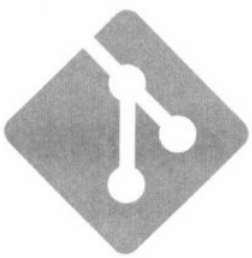
Git

从入门到精通

高见龙 著

- 适合新手上路，老手也可以从中学到正确的观念
- 树立正确的Git观念，可以让你在工作中选用正确的Git指令
- 终端机指令搭配图形界面工具，使学习效率倍增
- 不只教你如何用，还能让你知道自己在用什么，以及为什么要这么用

北京大学出版社
PEKING UNIVERSITY PRESS



Git

从入门到精通

高见龙 著

北京大学出版社
PEKING UNIVERSITY PRESS

内 容 提 要

Git是一款让人一开始觉得很容易学，但却很难精通的工具。本书除了介绍Git的相关知识外，还会模拟各种常见的状况，让读者知道应该在什么时候使用什么指令。

本书共分11个章节，第1~3章介绍安装工具及环境，对于已经安装完成的读者可直接从第4章开始阅读。第5章介绍Git最基本的使用方式，虽然难度不高，但却是整个Git系统的基础。第6章介绍Git中常用的分支功能以及使用情境，第7~9章则是介绍如何修改现有的历史记录、使用标签，以及如何应对其他常见的状况。

前面的内容都是在自己的计算机上就可以完成的，从第10章开始介绍如何将自己计算机里的记录推一份到线上（GitHub）。最后一章（第11章）介绍团队开发时可能会使用的开发过程Git Flow。

市面上的参考书籍或网络教程大多是教大家如何通过终端机指令来学习Git，这让不少想学习Git的新手打了退堂鼓。本书除了教大家如何在终端机视窗中输入Git指令，还搭配了图形界面工具，缓和了读者的学习曲线，让读者更容易上手。

图书在版编目(CIP)数据

Git从入门到精通 / 高见龙著. — 北京：北京大学出版社，2019.12
ISBN 978-7-301-30587-4

Ⅰ. ①G… Ⅱ. ①高… Ⅲ. ①软件工具－程序设计 Ⅳ. ①TP311.561

中国版本图书馆CIP数据核字（2019）第133656号

书　　名 Git 从入门到精通
Git CONG RUMEN DAO JINGTONG

著作责任者 高见龙 著
责任编辑 吴晓月 孙宜
标准书号 ISBN 978-7-301-30587-4
出版发行 北京大学出版社
地　　址 北京市海淀区成府路205号 100871
网　　址 http://www.pup.cn 新浪微博：@北京大学出版社
电子信箱 pup7@pup.cn
电　　话 邮购部 010-62752015 发行部 010-62750672 编辑部 010-62570390
印 刷 者 大厂回族自治县彩虹印刷有限公司
经 销 者 新华书店
787毫米×1092毫米 16开本 15印张 341千字
2019年12月第1版 2019年12月第1次印刷
印　　数 0-4000册
定　　价 49.00元

前言
Preface

写在最前面——为你自己学 Git！

为什么要写这本书

在周星驰主演的电影《大话西游》中，至尊宝用月光宝盒便可穿越时空，回到过去救他的娘子。Git工具虽然无法真的让我们穿越时空（如果有请一定要让我知道，我要回到过去买大乐透），但对计算机工作者来说，它就像时光机一样神奇，可以让你回到指定的时间点去救回不小心被删除的文件。

Git看起来很容易学，但这只是表象，实际上Git是一款很容易上手，但却很难精通的工具。市面上的参考书籍或网络教程大多只会教大家从终端机指令来学习 Git，这让不少想学习Git的新手打了退堂鼓。

我也认同 Git 指令很重要，因为那是整个 Git 的基础，所以学习在终端机窗口敲打、输入 Git 指令是必经过程。但是如果可以搭配图形界面工具，就可以让这个学习曲线稍微缓和一些。所以本书除了 Git 指令的介绍外，也会使用图形界面工具（本书使用 SourceTree）辅助说明，让大家更容易上手。

因为本人个性的关系，在学习新事物的过程中如果有疑惑的地方，总是希望可以搞懂，否则知其然而不知其所以然，无法真正把一门技术搞懂，就会痛苦得睡不着觉。正因为这样，本书在撰写的过程中，即使是很简单的内容，也希望可以尽量解释清楚。希望本书不仅可以教大家如何用（How），也能让大家知道在用什么（What），以及为什么（Why）要这样用。

虽然本书是以中文撰写，但专有名词大多还是英文。之所以用英文来表示，除了因为每个人的翻译可能不一样或翻译之后没有原文贴切之外，最重要的一点，是希望大家能尽早习惯这些英文，因为在实际工作中，很多第一手的资料都是英文的，早点习惯英文对大家绝对是有帮助的。

谁适合本书

只要你对 Git 有兴趣，就可以学习本书。

如果日常工作中已经在使用 Git，那么本书大部分的内容对你来说应该是比较轻松的。不过即便这样，你仍然可以从本书中学到一些“本来以为 Git 是这样，但其实是那样”的理念。

本书内容

本书包括以下内容。

（1）常用 Git 指令介绍。

（2）各种 Git 的常见问题及使用情境。

（3）如何修改 Git 的历史记录。

（4）如何使用 GitHub 与其他人一起工作。

（5）日常工作中一般用不到，但对观念建立有帮助的冷知识。

你需要准备什么？

只要有一台可以工作的计算机（不限定操作系统）就够了。

如何使用这本书

本书主要分为以下几部分。

（1）环境安装与设定。

（2）开始使用 Git。

（3）使用分支。

（4）使用标签。

（5）修改历史记录。

（6）其他常见状况。

（7）使用 GitHub。

（8）使用 Git Flow。

虽然每个章节的内容多少都跟前面的章节有关，但也不一定要从第1章开始依序阅读（当然这也是一种方式），可根据需要跳过部分章节。

使用版本

本书使用的 Git 版本为 2.14.1，读者可以使用 git --version 指令来检测自己目前所使用的 Git 版本。

```
$ git --version
git version 2.14.1
```

如果是不同的版本，一样的指令或参数可能会有不同的执行结果。

程序代码惯例

在学习、使用 Git 时，经常要在终端机（Terminal）模式下输入指令。例如：

```
$ git add index
```

或者这样：

```
$ git commit -m "init commit"
[master (root-commit) 5d47270] init commit
2 files changed, 1 insertion(+)
create mode 100644 config/database.yml create mode 100644 index.html
```

最前面的 $ 符号是系统提示符，告诉大家这是一条需要在终端机环境下手动输入的指令，而它的下一行则是这条指令执行的结果。实际输入指令时不要跟着输入 $，否则可能会出现 command not found 的错误信息。

程序范例及错误更正

本书所有的范例在 Git 2.14.1 以及 macOS 10.12 操作系统环境下均已测试且可正常执行，部分范例可在我的 GitHub 账号取得。不过，由于软件的版本演进或者操作系统的不同，范例程序执行的结果可能会有些微的差异（甚至是错误）。若有任何问题，或者有哪里写错，还请大家不吝来信、留言指教。

最后，希望大家会喜欢这本书，一起来学习 Git 这个看似好学但又不容易学好的有趣工具。

关于学习

输入指令可能很“吓人”，但它很重要！

对学习 Git 的新手来说，打开终端机、输入 Git 指令是件“吓人”的事。

即使有像 SourceTree 或 GitHub Desktop 这类方便的图形界面工具可供使用，我个人仍强烈建议一定要了解 Git 的运作原理。而输入、执行 Git 指令，正是最容易了解 Git 运作的方法之一。

不要害怕输入指令，不要害怕那些看起来很“吓人”的信息，不然即使有图形界面工具，也可能不知道单击某个按钮之后会发生什么事，而导致不能正确地使用 Git。

观念很重要

很多人，包括我自己也是，在一开始学习 Git 的时候，心想只不过就是简单地学习 git add 和 git commit 之类的基本操作指令罢了。但其实这就犹如冰山一角，沉在水底下的比浮在水面上的要多得多，Git 的运作方式远比这些指令来得复杂。所以，如果可以建立正确的观念，遇到问题的时候就不会那么迷茫，就能知道该用什么指令来解决。

目录

Contents

第1章

Git入门

1.1 Git概述

1.1.1 什么是Git

如果你问那些正在使用 Git 工具的人“什么是 Git”，他们大多可能会回答“Git 是一种版本控制系统（Version Control System）”，专业一点的可能会说“Git 是一种分布式版本的版本控制系统”。这样的解释对没接触过 Git 的新手来说是没有任何意义的。到底什么是“版本”？要“控制”什么东西？什么又是“分布式”？接下来就为你一一讲解。

不管你是不是程序员，只要你的日常工作离不开计算机，那么每天可能都要新建、编辑、改动很多的文件。例如，如果你是一名人力资源部门主管，就会创建一个 resume 目录，专门用来保存面试者的资料。

如图 1-1 所示，随着时间的变化，一开始 resume 目录中只有 3 个文件，过两天增加到 5 个；不久之后，其中的 2 个被修改了；过了 3 个月后又增加到 7 个；最后又删掉了 1 个，变成 6 个。每次 resume 目录的状态变化，不管是新建或删除文件，抑或是改动文件内容，都称为一个“版本”，如图 1-1 中的版本 1 ~ 5。而所谓的“版本控制系统”，就是用来记录所有的这些状态变化的，使你可以像搭乘时光机一样，随时切换到过去某一“版本”的状态。

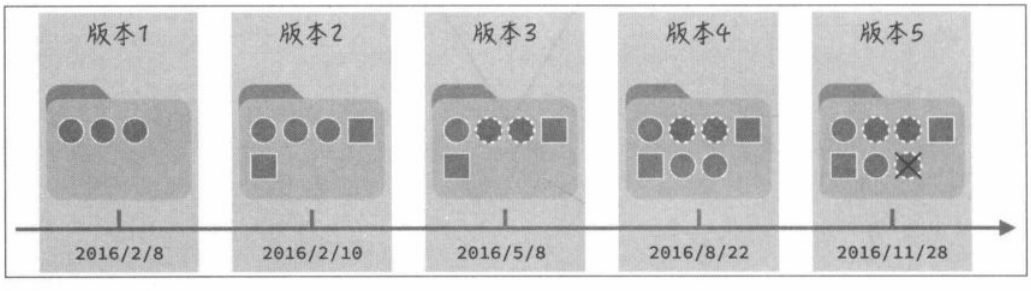


图 1-1

简单地说，Git 就像玩游戏的时候可以存储进度一样。例如，为了避免打头目输了而损失装备，或者打倒头目却没有掉落期望的珍贵装备，可以在每次去打头目之前都记录一下，以便在发生状况的时候回到旧进度，再挑战一次。

1.1.2 为什么要学习Git

先问个问题，大家平常怎样整理或备份文件？

以图 1-1 为例，最传统也是最方便的方式，便是使用“复制 + 粘贴法”。这样操作之后，可能会出现图 1-2 所示的画面。

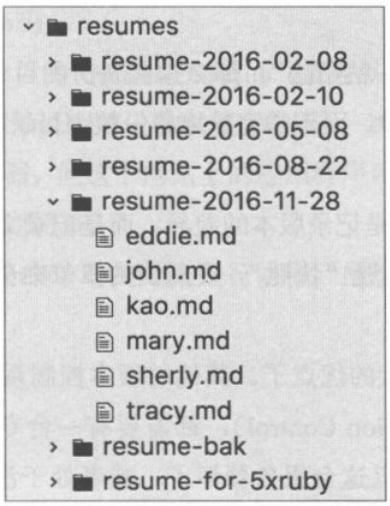


图 1-2

虽然一眼就可以看出每个“版本”的用途，但其他信息就不是那么明显了。例如，resume-2016-05-08 目录中的那两个改动过的文件都改了什么内容？ resume-2016-08-22 和 resume-2016-11-28 这两个目录有什么不一样的地方？ resume-bak 与其他的目录有什么不同？最麻烦的是，如果这个目录是和其他人共享的，而文件被其他人覆盖了，该怎么处理呢？

如果你在乎这些问题的答案，那么使用“版本控制系统”就是一个很不错的选择。通过这样的系统，可以清楚地记录每个文件是谁在什么时候加进来的、什么时候被修改或删除的。Git 就是这样一种版本控制系统，也是当前业界最流行的版本控制系统。

无论做任何工作，如果有 Git 帮你保留这些历史记录和证据，那么发生意外状况的时候你就能知道是从什么时候开始有问题的，以及该找谁负责，再也不用自己“背黑锅”了！

1.2 Git与其他版本控制系统的差异

1.2.1 Git的优点

那么 Git 到底有哪些厉害的地方，会让这么多人选择它呢？

1. 免费、开源

2005 年，为了管理 Linux 内核程序代码，Linux 内核的作者 Linus Trovalds 仅用了 10 天时间就开发出了 Git。粗略算来，至今已有十几年的历史了。除了可免费使用外，整个 Git 的源代码也可以在互联网上获取（当然 Git 的源代码也是用 Git 做版本控制的）。

2. 速度快、文件体积小

如果使用前面提到的“复制 + 粘贴法”，那么这些备份的目录会占用大量空间。其他的版控系统大多是记录每个版本之间的差异，而不是完整地备份整个目录，所以整个目录的大小不会快速地增加。

Git 的特别之处在于，它并不是记录版本的差异，而是记录文件内容的“快照”（snapshot），可以非常快速地切换版本。至于什么是“快照”，在后面的章节中会有更详细的介绍。

3. 分布式系统

对我来说，这可能是 Git 最大的优点了。其他的版本控制系统，比如 CVS 或 SVN 之类的集中式的版控系统（Centralize Version Control），都需要有一台专用的服务器，所有的更新都要与这台服务器沟通。也就是说，一旦这台服务器坏了，或者处于没有网络连线的环境下，就无法使用了。

而 Git 是一款分布式的版控系统（Distributed Version Control），虽然通常也会有共同的服务器，但即使在没有服务器或在没有网络的环境下，仍然可以使用 Git 进行版控，待服务器恢复正常运行或移到有网络的环境后再进行同步，不会受到影响。事实上，在使用 Git 的过程中，大多数的 Git 操作在计算机本机上就可以完成。

1.2.2 Git的缺点

如果非要说 Git 的缺点，那大概就是易学难精。虽然 Git 的指令非常多，而且有的指令有点复杂，但平常会用到的指令并不多。根据“80/20 法则”，大概 20% 的指令就足以应付 80% 的工作。

除了终端机（或命令提示符）环境下的 Git 指令外，还有很多实用的图形界面工具，让使用者不用输入复杂的指令就可以享用 Git 强大的功能。本书将使用终端机指令来解释概念，并以图形界面工具（如 SourceTree）来辅助说明 Git 是怎样运行的。

1.3 常见问题

1. 我是设计师，我也可以用Git吗

基本上，Git 只关心文件的“内容”，所以只要是文件，都可以使用 Git 来管理。只是设计生成的大多是 Photoshop 的 PSD 文件或 Illustrator 的 AI 文件，虽然 Git 也可以管理这些文件，但因为这些文件（二进制文件）不像常规文本文件那样可以一行一行地查看，也就无法那么精准地知道什么人在什么时候改了哪些字。但总体来说，Git 还是帮得上忙的，至少当文件不小心被覆盖或删除的时候，还可以找回旧版本的文件。

2. Git就是我之前看到过的GitHub吗

这是很多新手容易有的误会，以为 Git 就是 GitHub（或者认为 GitHub 就是 Git），甚至有的公司在招聘启事上明确写着“会使用 GitHub”。事实上，Git 是一款版本控制软件，而 GitHub 是一个商业网站，其本体是一个 Git 服务器，但这个网站上的应用程序可以让大家通过 Web 操作来完成一些原本需要复杂的 Git 指令才能做到的事。

本书后面也会介绍如何使用 GitHub 与其他人进行协作。虽然 GitHub 很好用，但别忘了 Git 才是它的本体。

第2章

环境安装

2.1 在Windows操作系统中安装Git

在 Windows 操作系统中安装 Git 之前，先从官方网站（https://git-scm.com/download/win）下载合适的 Git 版本，如图 2-1 所示。

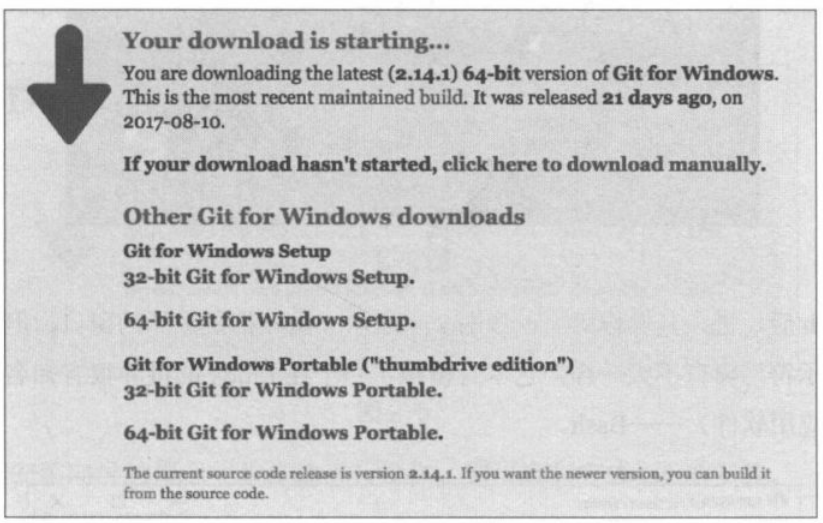


图 2-1

下载后，安装时只需根据提示一路单击“Next”按钮即可，如图 2-2 所示。

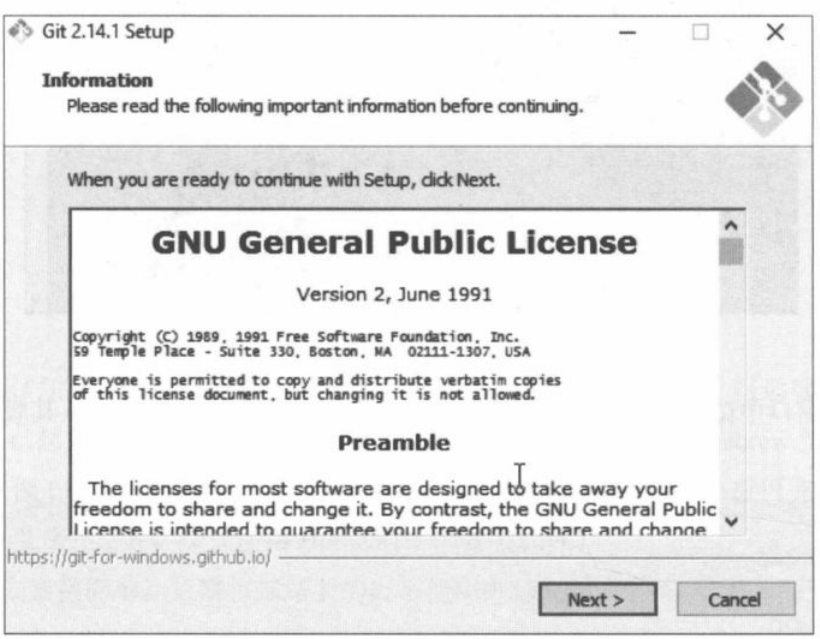


图 2-2

安装完成之后，选择“Git Bash”应用程序，如图 2-3 所示。

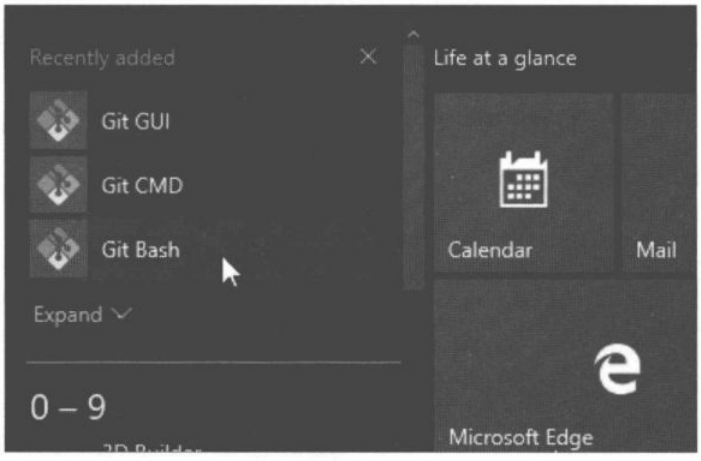


图 2-3

启动 Git Bash 后，进入其操作界面，如图 2-4 所示。虽然都是黑黑的窗口，但这个与 Windows 自带的“命令提示符”窗口不太一样，它本身模拟了一个在 Linux 的世界很有知名度的软件（其实不能算是常规的应用软件）——Bash。

图 2-4

这时可以在窗口中试着输入指令，验证一下 Git 是否安装完成，以及确认其版本信息。显示代码为：

```
$ which git
/mingw64/bin/git

$ git --version
git version 2.14.1.windows.1
```

如果看到类似的代码信息，就是安装成功了。