# 版本控制

## CVCS（集中式版本控制）和DVCS（分布式版本控制）各自的优缺点

CVCS的优点：管理方便，所有版本控制信息都记载在服务器上

CVCS的缺点：服务器发生故障时，将导致所有信息的丢失及不可恢复；而且在服务器不允许用户操作的前提下，用户信息将不可提交和更新

DVCS的优点：DVCS将会在每个用户下复制一个服务器仓库镜像到本地，有利于用户文件的更新和提交；并且服务器发生故障时将不会对其影响，有利于其恢复

DVCS的缺点：由于每个用户都对其服务器仓库进行镜像备份，这导致信息冗余，浪费磁盘空间。

## CVCS和DVCS的工作原理

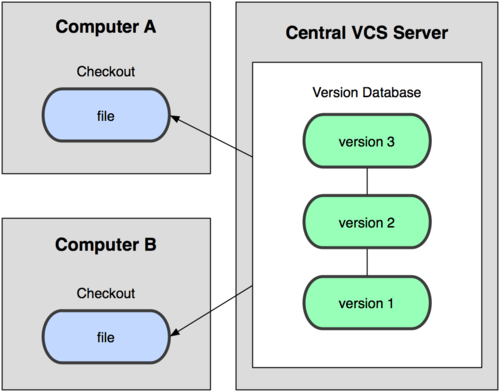


图1.1 、CVCS工作原理图



图1.2、DVCS的工作原理图

# Git的安装及配置

## 2.1、Git的配置文件简介

Git配置文件所在地：/etc/gitconfig和~/.gitconfig

/etc/gitconfig是全局的，所有用户共有

~/.gitconfig是局部的，只局限于本用户

## 2.2、Git用户名和邮件配置

用户名设置：git config [–global] user.name “用户名”

邮箱设置: git config [–global] user.email “邮箱”

其中输入—global表示配置在/etc/gotconfig文件中，否则表示配置在当前用户.gitconfig文件中

## 2.3、查看Git配置文件信息

查看git所有配置信息：git config –-list

查看git用户信息：git config user.name

# Git基本命令

## 3.1、Git仓库的创建或获取

创建Git仓库：git init

获取Git仓库：git clone [url]

## 3.2、Git更新、状态查看、差异、移动

检查当前文件状态：git status

将某个文件加入到暂存区：git add 文件名

提交更新：git commit

查看尚未暂存更新了哪部分：git diff

查看暂存和上次提交之间的差异：git diff –cached

将已跟踪过的文件一并提交，跳过git add步骤：git commit –a

移除git仓库里的文件：git rm 文件名

移除跟踪清单但不删除工作目录里的文件：git rm --cached 文件名

移动文件：git mv 源文件路径 目的文件路径

不理解的地方：

(1)、git commit -m “提交的描叙信息” 【已解决】

(2)、强制删除选项： –f怎么使用及其含义

(3)、忽略某些文件：.gitignore在哪个目录下

## 3.3、Git查看提交史

查看提交历史：git log

查看最近2次更新的内容差异：git log -p -2

git log 命令支持的选项:

| 选项 | 说明 |
| --- | --- |
| -p | 按补丁格式显示每个更新之间的差异。 |
| --word-diff | 按 word diff 格式显示差异。 |
| --stat | 显示每次更新的文件修改统计信息。 |
| --shortstat | 只显示 --stat 中最后的行数修改添加移除统计。 |
| --name-only | 仅在提交信息后显示已修改的文件清单。 |
| --name-status | 显示新增、修改、删除的文件清单。 |
| --abbrev-commit | 仅显示 SHA-1 的前几个字符，而非所有的 40 个字符。 |
| --relative-date | 使用较短的相对时间显示（比如，“2 weeks ago”）。 |
| --graph | 显示 ASCII 图形表示的分支合并历史。 |
| --pretty | 使用其他格式显示历史提交信息。可用的选项包括 oneline，short，full，fuller 和 format（后跟指定格式）。 |
| --oneline | --pretty=oneline --abbrev-commit 的简化用法。 |

git log 命令支持的选项掌握的模糊

## 3.3、Git撤销

撤销最后一次并重新提交：git commit --amend

取消已经暂存的文件：git reset HEAD 文件名

取消对文件的修改：git checkout -- 文件名

## 3.4、Git远程库的使用（增、删、改、查）

克隆远程仓库：git clone git://地址

查看当前远程库[对应的克隆地址]: git remote [-v]

添加远程库并为其起别名：git remote add [自定义远程库名] [git://地址]

从远程库中抓取数据：git fetch 别名|git://地址

将数据推送到远程库：git push 远程库名 文件名

把本地的 master 分支推送到origin 服务器上：git push origin master

查看远程仓库信息：git remote show 远程库名

远程库的重命名：git remote rename 原名 新名

## 3.5、Git标签

查看已有标签：git tag

标签过滤：git tag -l “v1.4.3.\*”

标签的类别没有掌握

# 四、Git分支

## 4.1、Git分支创建、切换

创建Git分支：git branch 分支名

切换到其他分支：git checkout 分支名

创建Git分支同时切换到改分支：git branch -b 文件名

合并当前分支与A分支并跳转到A：git merge 分支名A

删除分支名：git branch -d 分知名

## 4.2、Git分支查看、管理

查看所有分支：git branch

查看最后一个提交对象的信息：git branch -v

查看哪些分支已并入当前分支：git branch --merged

查看那些分支尚未合并：git branch --no-merged

## 4.3、远程分支更新、推送

远程分支的表示方法：远程仓库名/分支名

远程分支更新到现所处分支：git fetch 远程分支名

推送本地分支：git push 远程仓库名 分支名

新建分支A跟踪远程分支B：git checkout -b 分支名A 远程仓库名/分支名B

【或】git --track 远程仓库名/分支名

创建一个叫做“feature\_x”的分支，并切换过去：

git checkout -b feature\_x

切换回主分支：

git checkout master

再把新建的分支删掉：

git branch -d feature\_x

除非你将分支推送到远端仓库，不然该分支就是 不为他人所见的：

git push origin <branch>

要更新你的本地仓库至最新改动，执行：

git pull

以在你的工作目录中 获取（fetch） 并 合并（merge） 远端的改动

# 未解决的问题

（1）、不太理解有冲突分支的合并

# Git 极简教程

http://rogerdudler.github.io/git-guide/index.zh.html

# 忽略某些文件.gitignore

在仓库下新建.gitignore文件

内容如下：

### Windows ###

# Windows image file caches

Thumbs.db

ehthumbs.db

# Folder config file

Desktop.ini

# Recycle Bin used on file shares

$RECYCLE.BIN/

# Windows Installer files

\*.cab

\*.msi

\*.msm

\*.msp

### Eclipse ###

\*.pydevproject

.metadata

.gradle

bin/

tmp/

\*.tmp

\*.bak

\*.swp

\*~.nib

local.properties

.settings/

.loadpath

# External tool builders

.externalToolBuilders/

# Locally stored "Eclipse launch configurations"

\*.launch

# CDT-specific

.cproject

# PDT-specific

.buildpath

# sbteclipse plugin

.target

# TeXlipse plugin

.texlipse

### Java ###

\*.class

# Mobile Tools for Java (J2ME)

.mtj.tmp/

# Package Files #

\*.jar

\*.war

\*.ear

# virtual machine crash logs, see http://www.java.com/en/download/help/error\_hotspot.xml

hs\_err\_pid\*

### Maven ###

target/

pom.xml.tag

pom.xml.releaseBackup

pom.xml.versionsBackup

pom.xml.next

release.properties

/Servers