**实验五：CVS（SubVersion）软件版本数据库的安装**

要求：

1.每个人独立完成版本数据库的安装

2.设置软件开发小组中个人的权限

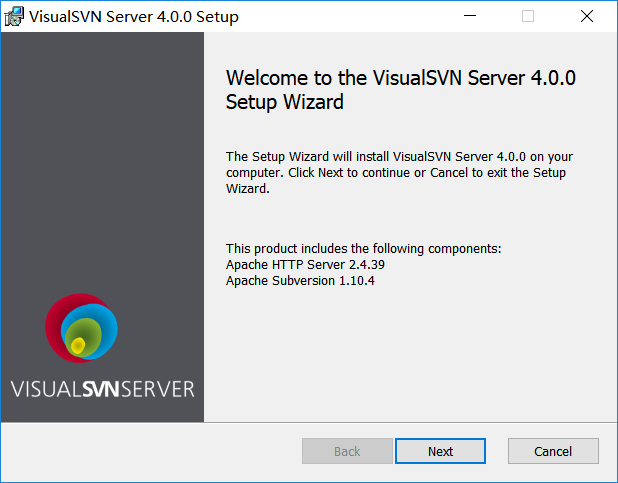
3.熟读软件使用手册

4.写下安装过程和体会

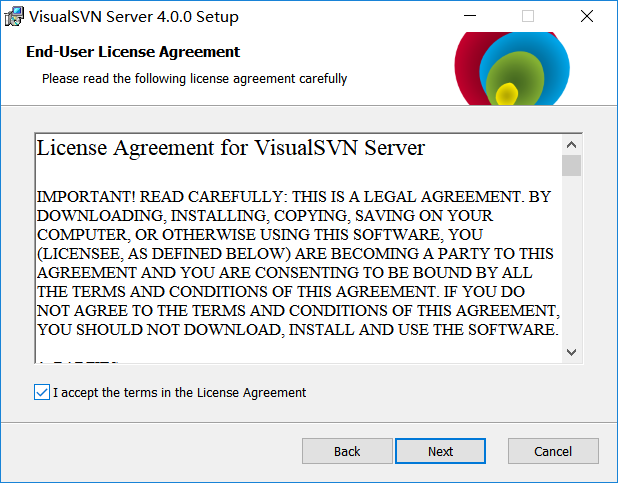
5.根据不同的权限至少熟悉20条以上指令，写下这下指令的使用方法以及体会。

**一、安装过程和体会**

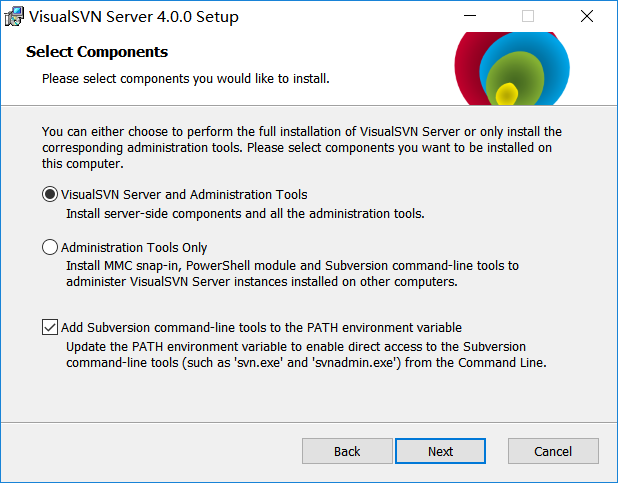
下载VisualSVN server,下载完成后双击安装，如下图：



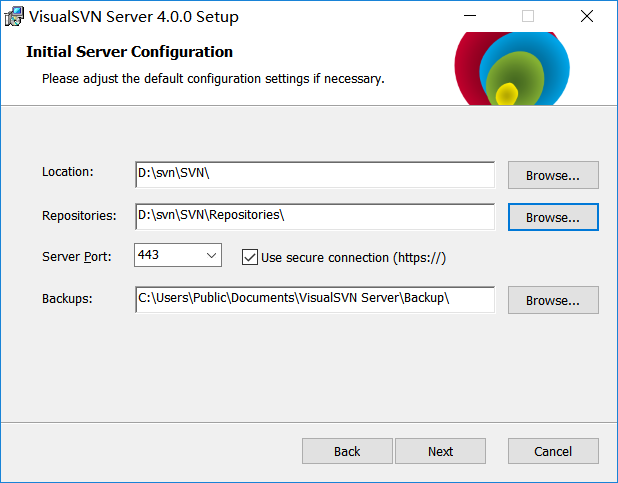
点击Next下一步，如下：



然后再点击Next项，下一步，如下：

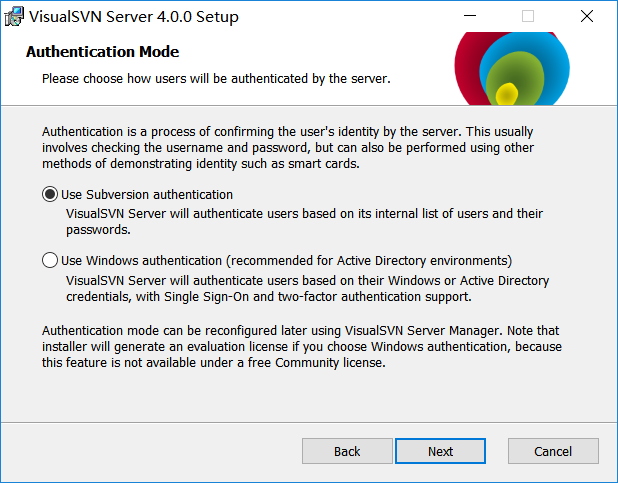


点击【Next】 如下：

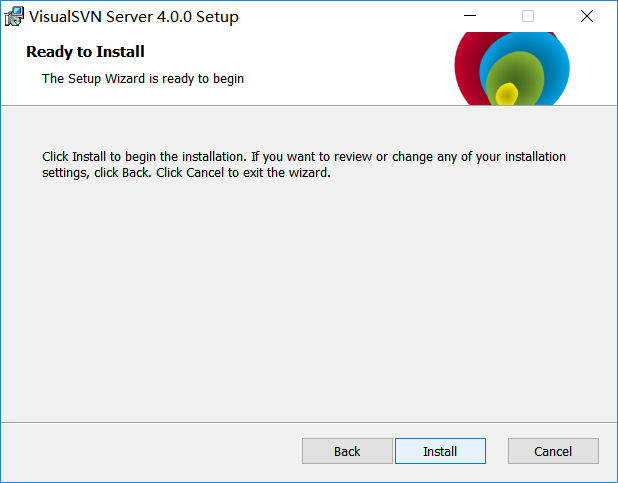


Location是指VisualSVN Server的安装目录,Repositorys是指定你的版本库目录.Server Port指定一个端口,Use secure connection勾山表示使用安全连接,

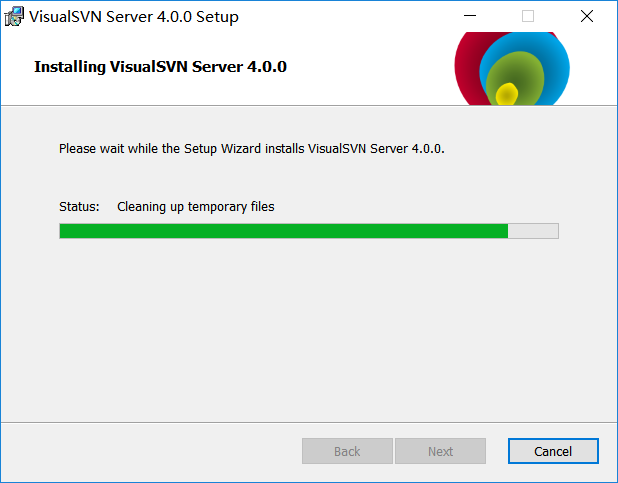
点击Next,进入下一步,如下图:



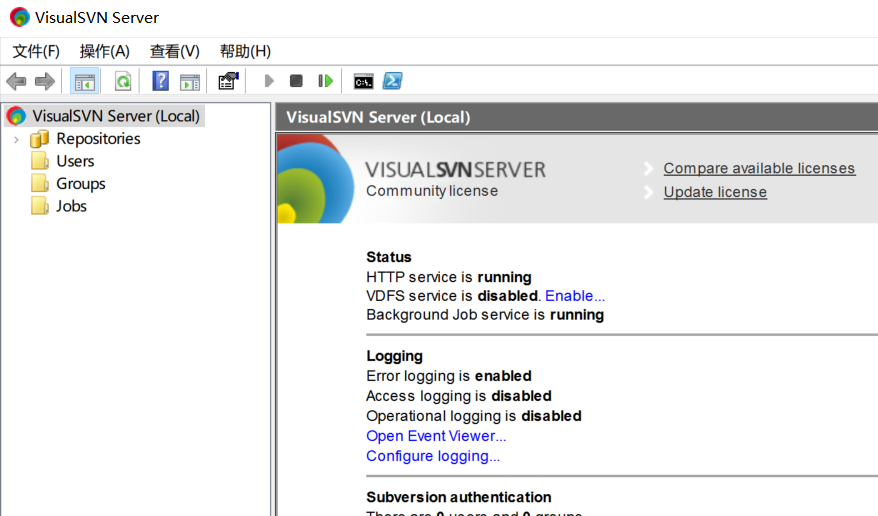
再点击【Install】,进入如下安装图：



等待安装完成后，点击【next】，进入下一步：如下图

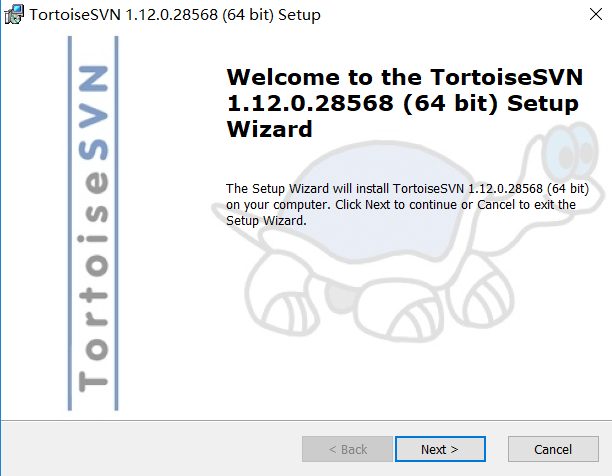


点击【Finish】即可完成安装。安装完成后,启动VisualSVN Server Manager,如图:



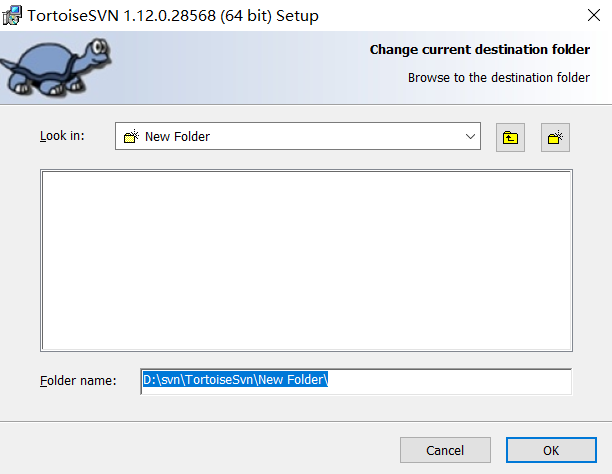
# 接下来进行TortoiseSVN的安装：

双击运行程序

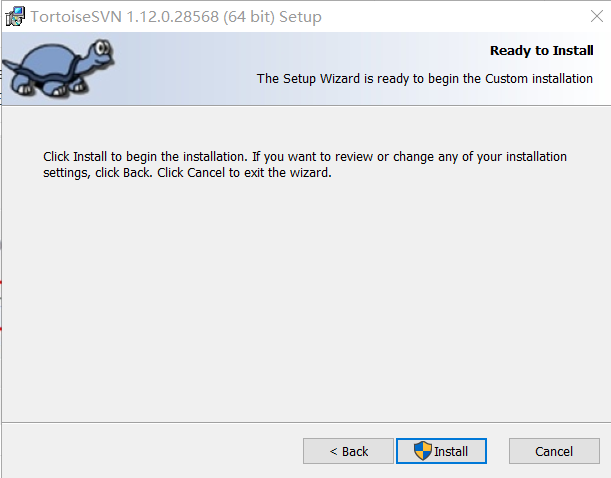


点击next进入下一步骤

点击OK，进入下一步骤



点击Install进入安装，然后等待完成安装就行了。

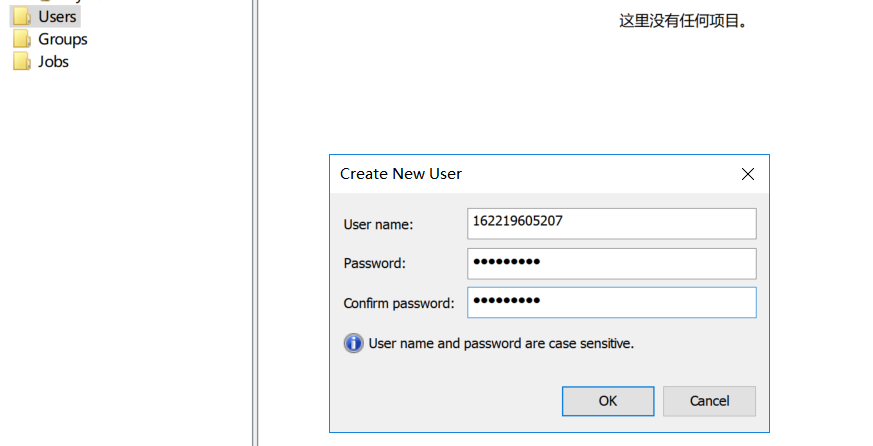


体会：本次安装没有什么难的地方，一直next就可以了，直到安装完成。

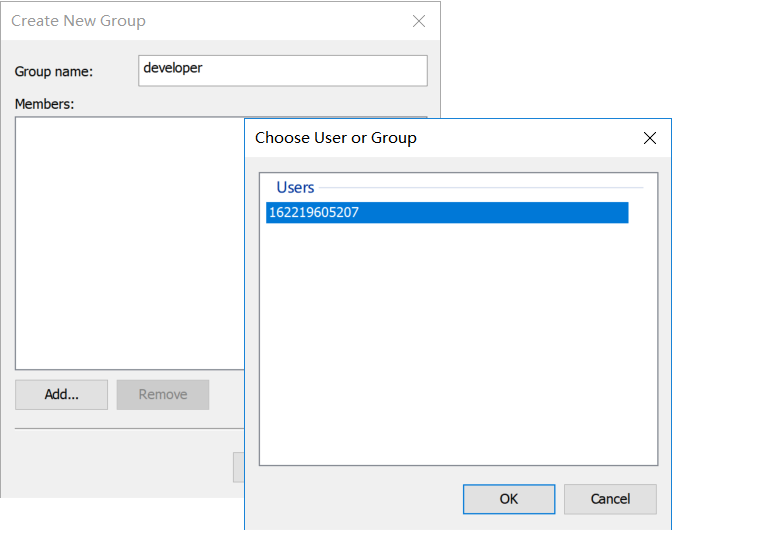
1. **创建个人权限**

建立用户并分配权限如下步骤，如图：

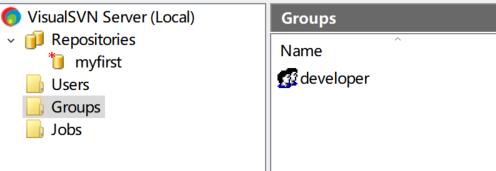
在VisualSVN Server Manager窗口的左侧右键单击用户组,选择Create User或者新建->User,如图:



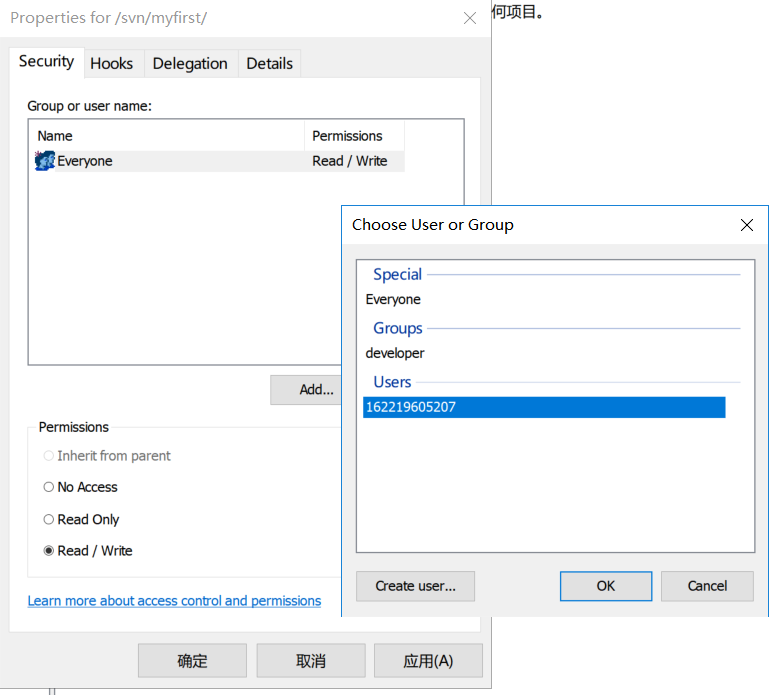
点击OK后进入下一步骤建立分组和建立用户差不多如图：



点击OK，进入下图：



分配权限前要建立版本库，我建立了myfirst的版本库，然后点击它，再点repositories如图：



点击确定，就完成了用户个人权限的分配。

**三、指令的使用说明和体会**

**1、**将文件checkout到本地文件夹

svn checkout path（path是server上的文件夹）

比如：svn checkout svn://192.168.1.1/pro/domain

简写：svn co

**2、**往版本号库中加入新的文件

svn add file

比如：svn add test.php(加入test.php)

svn add \*.php(加入当前文件夹下全部的php文件)

**3、**将修改的文件提交到版本号库

svn commit -m “LogMessage“ [-N] [--no-unlock] PATH(假设选择了保持锁，就使用–no-unlock开关)

比如：svn commit -m “add test file for my test“ test.php

简写：svn ci

**4、**加锁/解锁

svn lock -m “LockMessage“ [--force] PATH

比如：svn lock -m “lock test file“ test.php

svn unlock PATH

**5、**更新到某个版本号

svn update -r m path

比如：

svn update假设后面没有文件夹，默认将当前文件夹以及子文件夹下的全部文件都更新到最新版本号。

svn update -r 200 test.php(将版本号库中的文件test.php还原到版本号200)

svn update test.php(更新，于版本号库同步。假设在提交的时候提示过期的话，是由于冲突，须要先update，改动文件，然后清除svn resolved，最后再提交commit)

简写：svn up

**6、**查看文件或者文件夹状态

1）svn status path（文件夹下的文件和子文件夹的状态，正常状态不显示）

【?：不在svn的控制中；M：内容被改动；C：发生冲突；A：预定增加到版本号库；K：被锁定】

2）svn status -v path(显示文件和子文件夹状态)

第一列保持同样，第二列显示工作版本，第三和第四列显示最后一次改动的版本和改动人。

注：svn status、svn diff和 svn revert这三条命令在没有网络的情况下也能够运行的，原因是svn在本地的.svn中保留了本地版本号的原始拷贝。

简写：svn st

**7、**删除文件

svn delete path -m “delete test fle“

比如：svn delete svn://192.168.1.1/pro/domain/test.php -m “delete test file”

或者直接svn delete test.php 然后再svn ci -m ‘delete test file‘，推荐使用这样的

简写：svn (del, remove, rm)

**8、**查看日志

svn log path

比如：svn log test.php 显示这个文件的全部改动记录，及其版本的变化

**9、**查看文件具体信息

svn info path

比如：svn info test.php

**10、**比較差异

svn diff path(将改动的文件与基础版本号比較)

比如：svn diff test.php

svn diff -r m:n path(对版本号m和版本号n比較差异)

比如：svn diff -r 200:201 test.php

简写：svn di

**11、**将两个版本号之间的差异合并到当前文件

svn merge -r m:n path

比如：svn merge -r 200:205 test.php（将版本号200与205之间的差异合并到当前文件，可是一般都会产生冲突，须要处理一下）

**12、**SVN 帮助

svn help

svn help ci

**13、**版本号库下的文件和文件夹列表

svn list path

显示path文件夹下的全部属于版本号库的文件和文件夹

简写：svn ls

**14、**创建纳入版本号控制下的新文件夹

svn mkdir: 创建纳入版本号控制下的新文件夹。

使用方法: 1、mkdir PATH…

2、mkdir URL…

创建版本号控制的文件夹。

1、每个以工作副本 PATH 指定的文件夹，都会创建在本地端，而且增加新增

调度，以待下一次的提交。

2、每一个以URL指定的文件夹，都会透过马上提交于仓库中创建。

在这两个情况下，全部的中间文件夹都必须事先存在。

**15、**恢复本地改动

svn revert: 恢复原始未改变的工作副本文件 (恢复大部份的本地改动)。revert:

使用方法: revert PATH…

注意: 本子命令不会存取网络，而且会解除冲突的状况。可是它不会恢复

被删除的文件夹

**16、**代码库URL变更

svn switch (sw): 更新工作副本至不同的URL。

使用方法: 1、switch URL [PATH]

2、switch –relocate FROM TO [PATH...]

1、更新你的工作副本，映射到一个新的URL，其行为跟“svn update”非常像，也会将

server上文件与本地文件合并。这是将工作副本相应到同一仓库中某个分支或者标记的

方法。

2、改写工作副本的URL元数据，以反映单纯的URL上的改变。当仓库的根URL变动

(比方方案名或是主机名称变动)，可是工作副本仍旧对映到同一仓库的同一文件夹时使用

这个命令更新工作副本与仓库的相应关系。

**17、**解决冲突

svn resolved: 移除工作副本的文件夹或文件的“冲突”状态。

使用方法: resolved PATH…

注意: 本子命令不会依语法来解决冲突或是移除冲突标记；它仅仅是移除冲突的

相关文件，然后让 PATH 能够再次提交。

**18、**输出指定文件或URL的内容。

svn cat 目标[@版本号]…假设指定了版本号，将从指定的版本号開始查找。

svn cat -r PREV filename > filename (PREV 是上一版本号,也能够写详细版本号号,这样输出结果是能够提交的)

**19、**svn：'.'不是工作副本”，即当前文件夹不是工作副本，svn运行的文件夹须要在，SVN检出的版本号库裏面才干够运行，即当前运行svn的命令是svn检出的，必须存在.svn文件夹下的文件才干识别。

**20、**拷贝用户的一个未被版本化的目录树到版本库。

svn import 命令是拷贝用户的一个未被版本化的目录树到版本库最快的方法，如果需要，它也要建立一些中介文件。

**体会：**

通过这一次的实验，了解到了SVN = 版本控制 + 备份服务器，可以把SVN当成文件的备份服务器，更好的是，他可以帮助我们记住每次上传到这个服务器的档案内容。并且自动的赋予每次的变更一个版本。  
 通常，我们称用来存放上传档案的地方就做Repository。用中文来说，有点像是档案仓库的意思。不过，通常我们还是使用Repository这个名词。基本上，第一次我们需要有一个新增(add)档案的动作，将想要备份的档案放到Repository上面。日后，当要进行有任何修改时，都可以上传到Repository上面，上传已经存在且修改过的档案就叫做commit，也就是提交修改给SVN server的意思。针对每次的commit，SVN server都会赋予他一个新的版本。同时，也会把每次上传的时间记录下来。日后，因为某些因素，如果需要从Repository下载曾经提交的档案。我们可以直接选择取得最新的版本，也可以取得任何一个之前的版本。如果忘记了版本，还是可以靠记忆尝试取得某个日期的版本。这样就很方便我们对于文件的取用和修改了。通过熟悉这些常用的指令，我可以运用一些简单常用的指令来进行一些操作，让我明白了版本控制的重要性，通过SVN可以将文件恢复到之前的任何一个版本，这样就非常方便了，所以这一次的实验让我受益匪浅。