**关于SVN**

由 珍珍阿姨 创建，路飞 最后一次修改 2016-11-24 14:36:10

**SVN 教程**

Apache Subversion 通常被缩写成 SVN，是一个开放源代码的版本控制系统，Subversion 在 2000 年由 CollabNet Inc 开发，现在发展成为 Apache 软件基金会的一个项目，同样是一个丰富的开发者和用户社区的一部分。这个教程给你提供一个理解 SVN 系统，操作当前和历史版本的文件，比如代码、网页、文档。

**适用人群**

这个教程设计为了让对 SVN 感兴趣的软件专业人士简单方便地开始。完成这个教程，你将充分了解 SVN 让自己获得更高的水平的专业知识。

**学习前提**

在你继续本教程之前，你必须对简单的术语有一定的了解，比如源码，文档等等。因为在你的组织下处理各级软件项目，如果你有软件工作的知识在软件开发和软件测试流程那将是好的。

**版本信息**

书中演示代码基于以下版本：

| **语言/框架** | **版本信息** |
| --- | --- |
| SVN | 1.6.11 |

# SVN 基本概念

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:46

## 什么是版本控制系统(VCS)

**版本控制系统 (VCS)** 是一个软件，帮助软件开发人员团队工作并维持他们完整的工作历史。 **下面是版本控制系统(VCS) 的目标**

* 允许开发者们同时工作
* 不会重写每个人的改变
* 维持每个版本的全部的历史

**VCS 被分成两种**

* 集中版本控制系统 (CVCS) 和
* 分散或不集中的版本控制系统 (DVCS)

在这个教程里，我们只专注于集中的版本控制系统特别是 Subversion，Subversion 基于集中的版本控制系统，意味着使用统一的服务器让团队协作。

**版本控制的术语**

让我们先懂得一些在这个教程将用到的术语

* 仓库: 仓库是任何一个版本系统的核心，它是开发者们保存工作的总部，仓库不止处理文件还有历史记录，它需要访问网络，扮演服务器的角色，版本控制工具扮演客户端的角色，客户端可以连接仓库，那么他们就可以从仓库中存储或者提取。通过保存这些更改，一个客户端的更改可以被其他人检索到，一个客户端可以让其他人的更改作为一个工作副本。
* 主干：trunk 是主要开发所在的目录，经常被项目开发者们查看。
* 标签：tags 目录用于储存项目中被命名的快照，标签操作允许给予对仓库中特定版本一个描述和一个难忘的名字。比如，LAST\_STABLE\_CODE\_BEFORE\_EMAIL\_SUPPORT 比 Repository UUID: 7ceef8cb-3799-40dd-a067-c216ec2e5247 和Revision: 13 更令人难忘。
* 分支：分支操作用于创建开发的另一条线，当你想把开发进程复制进两个不同的方向是很有用的。比如，当你发布 5.0 版本时，你可能想从 5.0 的 bug 修复中分离出来创建一个开发 6.0 功能的分支。
* 工作副本：工作副本是仓库的一个快照。这个仓库被所有的成员共享，但人们不直接修改它，相反每个开发者检查这个工作副本，工作副本是一个私人的工作空间，这里开发者可以独立于其他成员做自己的工作。
* 提交更改：提交是一个保存更改的过程，从私人工作空间到中央服务器。提交后，更改对全部成员可用，通过更新工作副本其他开发者提取这些更改。提交是一个原子操作，要么全部提交成功要么回滚，用户绝不会看到一半完成提交。

# SVN 环境搭建

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:47

## SVN 环境搭建

Subversion 是一个受欢迎的开源的版本控制工具。他在互联网免费提供并且开源。大多数 GNU/Linux 发行版系统自带，所以它很有可能已经安装在你的系统上了。可以使用下面命令检查是否安装了。

[jerry@CentOS ~]$ svn --version

如果 Subversion 客户端没有安装，命令将报告错误，否则它将出现安装的软件版本

[jerry@CentOS ~]$ svn --version

-bash: svn: command not found

如果你使用基于 RPM 的 GNU/Linux，可以使用 yum 命令进行安装，安装成功之后，执行 svn --version命令。

[jerry@CentOS ~]$ su -

Password:

[root@CentOS ~]# yum install subversion

[jerry@CentOS ~]$ svn --version

svn, version 1.6.11 (r934486)

compiled Jun 23 2012, 00:44:03

如果你使用基于 Debian 的 GNU/Linux，使用 apt 命令进行安装。

[jerry@Ubuntu]$ sudo apt-get update

[sudo] password for jerry:

[jerry@Ubuntu]$ sudo apt-get install subversion

[jerry@Ubuntu]$ svn --version

svn, version 1.7.5 (r1336830)

compiled Jun 21 2013, 22:11:49

## Apache 安装

我们已经看到如何将 SVN 客户端安装到 GNU/Linux 上，让我们看看如何创建一个新的版本库让使用者们访问。

我们必须必须在服务器上安装 Apache httpd 模块和 svnadmin 工具。subversion 从 /etc/httpd/conf.d/subversion.conf 读取配置文件， subversion.conf 看起来像这个样子

LoadModule dav\_svn\_module modules/mod\_dav\_svn.so

LoadModule authz\_svn\_module modules/mod\_authz\_svn.so

<Location /svn>

DAV svn

SVNParentPath /var/www/svn

AuthType Basic

AuthName "Authorization Realm"

AuthUserFile /etc/svn-users

Require valid-user

</Location>

让我们创建 Subversion 用户，授权他们访问版本库，htpasswd 命令用于创建和更新用来保存用户名和密码的纯文本文件给 HTTP 用户提供基本身份认证。-c 选项创建一个密码文件，如果密码文件已经存在了，它将会被覆盖。这就是为什么 -c 只在第一次使用。-m 选项用于设置是否启用 MD5 加密密码。

## 用户安装

让我们创建 tom

[root@CentOS ~]# htpasswd -cm /etc/svn-users tom

New password:

Re-type new password:

Adding password for user tom

让我们创建 jerry

[root@CentOS ~]# htpasswd -m /etc/svn-users jerry

New password:

Re-type new password:

Adding password for user jerry

[root@CentOS ~]#

创建一个 Subversion 父目录保存所有的工作，(/etc/httpd/conf.d/subversion.conf)。

[root@CentOS ~]# mkdir /var/www/svn

[root@CentOS ~]# cd /var/www/svn/

## 版本库安装

创建一个名为 project\_repo 的版本库。svnadmin 命令用于创建一个新的版本库和一些其他目录保存数据。

[root@CentOS svn]# svnadmin create project\_repo

[root@CentOS svn]# ls -l project\_repo

total 24

drwxr-xr-x. 2 root root 4096 Aug 4 22:30 conf

drwxr-sr-x. 6 root root 4096 Aug 4 22:30 db

-r--r--r--. 1 root root 2 Aug 4 22:30 format

drwxr-xr-x. 2 root root 4096 Aug 4 22:30 hooks

drwxr-xr-x. 2 root root 4096 Aug 4 22:30 locks

-rw-r--r--. 1 root root 229 Aug 4 22:30 README.txt

让我们更改版本库的用户和组所有权。

[root@CentOS svn]# chown -R apache.apache project\_repo/

检查是否启用SELinux或没有使用SELinux状态工具

[root@CentOS svn]# sestatus

SELinux status: enabled

SELinuxfs mount: /selinux

Current mode: enforcing

Mode from config file: enforcing

Policy version: 24

Policy from config file: targeted

如果SELinux启用了，我们必须更改安全的上下文。

[root@CentOS svn]# chcon -R -t httpd\_sys\_content\_t /var/www/svn/project\_repo/

如果允许通过 HTTP 进行提交，执行下面命令。

[root@CentOS svn]# chcon -R -t httpd\_sys\_rw\_content\_t /var/www/svn/project\_repo/

更改这些配置后，我们重启 Apache 服务器。

[root@CentOS svn]# service httpd restart

Stopping httpd: [FAILED]

Starting httpd: httpd: apr\_sockaddr\_info\_get() failed for CentOS

httpd: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1 for ServerName

[ OK ]

[root@CentOS svn]# service httpd status

httpd (pid 1372) is running...

[root@CentOS svn]#

我们已经成功配置好了 Apache 服务器，现在我们将配置版本库，使用默认的授权文件给可信的用户访问，添加下列几行到 roject\_repo/conf/svnserve.conf 文件。

anon-access = none

authz-db = authz

照惯例，每个 SVN 项目都有主干，标签，分支在项目的 root 目录。

主干是主要开发和经常被开发者们查看的目录。

分支目录用于追求不同的开发方向。

让我们在项目版本库底下创建主干，标签，分支结构。

[root@CentOS svn]# mkdir /tmp/svn-template

[root@CentOS svn]# mkdir /tmp/svn-template/trunk

[root@CentOS svn]# mkdir /tmp/svn-template/branches

[root@CentOS svn]# mkdir /tmp/svn-template/tags

现在从 /tmp/svn-template 导入这些文件目录。

[root@CentOS svn]# svn import -m 'Create trunk, branches, tags directory structure' /tmp/svn-template/

Adding /tmp/svn-template/trunk

Adding /tmp/svn-template/branches

Adding /tmp/svn-template/tags

Committed revision 1.

[root@CentOS svn]#

完成了！我们已经成功创建版本库并允许 Tom 和 Jerry 访问，从现在开始他们可以所有版本库支持的操作了。

# SVN 生命周期

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:47

## SVN 生命周期

本章讨论了版本控制系统的生命周期。在后面的章节中，我们将会介绍每个操作对应的 SVN 命令。

## 创建版本库

版本库相当于一个集中的空间，用于存放开发者所有的工作成果。版本库不仅能存放文件，还包括了每次修改的历史，即每个文件的变动历史。

Create 操作是用来创建一个新的版本库。大多数情况下这个操作只会执行一次。当你创建一个新的版本库的时候，你的版本控制系统会让你提供一些信息来标识版本库，例如创建的位置和版本库的名字。

## 检出

Checkout 操作是用来从版本库创建一个工作副本。工作副本是开发者私人的工作空间，可以进行内容的修改，然后提交到版本库中。

## 更新

顾名思义，update 操作是用来更新版本库的。这个操作将工作副本与版本库进行同步。由于版本库是由整个团队共用的，当其他人提交了他们的改动之后，你的工作副本就会过期。

让我们假设 Tom 和 Jerry 是一个项目的两个开发者。他们同时从版本库中检出了最新的版本并开始工作。此时，工作副本是与版本库完全同步的。然后，Jerry 很高效的完成了他的工作并提交了更改到版本库中。

此时 Tom 的工作副本就过期了。更新操作将会从版本库中拉取 Jerry 的最新改动并将 Tom 的工作副本进行更新。

## 执行变更

当检出之后，你就可以做很多操作来执行变更。编辑是最常用的操作。你可以编辑已存在的文件来，例如进行文件的添加/删除操作。

你可以添加文件/目录。但是这些添加的文件目录不会立刻成为版本库的一部分，而是被添加进待变更列表中，直到执行了 commit 操作后才会成为版本库的一部分。

同样地你可以删除文件/目录。删除操作立刻将文件从工作副本中删除掉，但该文件的实际删除只是被添加到了待变更列表中，直到执行了 commit 操作后才会真正删除。

Rename 操作可以更改文件/目录的名字。“移动”操作用来将文件/目录从一处移动到版本库中的另一处。

## 复查变化

当你检出工作副本或者更新工作副本后，你的工作副本就跟版本库完全同步了。但是当你对工作副本进行一些修改之后，你的工作副本会比版本库要新。在 commit 操作之前复查下你的修改是一个很好的习惯。

Status 操作列出了工作副本中所进行的变动。正如我们之前提到的，你对工作副本的任何改动都会成为待变更列表的一部分。Status 操作就是用来查看这个待变更列表。

Status 操作只是提供了一个变动列表，但并不提供变动的详细信息。你可以用 diff 操作来查看这些变动的详细信息。

## 修复错误

我们来假设你对工作副本做了许多修改，但是现在你不想要这些修改了，这时候 revert 操作将会帮助你。

Revert 操作重置了对工作副本的修改。它可以重置一个或多个文件/目录。当然它也可以重置整个工作副本。在这种情况下，revert 操作将会销毁待变更列表并将工作副本恢复到原始状态。

## 解决冲突

合并的时候可能会发生冲突。Merge 操作会自动处理可以安全合并的东西。其它的会被当做冲突。例如，“hello.c” 文件在一个分支上被修改，在另一个分支上被删除了。这种情况就需要人为处理。Resolve 操作就是用来帮助用户找出冲突并告诉版本库如何处理这些冲突。

## 提交更改

Commit 操作是用来将更改从工作副本到版本库。这个操作会修改版本库的内容，其它开发者可以通过更新他们的工作副本来查看这些修改。

在提交之前，你必须将文件/目录添加到待变更列表中。列表中记录了将会被提交的改动。当提交的时候，我们通常会提供一个注释来说明为什么会进行这些改动。这个注释也会成为版本库历史记录的一部分。Commit 是一个原子操作，也就是说要么完全提交成功，要么失败回滚。用户不会看到成功提交一半的情况。

# SVN 检出过程

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:48

## SVN 检出过程

SVN提供了 checkout 命令来从版本库检出一个工作副本。下面的命令将会在当前工作副本中新建一个名为 project\_repo 的文件夹。不用担心版本库的 URL 地址是什么，大部分时间里，SVN 管理员会提供给你地址和访问权限的。

[tom@CentOS ~]$ svn checkout http://svn.server.com/svn/project\_repo --username=tom

以上命令将产生如下结果：

A project\_repo/trunk

A project\_repo/branches

A project\_repo/tags

Checked out revision 1.

每一次成功提交之后，修订版本号都会显示出来。如果你想查看更多关于版本库的信息，执行 info 命令。

[tom@CentOS trunk]$ pwd

/home/tom/project\_repo/trunk

[tom@CentOS trunk]$ svn info

以上命令将产生如下结果：

Path: .

URL: http://svn.server.com/svn/project\_repo/trunk

Repository Root: http://svn.server.com/svn/project\_repo

Repository UUID: 7ceef8cb-3799-40dd-a067-c216ec2e5247

Revision: 1

Node Kind: directory

Schedule: normal

Last Changed Author: jerry

Last Changed Rev: 0

Last Changed Date: 2013-08-24 18:15:52 +0530 (Sat, 24 Aug 2013)

[tom@CentOS trunk]$

# SVN 执行修改

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:48

## SVN 执行修改

Jerry 从版本库检出了最新的版本并开始在项目上工作。他在 trunk 目录下创建了一个 array.c 文件。

[jerry@CentOS ~]$ cd project\_repo/trunk/

[jerry@CentOS trunk]$ cat array.c

以上命令将产生如下结果：

#include <stdio.h>

#define MAX 16

int main(void) {

int i, n, arr[MAX];

printf("Enter the total number of elements: ");

scanf("%d", &n);

printf("Enter the elements\n");

for (i = 0; i < n; ++i) scanf("%d", &arr[i]);

printf("Array has following elements\n");

for (i = 0; i < n; ++i) printf("|%d| ", arr[i]);

printf("\n");

return 0;

}

他想在提交之前测试他的代码。

[jerry@CentOS trunk]$ make array

cc array.c -o array

[jerry@CentOS trunk]$ ./array

Enter the total number of elements: 5

Enter the elements

1

2

3

4

5

Array has following elements

|1| |2| |3| |4| |5|

他编译并测试了代码，一切正常，现在是时候提交更改了。

[jerry@CentOS trunk]$ svn status

? array.c

? array

SVN显示在文件名前显示“?”，因为它不知道如何处理这些文件。

在提交之前，Jerry 需要将文件添加到待变更列表中。

[jerry@CentOS trunk]$ svn add array.c

A array.c

现在让我们来用 status 命令来检查它。SVN在 array.c 文件前面显示了一个 **A**，它意味着这个文件已经被成功地添加到了待变更列表中。

[jerry@CentOS trunk]$ svn status

? array

A array.c

为了把 array.c 存储到版本库中，使用 commit -m 加上注释信息来提交。如果你忽略了 -m 选项， SVN会打开一个可以输入多行的文本编辑器来让你输入提交信息。

[jerry@CentOS trunk]$ svn commit -m "Initial commit"

Adding trunk/array.c

Transmitting file data .

Committed revision 2.

现在 array.c 被成功地添加到了版本库中，并且修订版本号增加了1。

# SVN 检查更改

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:49

## SVN 检查更改

Jerry 往仓库里添加了一个叫做 array.c 的文件。 Tom 签出最后一个版本后开始工作。

[tom@CentOS ~]$ svn co http://svn.server.com/svn/project\_repo --username=tom

上面的命令将会产生下面的效果

A project\_repo/trunk

A project\_repo/trunk/array.c

A project\_repo/branches

A project\_repo/tags

Checked out revision 2.

但是，他发现有人已经添加了代码，他很好奇是谁添加的，于是他用下面的命令检查 log 信息：

[tom@CentOS trunk]$ svn log

上面的命令将会产生下面的效果

------------------------------------------------------------------------

r2 | jerry | 2013-08-17 20:40:43 +0530 (Sat, 17 Aug 2013) | 1 line

Initial commit

------------------------------------------------------------------------

r1 | jerry | 2013-08-04 23:43:08 +0530 (Sun, 04 Aug 2013) | 1 line

Create trunk, branches, tags directory structure

------------------------------------------------------------------------

当 Tom 查看 Jerry 的代码时，他注意到了里面的一个 bug 。 Jerry 没有检查数组溢出，这会导致很严重的问题。所以 Tom 决定修复这个问题。在修改之后， array.c 将会是这个样子。

#include <stdio.h>

#define MAX 16

int main(void)

{

int i, n, arr[MAX];

printf("Enter the total number of elements: ");

scanf("%d", &n);

/\* handle array overflow condition \*/

if (n > MAX) {

fprintf(stderr, "Number of elements must be less than %d\n", MAX);

return 1;

}

printf("Enter the elements\n");

for (i = 0; i < n; ++i)

scanf("%d", &arr[i]);

printf("Array has following elements\n");

for (i = 0; i < n; ++i)

printf("|%d| ", arr[i]);

printf("\n");

return 0;

}

Tom 想使用 status 操作来看看将要生效的更改列表

[tom@CentOS trunk]$ svn status

M array.c

array.c 文件已经被修改，Subversion 会在修改过的文件前面加一个字母 M 。接下来 Tom 编译测试了他的代码，并且工作良好。在提交更改前，他想要再次检查他的更改。

[tom@CentOS trunk]$ svn diff

Index: array.c

===================================================================

--- array.c (revision 2)

+++ array.c (working copy)

@@ -9,6 +9,11 @@

printf("Enter the total number of elements: ");

scanf("%d", &n);

+ if (n > MAX) {

+ fprintf(stderr, "Number of elements must be less than %d\n", MAX);

+ return 1;

+ }

+

printf("Enter the elements\n");

for (i = 0; i < n; ++i)

Tom 在 array.c 文件中添加了几行代码,Subversion 会在新添加的这几行代码前面添加 ＋ 号标记，现在，他已经准备好提交他的代码。

[tom@CentOS trunk]$ svn commit -m "Fix array overflow problem"

上面的命令将会产生下面的效果

Sending trunk/array.c

Transmitting file data .

Committed revision 3.

Tom 的更改被成功得提交到了仓库中。

# SVN 更新过程

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:49

## SVN 更新过程

Jerry 提交了他第一个版本的代码. 但是他想他应该写两个函数用来接收输入和显示数组，在修改之后， array.c 看起来像是下面这样。

#include <stdio.h>

#define MAX 16

void accept\_input(int \*arr, int n) {

int i;

for (i = 0; i < n; ++i)

scanf("%d", &arr[i]);

}

void display(int \*arr, int n) {

int i;

for (i = 0; i < n; ++i)

printf("|%d| ", arr[i]);

printf("\n");

}

int main(void) {

int i, n, arr[MAX];

printf("Enter the total number of elements: ");

scanf("%d", &n);

printf("Enter the elements\n");

accept\_input(arr, n);

printf("Array has following elements\n");

display(arr, n);

return 0;

}

Jerry 编译和测试了他的代码，现在他准备提交他的更改。在此之前，他想要用下面的命令查看更改。

[jerry@CentOS trunk]$ svn diff

上面的命令将会产生下面的效果

Index: array.c

===================================================================

--- array.c (revision 2)

+++ array.c (working copy)

@@ -2,6 +2,24 @@

#define MAX 16

+void accept\_input(int \*arr, int n)

+{

+ int i;

+

+ for (i = 0; i & n; ++i)

+ scanf("%d", &arr[i]);

+}

+

+void display(int \*arr, int n)

+{

+ int i;

+

+ for (i = 0; i < n; ++i)

+ printf("|%d| ", arr[i]);

+

+ printf("\n");

+}

+

int main(void)

{

int i, n, arr[MAX];

@@ -10,15 +28,10 @@

scanf("%d", &n);

printf("Enter the elements\n");

+ accept\_input(arr, n);

- for (i = 0; i < n; ++i)

- scanf("%d", &arr[i]);

-

printf("Array has following elements\n");

- for (i = 0; i < n; ++i)

- printf("|%d| ", arr[i]);

-

- printf("\n");

+ display(arr, n);

return 0;

}

对于新增加的行， Subversion 在前面加上了 ＋ 号，并且用 - 号标记了删除掉的行。现在， Jerry 尝试使用下面的命令来提交他的更改：

[jerry@CentOS trunk]$ svn commit -m "Add function to accept input and to display array contents"

上面的命令将会产生下面的效果

Sending trunk/array.c

svn: Commit failed (details follow):

svn: File or directory 'array.c' is out of date; try updating

svn: resource out of date; try updating

Subversion 不会允许 Jerry 提交他的更改，因为 Tom 已经修改了仓库，所以 Jerry 的工作副本已经失效。为了避免两人的代码被互相覆盖，Subversion 不允许他进行这样的操作。Jerry 在提交他的更改之前必须先更新工作副本。所以他使用了 update 命令，如下：

[jerry@CentOS trunk]$ svn update

G array.c

Updated to revision 3.

Subversion 在这个文件前面加上了字母 G 标记, 这意味着这个文件是被合并过的。

[jerry@CentOS trunk]$ svn diff

上面的命令将会产生下面的效果

Index: array.c

===================================================================

--- array.c (revision 3)

+++ array.c (working copy)

@@ -2,6 +2,24 @@

#define MAX 16

+void accept\_input(int \*arr, int n)

+{

+ int i;

+

+ for (i = 0; i < n; ++i)

+ scanf("%d", &arr[i]);

+}

+

+void display(int \*arr, int n)

+{

+ int i;

+

+ for (i = 0; i < n; ++i)

+ printf("|%d| ", arr[i]);

+

+ printf("\n");

+}

+

int main(void)

{

int i, n, arr[MAX];

@@ -15,15 +33,10 @@

}

printf("Enter the elements\n");

+ accept\_input(arr, n);

- for (i = 0; i < n; ++i)

- scanf("%d", &arr[i]);

-

printf("Array has following elements\n");

- for (i = 0; i < n; ++i)

- printf("|%d| ", arr[i]);

-

- printf("\n");

+ display(arr, n);

return 0;

}

Subversion 只展示出了 Jerry 的更改，但是 array.c 文件被合并了。如果你仔细观察，Subversion 现在展示的版本号是3。在之前的输出中，它展示的版本号是2。只是展示出了谁对其进行了更改和更改的目的。

jerry@CentOS trunk]$ svn log

------------------------------------------------------------------------

r3 | tom | 2013-08-18 20:21:50 +0530 (Sun, 18 Aug 2013) | 1 line

Fix array overflow problem

------------------------------------------------------------------------

r2 | jerry | 2013-08-17 20:40:43 +0530 (Sat, 17 Aug 2013) | 1 line

Initial commit

------------------------------------------------------------------------

r1 | jerry | 2013-08-04 23:43:08 +0530 (Sun, 04 Aug 2013) | 1 line

Create trunk, branches, tags directory structure

------------------------------------------------------------------------

现在 Jerry 的工作目录是和仓库同步的，他现在可以安全地提交更改了。

[jerry@CentOS trunk]$ svn commit -m "Add function to accept input and to display array contents"

Sending trunk/array.c

Transmitting file data .

Committed revision 4.

# SVN 修复错误

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:51

## SVN 修复错误

假设 Jerry 意外地更改了 array.c 文件而导致编译错误，他想放弃修改。在这种状况下，‘revert’ 操作将派上用场。revert 操作将撤销任何文件或目录里的局部更改。

[jerry@CentOS trunk]$ svn status

上面的命令将会产生下面的效果

M array.c

让我们尝试创建一个数组，如下：

[jerry@CentOS trunk]$ make array

上面的命令将会产生下面的效果

cc array.c -o array

array.c: In function ‘main’:

array.c:26: error: ‘n’ undeclared (first use in this function)

array.c:26: error: (Each undeclared identifier is reported only once

array.c:26: error: for each function it appears in.)

array.c:34: error: ‘arr’ undeclared (first use in this function)

make: \*\*\* [array] Error 1

Jerry 在 array.c 文件里执行了‘revert’操作。

[jerry@CentOS trunk]$ svn revert array.c

Reverted 'array.c'

[jerry@CentOS trunk]$ svn status

[jerry@CentOS trunk]$

现在开始编译代码。

[jerry@CentOS trunk]$ make array

cc array.c -o array

进行 revert 操作之后，他的文件恢复了原始的状态。 revert 操作不单单可以使单个文件恢复原状，而且可以使整个目录恢复原状。恢复目录用 －R 命令，如下。

[jerry@CentOS project\_repo]$ pwd

/home/jerry/project\_repo

[jerry@CentOS project\_repo]$ svn revert -R trunk

现在，我们已经知道如何撤销更改。但是，假使你想恢复一个已经提交的版本怎么办！Version Control System 工具不允许删除仓库的历史纪录。为了消除一个旧版本，我们必须撤销旧版本里的所有更改然后提交一个新版本。这种操作叫做 reverse merge。

假设 Jerry 添加了一段线性搜索操作的代码，核查之后，他提交了更改。

[jerry@CentOS trunk]$ svn diff

Index: array.c

===================================================================

--- array.c (revision 21)

+++ array.c (working copy)

@@ -2,6 +2,16 @@

#define MAX 16

+int linear\_search(int \*arr, int n, int key)

+{

+ int i;

+

+ for (i = 0; i < n; ++i)

+ if (arr[i] == key)

+ return i;

+ return -1;

+}

+

void bubble\_sort(int \*arr, int n)

{

int i, j, temp, flag = 1;

[jerry@CentOS trunk]$ svn status

? array

M array.c

[jerry@CentOS trunk]$ svn commit -m "Added code for linear search"

Sending trunk/array.c

Transmitting file data .

Committed revision 22.

Jerry 很好奇 Tom 以前写的代码。所以他检查了 Subversion 的 log 信息。

[jerry@CentOS trunk]$ svn log

上面的命令将会产生下面的效果

------------------------------------------------------------------------

r5 | tom | 2013-08-24 17:15:28 +0530 (Sat, 24 Aug 2013) | 1 line

Add binary search operation

------------------------------------------------------------------------

r4 | jerry | 2013-08-18 20:43:25 +0530 (Sun, 18 Aug 2013) | 1 line

Add function to accept input and to display array contents

查看 log 信息之后，Jerry 意识到他犯了个严重的错误。因为 Tom 已经写了比线性搜索更好的二分法搜索，Jerry 发现自己的代码很冗余，他决定撤销之前对版本的修改。首先，找到仓库的当前版本，现在是版本 22，我们要撤销回之前的版本，比如版本 21。

[jerry@CentOS trunk]$ svn up

At revision 22.

[jerry@CentOS trunk]$ svn merge -r 22:21 array.c

--- Reverse-merging r22 into 'array.c':

U array.c

[jerry@CentOS trunk]$ svn commit -m "Reverted to revision 21"

Sending trunk/array.c

Transmitting file data .

Committed revision 23.

# SVN 解决冲突

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:51

## SVN 解决冲突

Tom决定给他的工程添加一个 README 文件，于是他创建了这个文件并在其中添加了 TODO 列表。添加完成之后，该文件的存放处位于 revision 6.

[tom@CentOS trunk]$ cat README

/\* TODO: Add contents in README file \*/

[tom@CentOS trunk]$ svn status

? README

[tom@CentOS trunk]$ svn add README

A README

[tom@CentOS trunk]$ svn commit -m "Added README file. Will update it's content in future."

Adding trunk/README

Transmitting file data .

Committed revision 6.

Jerry 检出了位于 revision 6 最后的代码，并且他直接立刻开始了工作。几个小时以后，Tom 更新了 README 文件并且提交了他所修改的地方。修改的 README 将会看上去像这个样子。

[tom@CentOS trunk]$ cat README

\* Supported operations:

1) Accept input

2) Display array elements

[tom@CentOS trunk]$ svn status

M README

[tom@CentOS trunk]$ svn commit -m "Added supported operation in README"

Sending trunk/README

Transmitting file data .

Committed revision 7.

现在，仓库位于修改版本 7，并且 Jerry 的工作副本已经过期。Jerry 也更新 README 文件并且试图提交他的更改。

Jerry 的 README 文件将会看上去像这个样子：

[jerry@CentOS trunk]$ cat README

\* File list

1) array.c Implementation of array operation.

2) README Instructions for user.

[jerry@CentOS trunk]$ svn status

M README

[jerry@CentOS trunk]$ svn commit -m "Updated README"

Sending trunk/README

svn: Commit failed (details follow):

svn: File or directory 'README' is out of date; try updating

svn: resource out of date; try updating

## 第一步：视图冲突

Subversion 已经检测出 README 自上次更新后文件已经更改。所以，Jerry 必须更新他的工作副本。

[jerry@CentOS trunk]$ svn up

Conflict discovered in 'README'.

Select: (p) postpone, (df) diff-full, (e) edit,

(mc) mine-conflict, (tc) theirs-conflict,

(s) show all options:

Subversion 提示说有一个冲突在 README 文件，并且 Subversion 并不知道如何解决这个问题。于是 Jerry 选择 df 选项来检查冲突。

[jerry@CentOS trunk]$ svn up

Conflict discovered in 'README'.

Select: (p) postpone, (df) diff-full, (e) edit,

(mc) mine-conflict, (tc) theirs-conflict,

(s) show all options: df

--- .svn/text-base/README.svn-base Sat Aug 24 18:07:13 2013

+++ .svn/tmp/README.tmp Sat Aug 24 18:13:03 2013

@@ -1 +1,11 @@

-/\* TODO: Add contents in README file \*/

+<<<<<<< .mine

+\* File list

+

+1) array.c Implementation of array operation.

+2) README Instructions for user.

+=======

+\* Supported operations:

+

+1) Accept input

+2) Display array elements

+>>>>>>> .r7

Select: (p) postpone, (df) diff-full, (e) edit, (r) resolved,

(mc) mine-conflict, (tc) theirs-conflict,

(s) show all options:

## 第二步：推迟冲突

接下来 Jerry 用 postpone（p） 来解决冲突。

Select: (p) postpone, (df) diff-full, (e) edit, (r) resolved,

(mc) mine-conflict, (tc) theirs-conflict,

(s) show all options: p

C README

Updated to revision 7.

Summary of conflicts:

Text conflicts: 1

在用文档编辑器打开 README 文件后，Jerry 意识到 Subversion 已经包含了 Tom 的代码和他的代码，并被冲突标示包裹了起来。

[jerry@CentOS trunk]$ cat README

<<<<<<< .min

\* File list

1) array.c Implementation of array operation.

2) README Instructions for user.

=======

\* Supported operations:

1) Accept input

2) Display array elements

>>>>>>> .r7

Jerry 想让 Tom 的更改跟他的保持一致，所以他决定移除包含冲突标识的行。

所以，更改后的 README 将会是这个样子。

[jerry@CentOS trunk]$ cat README

\* File list

1) array.c Implementation of array operation.

2) README Instructions for user.

\* Supported operations:

1) Accept input

2) Display array elements

Jerry 解决了冲突并试图再次提交。

[jerry@CentOS trunk]$ svn commit -m "Updated README"

svn: Commit failed (details follow):

svn: Aborting commit: '/home/jerry/project\_repo/trunk/README' remains in conflict

[jerry@CentOS trunk]$ svn status

? README.r6

? README.r7

? README.mine

C README

## 第三步：解决冲突

在上面的提交中，字母 C 指示说有一个冲突在 README 文件。Jerry 解决了冲突但并没有告诉 Subversion 已经解决了冲突。 他使用了 resolve 命令通知 Subversion 冲突的解决。

[jerry@CentOS trunk]$ svn resolve --accept=working README

Resolved conflicted state of 'README'

[jerry@CentOS trunk]$ svn status

M README

[jerry@CentOS trunk]$ svn commit -m "Updated README"

Sending trunk/README

Transmitting file data .

Committed revision 8.

# SVN 标签

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:52

## SVN 标签

版本管理系统支持tag选项，通过使用tag的概念，我们可以给某一个具体版本的代码一个更加有意义的名字。标签允许给某一个具体版本的代码一个描述性强，难忘的名字。举个例子：BASIC\_ARRAY\_OPERATIONS 就比修改版本 7更有意义。

让我们来看一个 tag 标签的例子。Tom为了能更好的审查代码，决定创建一个tag。

[tom@CentOS project\_repo]$ svn copy --revision=4 trunk/ tags/basic\_array\_operations

上面的命令将会产生出下面的结果。

A tags/basic\_array\_operations/array.c

Updated to revision 4.

A tags/basic\_array\_operations

上面的代码成功完成，新的目录将会被创建在 tags 目录下。

[tom@CentOS project\_repo]$ ls -l tags/

total 4

drwxrwxr-x. 3 tom tom 4096 Aug 24 18:18 basic\_array\_operations

Tom想要在提交前双击。状态选项显示 tag 选项成功，所以他可以安全的提交他的更改。

[tom@CentOS project\_repo]$ svn status

A + tags/basic\_array\_operations

[tom@CentOS project\_repo]$ svn commit -m "Created tag for basic array operations"

Adding tags/basic\_array\_operations

Committed revision 5.

# SVN 分支

由 珍珍阿姨 创建， 最后一次修改 2016-08-12 21:14:52

## SVN 分支

Branch 选项会给开发者创建出另外一条线路。当有人希望开发进程分开成两条不同的线路时，这个选项会非常有用。我们先假设你已经发布了一个产品的 1.0 版本，你可能想创建一个新的分支，这样就可以不干扰到 1.0 版本的bug修复的同时，又可以开发2.0版本。

在这一节，我们将看到如何创建，穿过和合并分支。Jerry 因为代码冲突的事情不开心，所以他决定创建一个新的私有分支。

[jerry@CentOS project\_repo]$ ls

branches tags trunk

[jerry@CentOS project\_repo]$ svn copy trunk branches/jerry\_branch

A branches/jerry\_branch

[jerry@CentOS project\_repo]$ svn status

A + branches/jerry\_branch

[jerry@CentOS project\_repo]$ svn commit -m "Jerry's private branch"

Adding branches/jerry\_branch

Adding branches/jerry\_branch/README

Committed revision 9.

[jerry@CentOS project\_repo]$

现在 Jerry 在自己的分支下开始工作。他给序列添加了 sort 选项。Jerry 修改后的代码如下：

[jerry@CentOS project\_repo]$ cd branches/jerry\_branch/

[jerry@CentOS jerry\_branch]$ cat array.c

上面的代码将会产生下面的结果：

#include <stdio.h>

#define MAX 16

void bubble\_sort(int \*arr, int n)

{

int i, j, temp, flag = 1;

for (i = 1; i < n && flag == 1; ++i) {

flag = 0;

for (j = 0; j < n - i; ++j) {

if (arr[j] > arr[j + 1]) {

flag = 1;

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

void accept\_input(int \*arr, int n)

{

int i;

for (i = 0; i < n; ++i)

scanf("%d", &arr[i]);

}

void display(int \*arr, int n)

{

int i;

for (i = 0; i < n; ++i)

printf("|%d| ", arr[i]);

printf("\n");

}

int main(void)

{

int i, n, key, ret, arr[MAX];

printf("Enter the total number of elements: ");

scanf("%d", &n);

/\* Error handling for array overflow \*/

if (n >MAX) {

fprintf(stderr, "Number of elements must be less than %d\n", MAX);

return 1;

}

printf("Enter the elements\n");

accept\_input(arr, n);

printf("Array has following elements\n");

display(arr, n);

printf("Sorted data is\n");

bubble\_sort(arr, n);

display(arr, n);

return 0;

}

Jerry 编译并且测试了他的代码，准备提交他的更改。

[jerry@CentOS jerry\_branch]$ make array

cc array.c -o array

[jerry@CentOS jerry\_branch]$ ./array

上面的命令将会产生如下的结果：

Enter the total number of elements: 5

Enter the elements

10

-4

2

7

9

Array has following elements

|10| |-4| |2| |7| |9|

Sorted data is

|-4| |2| |7| |9| |10|

[jerry@CentOS jerry\_branch]$ svn status

? array

M array.c

[jerry@CentOS jerry\_branch]$ svn commit -m "Added sort operation"

Sending jerry\_branch/array.c

Transmitting file data .

Committed revision 10.

同时，越过主干，Tom 决定实现 search 选项。Tom 添加了 search 选项而添加代码，他的代码如下：

[tom@CentOS trunk]$ svn diff

上面的命令将会产生下面的结果：

Index: array.c

===================================================================

--- array.c (revision 10)

+++ array.c (working copy)

@@ -2,6 +2,27 @@

#define MAX 16

+int bin\_search(int \*arr, int n, int key)

+{

+ int low, high, mid;

+

+ low = 0;

+ high = n - 1;

+ mid = low + (high - low) / 2;

+

+ while (low <= high) {

+ if (arr[mid] == key)

+ return mid;

+ if (arr[mid] > key)

+ high = mid - 1;

+ else

+ low = mid + 1;

+ mid = low + (high - low) / 2;

+ }

+

+ return -1;

+}

+

void accept\_input(int \*arr, int n)

{

int i;

@@ -22,7 +43,7 @@

int main(void)

{

- int i, n, arr[MAX];

+ int i, n, ret, key, arr[MAX];

printf("Enter the total number of elements: ");

scanf("%d", &n);

@@ -39,5 +60,16 @@

printf("Array has following elements\n");

display(arr, n);

+ printf("Enter the element to be searched: ");

+ scanf("%d", &key);

+

+ ret = bin\_search(arr, n, key);

+ if (ret < 0) {

+ fprintf(stderr, "%d element not present in array\n", key);

+ return 1;

+ }

+

+ printf("%d element found at location %d\n", key, ret + 1);

+

return 0;

}

After reviewing, he commits his changes.

[tom@CentOS trunk]$ svn status

? array

M array.c

[tom@CentOS trunk]$ svn commit -m "Added search operation"

Sending trunk/array.c

Transmitting file data .

Committed revision 11.

但是 Tom 好奇 Jerry 在他自己的私有分支中干了什么：

[tom@CentOS trunk]$ cd ../branches/

[tom@CentOS branches]$ svn up

A jerry\_branch

A jerry\_branch/array.c

A jerry\_branch/README

[tom@CentOS branches]$ svn log

------------------------------------------------------------------------

r9 | jerry | 2013-08-27 21:56:51 +0530 (Tue, 27 Aug 2013) | 1 line

Added sort operation

------------------------------------------------------------------------

通过查看 Subversion 的 log 信息，Tom 发现 Jerry 依赖 ‘sort’ 选项。Tom 决定增添用折半查找，期望数据总是根据种类进行分类。但是如果用户提供的数据是没有进行分类呢？在那种情况下，折半查找将会失效。所以他决定接着 Jerry 的代码，在搜索选项前先进性分类。所以他告诉 Subversion 合并 Jerry 的分支到主干中去。

[tom@CentOS trunk]$ pwd

/home/tom/project\_repo/trunk

[tom@CentOS trunk]$ svn merge ../branches/jerry\_branch/

--- Merging r9 through r11 into '.':

U array.c

在融合后，array.c 会看上去是这个样子：

[tom@CentOS trunk]$ cat array.c

上面的代码将会产生下面的结果：

#include <stdio.h>

#define MAX 16

void bubble\_sort(int \*arr, int n)

{

int i, j, temp, flag = 1;

for (i = 1; i < n && flag == 1; ++i) {

flag = 0;

for (j = 0; j < n - i; ++j) {

if (arr[j] > arr[j + 1]) {

flag = 1;

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

int bin\_search(int \*arr, int n, int key)

{

int low, high, mid;

low = 0;

high = n - 1;

mid = low + (high - low) / 2;

while (low <= high) {

if (arr[mid] == key)

return mid;

if (arr[mid] > key)

high = mid - 1;

else

low = mid + 1;

mid = low + (high - low) / 2;

}

return -1;

}

void accept\_input(int \*arr, int n)

{

int i;

for (i = 0; i < n; ++i)

scanf("%d", &arr[i]);

}

void display(int \*arr, int n)

{

int i;

for (i = 0; i < n; ++i)

printf("|%d| ", arr[i]);

printf("\n");

}

int main(void)

{

int i, n, ret, key, arr[MAX];

printf("Enter the total number of elements: ");

scanf("%d", &n);

/\* Error handling for array overflow \*/

if (n > MAX) {

fprintf(stderr, "Number of elements must be less than %d\n", MAX);

return 1;

}

printf("Enter the elements\n");

accept\_input(arr, n);

printf("Array has following elements\n");

display(arr, n);

printf("Sorted data is\n");

bubble\_sort(arr, n);

display(arr, n);

printf("Enter the element to be searched: ");

scanf("%d", &key);

ret = bin\_search(arr, n, key);

if (ret < 0) {

fprintf(stderr, "%d element not present in array\n", key);

return 1;

}

printf("%d element found at location %d\n", key, ret + 1);

return 0;

}

经过编译和测试后，Tom 提交了他的更改到仓库。

[tom@CentOS trunk]$ make array

cc array.c -o array

[tom@CentOS trunk]$ ./array

Enter the total number of elements: 5

Enter the elements

10

-2

8

15

3

Array has following elements

|10| |-2| |8| |15| |3|

Sorted data is

|-2| |3| |8| |10| |15|

Enter the element to be searched: -2

-2 element found at location 1

[tom@CentOS trunk]$ svn commit -m "Merge changes from Jerry's code"

Sending trunk

Sending trunk/array.c

Transmitting file data .

Committed revision 12.

[tom@CentOS trunk]$