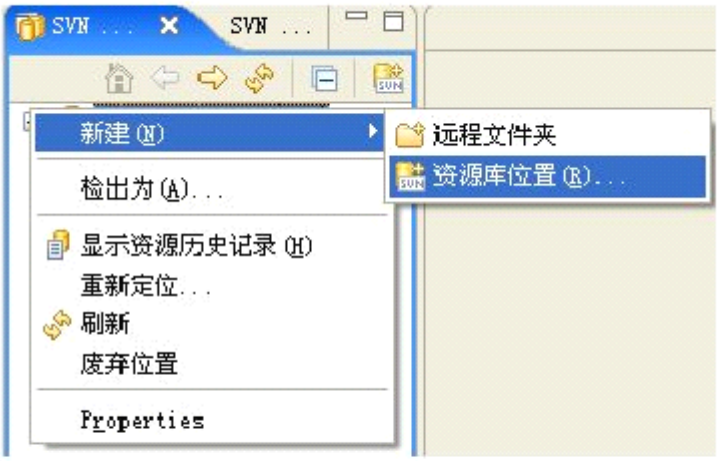
二、Use Subclipse in Eclipse 3.x（使用）

先向版本库中加入一个 Eclipse 工程，我这里是随便搭建的一个 WEB 工程 SVNtest，将它导入版本库 svn://localhost/ts（这使用本机装的版本库，其他机器就要打 ip 地址，之前说过）。导入后用版本库浏览器找到你导入的工程目录，把其下 WebRoot\WEB-INF\classes 目录删掉（注意：不删出以后会报错，原因后面再讨论）这些初始化的操作都用客户端工具完成的，毕竟插件的功能还不是那么强大。

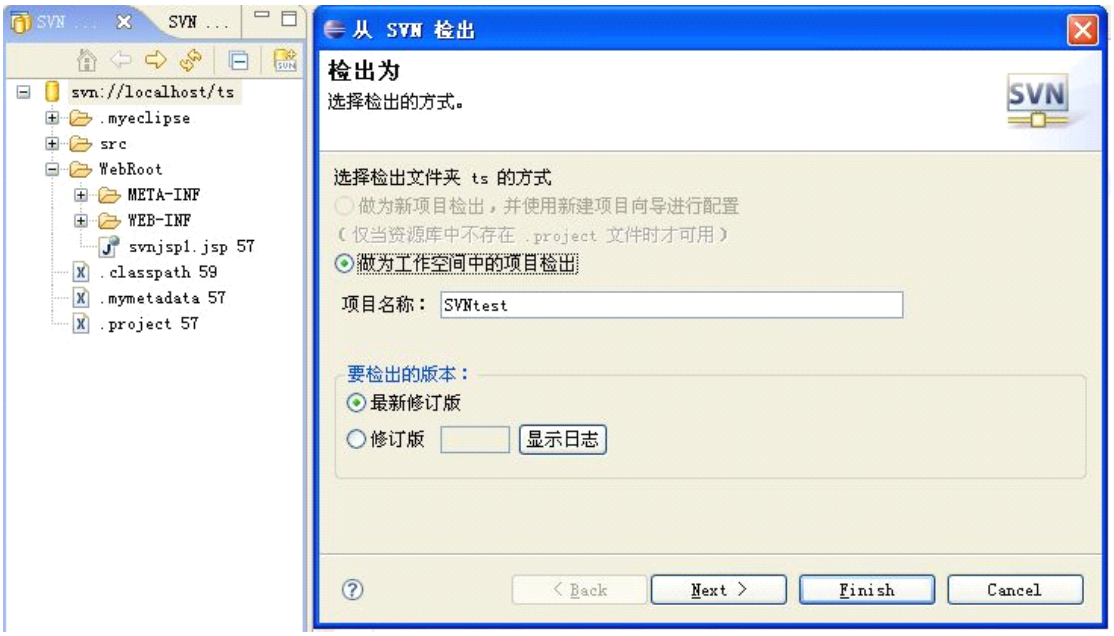




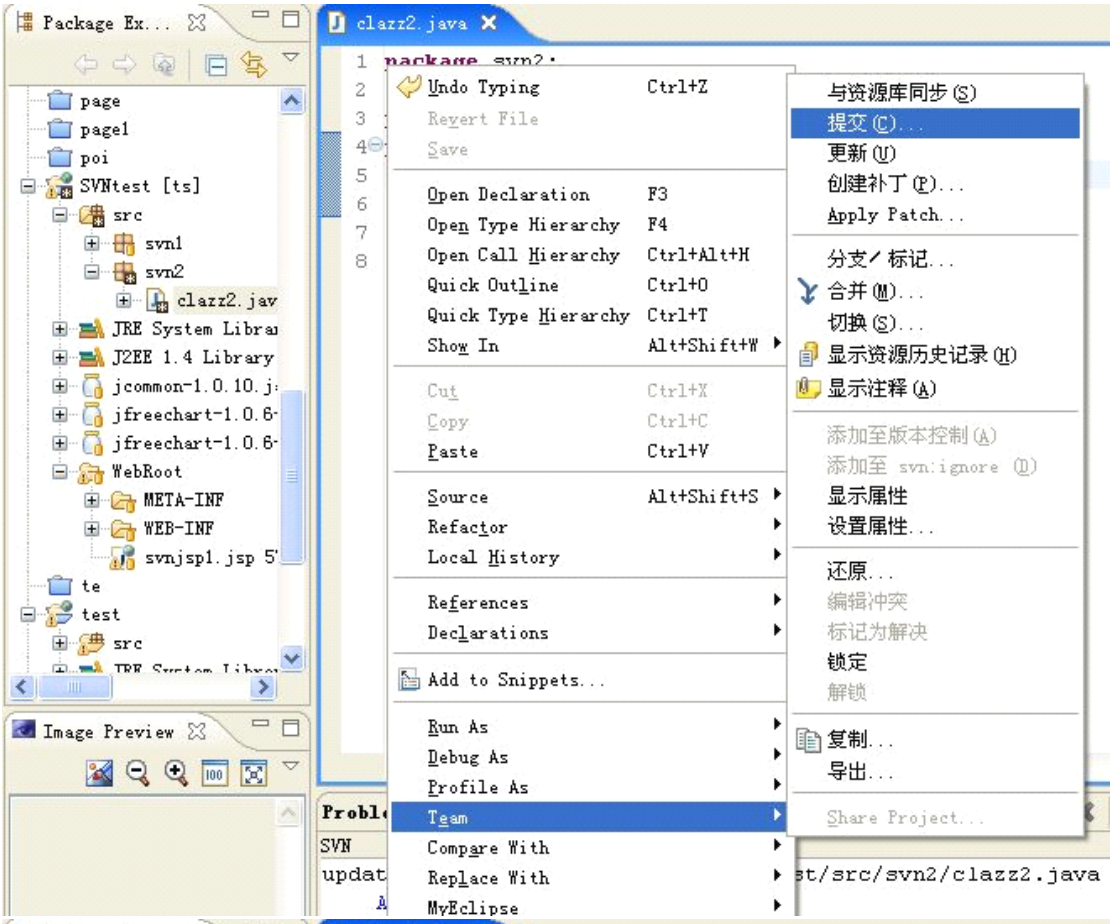
做好以上的准备后打开 Eclipse 编译器，点击编译器右上角的 Open Perspective 打开 SVN 资源库界面，新建一个资源库

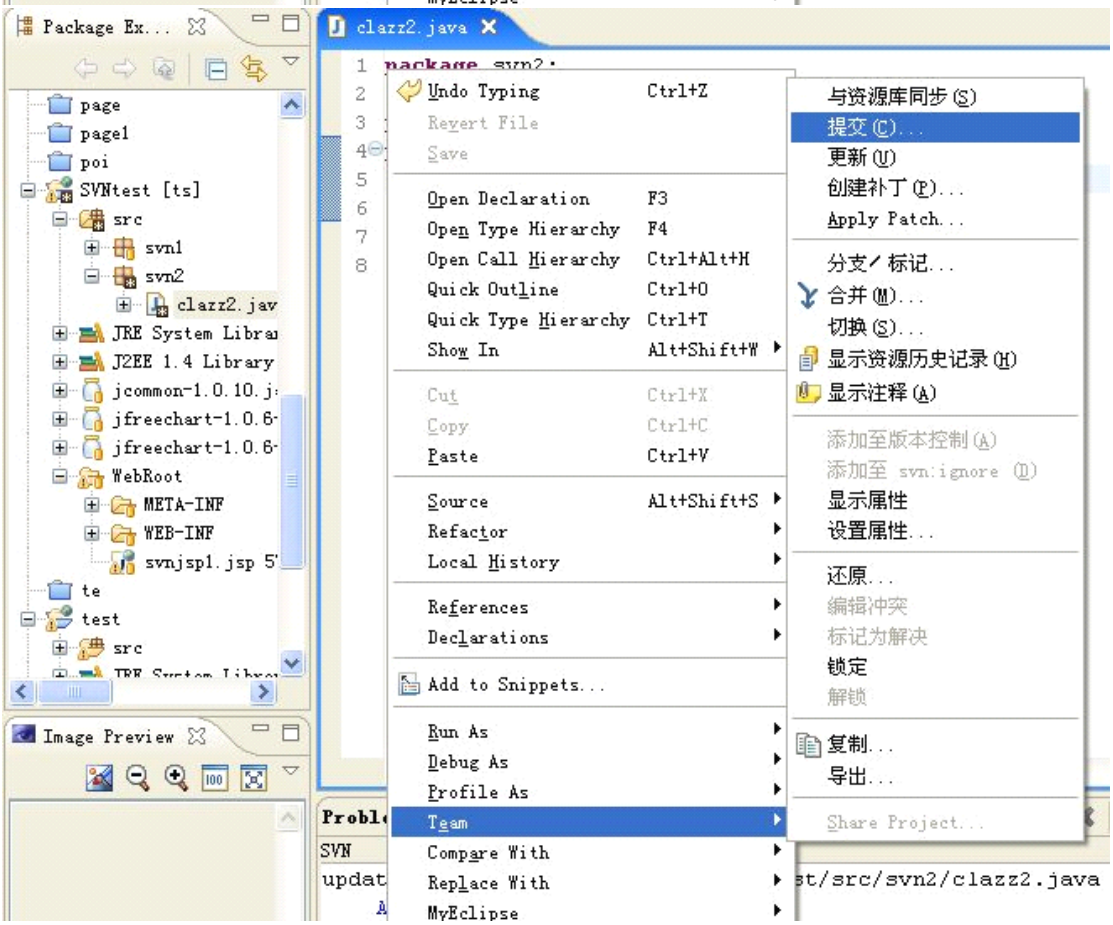


选择资源库的位置，这里我们就用刚才搭好的 svn://localhost/ts 作为工程目录，点击 Finish 后如果成功则会看到版本服务器中工程的树形结构了（可能需要用户密码验证）。在 svn://localhost/ts 根目录上点右键，选择“验出”（英文版的可能叫 Check Out），在弹出窗口中选择“作为空间的项目验出”，下边的项目名称随便叫，我这里还是用 SVNtest 作为项目名称。



一切没有问题了点击 Finish。之后它就会从服务器上把工程下到你本地了。好啦，来到Eclipse编辑栏会发现刚才下的工程文件都会带有SVN版本控制标记了，再去 Workspace 看看，也会发现下载的工程文件夹，并且也有控制标记。 而且里我们会发现在 SVN 控制下的文件多出来一个 Team 选单，打开一看原来就是一个客户端工具呀，提交，更新，同步，合并，切换……我们再熟悉不过了，自己试试吧 ^_^。


Package Ex...
clazz2.java
page
page1
poi
SVNtest [ts]
src
svn1
svn2
clazz2.jav
JRE System Libra
J2EE 1.4 Library
jcommon-1.0.10.j
jfreechart-1.0.6
jfreechart-1.0.6
WebRoot
META-INF
WEB-INF
svnjsp1.jsp 5
te
test
src
Image Preview
Undo Typing Ctrl+Z
Revert File
Save
Open Declaration F3
Open Type Hierarchy F4
Open Call Hierarchy Ctrl+Alt+H
Quick Outline Ctrl+O
Quick Type Hierarchy Ctrl+T
Show In Alt+Shift+W
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Source Alt+Shift+S
Refactor
Local History
References
Declarations
Add to Snippets...
Run As
Debug As
Profile As
Team
Compare With
Replace With
MyEclipse
与资源库同步(S)
提交(C)...
更新(U)
创建补丁(P)...
Apply Patch...
分支/标记...
合并(M)...
切换(S)...
显示资源历史记录(H)
显示注释(A)
添加至版本控制(A)
添加至 svn:ignore (D)
显示属性
设置属性...
还原...
编辑冲突
标记为解决
锁定
解锁
复制...
导出...
Share Project...
SVN
st/src/svn2/clazz2.java


Package Ex...
clazz2.java
page
page1
poi
SVNtest [ts]
src
svn1
svn2
clazz2.jav
JRE System Libra
J2EE 1.4 Library
jcommon-1.0.10.j
jfreechart-1.0.6
jfreechart-1.0.6
WebRoot
META-INF
WEB-INF
svnjsp1.jsp 5
te
test
src
Image Preview
Undo Typing Ctrl+Z
Revert File
Save
Open Declaration F3
Open Type Hierarchy F4
Open Call Hierarchy Ctrl+Alt+H
Quick Outline Ctrl+O
Quick Type Hierarchy Ctrl+T
Show In Alt+Shift+W
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Source Alt+Shift+S
Refactor
Local History
References
Declarations
Add to Snippets...
Run As
Debug As
Profile As
Team
Compare With
Replace With
MyEclipse
与资源库同步(S)
提交(C)...
更新(U)
创建补丁(P)...
Apply Patch...
分支/标记...
合并(M)...
切换(S)...
显示资源历史记录(H)
显示注释(A)
添加至版本控制(A)
添加至 svn:ignore (D)
显示属性
设置属性...
还原...
编辑冲突
标记为解决
锁定
解锁
复制...
导出...
Share Project...
SVN
st/src/svn2/clazz2.java

顺便说一下刚才为什么要删除 class 目录，起初我没有删掉它，更改提交没有问题，但在更新时总是报错：……Working copy not locked; this is probably a bug, please report svn:……，提示就是 class 目录下的.svn 有毛病，到 http://www.tigris.org 查找解决方法，有一段描述：

This message is coming straight out of the Subversion library, so technically it is Subversion asking you to report the problem to them. This error message is kind of their general error message when something really unexpected happens. In the case of Eclipse, the problem is almost always one specific thing. The problem is that your Eclipse build folder was versioned and added to your repository. What happens is that when Eclipse does a full build it will delete everything in this folder, including the ".svn" metadata folder. When Subversion cannot find this folder it issues the above error.
The solution is to delete this folder from your repository, which you can do from the SVN Repositories view. Then try deleting the folder from your working copy and perforiming an update. You might need to checkout your project again. Once you have a valid project again, be sure to add the build folder to the svn:ignore property of its parent folder so that the problem does not happen again.
If this is not your problem, then as best as you can try to figure out what might have led up to having this problem and report it on th Subversion users@subversion.tigris.org mailing list

大意：重新部署工程时会将文件夹 WEB-INF/classes 下的内容全删了，包括".svn"这个文件夹，从面导致 SVN 找不到这个文件夹的信息，于是报错。 解决的方法是在 SVN 仓库内将 classes 下的内容清空，然后在确保本地工程已完全提交的情况下，将本地工程包括文件删除，再从 SVN 仓库中重新取出。 导致此问题的原因善不明确，估计是往这个文件夹手动添加了文件。

安装方面就介绍到此吧，如果有什么问题在联系我吧。

相关资源
Subversion http://subversion.tigris.org/
TortoiseSVN http://tortoisesvn.net/downloads http://tortoisesvn.tigris.org/
Svn1ClickSetup http://svn1clicksetup.tigris.org/
Subclipse http://subclipse.tigris.org/
Subversion 中文站 http://www.subversion.org.cn/

四，SVN 的权限控制详细讲解

本章将详细介绍前一章所涉及的两个配置文件， svnserve.conf 和 authz.conf，通过对配置逐行的描述，来阐明其中的一些细节含义。除此之外的其他配置、安装等内容，不是本文重点，读者若有什么疑问，请参考后面“参考文献”中列出的一些文档。

这里首先要注意一点，任何配置文件的有效配置行，都 **不允许存在前置空格** ，否则程序可能会出错，给你一个 ``Option expected`` 的提示。也就是说，如果你直接从本文的纯文本格式中拷贝了相关的配置行过去，需要手动将前置的 4 个空格全部删除。当然了，如果你觉得一下子要删除好多行的同样数目的前置空格是一件苦差使，那么也许 UltraEdit 的“Column Mode”编辑模式，可以给你很大帮助。

svnserve.conf

``arm\conf\svnserve.conf`` 文件，是 svnserve.exe 这个服务器进程的配置文件，我们逐行解释如下。

首先，我们告诉 svnserve.exe，用户名与密码放在 passwd.conf 文件下。当然，你可以改成任意的有效文件名，比如默认的就是 passwd::

```
password-db = passwd.conf
```

接下来这两行的意思，是说只允许经过验证的用户，方可访问代码库。 那么哪些是“经过验证的”用户呢？噢，当然，就是前面说那些在 passwd.conf 文件里面持有用户名密码的家伙。这两行的等号后面，目前只允许 read write none 三种值，你如果想实现一些特殊的值，比如说“read-once”之类的，建议你自己动手改源代码，反正它也是自由软件::

```
anon-access = none
auth-access = write
```

接下来就是最关键的一句呢，它告诉 svnserve.exe，项目目录访问权限的相关配置是放在 authz.conf 文件里::

```
authz-db = authz.conf
```

当然，svn 1.3.2 引入本功能的时候，系统默认使用 authz 而不是 authz.conf 作为配置文件。不过可能由于鄙人是处女座的，据说有着强烈的完美主义情结，看着 svnserve.conf 有后缀而 passwd 和 authz 没有就是不爽，硬是要改了。

上述的 passwd.conf 和 authz.conf 两个文件也可以作为多个代码库共享使用，我们只要将它们放在公共目录下，比如说放在 ``D:\svn`` 目录下，然后在每个代码库的 svnserve.conf 文

件中，使用如下语句::

```
password-db = ..\..\passwd.conf
authz-db = ..\..\authz.conf
```

或者::

```
password-db = ../../passwd.conf
authz-db = ../../authz.conf
```

这样就可以让多个代码库共享同一个用户密码、目录控制配置文件，这在有些情况下是非常方便的。

authz.conf 之用户分组

``arm\conf\authz.conf`` 文件的配置段，可以分为两类， ``[group]`` 是一类，里面放置着所有用户分组信息。其余以 ``[arm:/]`` 开头的是另外一类，每一段就是对应着项目的一个目录，其目录相关权限，就在此段内设置。

首先，我们将人员分组管理，以便以后由于人员变动而需要重新设置权限时候，尽量少改动东西。我们一共设置了 5 个用户分组，分组名称统一采用 ``g_`` 前缀，以方便识别。当然了，分组成员之间采用逗号隔开::

```
[groups]
# 任何想要查看所有文档的非本部门人士
g_vip = morson

# 经理
g_manager = michael

# 北京办人员
g_beijing = scofield

# 上海办人员
g_shanghai = lincon

# 总部一般员工
g_headquarters = rory, linda

# 小秘，撰写文档
g_docs = linda
```

注意到没有， linda 这个帐号同时存在“总部”和“文档员”两个分组里面，这可不是我老眼昏花写错了，是因为 Subversion 允许我这样设置。它意味着，这个家伙所拥有的权限，将会比他的同事 rory 要多一些，这样的确很方便。具体多了哪些呢？请往下看！

authz.conf 之项目根目录

接着，我们对项目根目录做了限制，该目录只允许 arm 事业部的经理才能修改，其他人都只能眼巴巴的看着::

```
[arm:/]
@g_manager = rw
* = r
```

- ``[arm:/]`` 表示这个目录结构的相对根节点，或者说是 arm 项目的根目录。其中的 arm 字样，其实就是代码库的名称，即前面用 svnadmin create 命令创建出来的那个 arm。

- 这里的 ``@`` 表示接下来的是一个组名，不是用户名。因为目前 g_manager 组里面只有一个 michael，你当然也可以将 ``@g_manager = rw`` 这一行替换成 ``michael = rw`` ，而表达的意义完全一样。

- ``*`` 表示“除了上面提到的那些人之外的其余所有人”，也就是“除了部门经理外的其他所有人”，当然也包括总经理那个怪老头

- ``* = r`` 则表示“那些人只能读，不能写”

authz.conf 之项目子目录

然后，我们要给总部人员开放日志目录的读写权限::

```
[arm:/diary/headquarters]
@g_manager = rw
@g_headquarters = rw
@g_vip = r
* =
```

这个子目录的设置有些特色，因为从需求分析中我们知道，这个子目录的权限范围要比其父目录小，它不允许除指定了的之外其他任何人访问。在这段设置中，我们需要注意以下几点：

- 我敢打赌，设计 svn 的家伙们，大部分都是在类 unix 平台下工作，所以他们总喜欢使用 ``/`` 来标识子目录，而完全忽视在 MS Windows 下是用 ``\`` 来做同样的事情。所以这儿，

为了表示 ``diary\headquarters`` 这个目录，我们必须使用 ``[arm:/diary/headquarters]`` 这样的格式。当然如果你一定要用 ``\`` ，那么唯一的结果就是，Subversion 会将你的这部分设置置之不理，全当没看到。

- 这里最后一行的 ``* =`` 表示，除了经理、总部人员、特别人士之外，任何人都被禁止访问本目录。这一行是否可以省略呢？不行，因为 **权限具备继承性** ，子目录会自动拥有父目录的权限。若没有这一行，则所有帐号都可以读取 ``/diary/headquarters`` 目录下的文件。因为虽然我们并没有设置这个目录的父目录权限，可是默认的规则使得 ``/diary`` 目录的权限与根目录完全一样，从而让其余帐号获得对 ``/diary/headquarters`` 目录的 r 权限。所以简单来说， ``* =`` 这一句的目的，就是割断权限继承性，使得管理员可以定制某个目录及其子目录的权限，从而完全避开其父目录权限设置的影响。

- 之所以这儿需要将 ``@g_vip = r`` 一句加上，就是因为存在上述这个解释。如果说你没有明确地给总经理授予读的权力，则他会和其他人一样，被 ``* =`` 给排除在外。

- 如果众位看官中间，有谁玩过防火墙配置的话，可能会感觉上述的配置很熟悉。不过这里有一点与防火墙配置不一样，那就是各个配置行之间，没有 **先后顺序** 一说。也就是说，如果我将本段配置的 ``* =`` 这一行挪到最前面，完全不影响整个配置的最终效果。

接下来我们看看这一段::

```
[arm:/ref]
@g_manager = rw
@g_docs = rw
* = r
```

这里的主要看点，就是 g_docs 组里面包含了一个 linda 帐号，她也同时在 g_headquarters 组里面出现，这就意味着， linda 将具备对 ``/ref`` 和 ``diary\headquarters`` 两个目录的读写权限。

authz.conf 之目录表示法

在前面的描述中，我们都采用 ``[repos:/some/dir]`` 这样的格式来表示项目的某个目录，比如上一小节中的 ``[arm:/diary/headquarters]`` 。而实际上， Subversion 允许你采用 ```[/some/dir]`` 这样的格式，即不指定代码库的方式来表示目录，此时的目录就匹配所有项目。

对于使用 svnserve 的用户来说，只有当 svnserve 运行的时候使用了 ``-r`` 参数，并且让多个代码库共享同一个目录权限文件（即 authz.conf 或 authz）时，不指明代码库名称才有可能惹麻烦。一般情况下，我们对每个代码库都会独立使用配置文件，毕竟每个项目的目录结构，都有很大不同，混在一起意义不大。因此一般来说，为简洁起见，都可以不指明代码库名称。本文全都指明了代码库名称，主要是为了将来扩展成同一个配置文件，以方便配合 Apache 服务器。

对于使用 Apache 的用户来说，它们二者可有着很大的不同，因为此时往往习惯于使用一个公共的目录权限配置文件。如果你使用了 SVNParentPath 指令，则指定版本库的名字是很重要的，因为假若你使用后者，那么 ``[/some/dir]`` 部分就会与所有代码库项目的 ``[/some/dir]`` 目录匹配。如果你使用 SVNPath 指令，则这两种表示方式就没有什么区别了，毕竟只有一个版本库。

authz.conf 的其他注意点
---

1. 父目录的 ``r`` 权限，对子目录 ``w`` 权限的影响

把这个问题专门提出来，是因为在 1.3.1 及其以前的版本里面，有个 bug，即某个帐号为了对某个子目录具备写权限，则必须对其父目录具备读权限。因此现在使用了 1.3.2 及其更高的版本，就方便了那些想在一个代码库存放多个相互独立的项目的管理员，来分配权限了。比如说央舜公司建立一个大的代码库用于存放所有员工日志，叫做 diary，而 arm 事业部只是其中一个部门，则可以这样做::

```
[diary:/]
@g_chief_manager = rw

[diary:/arm]
@g_arm_manager = rw
@g_arm = r
```

这样，对于所有 arm 事业部的人员来说，就可以将 svn://192.168.0.1/diary/arm 这个 URL 当作根目录来进行日常操作，而完全不管它其实只是一个子目录，并且当有少数好奇心比较强的人想试着 checkout 一下 svn://192.168.0.1/diary 的时候，马上就会得到一个警告“Access denied”，哇，太酷了。

2. 默认权限

如果说我对某个目录不设置任何权限，会怎样？马上动手做个试验，将::

```
[diary:/]
@g_chief_manager = rw
```

改成::

```
[diary:/]
# @g_chief_manager = rw
```

这样就相当于什么都没有设置。在我的 svn 1.3.2 版本上，此时是禁止任何访问。也就是说，如果你想要让某人访问某目录，你一定要显式指明这一点。这个策略，看起来与防火墙的策略是一致的。

3. 只读权限带来的一个小副作用

若设置了::

```
[arm:/diary]
* = r
```

则 Subversion 会认为，任何人都不允许改动 diary 目录，包括删除、 **改名** ，和 **新增** 。

也就是说，如果你在项目初期创建目录时候，一不小心写错目录名称，比如因拼写错误写成 dairy，以后除非你改动 authz.conf 里面的这行设置，否则无法利用 svn mv 命令将错误的目录更正。

4. anon-access 属性对目录权限的影响

你想将你的代码库开放给所有人访问，于是你就开放了匿名访问权限，在 svnserve.conf 文件中添加一行： ``anon-access=read`` 。可是对于部分目录，你又不希望别人看到，于是针对那些特别目录，你在 authz.conf 里面进行配置，添加了授权访问的人，并添加了 ``* =`` 标记。你认为一切 OK 了，可是你缺发现，那个特别目录却无法访问了，总是提示 ``Not authorized to open root of edit operation`` 或者 ``未授权打开根进行编辑操作`` 。你再三检查你配置的用户名与密码，确认一切正确，还是无法解决问题。

原来，Subversion 有个小 bug ，当 ``anon-access=read`` 并且某个目录有被设置上 ``* =`` 标记，则会出现上述问题。这个 bug 在当前最新版本上（v1.4）还存在，也许在下一版本内可以被改正吧。

解决的办法是，在 svnserve.conf 中，将 anon-access 设置成 none 。

改进
====

对中文目录的支持

上午上班的时候，Morson 来到 Michael 的桌子前面，说道：“你是否可以将我们的北京办、上海办目录，改成用中文的，看着那些拼音我觉得很难受？”Michael 心想，还好这两天刚了解了一些与 unicode 编码相关的知识，于是微笑地回答：“当然可以，你明天下午就可以看到中文目录名称了。”

1. 使用 svn mv 指令，将原来的一些目录改名并 commit 入代码库，改名后的目录结构如下::

```
arm
├─工作日志
│  ├─总部人员
│  ├─北京办
│  └─上海办
├─公司公共文件参考目录
└─临时文件存放处
```

2. 修改代码库的 authz.conf 文件，将相应目录逐一改名

3. UTF-8 格式的 authz.conf 文件，以及 BOM

将配置文件转换成 UTF-8 格式之后，Subversion 就能够正确识别中文字符了。但是这里需要注意一点，即必须保证 UTF-8 文件不包含 BOM 。BOM 是 Byte Order Mark 的缩写，指 UNICODE 文件头部用于指明高低字节排列顺序的几个字符，通常是 ``FF FE`` ，而将之用 UTF-8 编码之后，就是 ``EF BB BF`` 。由于 UTF-8 文件本身不存在字节序问题，所以对 UTF-16 等编码方式有重大意义的 BOM，对于 UTF-8 来说，只有一个作用——表明这个文件是 UTF-8 格式。由于 BOM 会给文本处理带来很多难题，所以现在很多软件都要求使用不带 BOM 的 UTF-8 文件，特别是一些处理文本的软件，如 PHP、 UNIX 脚本文件等，svn 也是如此。

目前常用的一些文本编辑工具中，MS Windows 自带的“记事本”里面，“另存为”菜单保存出来的 UTF-8 格式文件，会自动带上 BOM 。新版本 UltraEdit 提供了选项，允许用户选择是否需要 BOM，而老版本的不会添加 BOM。请各位查看一下自己常用的编辑器的说明文件，看看它是否支持这个功能。

对于已经存在 BOM 的 UTF-8 文件，比如说就是微软“记事本”弄出来的，我们可以利用 UltraEdit 来将 BOM 去掉。方法是，首先利用“UTF-8 TO ASCII”菜单将文件转换成本地编码，通常是 GB2312 码，然后再使用“ASCII TO UTF-8(UNICODE Editing)”来转换到 UTF-8 即可。当然，这么操作之前，你肯定得先保证，你的 UltraEdit 保存出来的 UTF-8 文件的确是不带 BOM 的。

Subversion 为什么讨厌 BOM 呢？我不知道，毕竟我也只是一个普通用户，不是开发人员。如果你感兴趣，并且英文够好的话，不妨参考一下这个讨论： http://subversion.tigris.org/ser ... ers&msgNo=51334

安装 ApacheSVN 服务器教程（转载）

Subversion 的设计包括一个抽象的网络层，这意味着版本库可以通过各种服务器进程访问。理论上讲，Subversion 可以使用无限数量的网络协议实现，目前实践中存在着两种服务器。

? SVNServer：svnserve 是一个小的（也叫轻型的）、独立服务器，使用自己定义的协议和客户端。（作者注：以下称这种服务器为“svnserver 服务器”，上面的安装配置就是安装 svnserver 服务器。）

? ApacheSVN：Apache 是最流行的 web 服务器，通过使用 mod_dav_svn 模块，Apache 可以访问版本库，并且可以使客户端使用 HTTP 的扩展协议 WebDAV/DeltaV 进行访问。（作者注：以下称这种服务器为“ApacheSVN 服务器”）

通过 Http 协议访问版本库是 Subversion 的亮点之一。ApacheSVN 服务器具备了许多 svnserve 服务器没有的特性，使用上更加灵活，但是有一点难于配置，灵活通常会带来复杂性。

由于 Subversion 需要版本化的控制，因此标准的 Http 协议不能满足需求。要让 Apache 与 Subversion 协同工作，需要使用 WebDAV（Web-based Distributed Authoring and Versioning：）Web 分布式创作和版本控制）。WebDAV 是 HTTP 1.1 的扩展，关于 WebDAV 的规范和工作原理，可以参考 IETF RFC 2518 （http://www.ietf.org/rfc/rfc2518.txt）。

一、必备条件

为了让你的版本库使用 HTTP 网络，你必需具备以下几个条件：

1、配置好 httpd 2.2.x，并且使用 mod_dav 启动。
2、为 mod_dav 安装 mod_dav_svn 插件。
3、配置你的 httpd.conf，使 http 协议能访问版本库。

下面以我的配置过程详细讲解。

环境：
OS：Windows XP SP2
Web：Apache 2.2.6
SVN：svn-win32-1.4.6

二、安装

1、安装 Apache
具体安装方法见：《Windows 下安装 Apache 2.2.x》

2、安装 Subversion

将下载下来的 svn-win32-1.4.6.zip 直接解压即可，比如我解压到 e:\subversion 。
从 Subversion 安装目录的 bin 子目录将 intl3_svn.dll、libdb44.dll、mod_authz_svn.so、mod_dav_svn.so 拷贝到 Apache 的模块目录（Apache 安装目录的 modules 文件夹）。

三、基本的 Apache 配置

修改 Apache 的配置文件 httpd.conf ，使用 LoadModule 来加载 mod_dav_svn 模块。

将
#LoadModule dav_module modules/mod_dav.so

改成：
LoadModule dav_module modules/mod_dav.so

即去掉前面的“#”号。

添加：
LoadModule dav_svn_module modules/mod_dav_svn.so

一定确定它在 mod_dav 之后。

现在你已经设置了 Apache 和 Subversion，但是 Apache 不知道如何处理 Subversion 客户端，例如 TortoiseSVN。为了让 Apache 知道哪个目录是用来作为 Subversion 版本库，你需要使用编辑器（例如记事本）编辑 Apache 的配置文件。

在配置文件最后添加如下几行：

```
<Location / repository>
  DAV svn
  SVNPath e:/svn/repos1
</Location>
```

这个配置告诉 Apache 首先需要启用 dav_module，然后加载 dav_svn_module 。版本库对外的 URL 是：http://服务器 IP/repository ，所有的 Subversion 版本库在物理上位于 e:/svn/repos1 。
配置完毕后重新启动 Apache，打开浏览器，输入 http://服务器 IP/repository 将会看到如下画面：

这表示 Apache 的 dav_svn 模块已经可以正常工作了。用户可以使用任何一种

Subversion 的客户端通过 Http 协议访问你的版本库。

如果想要指定多个版本库，可以用多个 Location 标签，也可以使用 SVNParentPath 代替 SVNPath,例如在 e:\svn 下有多个版本库 repos1,repos2 等等，用如下方式指定：

```
<Location /repository>
        DAV svn
        SVNParentPath e:/svn
</Location>
```

“SVNParentPath e:/svn ” 表示 e:\svn 下的每个子目录都是一个版本库。可以通过 http://服务器 IP/repository/repos1，http://服务器 IP/repository/repos2 来访问。

现在你的版本库任何人都可以访问，并且有完全的写操作权限。也就是说任何人都可以匿名读取，修改，提交，以及删除版本库中的内容(注：这时不需要配置 E:\svn\repos\conf\svnserve.conf 文件，并且也不需要启动 E:\subversion\bin\svnserve.exe。因为提交是通过 Apache 的 dav 模块处理的，而不是由 svnservice 处理。)。我们用 TortoiseSVN 客户端验证即知。
显然大部分场合这是不符合需求的。那么如何进行权限设置呢，Apache 提供了基本的权限设置：

四、认证选项
1、基本 HTTP 认证
最简单的客户端认证方式是通过 HTTP 基本认证机制，简单的使用用户名和密码来验证一个用户的身份。Apache 提供了一个 htpasswd 工具来管理一个用户文件，这个文件包含用户名和加密后的密码，这些就是你希望赋予 Subversion 特别权限的用户。htpasswd 可以在 Apache 的 bin 安装目录下找到。具体使用方法如下：

创建用户文件：
```
htpasswd -c /etc/svn/passwordfile username
```

添加新用户（-m 表示以 MD5 加密密码）：
```
htpasswd [-m] /etc/svn/passwordfile Newusername
```

更改用户密码：
```
htpasswd [-m] /etc/svn/passwordfile username
```

删除用户（要用大写的 D ）：
```
htpasswd  - D /etc/svn/passwordfile username
```

接下来修改 httpd.conf，在 Location 标签中加入如下内容：

```
AuthType Basic
AuthName "svn repos"
AuthUserFile E:/usr/Apache2.2/bin/passwd
Require valid-user
```

说明：
AuthType Basic：启用基本的验证，比如用户名/密码对。
AuthName "svn repos"：当一个认证对话框弹出时，出现在认证对话框中的信息。（最好用英文，TortoiseSVN 不支持中文，安装语言包除外。）
AuthUserFile E:/usr/Apache2.2/bin/passwd：指定 E:\usr\Apache2.2\bin\passwd 为用户文件，用来验证用户的用户名及密码。
Require valid-user：限定用户只有输入正确的用户名及密码后才能访问这个路径

重新启动 Apache ，打开浏览器访问版本库。Apache 会提示你输入用户名和密码来认证登陆了，现在只有 passwd 文件中设定的用户才可以访问版本库。也可以配置只有特定用户可以访问，替换上述 "Require valid-user" 为 "Require user tony robert" 将只有用户文件中的 tony 和 robert 可以访问该版本库。

有的时候也许不需要这样严格的访问控制，例如大多数开源项目允许匿名的读取操作，而只有认证用户才允许写操作。为了实现更为细致的权限认证，可以使用 Limit 和 LimitExcept 标签。例如：

```
<LimitExcept GET PROPFIND OPTIONS REPORT>
  require valid-user
</LimitExcept>
```

以上配置将使匿名用户有读取权限，而限制只有 passwd 中配置的用户可以使用写操作。
如果这还不能满足你的要求，你希望精确的控制版本库目录访问，可以使用 Apache 的 mod_authz_svn 模块对每个目录进行认证操作。

2、用 mod_authz_svn 进行目录访问控制
首先需要让 Apache 将 mod_authz_svn 模块加载进来。在 Subversion 的安装目录中找到 mod_auth_svn 模块，将其拷贝到 Apache 安装目录的 modules 子目录下。修改 httpd.conf 文件，添加：

```
LoadModule authz_svn_module modules/mod_authz_svn.so
```

现在可以在 Location 标签中使用 authz 的功能了。一个基本的 authz 配置如下：

```
<Location /repository>
  DAV svn
  SVNParentPath e:/svn

  # our access control policy
  AuthzSVNAccessFile E:/usr/Apache2.2/bin/accesspolicy.conf

  # try anonymous access first, resort to real
  # authentication if necessary.
  Satisfy Any
  Require valid-user

  # how to authenticate a user
  AuthType Basic
  AuthName "Subversion repository"
  AuthUserFile E:/usr/Apache2.2/bin/passwd
</Location>
```

AuthzSVNAccessFile 指向的是 authz 的策略文件，详细的权限控制可以在这个策略文件中指定。访问文件 accesspolicy.conf 的语法与 svnserve.conf 和 Apache 的配置文件非常相似，以（#）开头的行会被忽略；在它的简单形式里，每一小节命名一个版本库和一个里面的路径；认证用户名是在每个小节中的选项名；每个选项的值描述了用户访问版本库的级别：r（只读）或者 rw（读写），如果用户没有提到或者值留空，访问是不允许的； * 表示所有用户，用它控制匿名用户的访问权限；@符号区分组和用户。如：

```
[groups]
committers = paulex, richard
developers = jimmy, michel, spark, sean

[/]
* = r
@committers = rw

[/branches/dev]
@developers = rw

[/tags]
tony = rw
[/private]
* =
@committers= r
```

使用 SVNParentPath 代替 SVNPath 来指定多个版本库的父目录时，其中所有的

版本库都将按照这个策略文件配置。例如上例中 tony 将对所有版本库里的 /tags 目录具有读写权限。如果要对具体每个版本库配置，用如下的语法：

```
[groups]
project1_committers = paulex, richard
project2_committers = jimmy, michel, spark, tony, Robert

[repos1:/]
* = r
@ project1_committer = rw

[repos2:/]
* = r
@ project2_committer = rw
```

这样 repos1 的 project1_committer 组只能对 repos1 版本库下的文件具有写权限而不能修改版本库 repos2 ，同样 repos2 的 project2_commiter 组也不能修改 repos1 版本库的文件。

FAQ：

1、路径或权限不足时将出现错误信息提示：

```
http://localhost （路径不对）
Error * PROPFIND request failed on '/' PROPFIND of '/': 200 OK
(http://localhost)

http://localhost/svn （权限不足）
Error * PROPFIND request failed on '/svn' PROPFIND of '/svn': 403
Forbidden (http://localhost)

http://localhost/svn/repos （正常显示）

http://localhost/repos （权限不允许）
Error * PROPFIND request failed on '/repos' PROPFIND of '/repos': 405
Method Not Allowed (http://localhost)
```

2、不启动 E:\subversion\bin\svnserve.exe ，但启动了 ApacheSVN ，访问（tortoiseSVN -> Repo - browser）或提交（SVN Commit）情形如下：

现象：svn://localhost/svn/repos 不能访问或提交，提示：Error * Can't connect to host 'localhost': 由于目标机器积极拒绝，无法连接。 但 file:///e:/svn/repos 和 http://localhost/svn/repos 可以访问或提交。

原因：svn:// 是独立服务器 svnserver 自己的协议。file:/// 是本地访问，即服务器端和客户端在一个机器上。

摘要：Subversion 版本控制使用中的常见问题及解决方法。

常见问题及解决方法

1、路径或权限不足时将出现错误信息提示：

http://localhost (路径不对)

Error * PROPFIND request failed on '/' PROPFIND of '/': 200 OK (http://localhost)

http://localhost/svn (权限不足)

Error * PROPFIND request failed on '/svn' PROPFIND of '/svn': 403 Forbidden (http://localhost)

http://localhost/svn/repos (正常显示)

http://localhost/repos (权限不允许)

Error * PROPFIND request failed on '/repos' PROPFIND of '/repos': 405 Method Not Allowed (http://localhost)

解决办法是填写正确的路径或给予适当的权限。

2、不启动 E:\subversion\bin\svnserve.exe ，但启动了 ApacheSVN ，访问(tortoiseSVN -> Repo - browser)或提交(SVN Commit)情形如下：

现象：svn://localhost/svn/repos 不能访问或提交，提示：Error * Can't connect to host 'localhost': 由于目标机器积极拒绝，无法连接。 但 file:///e:/svn/repos 和 http://localhost/svn/repos 可以访问或提交。

原因：svn:// 是独立服务器 svnserver 自己的协议。file:/// 是本地访问，即服务器端和客户端在一个机器上。

解决方法：使用 http 方式访问。

3、设置 SVNListParentPath 后浏览出现 Forbidden

配置 svn 服务器为“多库”后(“单库”、“多库”的概念见 http://bbs.iusesvn.com/thread-157-1-1.html)，我们想让浏览器显示父目录列表，这样，访问者可以直观的看到在版本库根目录下有哪些版本库。配置时加入“SVNListParentPath on”，比如我的配置文件：

```
DAV svn
SVNListParentPath on
#SVNPath e:/svn/repos1
SVNParentPath e:/svn

# 访问控制策略
AuthzSVNAccessFile E:/usr/Apache2.2/bin/accesspolicy.conf

# 首先采取匿名，如有必要采取认证
Satisfy Any
Require valid-user

# 授权类型
AuthType Basic
# 认证时的提示信息，出现在对话框中
AuthName "UserFile Auth"
# 用户文件，存储帐号
AuthUserFile E:/usr/Apache2.2/bin/passwd.conf
```

这时，如果我们在浏览器中输入“http://svn 服务器 IP/repos/repos1”(假设存在“repos1”版本库)，那么我们可以自己看到“repos”版本库的列表。但是如果我们想查看版本库的根目录，看看有哪些版本库，输入“http://svn 服务器 IP/repos”，回车。却出现：

```
Forbidden

You don't have permission to access /repos on this server.
```

查看访问控制策略文件“accesspolicy.conf”，噢，原来没有给根目录加权限。添加权限，整个配置看起来如下：

```
[groups]

admin = indian

test = test

[/]

* = r

[repos1:/]

* = r

@admin = rw

[repos2:/]

@admin = rw

@test = r

* =
```

再次输入“http://svn 服务器 IP/repos”，回车，还是出现禁止访问的信息。问题出在“<Location /repos>”，把它改成“<Location /repos/>”，即在后面加个斜线“/”，问题得到解决。