

# Git tutorial

LeeMoo

2015 年 5 月 20 日



# Contents

0.0.1	5
<b>1 起步</b>	<b>7</b>
1.1 关于版本控制	7
1.1.1 关于版本控制	7
1.1.2 本地版本控制系统	7
1.2 Git 简史	9
1.3 Git 基础	9
1.4 安装 Git	9
1.5 初次运行 Git 前的配置	9
1.6 获取帮助	9
1.7 小结	9
<b>2 Git 基础</b>	<b>11</b>
2.1 取得项目的 Git 仓库	11
2.2 记录每次更新到仓库	11
2.3 查看提交历史	11
2.4 撤消操作	11
2.5 远程仓库的使用	11
2.6 打标签	11
2.7 技巧和窍门	11
2.8 小结	11
<b>3 Git 分支</b>	<b>13</b>
3.1 何谓分支	13
3.2 分支的新建与合并	13
3.3 分支的管理	13

3.4	利用分支进行开发的工作流程 . . . . .	13
3.5	远程分支 . . . . .	13
3.6	分支的衍合 . . . . .	13
3.7	小结 . . . . .	13
<b>4</b>	<b>服务器上的 Git</b>	<b>15</b>
4.1	协议 . . . . .	16
4.2	在服务器上部署 Git . . . . .	16
4.3	生成 SSH 公钥 . . . . .	16
4.4	架设服务器 . . . . .	16
4.5	公共访问 . . . . .	16
4.6	GitWeb . . . . .	16
4.7	Gitolite . . . . .	16
4.8	Gitolite . . . . .	16
4.9	Git 守护进程 . . . . .	16
4.10	Git 托管服务 . . . . .	16
4.11	小结 . . . . .	16
<b>5</b>	<b>分布式 Git</b>	<b>17</b>
5.1	分布式工作流程 . . . . .	17
5.2	为项目做贡献 . . . . .	17
5.3	项目的管理 . . . . .	17
5.4	小结 . . . . .	17
<b>6</b>	<b>Git 工具</b>	<b>19</b>
6.1	修订版本 (Revision) 选择 . . . . .	19
6.2	交互式暂存 . . . . .	19
6.3	储藏 (Stashing) . . . . .	19
6.4	重写历史 . . . . .	19
6.5	使用 Git 调试 . . . . .	19
6.6	子模块 . . . . .	19
6.7	子树合并 . . . . .	19
6.8	总结 . . . . .	19
<b>7</b>	<b>自定义 Git</b>	<b>21</b>
7.1	配置 Git . . . . .	21
7.2	Git 属性 . . . . .	21

<i>CONTENTS</i>	5
7.3 Git 挂钩 . . . . .	21
7.4 Git 强制策略实例 . . . . .	21
7.5 总结 . . . . .	21
<b>8 Git 与其他系统</b>	<b>23</b>
8.1 Git 与 Subversion . . . . .	23
8.2 迁移到 Git . . . . .	23
8.3 总结 . . . . .	23
<b>9 git 内部原理</b>	<b>25</b>
9.1 底层命令 (Plumbing) 和高层命令 (Procelain) . . . . .	25
9.2 Git 对象 . . . . .	25
9.3 Git Refernces . . . . .	25
9.4 Packfiles . . . . .	25
9.5 The Refspec . . . . .	25
9.6 传输协议 . . . . .	25
9.7 维护及数据恢复 . . . . .	25
9.8 总结 . . . . .	25

## 0.0.1

The entire pro Git book, written by Scott Chacon and Ben Straub and published by Apress, is availble here. All conteent is licensed under the Creative Commons Attribution Non Commercial Share Alike 3.0 license. Print versions of the book are available on Amazon.com.



# Chapter 1

## 起步

### 1.1 关于版本控制

#### 1.1.1 关于版本控制

什么是版本控制? 我为什么要关心它呢? 版本控制是一种记录一个或若干个文件内容变化, 以便将来查阅特定版本修订情况的系统。在配中所展示的例子中, 我们仅对保存着软件源代码的广西文件作版本控制管理, 但实际上, 你可以对任何类型的文件进行版本控制。

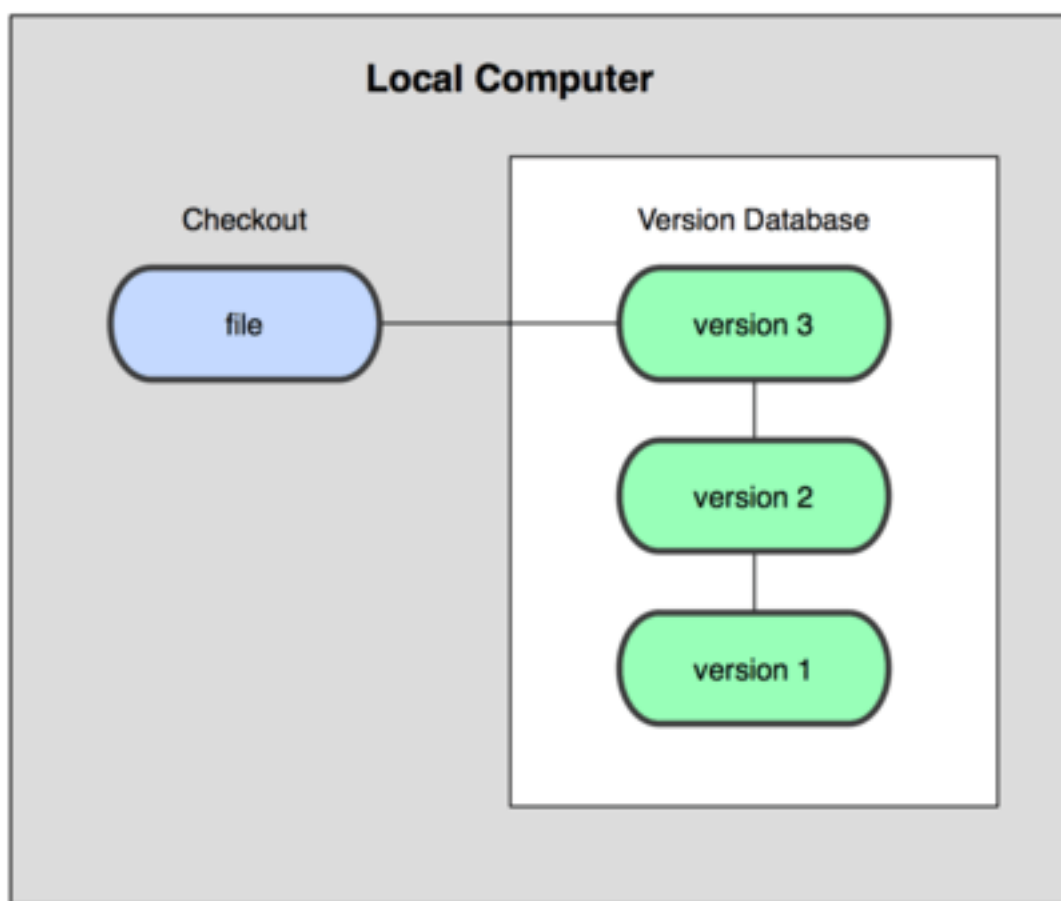
如果你是位图形或网页设计师, 可能会需要保存某一幅图片或页面布局文件的所有修订版本 (这或许是你非常渴望拥有的功能)。采用版本控制系统 (VCS) 是个明智的选择。有了它你就可以将某个文件回溯到之前的状态, 你可以比较文件的变化细节, 查出最后是谁修改了哪个地方, 从而找出导致怪异问题的原因, 又是谁在何时报告了某个功能缺陷等等。使用版本控制系统通常是意味着, 就算你乱来一气把整个项目的文件改的改删的删, 你也照样可以轻松恢复到原先的样子。但额外增加的工作量却微乎其微。

#### 1.1.2 本地版本控制系统

许多人习惯用复制整个项目目录的方式来保存不同的版本, 或许还会改名加上备份的时间以示区别。这么做唯一的好处就是简单。不过坏处也不少: 有时候会混淆所在的工作目录, 一旦弄错文件丢了数据就没法撤销恢复。

为了解决这个问题, 人们很久以前就开发了许多种本地版本控制系统, 大多都是

采用某种简单的数据库来记录文件的历次更新差异 (如图 1-1)。





## 1.2 Git 简史

## 1.3 Git 基础

## 1.4 安装 Git

## 1.5 初次运行 Git 前的配置

## 1.6 获取帮助

## 1.7 小结



## Chapter 2

# Git 基础

2.1 取得项目的 Git 仓库

2.2 记录每次更新到仓库

2.3 查看提交历史

2.4 撤消操作

2.5 远程仓库的使用

2.6 打标签

2.7 技巧和窍门

2.8 小结



## Chapter 3

# Git 分支

### 3.1 何谓分支

### 3.2 分支的新建与合并

### 3.3 分支的管理

### 3.4 利用分支进行开发的工作流程

### 3.5 远程分支

### 3.6 分支的衍合

### 3.7 小结





## Chapter 4

# 服务器上的 Git

### 4.1 协议

### 4.2 在服务器上部署 Git

### 4.3 生成 SSH 公钥

### 4.4 架设服务器

### 4.5 公共访问

### 4.6 GitWeb

### 4.7 Gitorious

### 4.8 Gitolite

### 4.9 Git 守护进程

### 4.10 Git 托管服务

### 4.11 小结



## Chapter 5

# 分布式 Git

### 5.1 分布式工作流程

### 5.2 为项目作贡献

### 5.3 项目的管理

### 5.4 小结



## Chapter 6

# Git 工具

6.1 修订版本 (Revision) 选择

6.2 交互式暂存

6.3 储藏 (Stashing)

6.4 重写历史

6.5 使用 Git 调试

6.6 子模块

6.7 子树合并

6.8 总结



## Chapter 7

# 自定义 Git

### 7.1 配置 Git

### 7.2 Git 属性

### 7.3 Git 挂钩

### 7.4 Git 强制策略实例

### 7.5 总结



## Chapter 8

# Git 与其他系统

### 8.1 Git 与 Subversion

### 8.2 迁移到 Git

### 8.3 总结





## Chapter 9

# git 内部原理

9.1 底层命令 (Plumbing) 和高层命令 (Porcelain)

9.2 Git 对象

9.3 Git References

9.4 Packfiles

9.5 The Refspec

9.6 传输协议

9.7 维护及数据恢复

9.8 总结