前言

Flask 是目前最流行的 Python Web 框架之一。自 2010 年开源以来，受到了越多越多的 Python 开发者的喜欢，其受欢迎程度不输于 Django。截止 2018 年 6 月，它在 GitHub 上已有近 36000 个 Star，2000 多位 Watchers，是目前 GitHub 中 Star 数最多的 Python Web 框架。

图 0-1 Flask 的 logo

附注 Flask 的图标虽然看起来很像辣椒，但其实它是角状的容器（powder horns）。

Flask 仅仅保留了 Web 框架的核心，其他的功能都交给扩展实现。如果没有合适的扩展，你甚至可以自己编写。Flask 不会给你做决定，也不会限制你的选择。它足够轻量，使用它你可以只用 5 行就编写一个最简单的 Web 程序，但却并不简陋，它能够适应各类项目项目的开发。

因为 Flask 的灵活性，越来越多的公司选择 Flask 作为 Web 框架，甚至开始从 Django 迁移到 Flask。使用 Flask 的公司在国外有 Netflix、Reddit、Twilio、Mailgun 等，在国内则有豆瓣、果壳、下厨房等，这说明 Flask 能经受大型项目的挑战，也足够灵活，能够适应各种需求。下图列出了部分使用 Flask 的公司：

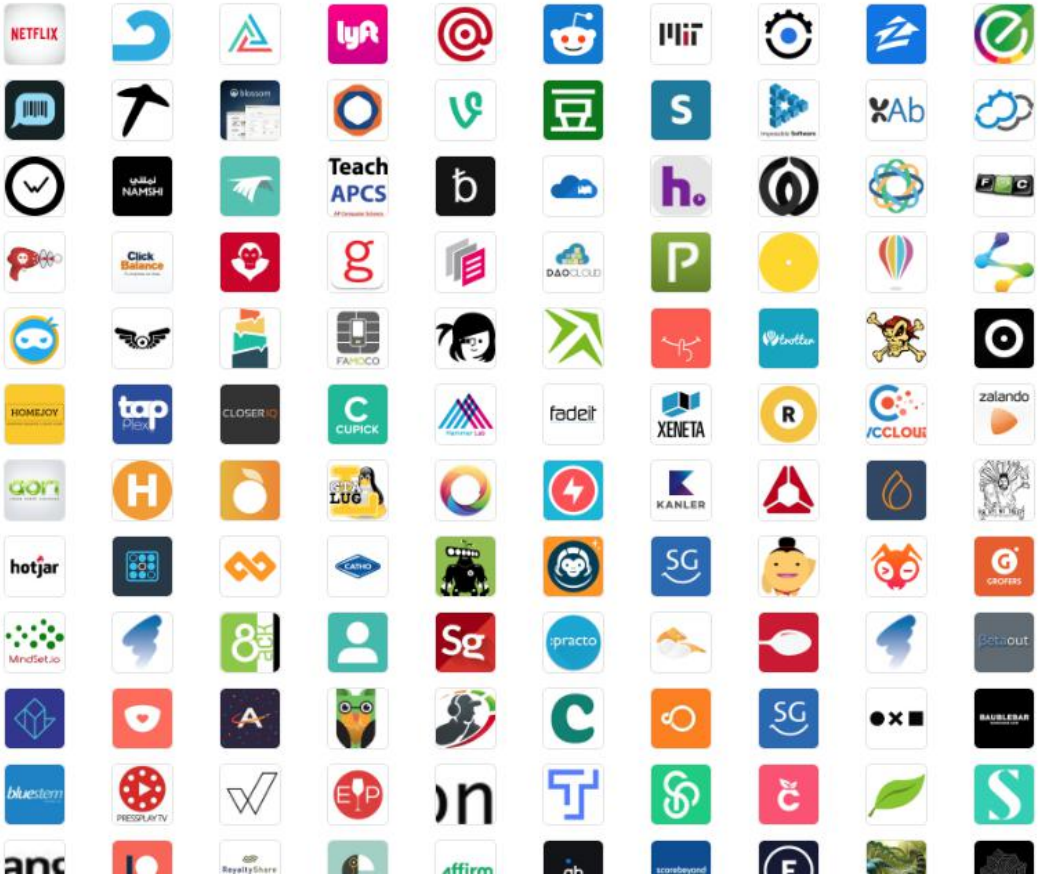

图 0-2 使用 Flask 的公司

附注 你可以在 StackShare 上查看完整的使用 Flask 的公司列表（https://stackshare.io/flask）。

在国内，越来越多的 Python 程序员开始关注和学习 Flask。对于国内程序员来说，相关书籍仅有一两本，内容上也过于陈旧和单薄，本书的编写可以填补这一空白。本书提供了学习 Flask 的完善路径，从基础内容到进阶实践，再到源码分析。同时也安排了丰富的示例程序，让读者可以通过亲自实践来更快的掌握 Flask 开发。

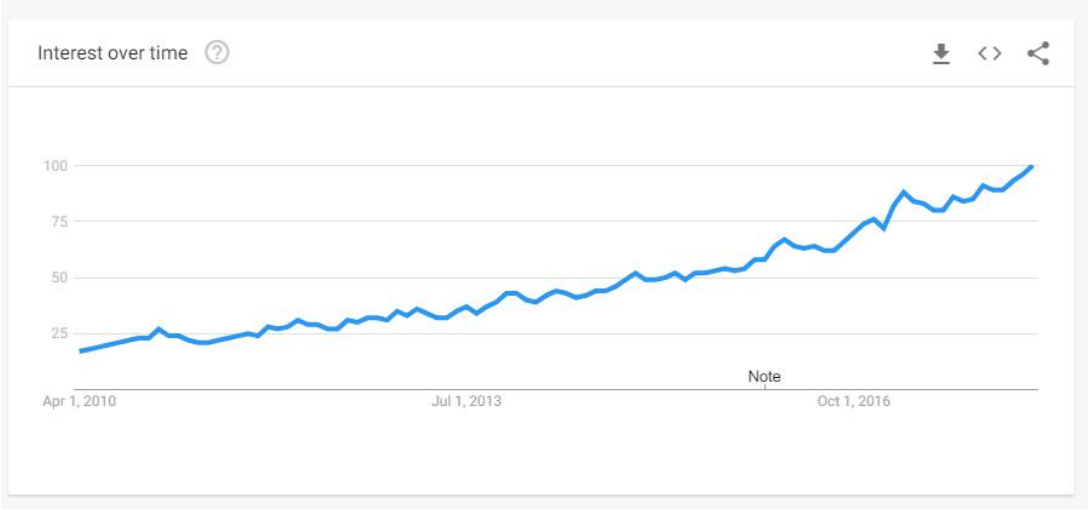

图 0-3 Flask 自 2010 年开源以来在 Google 上的搜索趋势

附注 参考来源：Google Trends

（https://trends.google.com/trends/explore?date=2010-04-01%202018-04-01&q=%2Fm%2F0dgs72v）

目标读者

在技术层面，本书适合所有的 Python 程序员（了解 Python 即可）阅读，包括已经学习过其他 Python Web 框架（比如 Django）的读者和没有接触过 Web 框架的读者。

在难度水平层面，本书适合新手以及中级读者阅读。新手会在这里学到 Flask 的基础内容，并且通过丰富完善的实例学习 Flask 开发的方方面面；中级读者则可以通过实践和阅读进阶内容来进一步提高 Flask 开发能力。

综上所述，本书主要适合以下几类读者：

- 了解 Python 基本语法，想要自己动手做网站的编程爱好者
- 熟悉 Python，想要从事 Python Web 开发的后端工程师、运维工程师和爬虫工程师
- 想要从 Django 等其他 Python Web 框架转向 Flask 的 Python 工程师

本书主要特点

本书主要有三个显著的特点：

- 内容全面

本书的内容覆盖了 Flask Web 开发的完整路径：从基础知识的学习，到不同类型和复杂程度的程序的编写，再到代码的测试优化以及 Flask 源码分析。在实践上，从基础的内容管理，到用户认证和权限管理，再到 Flask 与 JavaScript 的数据交互、Web API 的编写以及 WebSocket 的应用等等都会包括在内。

- 实践丰富

本书包含大量的代码片段，并附带多个完整可运行的示例程序。在本书第一个部分每一章都提供一个示例程序，第二部分则会通过介绍 5 个比较完善的 Flask 项目来学习各个方面的进阶知识，在第三部分还会通过一个真实的扩展来学习 Flask 扩展开发。通过将各类知识融入到实际的项目开发实践中，可以让你更直观的了解具体的代码实现，并且快速的应用到实际开发中。

- 内容最新

本书的另一个特点就是内容保证最新。本书中的代码和示例程序都基于 Flask 最新发布的稳定版 1.0。本书中使用的其他 Pyhton 包和前端框架（Bootstrap、Materialize 等）全部使用最新版本，并且对未来可能会有的变化会加以说明。

这些特点可以保证书中的内容在一定时间内不会过时。对于其他书籍或教程中关于 Flask 的误区，本书也会逐一纠正说明。

除了使用的工具保持最新，本书还引入了 Python 和 Flask 开发中的新变化，比如 Flask 的命令行系统，新的 Python 包管理工具（Pipenv），新的包上传工具（twine），新的 PyPI 站点（https://pypi.org），在 PyPI 上使用 Markdown 格式的 README……

本书核心内容

本书由四部分组成，分别为基本篇、实战篇、进阶篇和附录，共 16 章。本书的章节安排经过精心的设计，力求让读者可以循序渐进的掌握 Flask 开发基础知识和技巧。

第一部分：基础篇 介绍 Flask 开发相关的基础知识。

- 第 1 章：搭建开发环境，编写一个最小的 Flask 程序并运行它，了解 Flask 基本知识。

- 第 2 章：介绍了 Flask 与 HTTP 的交互方式以及相关的 Flask 功能。
- 第 3 章：Jinja2 模板的使用。
- 第 4 章：Web 表单的创建和表单数据的验证。
- 第 5 章：在 Flask 程序中使用数据库进行 CRUD 操作。
- 第 6 章：在 Flask 程序中发送电子邮件的几种方式。

第二部分：实战篇 通过几个示例程序来介绍 Flask 开发中的各类功能实现和技巧。

- 第 7 章：通过一个简单的留言板程序 SayHello 介绍 Web 开发基本流程和基本的项目管理方式，对第一部分的基础知识进行简单的回顾。
- 第 8 章 ：通过个人博客程序 Bluelog 介绍 CRUD 操作、用户认证、文章评论、管理后台等功能。
- 第 9 章：通过图片社交程序 Albumy 介绍用户注册和认证、用户权限管理、图片上传与处理、用户头像、复杂的数据库关系、复杂的数据库查询、全文搜索等内容。
- 第 10 章：通过待办事项程序 Todoism 介绍单页应用、国际化与本地化、Web API、OAuth 服务器端实现等内容。
- 第 11 章：通过聊天室程序 CatChat 介绍 Websocket 应用、OAuth 客户端实现（第三方登录）、Markdown 支持、代码语法高亮等内容。

第三部分：进阶篇 介绍 Flask 程序的部署流程：测试、性能优化、部署上线；介绍 Flask 开发的进阶话题：Flask 扩展开发和 Flask 源码与机制分析。

- 第 12 章：介绍 Flask 程序的自动化测试，包括单元测试和 UI 测试的编写、计算测试覆盖率和代码质量检查。
- 第 13 章：对 Flask 程序进行性能优化的主要措施，包括函数与数据库查询的性能分析、缓存的使用、静态文件优化。
- 第 14 章：介绍部署 Flask 程序前的准备，以及部署到 Linux 服务器和云平台 Heroku、PythonAnywhere 的完整流程。
- 第 15 章：通过扩展 Flask-Share 来介绍编写 Flask 扩展的完整流程，从创建项目到上传到 PyPI。
- 第 16 章：介绍了 Flask 的一些设计理念，包括底层 WSGI 的相关实现,并对各个主要功能点进行源码分析。

第四部分 附录

- 附录 A 一些相关 Flask 学习资源的推荐

阅读前的准备

在开始我们的 Flask 之旅前，还有一些准备工作要做。首先，你要有一台安装了 Python（https://www.python.org/）的电脑，只有菜谱是没法学会烹饪的。并且，你要了解 Python 的基础知识。

提示 本书的所有示例程序的代码通过了 Python2.7 和 Python3.6 的测试，建议你选用这两个版本。因为大多数 Python 包（包括 Flask）已经不再支持 Python 2.6 及以下版本，Python3.3 及以下版本，确保不要使用这些版本。另外，Python 官方社区将于一年零七个月后（2020 年 1 月 1 日）停止对 Python2 的维护，这或许可以作为你选择 Python 版本时的考量之一。

其次，本书有大量操作需要在命令行（CLI，Command Line Interface）下进行，所以你要熟悉你所在操作系统下的命令行。本书中会在涉及到操作系统特定的命令时给出提示，Windows 系统给出的命令是针对 CMD.exe，Linux 和 macOS 系统则对应 Bash。

最后，HTML、CSS、JavaScript 分别作为一个 Web 页面的结构层、表现层和行为层，是 Web 开发的基础，你需要对他们有基本的了解。任何一个 Web 程序都是由单个或多个 Web 页面以及页面上包含的内容以及按钮，表单等交互组件构成的。在本书中，我们会使用 Flask 操作 HTML 页面；为了让 HTML 页面更加美观，我们会使用 CSS 定义样式，为了简化编写样式的操作，我们会使用 CSS 框架，比如 Bootstrap（http://getbootstrap.com/）；为了让某些操作更加合理和方便，或是为程序增加动画效果，我们会使用 JavaScript 来操作页面元素；为了简化编写 JavaScript 的工作，我们会使用 JavaScript 库 jQuery（https://jquery.com/）。

附注 在 Web 开发中，大部分的程序离不开 JavaScript，有很多的页面逻辑和功能使用 JavaScipt 可以很方便简洁的实现。为了更多的介绍 Flask，本书尽量避免使用过多的 JavaScipt 代码。

如果你还不熟悉这些内容，那么可以通过下面的网站来快速入门：

- W3Schools：https://www.w3schools.com
- MDN Web 文档：https://developer.mozilla.org/docs/Web
- Codecademy：https://www.codecademy.com

使用示例程序

示例程序均使用 Git 来管理程序版本，为了便于获取，代码均托管在在线代码托管平台 GitHub（https://github.com/）上。Git（https://git-scm.com/）是最流行的开源 VCS（Version Control System，版本控制系统），大多数项目都使用它来追踪文本文件（代码）的变化。Git 非常易于上手，如果你还不熟悉它，可以访问 Try Git（https://try.github.io/）花费 15 分钟通过实际的操作来快速了解 Git。

你可以访问 Git 官网的下载页面（https://git-scm.com/downloads）了解不同操作系统的安装方法，安装成功后即可使用它来获取示例程序。下面介绍了两种使用示例程序的方式：

- 阅读示例程序

因为示例程序都托管在 GitHub 上，所以阅读示例程序最简单的方式是在浏览器中阅读。在对应的章节，我们会给出示例程序在 GitHub 上的仓库链接。

如果要在本地阅读，那么首先使用 git clone 命令把 Github 上的示例程序克隆（即复制）到本地，以本书的项目仓库为例：

```
$ git clone https://github.com/greyli/helloflask.git
```

提示 clone 命令后面的参数是远程 Git 仓库的 URL，最后的“.git”后缀也可以省略。这里的 URL 中的传输协议使用了 http(s)://协议，你也可以使用 git://协议，即 git://github.com/greyli/helloflask.git。本书中的示例程序都托管在 GitHub（https://github.com/）上。

使用 ls（即 List）命令（Windows 下使用 dir 命令）列出当前目录下的文件信息，你会看到当前目录中多了一个 helloflask 文件夹，这就是我们刚刚克隆下来的项目仓库。下面使用 cd（即 Change Directory）命令切换进这个文件夹：

```
$ cd helloflask
```

现在你可以使用你喜欢的文本编辑器打开项目文件夹，然后准备阅读。建议使用轻量的文本编辑器来阅读示例代码，比如 Atom（https://atom.io/）、Sublime Text（https://www.sublimetext.com/）或 Notepad++（https://notepad-plus-plus.org/）。

在对应章节的开始都会包含从 GitHub 克隆程序，创建虚拟环境并运行程序的基本步骤，你可

以一边阅读源码，一边实际尝试对应的程序功能。

示例程序根据章节内容设置了对应的标签，每个标签都对应了一个程序版本。你可以使用 git tag -n 命令查看当前项目仓库中包含的所有标签：

```
$ git tag -n
```

使用 git checkout 命令即可签出对应标签版本的代码，添加标签名作为参数，比如：

```
$ git checkout foo
```

在后面，书中会在每一次包含更改的章节提示应该签出的标签名。如果在执行新的签出命令之前，你对文件做了修改，那么需要使用 git reset 命令来撤销改动：

```
$ git reset --hard
```

注意 git reset 会删除本地修改，如果你希望修改示例程序源码并保存修改，可以参考下面的“改造示例程序部分”。

如果你想比较两个版本之间的变化，可以使用 git diff 命令，添加比较的两个标签作为参数，比如：

```
$ git diff foo bar
```

如果你想更直观的查看版本变化，可以使用下面的命令打开内置的 Git 浏览客户端：

```
$ gitk
```

除了内置的 Git 客户端，还有大量的第三方客户端可以使用，详情可以访问 https://git-scm.com/downloads/guis 查看。另外，你也可以访问 GitHub 的 Web 页面查看不同版本（标签）变化，查看某个项目的两个版本之间的变化可以访问 https://github.com/用户名/仓库名/compare/标签 A...标签 B，比如对 foo 和 bar 标签进行比较可以访问 https://github.com/greyli/helloflask/compare/foo...bar。

最后，你可以定期使用 git fetch 命令来更新本地仓库：

```
$ git fetch --all
$ git fetch --tags
```

```
$ git reset --hard origin/master
```

❑ 改造示例程序

只看菜谱是没法学会烹饪的，自己动手编写代码是学习 Flask 最有效的途径。你可以在阅读示例程序的同时编写自己的 Flask 程序，将书中的介绍和实际的示例程序代码作为参照。另外，你也可以创建一份示例程序的拷贝（派生，fork），这样你就可以自由的修改示例程序的源码，改造一个你自己版本的示例程序。创建派生仓库的主要步骤如下所示：

1）注册一个 GitHub 账号（https://github.com）。

2）访问示例程序的 GitHub 仓库页面（比如 https://github.com/greyli/helloflask），点击右上角的 Fork 按钮创建一个派生仓库，如图 0-4 所示。

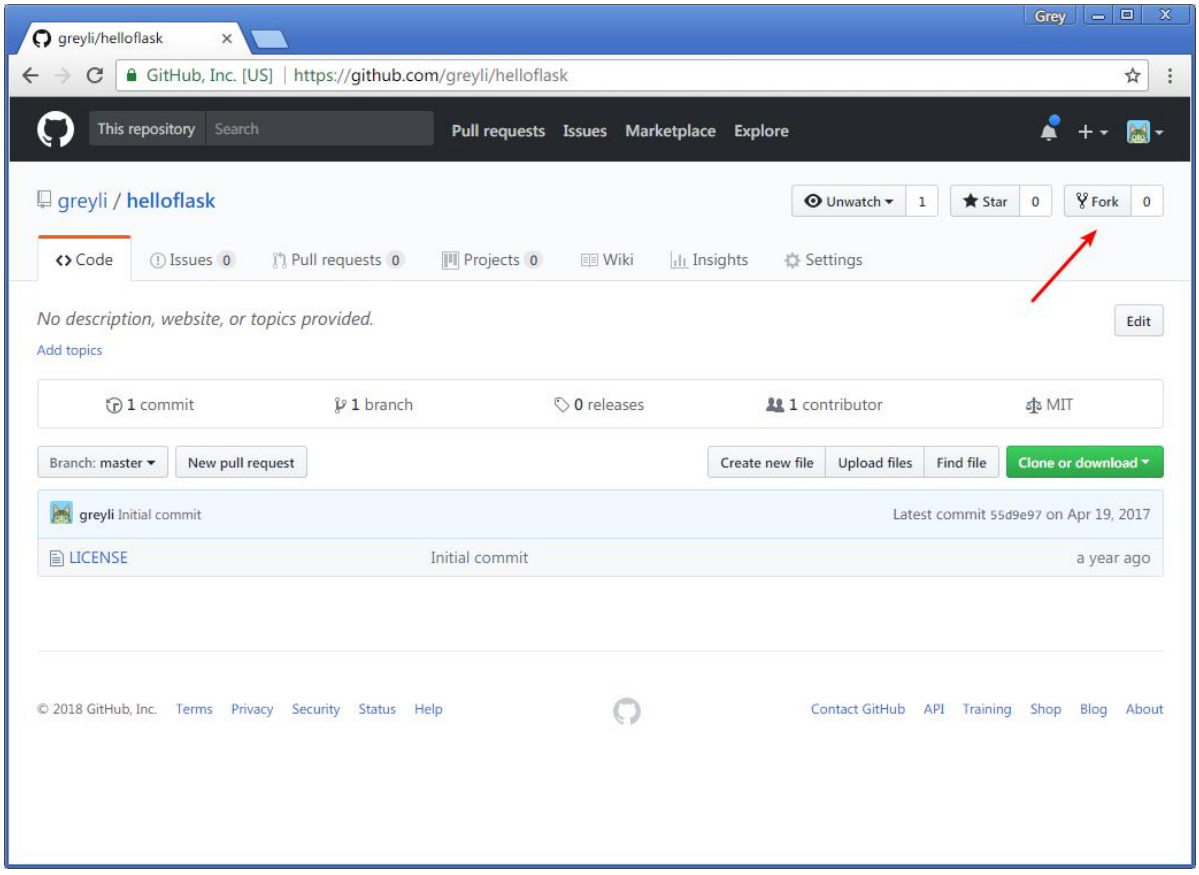

图 0-4 创建派生仓库

3）在本地使用 git clone 命令克隆新创建的派生的仓库，使用你的用户名构建 URL：

```
$ git clone https://github.com/你的用户名/helloflask.git
```

现在你可以在本地自由的修改实例程序，并提交到你的 GitHub 账户中的远程仓库中。

排版约定

Windows 中的命令提示符为“>”，而 Linux 或 macOS 中的命令提示符为“$”，本书中将统一使用美元符号（即“$”）作为命令提示符，比如：

```
$ cd hello
```

命令提示符为三个大于号（即“>>>”）表示 Python Shell 中输入的代码，比如：

```
>>> import os
```

“$”或“>>>”标记的文本下方没有命令提示符的文字表示输出的字符，不需要打出，比如：

```
$ cat hello.txt
Hello, Flask!
```

为了节省篇幅，本书中的代码片段没有严格按照 PEP8 的约定，比如类和函数之间的空行被缩减为 1 行。另外，出现过的导入语句和无关的代码块会被省略掉。代码中重复或不相干的部分，为了节省篇幅，都使用三个省略号代替，比如：

```
def do_someting():
    ...
    if foo:
        return False
    return True
```

代码、命令或 URL 中有时会使用“<”和“>”来标识演示内容，在实际输入中并不需要写出，比如：

```
https://github.com/<你的用户名>
```

因为在示例代码中通常会引入大量随机字符，这些随机字符包含下面的使用规则：

- 列表 1：spam、ham、eggs，
- 列表 2：foo, bar, baz, qux, quux, quuz, corge, grault, garply, waldo, fred
- 人名会使用 Grey Li 或 grey

- 网站会使用 helloflask.com 或 example.com
- 其他需要读者自己修改的占位字符会使用类似 your_password、you_email 的文本

最后，为了尽量让正文保持简洁，每一章所新涉及到的 Python 库都会在第一小节前汇总列出对应的版本和相关链接（比如主页、源码和文档）。因为大部分项目在 PyPI 上提供的介绍都不够完善，除非程序有独立的主页，否则会优先使用 GitHub 或 BitBucket 上的项目页面作为主页。

读者反馈与疑问

由于本人水平有限，编写时间也比较仓促，因此书中难免有错误或者不全面的地方，在此恳请读者朋友批评指正。

关于本书的疑问和反馈可以到本书在 GitHub 上的项目仓库 HelloFlask（https://github.com/greyli/helloflask）中创建 Issue；书中的错误或笔误可以修改仓库中的勘误文件（Errata.md）并提交 Pull Request。

对于示例程序的疑问、反馈和改进建议可以到示例程序在 GitHub 上的项目仓库提交 Issue 或 Pull Request，具体的网址可以在对应的章节看到。

当然，你也可以直接发邮件与我联系，我的邮箱是 withlihui@gmail.com。

本书的配套资源索引可以在本书的主页 http://helloflask.com/book 上看到。另外，你可以访问我的博客（http://greyli.com）或是知乎专栏“Hello, Flask!”（https://zhuanlan.zhihu.com/flask）阅读更多 Flask 文章。

致谢

首先，感谢机械工业出版社华章公司的杨福川老师和李艺老师。因为杨老师的信任，才让我有幸能够写作这本书。本书能够顺利完成，离不开两位老师的悉心指导，更离不开其他编辑的辛苦工作。

其次，感谢 Flask 社区和其他开源项目的贡献者们创造了这一切，也感谢在 Stack Overflow、GitHub、Reddit 和 Wikipedia 等网站贡献知识的开发者们。

最后，感谢我的父母和奶奶这段时间的支持和帮助，也感谢女友魏瑶和弟弟家辉对我的鼓励和陪伴。