Broadview
www.broadview.com.cn

人生苦短，我用Python。
在项目实践中学习Django，简单快速建网站！

齐伟 编著

“跟老齐学Python”系列图书之一。
面向已经“轻松入门”之后的读者。
以实际Web开发项目为主线，既综合应用Python基础知识，又学习新的网站开发技能！

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Broadview®
www.broadview.com.cn
博文视点·IT出版旗舰品牌
技术凝聚实力·专业创新出版

书不在多，实用就行！
无数网友好评，易学易懂的Python应用经典！

注重实战： 专注于实战，以项目为主线，帮助读者轻松掌握Python应用。

通俗易懂： 轻松幽默，毫无做作、晦涩之感，让编程不枯燥、不乏味。

深入浅出： 指导读者理解Python编程思想，领略Python魅力，授之以渔。

◆简单的博客系统
Django 起步
编写博客的数据模型类
显示博客信息
◆用户管理
自定义模板和静态文件位置
用户登录
用内置方法实现登录和退出
用户注册
关于密码的操作
维护个人信息
◆文章管理和展示
管理文章栏目
发布和显示文章
删除和修改文章
文章展示
◆拓展网站功能
梳理已有功能
查看作者全部文章
为文章点赞
文章的阅读次数
文章的评论功能
多样化显示
管理和应用文章标签
◆收集和展示图片
收集网络图片
展示图片
◆中场休整
◆创建在线学习应用
基于类的视图
管理课程标题
发布和学习课程内容
◆结束和开始

博文视点Broadview

新浪微博 weibo.com
@博文视点Broadview

策划编辑：高洪霞
责任编辑：牛　勇
封面设计：李　玲

上架建议：程序设计>Python

ISBN 978-7-121-32572-4

9 787121 325724 >

定价：69.00元

跟老齐学 Python

Django实战

齐伟 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书是以 Python 为基础进行 Web 应用开发的进阶读物。书中以一个实例项目为主线，在实践中边学边做，理论联系实际。每节都配有思维导图，使读者对项目需求一目了然；每章都有知识点和文档导读，引导读者“知其所以然”。相信认真阅读本书的读者，不仅能够得到“鱼”，更能得到“渔”，从而具备独立开发的能力。

本书适合已经具有 Python 基础技能、进行 Web 应用开发的读者。

如有此意愿，但尚缺乏 Python 基础技能者，推荐阅读《跟老齐学 Python：轻松入门》。

图书在版编目（CIP）数据

跟老齐学 Python：Django 实战 / 齐伟编著. —北京：电子工业出版社，2017.10
ISBN 978-7-121-32572-4

Ⅰ. ①跟… Ⅱ. ①齐… 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（2017）第 209727 号

策划编辑：高洪霞
责任编辑：牛 勇
印 刷：三河市双峰印刷装订有限公司
装 订：三河市双峰印刷装订有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编：100036
开 本：787×1092 1/16 印张：17 字数：457 千字
版 次：2017 年 10 月第 1 版
印 次：2017 年 10 月第 1 次印刷
定 价：69.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：（010）51260888-819，faq@phei.com.cn。

前言

是的，你应该开始阅读本书了。

如果读者已经通过《跟老齐学 Python：轻松入门》这本书完成了 Python 入门，那么接下来就要用 Python 做一些实际项目了，做什么呢？可以做的东西真不少，比如数据统计、爬虫、机器学习、人工智能等，当然还有不可或缺的 Web 开发。

本书就是要带领读者：

- 学习 Django；
- 完成一个项目；
- 边做项目边学知识。

因此，本书按照完成一个项目从易到难的顺序，以项目为主线逐渐展开对 Django 的学习。Django 是一种在 Web 开发中应用非常广泛的开发框架，它能够让程序员非常快捷地建设一个网站，并且支持网站开发中各种复杂的业务。如果需要快速建设一个网站，并且希望后期还能通过迭代开发实现网站功能和性能的提升，那么 Django 是一个不错的选择。

Django 的官方网站中有不少内容完备的文档，当然，这种完备不是针对初学者而言的。对于初学者来说，也许官方文档太简要了，因此本书的作用就是将初学者领入 Django 开发的大门，并引导读者熟悉使用 Django 官方文档。当读者阅读并练习完本书的项目后，就能够使用官方文档进行开发了，本书的目的也就达到了。

本书从始至终贯穿着一个项目实践，将开发实战中常用的 Django 知识串联起来，因此它不像课本那样按部就班地把每个知识点都罗列出来，而是根据项目需要，将技能融汇在实战之中。此外，如果仅从开发实践的角度学习 Django，难免会出现就问题而解决问题的现象，缺少系统和深入的学习。为此，每节中均配有“知识点”，用以扩展或者系统化有关知识，特别是在“文档导读”中会向读者提供有关网络资料，试图让读者知其然并知其所以然。

在项目进行中，读者可能会遇到暂时无法理解的部分，不要纠结于一时之惑，而要跟着项目继续进行下去，或许到后面就能理解前面的内容了——“发展是硬道理”。

如果正在阅读本书的你是 Web 开发的初学者，此前没有做过任何网站开发，那么笔者特别建议遵循如下学习方法。

- 严格按照本书要求，将所有代码都认真敲过。不要复制代码（本书提供的源代码，仅仅

是一个参考），要自己一个字母一个字母地敲进去。

- 将自己的调试结果与本书所示结果进行对照，看看是否实现了预期的功能。
- 在调试不能通过时，首先要认真检查自己的代码，包括但不限于拼写问题、缩进问题等，若还不能解决，可通过 Google 搜索该异常（错误），请相信，你的错误别人也同样遇到过。
- 本书支持网站 www.itdiffer.com 和 QQ 群（26913719）也是读者学习的助手。
- 本书提示读者阅读的官方文档，一定要认真阅读。
- 本书实际上是一个 Django 项目，项目中包含了多个应用。如果读者按照本书的要求，第一遍就实现了应用的功能，那只能说明本书描述无误且你的阅读能力尚可，并不代表你已经掌握了什么。只有当你把刚刚调试通过的应用删除，然后完全靠自己的记忆和理解，同样实现了该应用的功能，才能说明你掌握了相关知识技能。再次强调，要理解相关内容，就必须多重复几遍。

因为本书是以“项目为主线”的，所以不会面面俱到地将 Django 的所有内容介绍一遍，而是根据项目的功能需要，选择性地使用和介绍有关内容。对于具体功能的实现，书中所给出的代码也非唯一写法，读者可以使用其他更好的方式实现。

在实际的 Web 开发项目中，除使用 Django 框架外，还会用到 HTML、CSS、JavaScript 等知识，从而做出一个看起来有点设计感的页面。虽然说“人不可貌相”，但使用 Django 做出一个漂亮的前端，还是能够让人感觉很舒服的。因此，书中也涉及一些前端内容，读者在阅读时，如果缺少相关知识，可以实时补充。当然，前端知识并不是学习本书所必备的，因为凭借读者的聪明才智和无所不包的网络，掌握应付本书所需要的前端知识是非常容易的。

读者可以在 GitHub 上（https://github.com/qiwsir/DjangoPracticeProject）获得本书的所有代码，这些代码是最终结果，希望不会给读者的学习带来不便。

感谢为本书的面市提供帮助的编辑们，感谢我的妻子在本书编写过程中提供的支持。

齐　伟

2017 年 8 月

目录

第 1 章　简单的博客系统 ····· 1

1.1　Django 起步 ····· 1

1.1.1　Django 简介 ····· 1

1.1.2　安装 Django ····· 3

1.1.3　创建项目 ····· 4

1.1.4　创建应用 ····· 6

1.1.5　网站配置 ····· 11

1.1.6　知识点 ····· 12

1.2　编写博客的数据模型类 ····· 12

1.2.1　数据模型类 ····· 13

1.2.2　发布博客文章 ····· 17

1.2.3　知识点 ····· 21

1.3　显示博客信息 ····· 23

1.3.1　显示文章标题 ····· 24

1.3.2　查看文章内容 ····· 28

1.3.3　知识点 ····· 32

第 2 章　用户管理 ····· 35

2.1　自定义模板和静态文件位置 ····· 35

2.1.1　自定义模板位置 ····· 36

2.1.2　自定义静态文件位置 ····· 36

2.1.3　通用静态文件和基础模板 ····· 37

2.1.4　重置管理后台模板 ····· 40

2.1.5　知识点 ····· 42

2.2　用户登录 ····· 43

2.2.1　创建应用 ····· 44

2.2.2　理解表单类 ····· 45

2.2.3　登录的视图函数 ····· 47

2.2.4　登录的前端界面 ····· 49

2.2.5 知识点 ……53
2.3 用内置方法实现登录和退出 ……54
2.3.1 内置的登录方法 ……55
2.3.2 判断用户是否登录 ……58
2.3.3 内置的退出方法 ……59
2.3.4 知识点 ……60
2.4 用户注册 ……62
2.4.1 简单注册 ……62
2.4.2 增加注册内容 ……65
2.4.3 管理新增的注册内容 ……70
2.4.4 知识点 ……71
2.5 关于密码的操作 ……72
2.5.1 修改密码 ……73
2.5.2 重置密码 ……78
2.5.3 利用第三方应用重置密码 ……84
2.5.4 知识点 ……87
2.6 维护个人信息 ……88
2.6.1 个人信息的数据模型类和表单类 ……88
2.6.2 展示个人信息 ……90
2.6.3 编辑个人信息 ……93
2.6.4 上传和裁剪头像图片 ……97
2.6.5 优化头像上传功能 ……105
2.6.6 对个人信息进行管理 ……107
2.6.7 知识点 ……108
第 3 章 文章管理和展示 ……110
3.1 管理文章栏目 ……110
3.1.1 设置栏目 ……110
3.1.2 编辑栏目 ……118
3.1.3 删除栏目 ……120
3.1.4 知识点 ……122
3.2 发布和显示文章 ……125
3.2.1 简单的文章发布 ……126
3.2.2 使用 Markdown ……131
3.2.3 文章标题列表 ……133
3.2.4 知识点 ……140
3.3 删除和修改文章 ……141
3.3.1 删除 ……142
3.3.2 修改 ……143
3.3.3 设置分页功能 ……147
3.3.4 知识点 ……149

3.4 文章展示……151
3.4.1 新写文章标题列表……151
3.4.2 重新编写“查看文章”功能……155
3.4.3 知识点……156
第 4 章 拓展网站功能……158
4.1 梳理已有功能……158
4.1.1 修改导航栏……158
4.1.2 修改登录和注册后的跳转……159
4.1.3 知识点……161
4.2 查看作者全部文章……161
4.2.1 查看某作者的文章列表……162
4.2.2 知识点……165
4.3 为文章点赞……167
4.3.1 修改数据模型类……167
4.3.2 编写视图函数……168
4.3.3 修改模板文件……169
4.3.4 知识点……172
4.4 文章的阅读次数……173
4.4.1 安装 Redis……174
4.4.2 在 Python 中使用 Redis……175
4.4.3 记录阅读次数……176
4.4.4 显示最“热”文章……177
4.4.5 知识点……179
4.5 文章的评论功能……180
4.5.1 数据模型类和表单类……180
4.5.2 实现评论功能……181
4.5.3 知识点……184
4.6 多样化显示……185
4.6.1 统计文章总数……186
4.6.2 最新发布的文章……188
4.6.3 评论最多的文章……189
4.6.4 自定义模板选择器……192
4.6.5 知识点……193
4.7 管理和应用文章标签……195
4.7.1 管理文章标签……195
4.7.2 发布文章时选择标签……200
4.7.3 在文章中显示文章标签……202
4.7.4 推荐相似文章……202
4.7.5 知识点……204

第 5 章　收集和展示图片 ……206

5.1　收集网络图片 ……206

5.1.1　创建图片相关类 ……207

5.1.2　收集和管理图片 ……210

5.1.3　完善图片管理功能 ……214

5.1.4　知识点 ……218

5.2　展示图片 ……219

5.2.1　瀑布流方式展示图片 ……219

5.2.2　查看图片的详细信息 ……223

5.2.3　知识点 ……224

第 6 章　中场休整 ……226

第 7 章　创建在线学习应用 ……227

7.1　基于类的视图 ……227

7.1.1　最简类视图 ……228

7.1.2　读取数据 ……230

7.1.3　初步了解 Mixin ……232

7.1.4　知识点 ……233

7.2　管理课程标题 ……234

7.2.1　判断用户是否登录 ……235

7.2.2　创建课程 ……237

7.2.3　删除课程 ……239

7.2.4　知识点 ……243

7.3　发布和学习课程内容 ……245

7.3.1　课程内容的数据模型 ……246

7.3.2　课程内容的表单类 ……249

7.3.3　课程内容的视图 ……250

7.3.4　查看课程内容 ……252

7.3.5　注册学习课程 ……257

7.3.6　知识点 ……260

第 8 章　结束和开始 ……262

第1章

简单的博客系统

从现在开始，请读者随我一起做一个项目，这个项目的名称叫做“多用户内容发布系统”。因为很多读者是来学习的，所以还要本着“循序渐进”、“深入浅出”的原则来做这个项目——这当然不是工程项目开发的原则，这是教学的原则。

本项目从博客开始。

博客，现在已经有人把它看做历史文物了，因为互联网的迅猛发展，10年前的事物就算古董了。尽管如此，博客依然可以作为一个学习的样本。本章将通过一个单用户的博客系统的开发，初步说明利用Django开发网站的基本步骤和网站的基本组成。当然，这仅仅是起步，并不意味着学完本章就理解Django了。不积跬步，无以至千里，最后的复杂系统也是由一个个简单的小功能堆砌起来的。

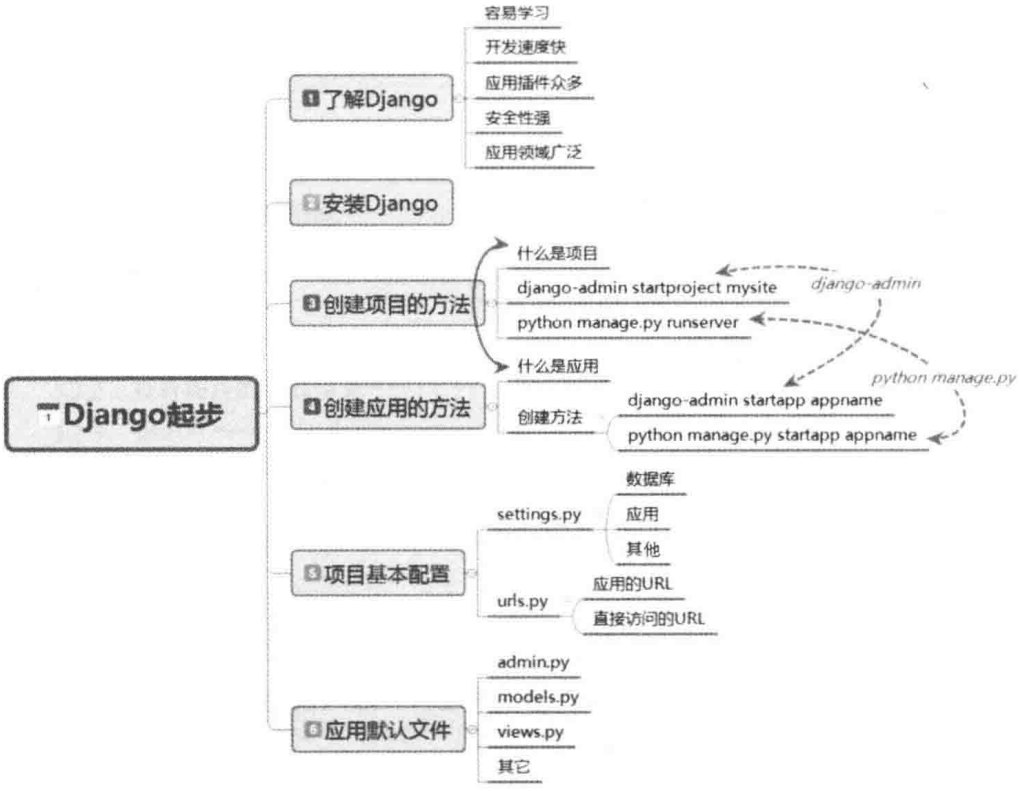
1.1 Django起步

Django这个词对一些人来说或许并不陌生，有一部荣获第85届奥斯卡金像奖的影片叫做*Django Unchained*，中文名被翻译为《被解救的姜戈》，这或许是Django首度被翻译为中文。但是，作为网站开发框架的Django跟这部电影没有任何关系。

Django的起步内容如下图所示。

1.1.1 Django简介

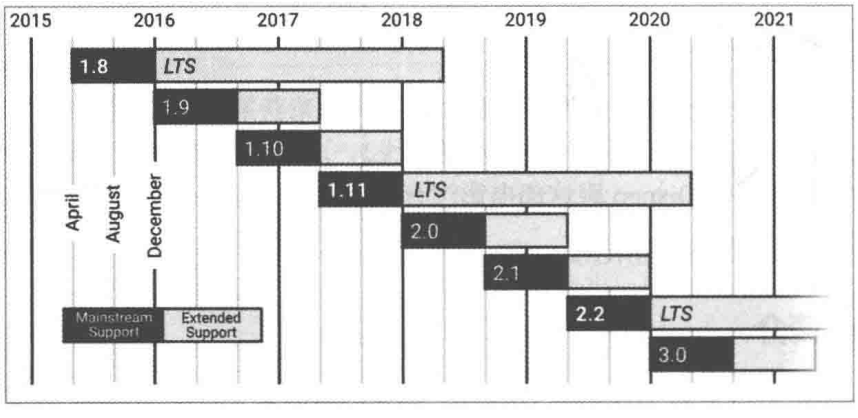
一般认为Django开发框架（简称Django）诞生的时间是2003年的金秋时节。此时千年古城苏州正桂花飘香，而大洋彼岸的美国有两位程序员在使用后来被冠名为Django的框架做网站开发，这是全世界首次使用。让我们感谢这两位程序员，他们是Adrian Holovaty和Simon Willison。2008年虽然遇到了金融危机，但依然挡不住技术的进步，Django有了自己的DSF（Django Software Foundation），后来Django的发展就由这个基金会来主导了。



对于 Django 的评价，如果借用李清照的《鹧鸪天·桂花》来表达，我认为是非常恰当的。

> 暗淡轻黄体性柔。情疏迹远只香留。何须浅碧深红色，自是花中第一流。
>
> 梅定妒，菊应羞。画栏开处冠中秋。骚人可煞无情思，何事当年不见收。

“自是花中第一流”，不仅仅是现在，未来依然蓬勃发展。在 Django 官方网站披露的信息中，可以看到其发展蓝图，如下图所示。



关于 Django 的特点，用官方网站（https://www.djangoproject.com/）上的大标题就可以概括了：“Django makes it easier to build better Web apps more quickly and with less code”。展开来说，就是如下几点：

- 容易上手，开发速度快；
- 囊括了网站开发中的用户管理、内容管理、网站地图、RSS 等常用的众多插件；

- 安全性强，比如 Django 默认解决了 SQL 注入、跨站攻击等问题；
- 应用广泛，类型多样化。使用 Django 开发的网站包括公司提供的各类在线服务网站、社会组织和政府机构网站等，其类型包括但不限于管理系统、社交网站、计算平台等。

就开发网站的框架而言，除 Django 外，在 Python 领域还有 Tornado、Flask 等，它们各有各的特点，但 Django 的应用范围最广。

接下来就开始 Django 之旅——虽有解救姜戈的惊心动魄，但无生命之忧。

1.1.2　安装 Django

一般情况下，读者所用的计算机操作系统上没有 Django，需要自己安装。

Django 是以 Python 为语言环境的，所以要先保证计算机上已经安装了 Python。读者如果对 Python 知之甚少，可以参阅《跟老齐学 Python：轻松入门》，这本书介绍了 Python 的基础知识。

Django 适用于 Python 3 和 Python 2.7 两种版本，如果你是一个新秀，并不是为了承接基于 Python 2.7 的旧项目开发，可以直接用 Python 3 进行 Django 的学习和开发。

本书的所有代码都是基于 Ubutnu 操作系统的 Python 3 撰写的。对于新秀，想要免除一些麻烦，不妨也构建同样的环境。如果读者非要在 Windows 操作系统里面折腾，也不是不可以，只不过本书不这么折腾了。

安装 Django 的最简单方法是输入以下代码（关于 pip，请参阅《跟老齐学 Python：轻松入门》有关章节说明）：

```
$ sudo pip install Django==1.10.1
```

安装中指定了版本，在撰写本书内容时，正式发布的最新版本是 1.10.1。读者在阅读时，或许已经有更新的版本了，可以修改版本号，或者不写版本号，默认安装软件源中已有的版本。不论读者安装的是什么版本，本书中的基本流程都是适用的，如果在某些细节上有所区别，请读者通过 Google 解决，一般情况下这种细节的区别很小。

安装好之后，可以先进入到 Python 交互模式中，查看一下所安装的 Django 版本。

```
$ python
Python 3.5.2 (default, Sep 10 2016, 08:21:44)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import django
>>> print(django.get_version())    #会打印出当前 Django 的版本号
1.10.1
```

上面的安装方法是一种简单而且常用的方法，除这种方法外，还可以下载 Django 源码进行安装，代码如下。

```
git clone https://github.com/django/django.git
```

如果系统中没有安装 Git，会提示用户安装。

这时，会在当前目录中看到一个名称为“django”的目录，里面是最新版本的 Django。

随后在当前目录中进行如下操作：

```
$ sudo pip install -e ./django
```

系统会提示 Django 已经安装成功的信息：“Successfully installed Django”。

通过这种方法得到的必然是 Django 的最新版本，而且是正在开发中的最新版，尚未正式发布——虽然如此，也是能够正常使用的，Django 框架的开发者不会把不能使用的代码发布出来。喜欢尝试的读者可以用上面这种方式安装。

安装成功之后，在 Python 交互模式中可以查看当前 Django 版本，显示结果中包含“dev”字样，说明我们使用的是开发版。

```
>>> print(django.get_version())
1.11.dev20160928182449
```

一般情况下，推荐使用 pip install Django 的方法来安装，其原因就是笔者提倡的保守主义。

基础设施已经建好，下面就要创建项目了。

1.1.3 创建项目

Django 中的“项目”（project）可以看做是一个专有名词，因为后面还有一个与之有关的名词“应用”（application）。

所谓“项目”，可以理解为一个网站。

先规划好将项目创建在什么地方，比如，笔者放在下面的目录里。

```
~/mysite$ pwd
/home/qiwsir/mysite
```

在这个目录中，创建一个 Django 项目。

```
~/mysite $ django-admin startproject mysite
```

看一下这个目录，多了一个 mysite 子目录，mysite 就是这个项目的名称。

```
$ ls
mysite
```

mysite 子目录里面的内容，就是我们创建的项目内容，如下图所示。

```
qiwsir@ubuntu:~/mysite$ tree
.
└── mysite
    ├── manage.py
    └── mysite
        ├── __init__.py
        ├── settings.py
        ├── urls.py
        └── wsgi.py

2 directories, 5 files
```

这是创建项目的一种方法。

可以把刚才建立的项目删除，即删除~/mysite/mysite/目录，然后尝试另外一种创建项目的方法，请读者注意观察两种方法的命令形式和结果。第二种创建项目的方法如下。

```
~/mysite$ django-admin startproject mysite .
```

这次在项目名称 mysite 后面有一个空格，然后还有一个句号（英文半角句号），如此也可以创建项目。创建之后，仔细观察目录结构，如下图所示，跟上面的目录结构进行对比，从而找出两种方式的区别。

```
qiwsir@ubuntu:~/mysite$ tree
.
├── manage.py
└── mysite
    ├── __init__.py
    ├── settings.py
    ├── urls.py
    └── wsgi.py

1 directory, 5 files
```

在所创建的项目目录中，有一个名称很特殊的文件__init__.py，熟悉 Python 的读者一定知道，其所在目录因它的存在而变成了一个 Python 包（package）（关于包、模块的详细内容请参阅《跟老齐学 Python：轻松入门》）。

此外，与 manage.py 在同一个级别上有一个目录 mysite。如果读者在创建项目时用其他的名称代替 mysite，那么就会出现以该名称命名的目录。

至此，已经建立了一个项目，也就意味着已经有一个网站的基本框架了，虽然现在还不能访问什么。

在后续的示例中，将使用以“django-admin startproject mysite”创建的项目结构。

准备就绪，执行下述操作。

```
$ python manage.py runserver
```

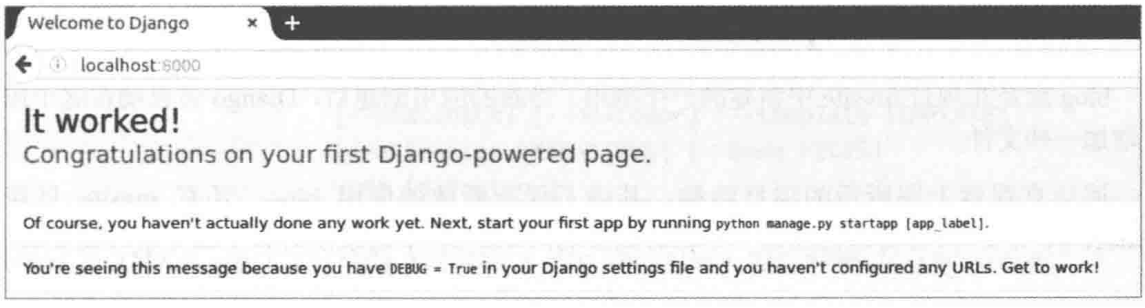
在本书中，为了明确说明目录或者文件的位置，以“./”表示项目根目录，比如上面指令中的 manage.py 文件，其在项目中的位置就是“./manage.py”；在子目录 mysite 中看到的 urls.py 文件，则用“./mysite/urls.py”路径表示。

执行上述指令后，如果一切正常，最终会看到下面的提示信息：

```
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

提示信息的第一行代码说明已经启动了一个服务，可以通过 http://127.0.0.1:8000/访问；提示信息的第二行代码说明了结束当前服务的方法——按 Ctrl+C 组合键。

打开浏览器，在地址栏中输入 http://127.0.0.1:8000 或者 http://localhost:8000，就会看到如下图所示的结果。



祝贺！祝贺！热烈祝贺！

已经看到了第一个网页，虽然它很简陋，但未来它将被不断优化，成为一个优秀的网站，读者会用它改变世界——程序员，天生就被赋予了改变世界的使命。某天某布斯找到你，“我现在只差一个程序员了”。为了改变世界，继续学习吧。

网站的成长方式就是不断地增加功能。在 Django 中，人们将完成某个或者某几个功能的集合称为一个“应用”，所以一个功能比较多的网站常常是由多个“应用”组成的，后面我们可以把注意力集中在一个个“应用”上。

1.1.4 创建应用

项目已经创建好，网站也有了，接下来要实现网站的具体功能。在 Django 中，人们把这些具体的功能称为“应用”（application）。

进入到刚才创建的项目目录中，即 manage.py 文件所在的目录，然后按照下面的代码执行。

```
qiwsir@ubuntu:~/mysite$ ls
db.sqlite3  manage.py  mysite
qiwsir@ubuntu:~/mysite$ python manage.py startapp blog    #①
qiwsir@ubuntu:~/mysite$ ls
blog  db.sqlite3  manage.py  mysite
```

从上述代码中可以看出，执行了语句①之后，在目录中多了一个 blog。如果看看这时候的目录结构，会看到 blog 里面已经有默认的文件和目录了，如下图所示。

```
qiwsir@ubuntu:~/mysite$ tree
.
├── blog
│   ├── admin.py
│   ├── apps.py
│   ├── __init__.py
│   ├── migrations
│   │   └── __init__.py
│   ├── models.py
│   ├── tests.py
│   └── views.py
├── db.sqlite3
├── manage.py
└── mysite
    ├── __init__.py
    ├── __pycache__
    │   ├── __init__.cpython-35.pyc
    │   ├── settings.cpython-35.pyc
    │   ├── urls.cpython-35.pyc
    │   └── wsgi.cpython-35.pyc
    ├── settings.py
    ├── urls.py
    └── wsgi.py

4 directories, 17 files
```

blog 就是在项目 mysite 中新建的一个应用。当新的应用创建后，Django 会自动在这个应用中增加一些文件。

请认真观察上图所示的项目结构，其中不仅有新建的应用 blog，还有 mysite 目录和 manage.py 等文件。目前这已经是一个相对完善的网站结构，下面依次对各个部分进行简要说明。

1. manage.py

在了解 manage.py 之前，先讲解创建项目时用到的 django-admin.py，它是 Django 的任务管理命令行工具。可以通过下面的方式查看 django-admin 命令的帮助信息和所支持的命令行参数。

```
qiwsir@ubuntu:~/mysite$ django-admin
Type 'django-admin help <subcommand>' for help on a specific subcommand.
Available subcommands:
[django]
    check
    compilemessages
    createcachetable
    dbshell
    diffsettings
    dumpdata
    flush
    inspectdb
    loaddata
    makemessages
    makemigrations
    migrate
    runserver
    sendtestemail
    shell
    showmigrations
    sqlflush
    sqlmigrate
    sqlsequencereset
    squashmigrations
    startapp
    startproject
    test
testserver
```

请读者耐心地把这些命令行参数都看一下，留个印象。

看到 startapp 和 startproject 了吧！在创建项目的命令中，我们使用了 startproject。这里还有一个 startapp，这个参数在刚刚创建的应用中出现了，但不是用在 django-admin 后面，而是用于 python manage.py 的参数，那么这里的 startapp 是什么意思呢？

```
qiwsir@ubuntu:~/mysite$ django-admin help startapp
usage: django-admin startapp [-h] [--version] [-v {0,1,2,3}]
                             [--settings SETTINGS] [--pythonpath PYTHONPATH]
                             [--traceback] [--no-color] [--template TEMPLATE]
                             [--extension EXTENSIONS] [--name FILES]
                             name [directory]

Creates a Django app directory structure for the given app name in the current
directory or optionally in the given directory.
```