# 第1章 起步

## 1.1 关于版本控制

### 1.1.1 本地版本控制系统

### 1.1.2 集中化的版本控制系统

### 1.1.3 分布式版本控制系统

## 1.2 Git 的历史

## 1.3 Git 基础要点

### 1.3.1 直接快照，而非比较差异

### 1.3.2 近乎所有操作都可本地执行

### 1.3.3 时刻保持数据完整性

### 1.3.4 多数操作仅添加数据

### 1.3.5 三种状态

## 1.4 安装 Git

### 1.4.1 从源代码安装

### 1.4.2 在 Linux 上安装

### 1.4.3 在 Mac 上安装

### 1.4.4 在 Windows 上安装

## 1.5 初次运行 Git 前的配置

### 1.5.1 用户信息

第一个要配置就是你个人的**用户名称**和**电子邮件地址**。每次Git提交时都会引用这2条信息，说明了谁提交了更新，所以会随更新内容一起被永久纳入历史记录：

$ git config --global user.name "John Doe"

$ git config --global user.email johndoe@example.com

如果用了 --global 选项，那么更改的配置文件就是位于你**用户主目录**下的那个，以后你所有的项目都会默认使用这里配置的用户信息。

如果要在某个特定的项目中使用其他用户名称或者电子邮件，只要去掉 --global 选项重新配置即可，新的设置保存在当前项目的 .git/config 文件里。

### 1.5.2 文本编辑器

接下来要设置的是默认使用的**文本编辑器**。Git 需要你输入一些额外消息的时候，会自动调用一个外部文本编辑器给你用。默认会使用操作系统指定的默认编辑器，一般可能会是 Vi 或者 Vim。如果你有其他偏好，比如 Emacs 的话，可以重新设置：

$ git config --global core.editor emacs

### 1.5.3 差异分析工具

还有一个比较常用的是，在解决合并冲突时使用哪种差异分析工具。比如要改用 vimdiff 的话：

$ git config --global merge.tool vimdiff

### 1.5.4 查看配置信息

要检查已有的配置信息，可以使用 git config --list 命令：

$ git config --list

user.name=Scott Chacon

user.email=schacon@gmail.com

color.status=auto

color.branch=auto

color.interactive=auto

color.diff=auto

...

有时候会看到重复的变量名，那就说明它们来自不同的配置文件（比如 /etc/gitconfig 和 ~/.gitconfig），不过最终 Git 实际采用的是最后一个。

也可以直接查阅某个环境变量的设定，只要把特定的名字跟在后面即可，像这样：

$ git config user.name

Scott Chacon

## 1.6 获取帮助

想了解 Git 的各式工具该怎么用，可以阅读它们的使用帮助，方法有三：

$ git help <verb>

$ git <verb> --help

$ man git-<verb>

比如，要学习 config 命令可以怎么用，运行：

$ git help config

## 1.7 小结

git config命令的使用：

(1) 配置用户名和电子邮件地址

git config --global user.name ‘leqingsheng’

git config --global user.email 123@qq.com

注意：--global 的使用

(2) 配置

git config --global

# 第2章 Git 基础

1. 如何初始化一个新的代码仓库，做一些适当的配置；
2. 如何开始或停止跟踪某些文件；
3. 如何暂存或提交某些更新；
4. 如何让 Git 忽略某些文件，或是名称符合特定模式的文件；
5. 如何既快且容易地撤消犯下的小错误；
6. 如何浏览项目的更新历史，查看某两次更新之间的差异；
7. 如何从远程仓库拉数据下来或者推数据上去；

## 2.1 取得项目的 Git 仓库

### 2.1.1 从当前目录初始化

### 2.1.2 从现有仓库克隆

## 2.2 记录每次更新到仓库

### 2.2.1 检查当前文件状态

### 2.2.2 跟踪新文件

### 2.2.3 暂存已修改文件

### 2.2.4 忽略某些文件

### 2.2.5 查看已暂存和未暂存的更新

### 2.2.6 提交更新

### 2.2.7 跳过使用暂存区域

### 2.2.8 移除文件

### 2.2.9 移动文件

## 2.3 查看提交历史

### 2.3.1 限制输出长度

### 2.3.2 使用图形化工具查阅提交历史

## 2.4 撤消操作

### 2.4.1 修改最后一次提交

### 2.4.2 取消已经暂存的文件

### 2.4.3 取消对文件的修改

## 2.5 远程仓库的使用

### 2.5.1 查看当前的远程库

### 2.5.2 添加远程仓库

### 2.5.3 从远程仓库抓取数据

### 2.5.4 推送数据到远程仓库

### 2.5.5 查看远程仓库信息

### 2.5.6 远程仓库的删除和重命名

## 2.6 打标签

### 2.6.1 列显已有的标签

### 2.6.2 新建标签

### 2.6.3 含附注的标签

### 2.6.4 签署标签

### 2.6.5 轻量级标签

### 2.6.6 验证标签

### 2.6.7 后期加注标签

### 2.6.8 分享标签

## 2.7 技巧和窍门

### 2.7.1 自动完成

### 2.7.2 Git 命令别名

## 2.8 小结