每章开头给出了克隆相应版本库的命令

版本控制系统好比银行保险箱

“\”-换行连接符

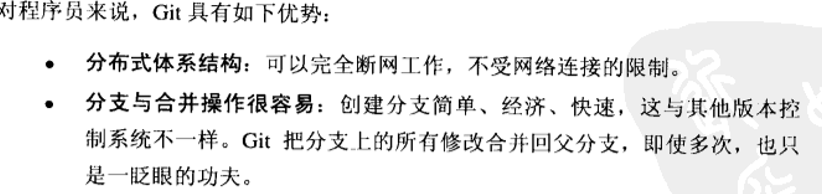
SVN——subversion 开源代码的版本控制系统 软件

Git

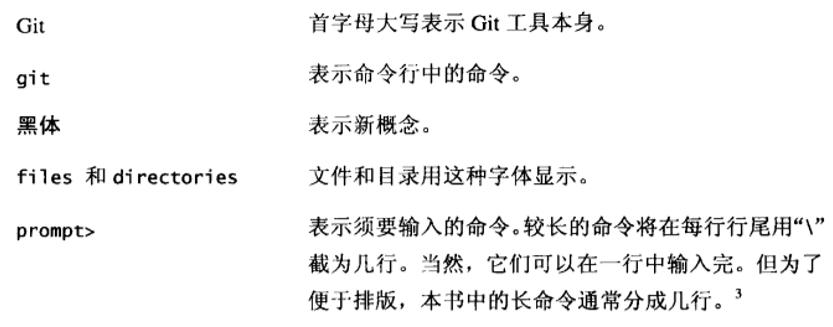
DVCS——Distribute Version Control System 分布式版本控制系统

VCS——version control System 版本控制系统

CVS——Concurrent Version System （concurrent 并发协作一致）C/S代码版本控制软件







全书结构

第一篇:介绍版本控制工具Hello Git

第二篇:展开介绍Git

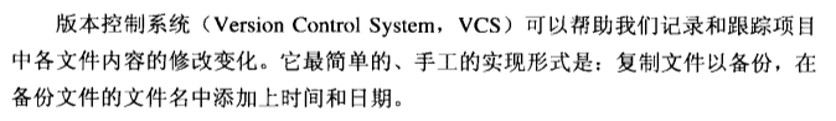
1. Git基础
2. 理解使用分支
3. 查询Git历史记录 第六章
4. 与远程版本库协作 第七章 P91
5. 技巧
6. 高阶功能 第九章 P115

第三篇：系统管理

1. 迁移
2. 使用Gitosis管理Git服务器

第四篇：附录

1. 快速查阅常用命令及其使用方法



直接访问版本控制系统》》集中式》》分布式版本控制系统

分布式版本控制系统 每个人都会在本地有自己的版本库，所有历史记录都在本地的版本库中，提交代码在本地库中。

上传到版本主库：通过Git推入

生成少量修改的补丁包，补丁包由维护人员跟新版本库。

Git中版本库不在服务器中，存储在本地Working Tree 的.git目录

.git目录的父目录是工作目录树

上游版本库（upstream repository）共享资源。

版本库——版本库中存储所有内容存储在本地.git目录中。

* 版本库断面视图 工作目录树 《《 目录文件

工作目录树Working Tree（工作拷贝 Working Copy）——程序员开发的地方，包括项目需要的所有文件

检出——更新工作目录树

1.4代码修改与文件同步

* + 修改内容后必须进行单元测试一确保修改不会产生负面影响
  + 每次修改会新增版本，对日志信息进行改动
  + Push 上游版本库（upstream repository）使其他人共享
    - Push 把自己版本库推入到另一个版本库中。

1.5 跟踪项目、目录和文件

* 众多版本控制工具以文件为单位存储，Git以记录追踪整个文件的内容。例如.py文件中的函数。
  + 降低存储历史版本所需空间。

Git支持所有项目放在一个版本库中，每个项目一个目录，公用一个版本树。

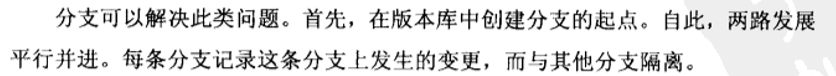
每个版本库中只存储一个项目。

树，目录——开发的地方 库存储source的地方

* 标签——记录下版本库在特定历史时刻的断面视图，便于日后查找和恢复

帮助使用者跟踪版本的历史，标识库中难读难记的内部版本号。

* 分支branch——VCS提供跟踪并进行演进的不同轨迹的方法。



分支存在时长

分支可以长期存在——记录项目的不同发布版本开发。

存在数小时——项目改动。

合并——分支和分支Merge

Git比较各分支上的变化，确定变化的发生，将两个以上分支合并在一起。

变化在不同部分可以自动合并，否则提示冲突（confict）等待人工介入（或冲突处理）

合并跟踪（merge tracking）——记录跟踪哪些提交合并到一起

锁机制——分支合并控制机制

严格锁——一个文件一次只有一个人可以进行操作。

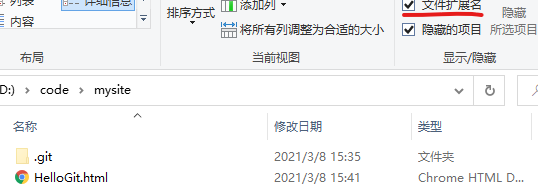
乐观锁——合并之前提取文件发生了修改需要后者把改动拖入检查是否发生conflict

Git命令 开始键入命令

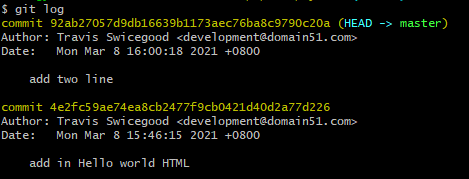
* git config命名配置一些参数
* 
* 
* 
* 

35min看完第二章开心

Git创建第一项目 （Git概览）

* 创建版本库
  + cd 目录名
  + git init
    - 
    - 

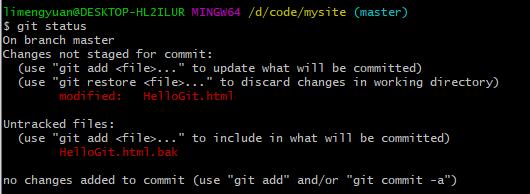
.git文件不可见解决办法

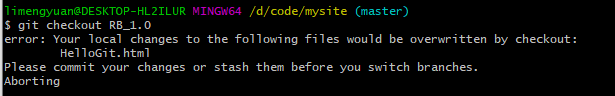
* 添加与修改文件
  + git add 文件名 ——添加文件到版本库的索引（暂存区）
  + git commit 创建一个提交记录并添加提交者信息
  + -m参数添加提交留言
  + git log 查看日志文件
    - 

第一行的SHA-1码是都一无二的（不同的提交号对应不同的提交）

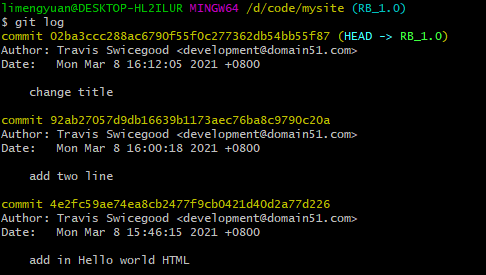
SHA-1

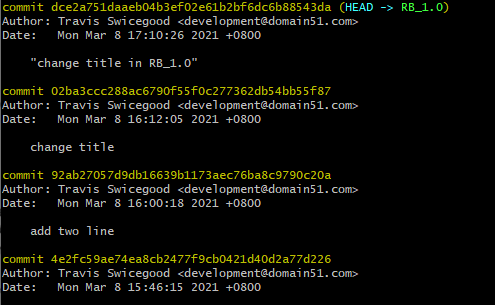
* + 修改 git status ——显示改变过的文件 如果提交需要暂存（stage）修改



* + 三个地方存放代码
    - 工作目录树
    - 索引index（暂存区 staging area） 工作目录树和版本库。存放准备提交到版本库中的修改。（git add 命令）
    - 版本库
* 创建新分支 分支为要发布的代码保留一份拷贝
  + Topic Branch 支持项目的不同发布版本的分支
  + git branch 新分支的名称 父分支的名称/标签点（从标签点位置创建分支）
  + git branch -d 分支名 删除分支
  + git checkout 分支名 切换到不同分支
  + 没有提交切换分支抛出异常
    - 

■提交后可以切换分支 不同分支根据操作打印不同日志

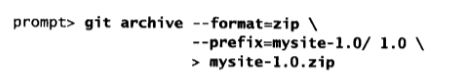




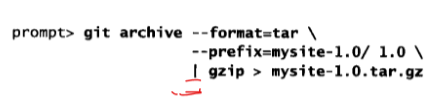
* 打标签并整理版本库
  + git tag 标签名 打标签点
  + git tag 没有参数查看标签点
  + 变基命令可以将分支合并——将一个分支末梢在另一个分支末梢上重现
    - 变基
    - git rebase 后面的参数希望变基的分支

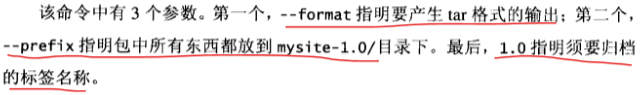
■

* 打包 创建项目源代码的发布包
  + 创建ZIP包



* 创建tar包





1.0的位置可以是标签也可以是分支名。（1.0是打包文件名 如果没有则打空包）。

* 克隆版本库



第一个参数目标远程仓库 第二个参数拷贝到的目标文件夹

git add

git commit [-m]

git status

git log

git branch

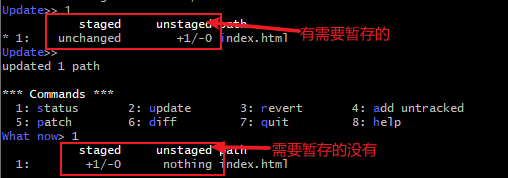
git tag

git rebase

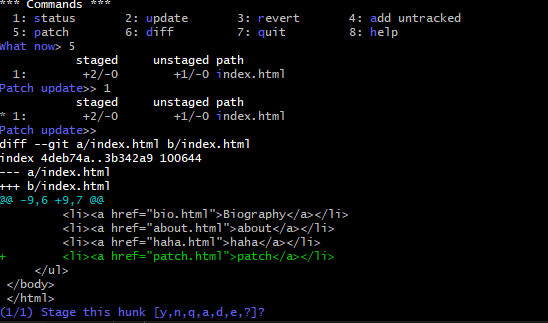
git clone

第四章 添加与提交：Git基础

* 添加文件
  + 暂存的变更（stage change）工作目录树中打算提交到版本库中的变更
  + 暂存操作更新Git内部索引，索引称为暂存区。
    - 用于明确哪些提交哪些不用
    - git add -I 选择要暂存的文件或文件内容



patch模式可以灵活选择单个或多个文件。（git add -p 直接进入）



* 提交修改（暂存区可以看作是Git的缓冲区，从名字就可以看出来）
  + git commit -m “…” -a 将工作目录树中将跟踪的文件提交到版本库中。
  + git commit -m “changes to some-file” some-file 提交指定文件
* 观察历史
* 管理文件

自定义命令

$ git config --global alias.ci "commit"git

文本编辑工具是Vim

第七章 访问远程版本库

* **协议**
  + git协议无需加密，匿名访问
  + git速度最快
  + SSH安全第一 需要用户具备一定权限
  + HTTP/HTTPS协议，通过防火墙，易于架设 需要WebDav服务
* 远程版本库包括克隆之前全部内容
* 版本库同步
  + git fetch
    - 更新本地版本库中的远程分支。但是不会作出修改
  + git pull [远程版本库的名称] [需要托入的远程分支]
    - 完成取来+合并
  + git branch -r
    - 查看远程分支
  + origin/
    - 远程版版本库上的分支名称，默认远程版本库别名(自动生成)
  + git push [repo] [refspec]
    - 默认推到默认版本库origin中，并把本地库中的分支推入到远程版本对应分支上。
    - 推到远程版本库对应分支
    - refspec--分支，或标签
  + git remote add [nickname] [repo-url]
    - 给远程版本库其别名
    - git remote
      * 无参查看远程版本库别名

第十一章 Gitosis管理Git服务器

仓库说进就进

默认没有登录用户名和密码

Rich文本粘贴粘代码挺好

git clone https ……超过TLS 通过更换ISP解决