名词说明：

WC：Working Copy 你的工作区

Versioned：受控的；受版本控制的

**SVN是什么？**

* SVN是开源的版本控制系统。
* 比CVS更多的特性。一个更好的CVS？因此SVN具有大部分CVS拥有的特性。
* 不仅仅是文件受控，目录也加入版本控制。
* 复制，删除，重命名这些操作都是受控的。
* 特殊作用的元数据（属性）。
* 提交事务原子性。提交完成之前任何一个部分都不会正真生效。版本号基于提交，而不是基于文件。提交时的log message也是依附于提交的那个版本。
* 创建分支和标记非常简单。简单的通过copy命令就可以创建分支和标记。
* 合并跟踪。在合并时协助你处理所有的更改点。
* 文件锁定。svn支持文件锁定修改策略。
* Unix的link可以受控了。前提是WC必须在Unix下。
* 可选的Apache network server，基于WEBDAV/DeltaV 协议。熟悉Apache的管理员会很喜欢这一点。
* 内置的server选择（svnserve）。如果不想用Apache，可以试试Svn自己的server：svnserve。同样也能提供授权和验证，ssh通道等功能。
* 方便解析的输出。svn的输出尽量做到方便阅读和方便机器解析。
* 冲突解决交互性良好。svn命令行客户端提供多种不同的方式解决冲突。
* svn提供一个实用工具，svnsync来实现从主库到只读附属库的同步。
* 持续写入代理功能让只读的附属库专门处理所有的读请求。所有的写请求交给主库处理。这个功能只在使用Apache WebDAV server的时候才有效。
* 基于本地化的客户机服务器体系，分层的库，明晰的API方便扩展。
* 高效处理二进制文件。
* 性能消耗与更改点的数量成正比。
* svn的api可以和多种语言集成。Python，Perl,Java,Ruby（svn本身是用C写的）
* ChangeLists功能。

Svn的安装分客户端和服务端。

你可以在如下地址找到下载：http://subversion.tigris.org/

这篇文章主要介绍的是svn客户端的命令，你至少需要安装客户端。默认就是以SVN做版本控制的。如果你不想在自己机器上安装服务版，google code是个练习svn命令的好地方。

http://code.google.com上申请托管项目很简单，但是现在不支持svn lock。目前最大支持1G空间。

以下是svn客户端常用命令一览：

**svn add [path]**

别名：无

描述：添加文件或目录到你的wc，打上新增标记。这些文件会在下一次你提交wc的时候提交到svn服务器。

在提交前，你也可以用svn revert撤销新增的文件。

访问库：否

eg: svn add file.cpp

**svn blame Target[@REV]**

别名：praise,annotate,ann

描述：显示某个已受控文件的每一行的最后修改版本和作者

访问库：是

eg: svn blame file.cpp

eg: svn blame --xml file.cpp ##加上xml参数可以以xml格式显示每一行的属性。

**svn cat TARGET[@REV]**

别名：无

描述：输出指定目标的内容，这里的目标一般是文件。

访问库：是

eg:svn cat file.cpp

eg:svn cat file.cpp -r 2 ##显示版本号为二的file.cpp内容。

eg:svn cat file.cpp --revision HEAD ##显示最新版本的file.cpp内容。

**svn changelist CLNAME TARGET...**

**svn changelist --remove TARGET**

别名：cl

描述：可以将wc中的文件从逻辑上分组.

访问库：否

eg:svn cl clName file.cpp file2.cpp file3.cpp ##将file.cpp等三个文件加入名叫clName的changelist

eg:svn commit --changelist clName -m "ci" ##将clName下的所有文件提交

**svn checkout URL[@REV]... [PATH]**

别名：co

描述：检出

访问库：否

eg:svn checkout file:///var/svn/repos/test file:///var/svn/repos/quiz working-copies

eg:svn checkout -r 2 file:///var/svn/repos/test mine ##check out 版本号为2的项目

**svn cleanup [PATH...]**

别名：无

描述：递归的清理WC中过期的锁和未完成的操作。

访问库：否

eg：svn cleanup

**svn commit [PATH...]**

别名：ci

描述：把你WC的更改提交到仓库

访问库：是

eg：svn commit -m "added howto section." ##默认情况下提交必须提供log message

**svn copy SRC[@REV]... DST**

别名：cp

描述:copy操作可以从WC到WC；WC到URL；URL到WC；URL到URL。现在SVN只支持同一个仓库内文件的拷贝，不允许跨仓库操作。

访问库：是

eg：svn copy -r 11 file:///var/svn/repos/test/trunk \

file:///var/svn/repos/test/tags/0.6.32-prerelease \

-m "Forgot to tag at rev 11"

##copy命令是创建分支和标记的常用方式。copy到url的操作隐含了提交动作，所以需要提供log messages。

**svn delete PATH...**

别名：del，remove，rm

描述：删除

访问库：如果PATH是库地址时会，删除WC内的文件不会。

eg：svn del localfile.cpp ##删除WC里的文件，在下一次提交WC的时候才会真正在仓库里将对应文件删除。

eg: svn del file:///var/svn/repos/test/yourfile ##删除仓库里的文件

**svn diff**

别名：di

描述：用来比较并显示修改点。

访问库：

eg：svn diff ##最常用的方式，用来显示WC基于最近一次更新以后的所有的本地修改点。

eg：svn diff -r 301 bin ## 比较WC和版本301中的bin目录的修改点

eg：svn diff -r 3000:3500 file:///var/svn/repos/myProject/trunk ##比较库里主干3000版和3500版的差异。

eg：svn diff --summarize --xml http://svn.red-bean.com/repos/test@r2 http://svn.red-bean.com/repos/test ##--summarize --xml 参数将差异情况以xml文档的方式显示出来。

**svn export [-r REV] URL[@PEGREV] [PATH]**

**svn export [-r REV] PATH1[@PEGREV] [PATH2]**

别名：无

描述：导出一个干净的目录树，不包含所有的受控信息。可以选择从URL或WC中导出。

访问库：如果访问的是URL则会。

eg：svn export file:///var/svn/repos my-export ##导出到my-export目录。

**svn help — Help!**

别名：?,h

描述：不用解释了

访问库：否。

**svn import [PATH] URL**

别名：无

描述：导入本地一个目录到库中。但是导入后，本地的目录并不会处于受控状态。

访问库：是。

eg：svn import -m "New import" myproj http://svn.myProject.com/repos/trunk/misc

**svn info [TARGET[@REV]...]**

别名：无

描述：显示指定WC和URL信息。

访问库：仅当访问的是库路径时。

eg：svn info --xml http://svn.myProject.com/repos/test ##将信息以xml格式显示。

**svn list [TARGET[@REV]...]**

别名：ls

描述：显示目标下的文件和目录列表。

访问库：如果访问的是库地址就会。

eg：svn list --verbose file:///var/svn/repos ##--verbose参数表示显示详细信息。

**svn lock TARGET...**

别名：无

描述：对目标获得修改锁。如果目标已被其他用户锁定，则会抛出警告信息。用--force参数强制从其他用户那里获得锁。

访问库：是

eg：svn lock --force tree.jpg

**svn log [PATH]**

**svn log URL[@REV] [PATH...]**

别名：无

描述：从库中显示log消息。log消息代码 A ：added D：deleted M：modified R：replaced

访问库：是

eg：svn log -v http://svn.myProject.com/repos/test/ foo.c bar.c ##详细显示指定URL的库中foo.c和bar.c所有版本的log信息。

eg：svn log -r 14:15 ##显示当前WC的14和15版本log信息。

eg：##如果版本号不连续，只能采用如下方式。

$ svn log -r 14 > mylog

$ svn log -r 19 >> mylog

$ svn log -r 27 >> mylog

**svn merge sourceURL1[@N] sourceURL2[@M] [WCPATH]**

**svn merge sourceWCPATH1@N sourceWCPATH2@M [WCPATH]**

**svn merge [[-c M]... | [-r N:M]...] [SOURCE[@REV] [WCPATH]]**

别名：无

描述：合并两个受控源的不同之处，存放到一个WC里。

访问库：只有当访问库地址时。

eg：svn merge --reintegrate http://svn.example.com/repos/calc/branches/my-calc-branch ##合并分支上的改变项到WC，往往用于分支合并到主干。

eg：svn merge -r 156:157 http://svn.example.com/repos/calc/branches/my-calc-branch ##将制定URL版本156到157的所有更新合并到WC。

**svn mkdir PATH...**

**svn mkdir URL...**

别名：无

描述：在WC或库路径创建目录

访问库：只有当访问库地址时。

eg：svn mkdir newdir

**svn move SRC... DST**

别名：mv, rename, ren

描述：等同于svn copy命令跟个svn delete命令。WC到URL的重命名是不被允许的。

访问库：只有当访问库地址时。

eg：svn move foo.c bar.c ##将foo.c改名成bar.c。

**svn propdel PROPNAME [PATH...]**

**svn propdel PROPNAME --revprop -r REV [TARGET]**

别名：pdel, pd

描述：从受控文件，目录等删除属性。第二种是删除某个指定版本上的附加属性。

访问库：只有当访问库地址时。

eg：svn propdel svn:mime-type someFile ##从someFile上移除svn:mime-type这个属性。

**svn propedit PROPNAME TARGET...**

**svn propedit PROPNAME --revprop -r REV [TARGET]**

别名：pedit, pe

描述：编辑属性

访问库：只有当访问库地址时。

eg：svn propedit svn:keywords file.c ##修改file.c上的svn:keywords属性。

**svn propget PROPNAME [TARGET[@REV]...]**

**svn propget PROPNAME --revprop -r REV [URL]**

别名：pget,pg

描述：从文件，目录或版本取得指定属性的值。

访问库：只有当访问库地址时。

eg：svn propget svn:keywords file.c ##从file.c中取得svn:keywords属性的值

**svn proplist [TARGET[@REV]...]**

**svn proplist --revprop -r REV [TARGET]**

别名：plist, pl

描述：列出文件、目录或版本上的所有附加属性

访问库：只有当访问库地址时。

eg：svn proplist --verbose file.c

**svn propset PROPNAME [PROPVAL | -F VALFILE] PATH...**

**svn propset PROPNAME --revprop -r REV [PROPVAL | -F VALFILE] [TARGET]**

别名：pset,ps

描述：给文件、目录或版本附加属性并赋值

访问库：只有当访问库地址时。

eg：svn propset svn:mime-type image/jpeg file.jpg ##给file.jpg附加属性svn:mime-type 其值为image/jpeg

eg:svn propset --revprop -r 25 svn:log "Journaled about trip to New York."

##给版本25补上log message

eg:svn propset svn:ignore '.classpath' .

##在本地忽略掉.classpath文件

**svn resolve PATH...**

别名：无

描述：将冲突的文件标记为已解决，并且删掉冲突产生的临时文件。注意这个命令并不是能把冲突解决，解决冲突还是得靠人工。

访问库：否

eg：svn resolve --accept mine-full foo.c ##1.5版本后，加上--accept参数，尝试自动处理冲突。

**svn resolved PATH...**

别名：无

描述：已过时，被resolve --accept取代。去除冲突的状态和冲突临时文件。

访问库：否

**svn revert PATH...**

别名：无

描述：还原WC中所有的本地更改。

访问库：否

eg：svn revert --depth=infinity . ##将整个目录所有文件还原

**svn status [PATH...]**

别名：stat, st

描述：输出WC中文件和目录的状态。如果WC提交，这些状态就会同步到库里。

一般状态有 ' ' 没有修改

'A' 新增

'D' 删除

'M' 修改

'R' 替代

'C' 冲突

'I' 忽略

'?' 未受控

'!' 丢失，一般是将受控文件直接删除导致

访问库：加上--show-updates参数时会

eg：svn status wc

**svn switch URL[@PEGREV] [PATH]**

**svn switch --relocate FROM TO [PATH...]**

别名：sw

描述：将WC转向一个其他的库地址同步

访问库：是

eg：svn sw http://svn.myProject.com/repos/trunk/vendors . ##将当前WC切换到另一个URL

**svn unlock TARGET...**

别名：无

描述：解锁

访问库：是

eg：svn unlock somefile

**svn update [PATH...]**

别名：up

描述：更新WC，更新反馈有如下几种分类。

A 新增

B 锁破坏

D 删除

U 更新

C 冲突

G 合并

E 存在的

访问库：是

eg：svn up -r22 ##更新到一个指定版本

ps:如何去除SVN中保存的用户授权密码

在Subversion安装目录下找到auth/svn.simple目录，将下面的文件删除即可。

如果在乌龟中，可以setting->saved data->Authentication Data 点 clear 即可。