**项目管理工具**

**svn**

# 教学目标

1、掌握svn服务端、svn客户端、svn eclipse插件安装方法

2、掌握svn的基本使用方法

# svn介绍

## 项目管理中的版本控制问题

通常软件开发由多人协作开发，如果对代码文件、配置文件、文档等没有进行版本控制，将会出现很多问题：

* 备份多个版本，占用磁盘空间大
* 解决代码冲突困难
* 容易引发BUG
* 难于追溯问题代码的修改人和修改时间
* 难于恢复至以前正确版本
* 无法进行权限控制
* 项目版本发布困难

总结：SVN是什么？它的特点是什么？

SVN是项目管理版本控制工具

它的特点：

* + 备份多个版本，占用磁盘空间大
  + 解决代码冲突困难
  + 容易引发BUG
  + 难于追溯问题代码的修改人和修改时间
  + 难于恢复至以前正确版本
  + 无法进行权限控制
  + 项目版本发布困难

## 什么是版本控制

版本控制(Revision control)是维护工程蓝图的标准做法，能追踪工程蓝图从诞生一直到定案的过程。是一种记录若干文件内容变化，以便将来查阅特定版本修订情况的系统。

## svn是什么？

SVN（Subversion）是近年来崛起的版本管理工具，在当前的开源项目里(J2EE)，几乎95%以上的项目都用到了 SVN。Subversion 项目的初衷是为了替换当年开源社区最为流行的版本控制软件 CVS，在 CVS的功能的基础上有很多的提升同时也能较好的解决 CVS 系统的一些不足。

Svn设计的初衷就是为了取代CVS

SVN支持B/S系统 browers浏览器VS服务器server

也支持C/S系统 client客户端VS服务器server

CVS支持C/S系统

当前互联网用的比较多版本控制工具svn,另外一个是Git

Git支持分布式的版本控制工具

举例：百度把一个外包项目，下发到一个外包公司

外包公司想节约成本 只招5个大牛 项目经理 架构师。。

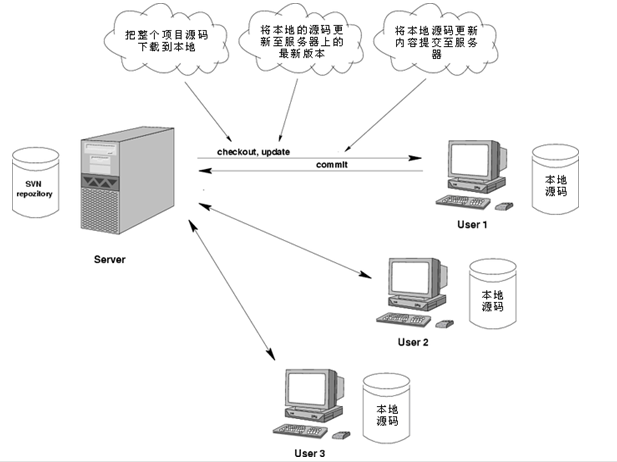
程序员从2 3线城市去招聘

从效率上没有git快。

从应用角度上 svn不支持分布式 git支持

## svn的使用方法

svn是基于客户/服务器模式：



在server上部署工程代码（shop1609C）。

1、开发人员1将服务器上shop1609C下载到本地(通过网络远程下载)

提交本地代码

2、开发人员1修改完成，将本地代码提交到服务器(通过网络远程提交)

3、开发人员2将服务器上shop1609C下载到本地(通过网络远程下载)

4、开发人员2修改完成，将本地代码提交到服务器(通过网络远程提交)

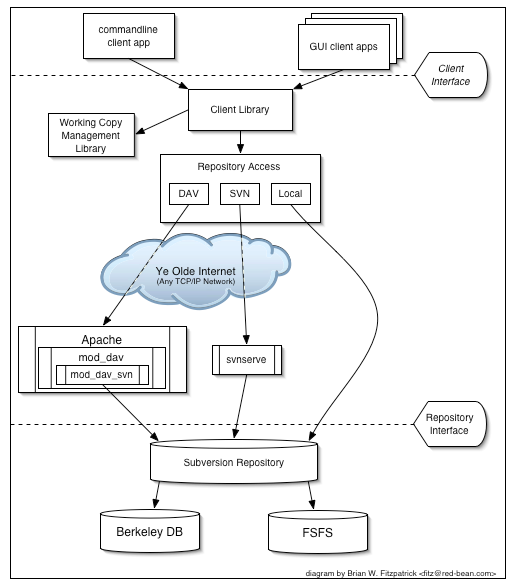
* 复制-修改-合并方案(Subversion默认的模式)

在这种模型里，每一个客户读取项目配置库建立一个私有工作副本——版本库中文件和目录的本地映射。用户并行工作，修改各自的工作副本，最终，各个私有的复制合并在一起，成为最终的版本，这种系统通常可以辅助合并操作，但是最终要靠人工去确定正误。

* 锁定-修改-解锁方案

在这样的模型里，在一个时间段里配置库的一个文件只允许被一个人修改。 此模式不适合软件开发这种工作。

## svn服务器的工作方式



服务端两种数据存储方法

将数据存储在仓库中。

1、将工程代码存储到数据库

2、将工程代码存储到文件系统（一般使用）

客户获取服务端数据：

1、客户直接连接服务端仓库(一般由svn专业管理员直接操作svn仓库的文件)

2、程序员在本地安装一个客户端，通过c/s方式连接svn服务端（一般使用）

3、程序员在本地安装一个客户端（使用浏览器），通过b/s方式连接svn服务端，如果采用b/s方式，svn软件需要集成apache（是一个web服务器，对外提供http服务）（一般使用）

独立服务器方式：

svnserve

借助Apache方式：

mod\_dav\_svn

SVN版本数据存储方式

BDB （Berkeley DB）

数据库方式

FSFS

文件方式 （推荐）

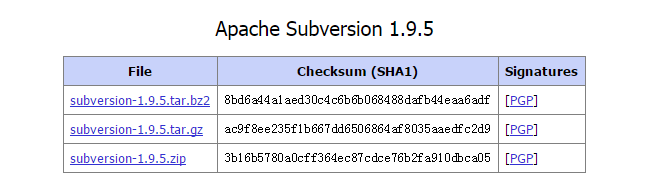
# svn服务端安装配置（2种）

## 两种服务端安装包

### 官方安装包

官方网站：http://subversion.apache.org/

下载：http://subversion.apache.org/download.cgi

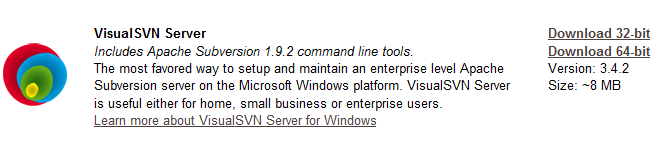


官方提供的服务端安装包，安装后需要通过命令行操作，适用于专业配置管理员使用。

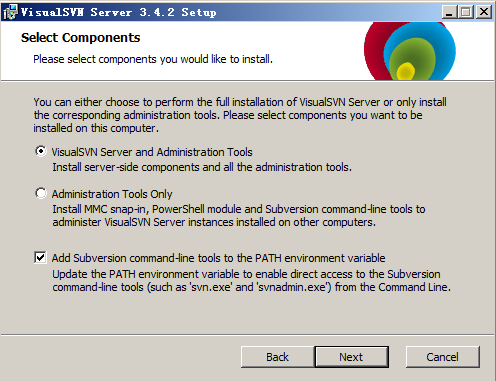
### 图形化服务端

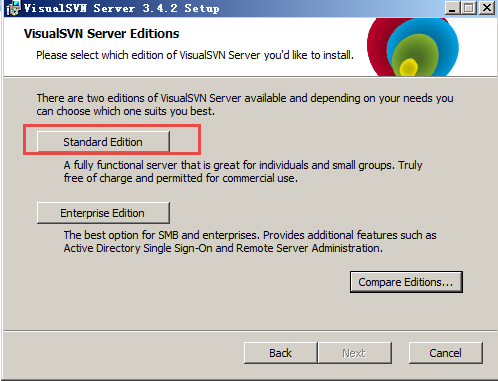
志愿者开发的图形化操作界面的svn服务端，它适用于普通软件开发人员使用。

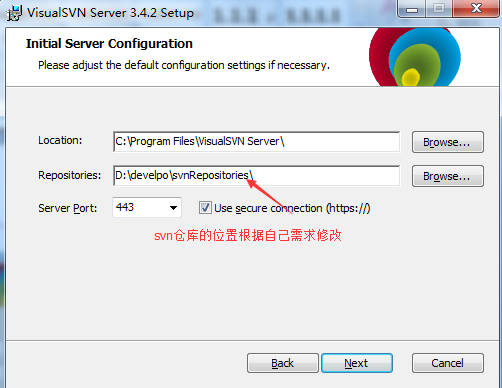
下载地址：https://www.visualsvn.com/downloads/



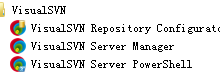
## 安装图形化服务端







查看程序菜单：



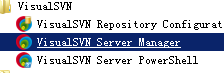
查看服务，VisualSVN成功启动：

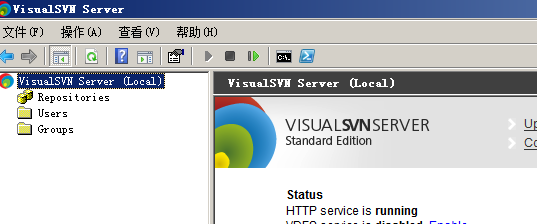


## 创建仓库

svn服务端创建完成需要创建仓库，仓库中存放要版本控制的文件。

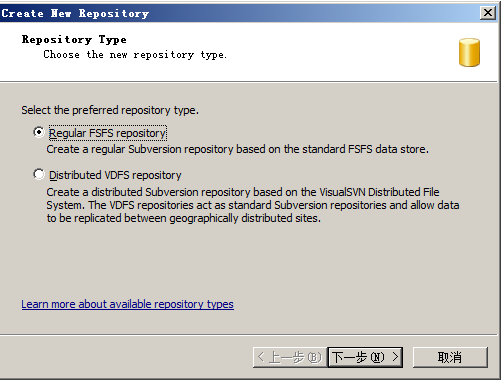
通过开始菜单进入VisualSVN server manager：

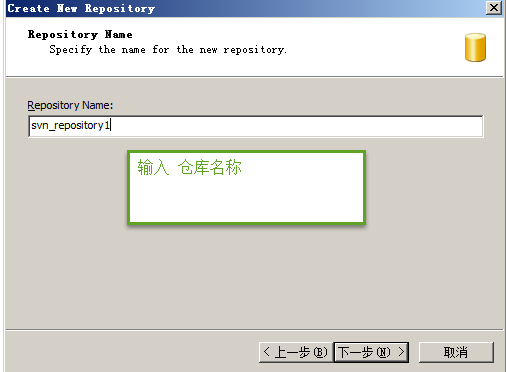




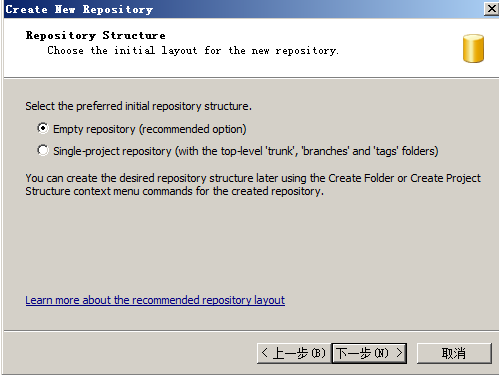


选择文件存储方式：

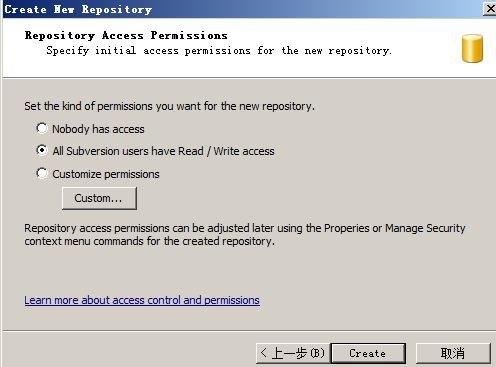




创建一个空的仓库：

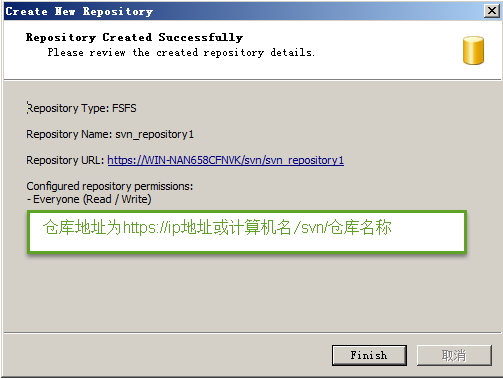


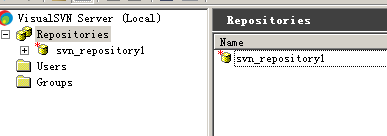
设置用户访问仓库权限：



仓库创建成功：

仓库地址为https://ip地址或计算机名/svn/仓库名称





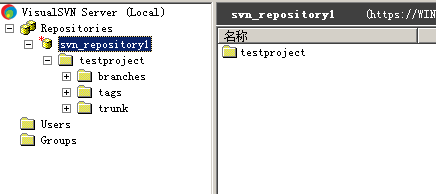
## 创建工程目录

仓库中存放开发项目代码、文档等，需要创建一个工程目录。





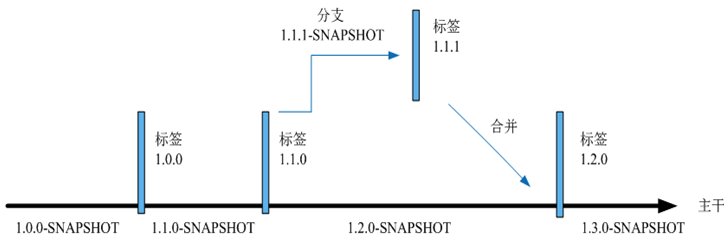
创建成功：



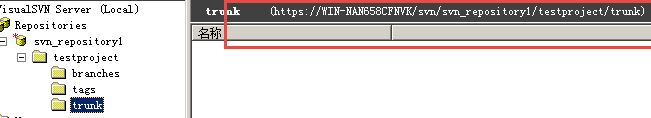
trunk：项目开发代码的主体，是从项目开始直到当前都处于活动的状态，从这里可以获得项目最新的源代码以及几乎所有的变更历史信息。

branches：从trunk的某个点分离出来的代码拷贝，通常可以在不影响主干的前提下在这里进行重大bug的修改，或者做实验性的开发，以及定制功能开发等。如果分支达到了预期的目的，通常可以被合并（Mgerge）到主干中。

tag：用来表示trunk和branch的某个点的状态，以代表项目的某个稳定状态，通常为最终发布状态。



工程目录创建完成，查看它的svn地址：



拷贝svn地址：



# TortoiseSVN客户端

## svn客户端类型

svn客户端需要通过网络访问svn服务端提交文件、查询文件等，可通过以下客户端类型访问svn服务端：

1、使用Subversion提供的客户端命令

使用方式：在命令行下输入命令操作。

2、使用Torotise图形化界面操作（推荐）

3、使用Eclipse等开发工具插件操作（推荐）

## 下载安装

TortoiseSVN是Subversion版本控制系统的一个免费开源客户端，不需要为使用它而付费。

TortoiseSVN是 Subversion 的 Windows 扩展。它使你避免接触 Subversion 枯燥而且不方便的 Command Line。它完全嵌入 Windows Explorer，使用时只需在正常的窗口里右键操作就可以了

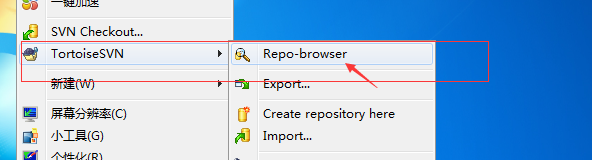


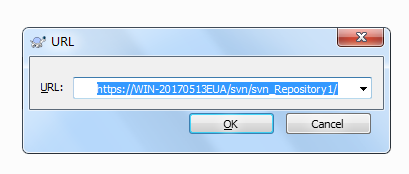
下载：http://tortoisesvn.net/downloads.html 提供 32位和64位不同版本，安装tortoiseSVN 需要修改客户端电脑右键菜单，安装后需要重启电脑。

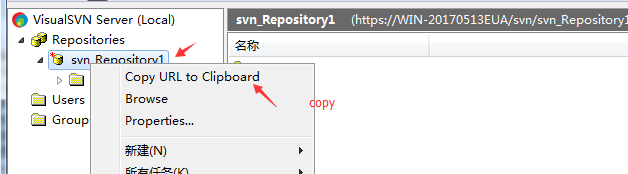


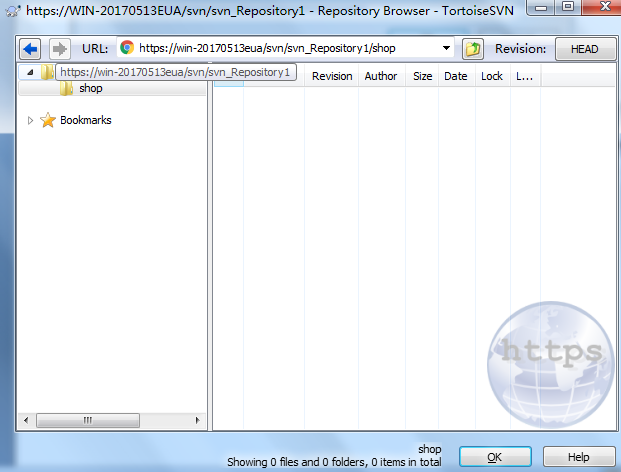
## 浏览仓库

使用Tortoise浏览svn服务端的仓库的内容：









# svn服务端权限控制（了解）

企业开发中，一个软件工程需要多个成员协助开发，还会有其它非开发人参与（指测试人员）。

安全控制，针对svn仓库每个目录，给每个成员分配权限。

分配权限流程：

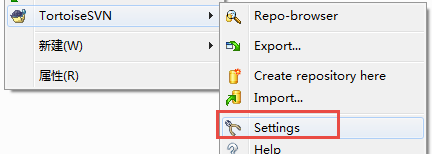
1、创建组，比如创建开发组、测试组

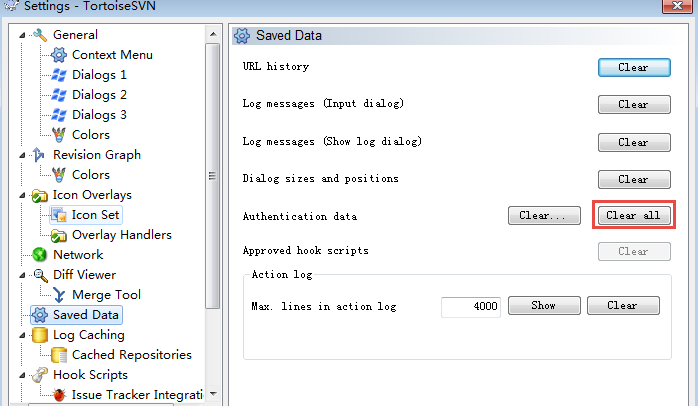
2、创建成员

3、给组分配成员

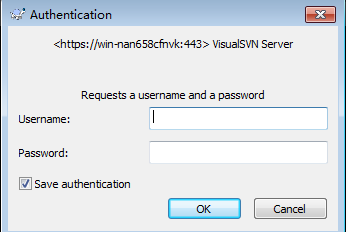
4、给组分配权限，组内成员也有了权限，

比如：开发组有读写权限，测试组有读权限





输入 账号和密码



没有权限访问：

# 权限管理(了解)

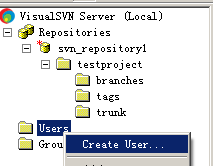
## 认证授权机制

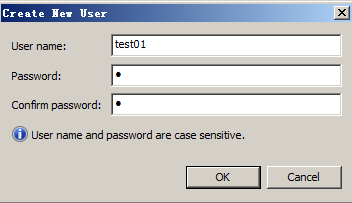
在企业开发中会为每位程序员、测试人员等相关人员分配一个账号，用户通过使用svn客户端连接svn服务时需要输入账号和密码，svn服务对账号和密码进行校验，输入正确可以继续访问，当用户访问仓库下某个目录时，svn服务对用户进行授权，如果用户拥有该目录的访问权限方可访问。

判断账号和密码输入是否正确的过程即认证过程。

判断用户是否拥有目录的读/写权限时即授权过程。

## 创建用户





查看已创建的用户：

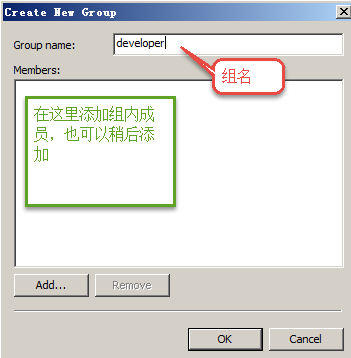


修改用户：

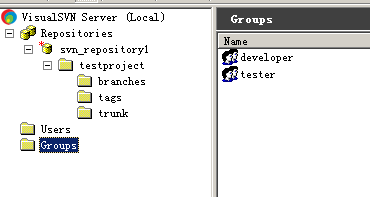


## 创建组

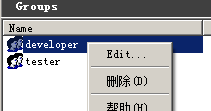




查看创建的组：



修改组：

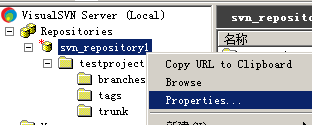


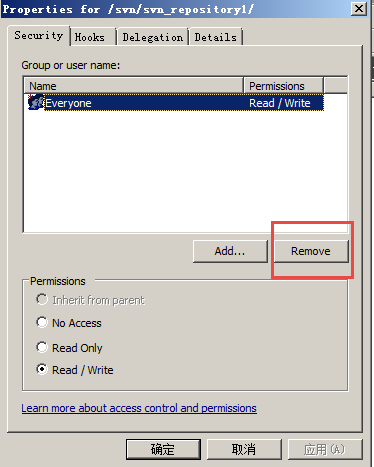
## 分配仅限

给仓库下的每个目录分配权限对访问进行控制。

### 删除默认权限

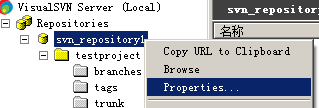
删除系统安装后默认权限：



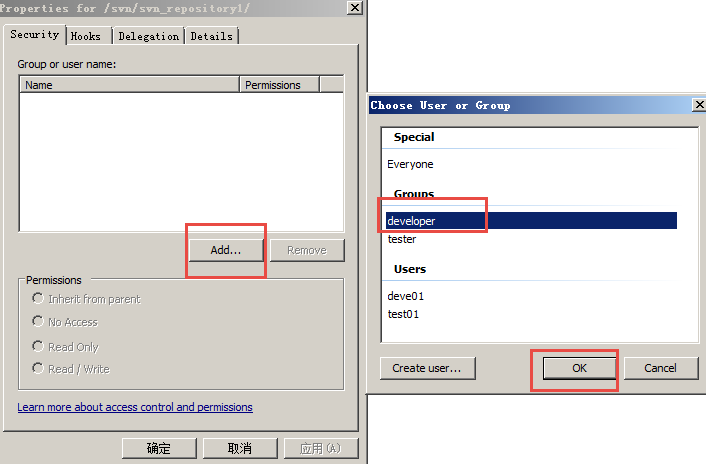


### 示例一：开发人员拥有读写权限

进入权限分配界面：

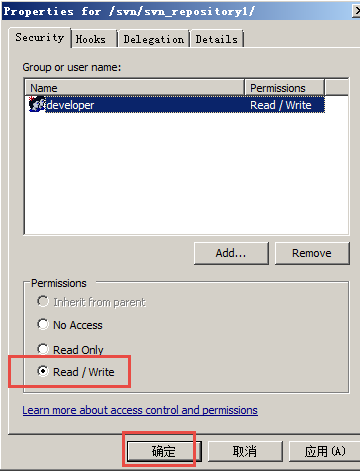


添加组或用户：

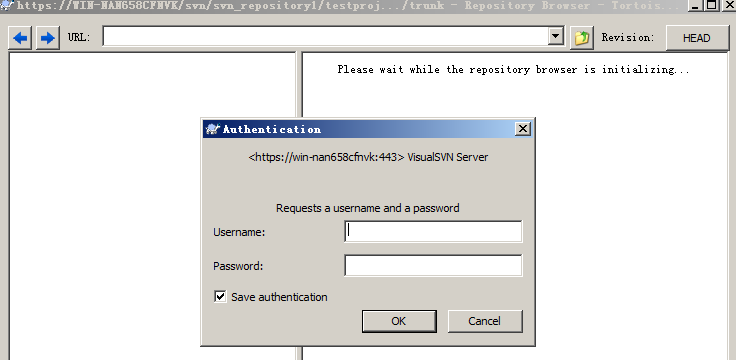


分配权限：

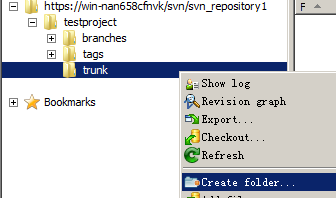
继承父目录权限、不可访问、读权限、读/写权限



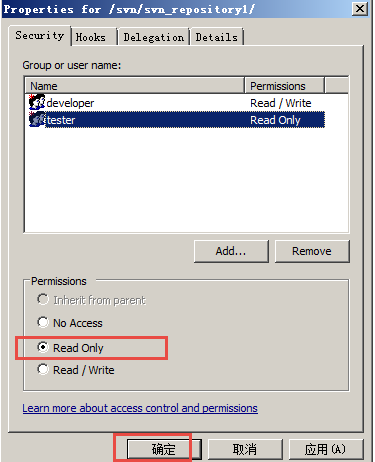
访问时输入账号：



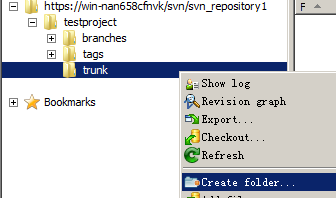
登陆测试是否有读/写权限：



### 示例二：测试人员拥有读权限



登陆测试是否有读/写权限：

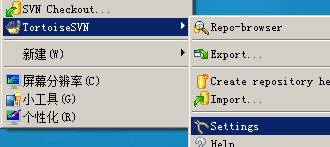


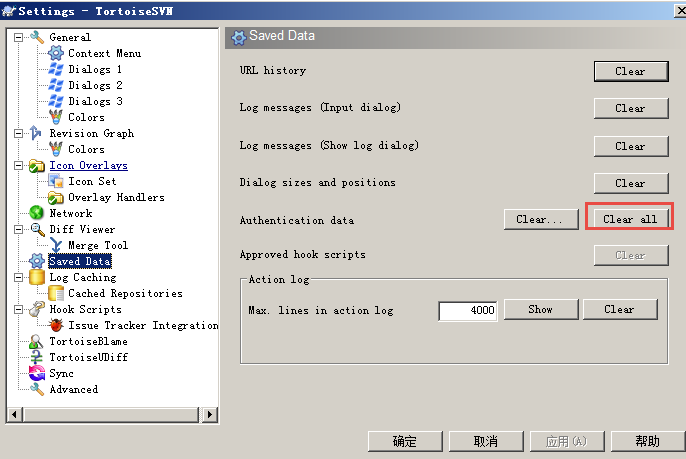
### 清除认证缓存

有几种情况需要清除认证缓存：

1、本地使用多个账号登陆，每次输入的账号和密码都不一样

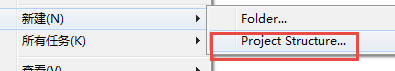
2、当账号密码修改后（建议清理）

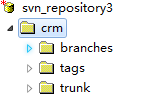




# 创建工程目录（了解）

创建一个完整的svn管理标准工程结构：





1、trunk：主干

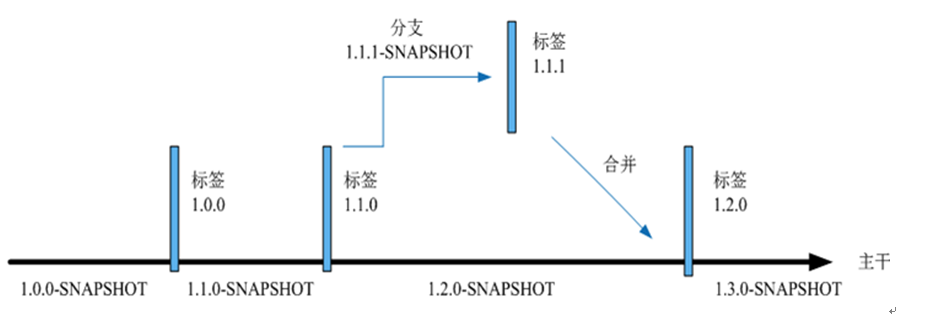
维护的代码，程序员从项目开发就开始针对主干中代码进行开发修改。

2、branches：分支

当软件出现重大bug需要修改时候，要新开一个分支，分支来源于主干或tags。 分支代码修改完成需要合并到主干。

3、tags：标签

针对软件的一个结点打一个标签，软件每个版本在tags目录中存在不同的目录。



# TortoiseSVN日常使用命令

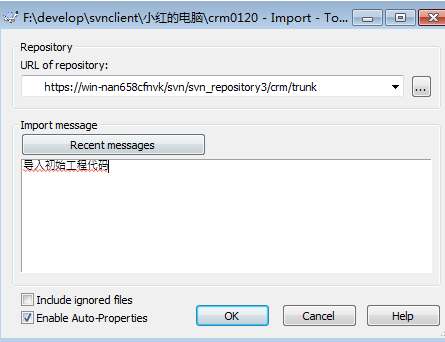
## 浏览仓库

Repo-browser ： 浏览仓库中资源信息

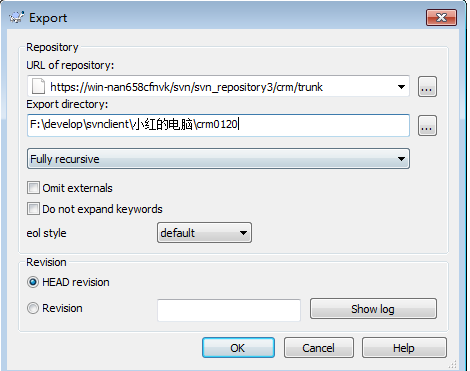


## 将工程导入到仓库

将工程代码导入trunk目录下https://win-nan658cfnvk/svn/svn\_repository3/crm/trunk



## 从svn导出工程

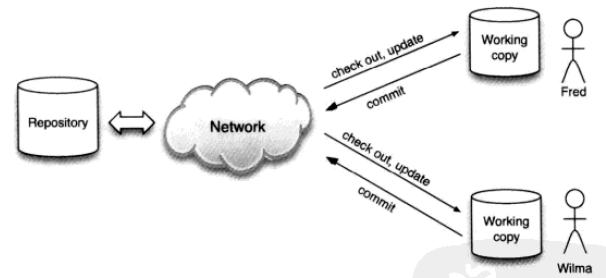


导出和checkout区别：

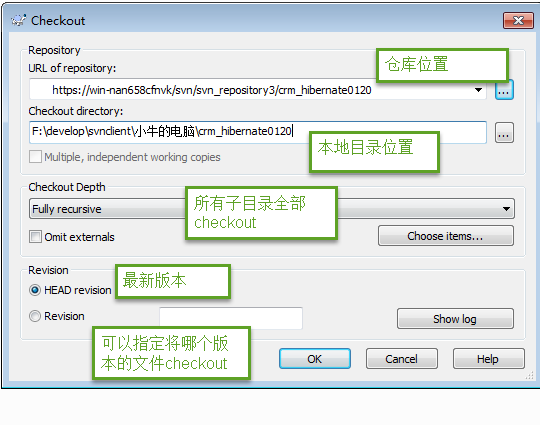
checkout：将代码从svn下载到本地，下载到本地目录受svn版本管理。

导出：不受版本控制

## 修改提交



1、将工程下载到本地一份，使用checkout



2、在本地修改checkout目录中的文件，如果修改了说明本地版本和svn服务器上文件版本不一致。

3、进行commit

右键文件，将此文件commit

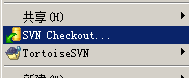
右键目录，显示此目录中版本一致的所有文件，选择文件commit

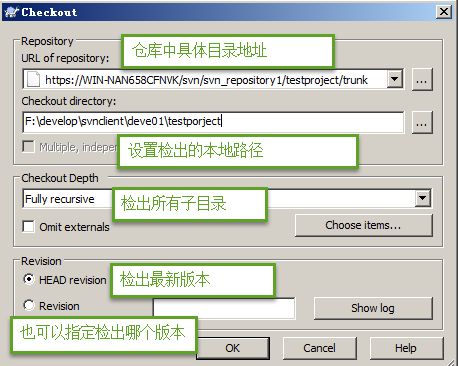
### Checkout

检出项目，复制项目的副本到本地。

从svn服务端下载一份副本到本地，目录受svn版本控制

在要检出的目录中右键：





### add

在检出的目录中添加文件：

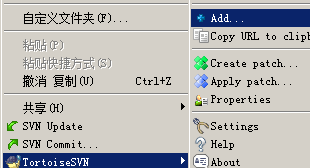


图标： 这是一个新文件

* Add to ignore list ：添加到忽略列表 （标记该文件不需要版本控制 ）



* Add ： 标记这个文件添加到服务器



 已经标记要添加到版本库

* 当前版本（本地版本）和svn服务端版本不一致

#### 添加文件提交步骤

1、将工程下载到本地一份，使用checkout

2、在checkout的目录中新增文件

方法1：先执行 add，此文件打上新增标记，再执行commit

好处：当打上add标识之后，在没有commit之前可以回滚

方法2：在checkout的目录中commit。

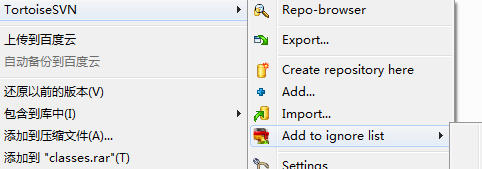
svn客户端自动执行add 和commit，此方法一旦执行不回滚。

如果要大批量添加文件或将整合工程上传到svn时候，注意选择哪些文件或目录不受版本控制：

在web工程中，对src、jsp等目录的文件需要由svn进行版本控制。

对于classes编译class文件不需要svn版本控制。

将classes目录添加到忽略版本控制列表中：



如果此目录为不受svn版本控制，此目录中添加或修改文件svn客户不作检测。

### Commit

当修改了之后要提交

文件如果被标识 为添加的新文件之后，要进行commit才生效

文件被标识删除之后，要进行commit才生效

### 回滚

文件被标识添加或删除，可以通过回滚操作，取消添加和删除

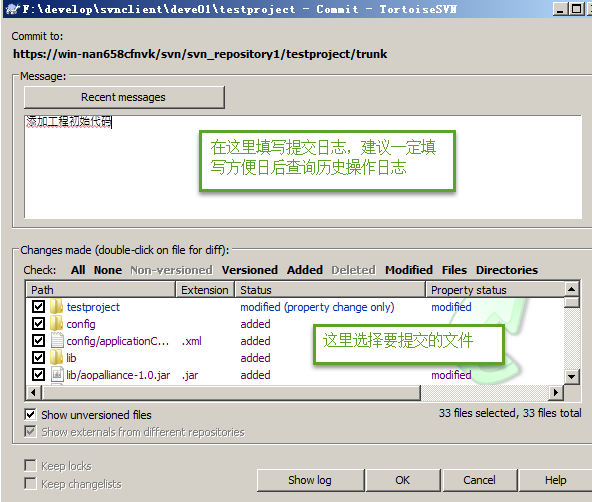


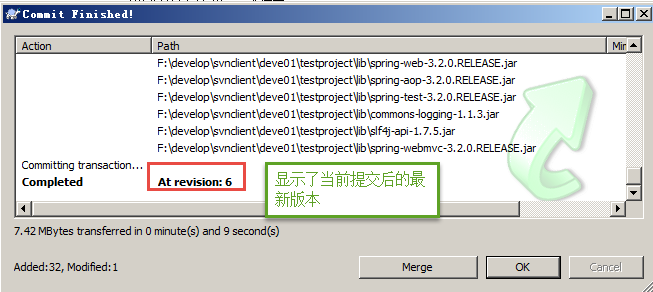
当检出目录或子目录中内容有修改，目录图标变为：

提交Commit 提交本地修改至svn服务器：

在检出目录或要提交修改的目录右键：







提交后目录中的内容与svn服务同步，目录图标变为：

### 将工程传到svn服务器

1、方式1 通过import导入方法

右键文件夹，Import导入，将此文件夹下的子目录及文件导入到仓库中

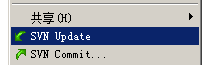
2、方式2，通过checkout，在要存储svn副本的位置，右键，checkout。

在checkout的目录有一个.svn的隐藏文件，在当前目录中右键，通过commit提交将新工程提交到svn服务器。

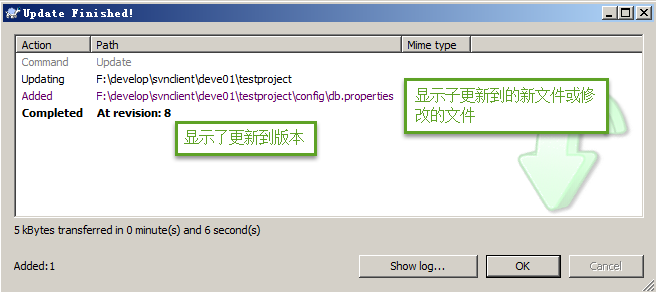
### update

更新仓库的文件到本地

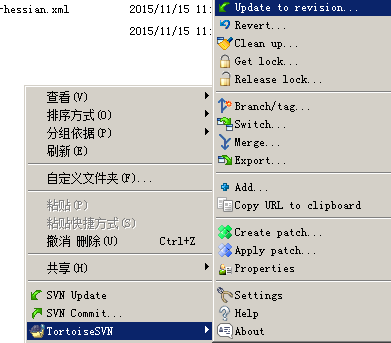
在检出目录或子目标或文件上右键：

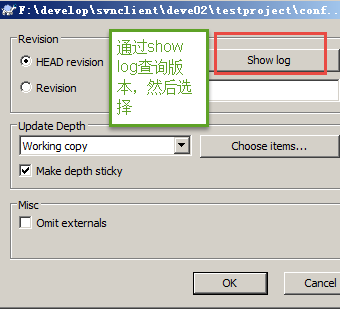


#### 更新到最新版本



#### 更新到指定版本





### Delete

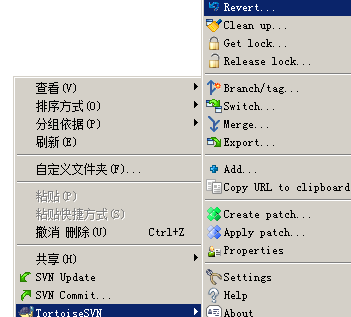
Delete ：删除版本库文件

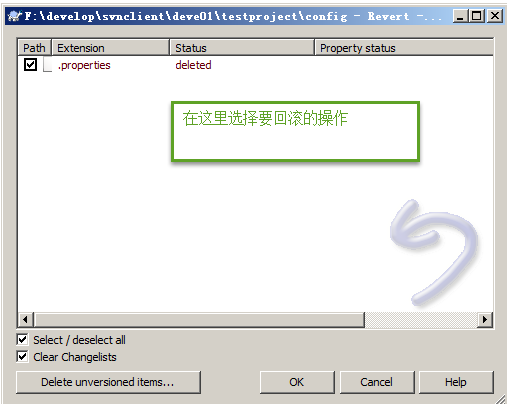
 标记删除后，本地文件删除，标记删除后需要提交。

### 恢复

在检出目录或子目录操作会记录操作日志，提交前可以回滚操作。

在要回滚的检出目录或子目录中右键：



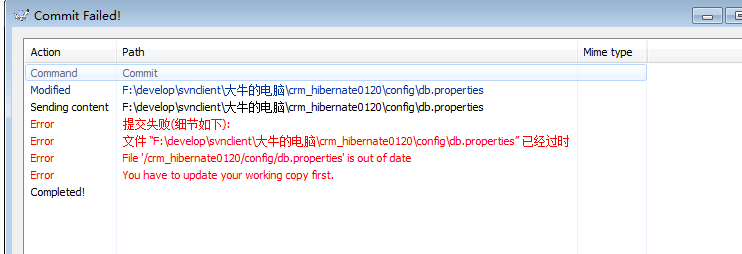


## 冲突处理

两个客户端同时修改同一个文件， 改动同一个位置，发生冲突情况

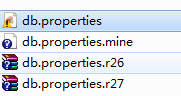
如果当commit 遇到文件已经过时，说明另一个人可能改动过 ----- update

1、修改文件后执行提交操作，报错：



原因：

当前修改文件版本不是在服务器上最新版本的基础上改的。



出现了三个新文件：

.mine 自己修改后的文件

.r26 自己修改前的文件

.r27 其它修改后的文件

处理方法1：通过图形界面查看哪些文件行内容有冲突

右键db.properties，编辑冲突



通过图形界面查询哪些行有冲突。

处理方法2：手工处理

手工修改成一个正确的内容，提交，要删除自动创建那三个文件。

### 如果避免冲突(重点)

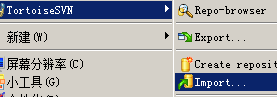
及时update更新，在最新版本的基础上去修改，修改后及时提交。

## 导入导出

* Export ：导出项目 ，和checkout区别 （checkout检出后文件，含有.svn隐藏文件夹， 会和SVN仓库交互， export导出，没有.svn隐藏文件夹）



* import 将本地资源导入到svn 服务器



## 锁(了解)

* Get Lock 获取文件修改锁



一旦文件加锁后，只能一个人在修改，其它人无法获取锁，进行修改



# eclipse的SVN插件使用

## svn插件安装

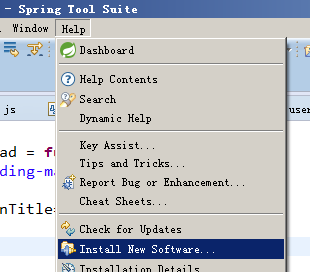
下载Subversion的eclipse插件

<http://subclipse.tigris.org/servlets/ProjectProcess?pageID=p4wYuA>

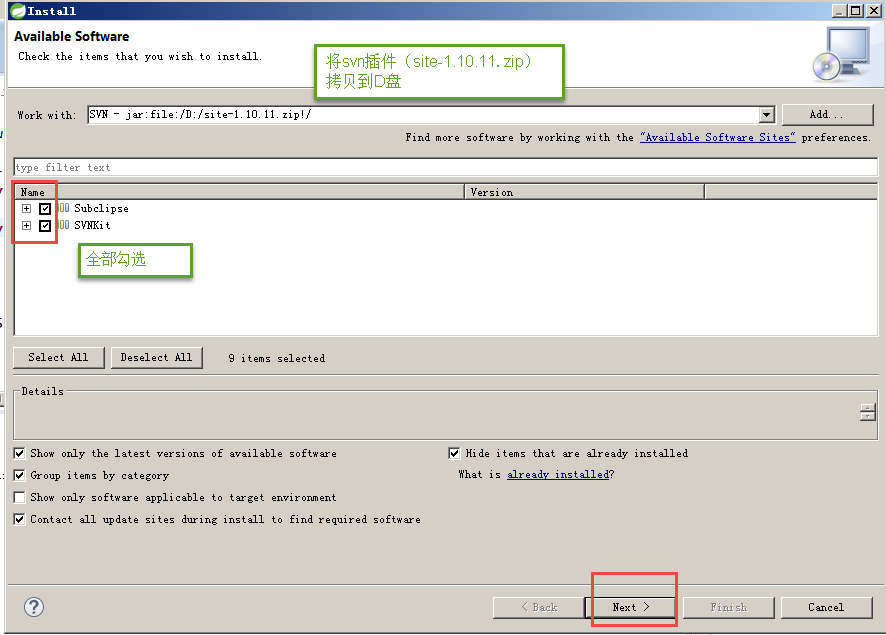
下载 site-1.10.11.zip，本教程将此zip放在D盘。

下边是在sts上安装svn插件步骤：

1、进入STS软件安装界面



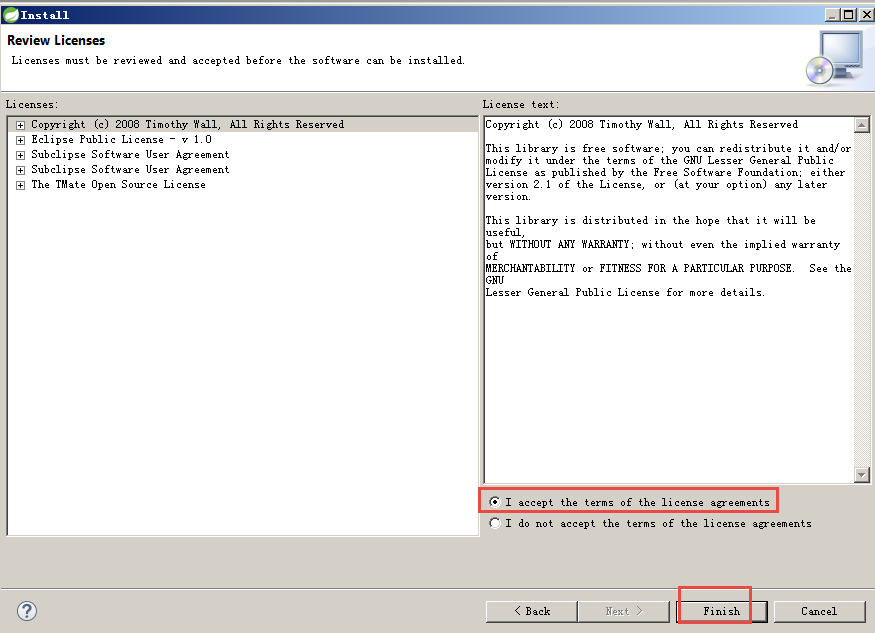
2、安装svn插件



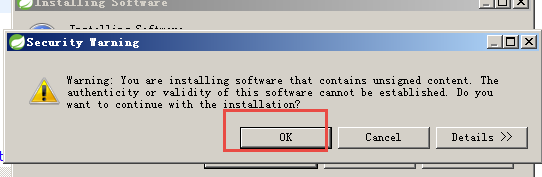
命令SVN - jar:file:/d:/site-1.10.11.zip!/

点击下一步。

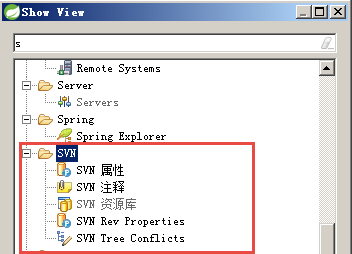
接受协议，完成：



出现提示，选择继续安装：



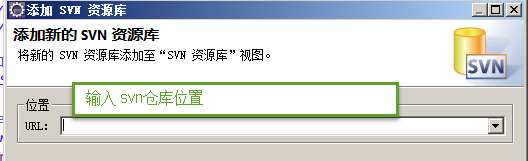
安装完成，查看STS视图有SVN选项说明安装成功：

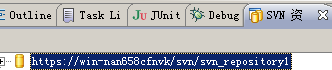


## 将项目共享到SVN

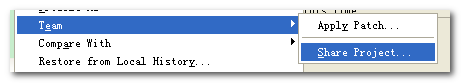
新建SVN仓库连接 ， 切换“SVN资源库” 视图



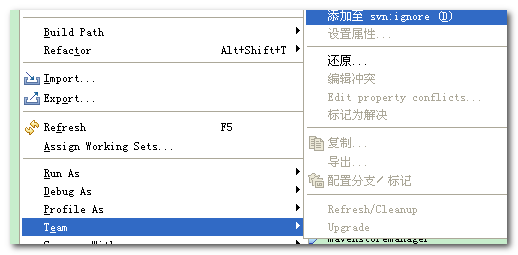




共享项目到SVN



注意： 共享后，SVN服务器上只有项目文件夹， 此时需要选择哪些资源不被管理！

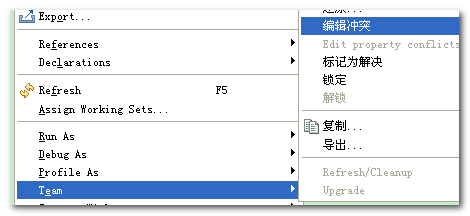


## 从svn检出

将svn管理项目检出到工作区



## 解决冲突



手动merge后，标记为解决

