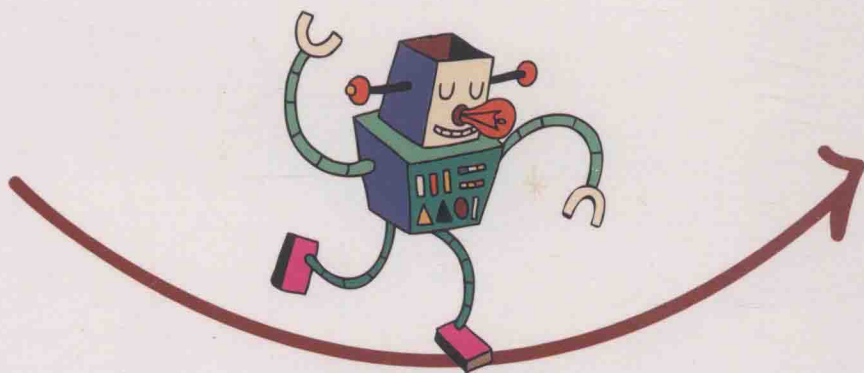


活用Django 2.0 Web Framework
快速构建移动网站

Python新手学 Django 2.0

架站的16堂课 第2版



何敏煌 林亮昀 编著



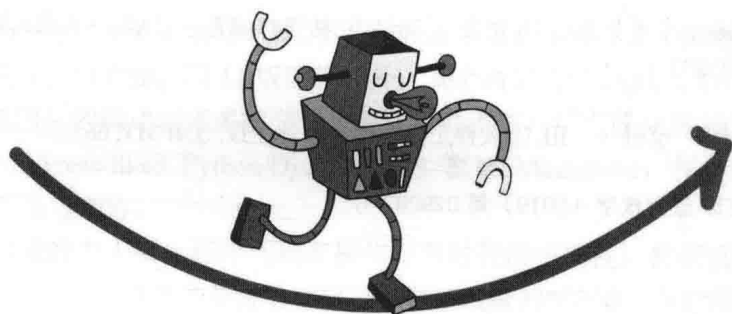
多个实用的网站实例，按图讲解实现的详细步骤，快速上手
全书使用Python 3，避开难懂的程序语法，更易于入门学习
学习如何连接到数据库、使用社交网站账号来实现验证机制，掌握实时运算处理的秘诀
使用Mezzanine内容管理平台快速打造CMS网站，助你轻松成为网络架站高手



清华大学出版社

Python新手学 Django 2.0

架站的16堂课 第2版



何敏煌 林亮昀 编著

清华大学出版社
北京

内 容 简 介

Python 是目前非常受欢迎的程序设计语言,本书通过对 Python 语言使用最多的 Django 2.0 Web Framework 的介绍,让读者可以轻松制作出全功能的动态网站。

本书分 4 部分,以 16 堂课来介绍 Python 新手使用 Django 架站的要点。第一部分(第 1~3 堂)以一个小型的个人博客网站为主轴,介绍如何快速建立一个实用的 Django 网站;第二部分(第 4~7 堂)是 Django 架构深入剖析,详细分析 Django 的 MVC/MTV 架构;第三部分(第 8~11 堂)为实用网站开发技巧;第四部分(第 12~16 堂)为实用网站开发教学,从设计、规划到实践,逐步指导读者在自己的主机环境下构建出有趣实用的内容。

本书既可作为希望快速上手 Python+Django 的初学者的参考书籍,也可作为 Python 培训学校在 Python+Django 方面的培训教程。

本书为荣钦科技股份有限公司授权出版发行的中文简体字版本。

北京市版权局著作权合同登记号 图字:01-2018-8435

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

Python 新手学 Django 2.0 架站的 16 堂课 / 何敏煌,林亮昀编著. —2 版. —北京:清华大学出版社,2019
ISBN 978-7-302-52332-1

I. ①P… II. ①何… ②林… 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 029083 号

责任编辑:夏毓彦

封面设计:王翔

责任校对:闫秀华

责任印制:丛怀宇

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座

邮 编:100084

社总机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者:三河市铭诚印务有限公司

经 销:全国新华书店

开 本:190mm×260mm

印 张:31.5

字 数:806 千字

版 次:2017 年 5 月第 1 版

2019 年 4 月第 2 版

印 次:2019 年 4 月第 1 次印刷

定 价:99.00 元

产品编号:081777-01

第 2 版序言

信息技术总是以飞快的脚步向前迈进着，世界各地的程序设计人员也一直兢兢业业持续不断地努力，使得各种软件的版本快速地演变，几乎到了目不暇接的程度。

在本书第 1 版上市不久，马上就面临了所有计算机书籍的宿命，书中用来示范的软件陆续开始升级成更新更高的版本。原本作者通过回复读者电子邮件以及在个人网页上新增文章说明的方式还算可以顺利地协助读者们解决问题，不过到 2017 年底，在一次教授《Python 数据分析》的课后，一位同学拿着本书的第 1 版前来询问 Django 部署问题时，我才真正意识到必须好好改版的这个事实。

平时的教学及研究生活较为繁重，好在有计算机与通信工程系林亮昀同学的协助，我才可以在比较短的时间内好好地更新、整理以及验证在 Python 3 和 Django 2.0 新架构下的教学材料，让这本书的“寿命”得以延长，有幸可以协助更多使用 Python/Django 架站的朋友们省去许多学习和摸索的时间。

在新改版的内容中，所有的程序代码都以 Python 3 重新编写，除了部分的章节因为模块的版本跟不上而继续保持使用 Django 1.11 版的架构外，其余内容均以 Django 2.0 版本为主。此外由于二级网络域名管理网站的构建对许多初学者来说实用性不高，所以就把第 14 堂课的内容置换为近来颇受欢迎的 WordPress-like Python/Django CMS 框架 Mezzanine，相信读者们看完之后对于 Django 会有更多想法与期待。

笔者已经尽力维持书上所有程序代码及操作示范过程的正确性，然而就像是所有的计算机书籍一样，许多第三方的服务网站以及模块套件各自有其改版的时间表，有些网站在对于已注册用户和初次登录的用户使用的界面并不完全相同，操作过程也不容易复制，因此在内容上也难免和各位读者自行学习时会有所差异，请大家在练习时稍加留意。

最后，感谢荣钦科技胡昭民总经理以及吴灿铭副总经理的厚爱与协助，使得本书得以如期改版发行。

前言

本书的主要目的在于介绍如何使用 Django 这个 Web Framework 在网络主机上架设一个全功能的网站。Django 是一个由 Python 编写的具有完整架站能力的 Web 网站框架，通过这个框架，只要短短几条指令，Python 的程序设计人员就可以轻松地建立一个正式网站所需要的骨架（框架），再从这个框架中开发出全功能的网站。

Python 语言充满了令人津津乐道的加速技巧，为了方便读者学习，本书尽量使用初学者容易理解的讲述方式，以期阅读本书的读者能够在最短的时间内跨过使用程序设计语言制作网站的门槛，马上以 Python 建立自己的特色网站，并在熟悉流程以及架构后，进一步提升网站的性能。

所以，只要你有 Python 的基本程序设计能力以及网站架构和运行的基本概念，基本上就有足够的能力通过本书来建立属于自己的动态网站——一个可以让你充分利用 Python 语言所有能力、连接数据库、使用社交网站账号验证机制、实时运算处理数据、充分实现所有“点子”的网站。

由于网站系统的版本更新迅速，因而本书所有网站范例均在更新的 Python 3.6 以及 Django 2.0 中测试无误（部分章节因为模块版本的问题，仍然是使用 Django 1.11 版），为了避免学习上的困扰，建议读者在学习时尽量以同样的版本练习（相同的主版本号即可），等熟练之后再根据需求升级版本。此外，一开始建立基本范例时也以本书提供的范例程序代码为主，等到有了一定的基础，再把读者自己的程序代码拿来使用，“在实践中学习”永远是学习程序设计的最佳方法。

改编说明

Python + Django 确实是迅速开发、设计、架设和部署网站的最佳组合。Django 是用如日中天的“胶水”语言 Python 写成的，是一个完全开放源代码的网站架构或网页框架（Web Framework）。Django 本身基于 MVC 模型，即 Model（模型）+ View（视图）+ Controller（控制器），因此天然具有 MVC 的出色基因：开发迅速、部署快、可重用性高、维护成本低等。

本书并非讲述如何使用 Python 程序设计语言进行网页的程序设计，也不是单独介绍 Django 框架及其核心组件，而是通过 16 堂课让读者迅速掌握使用 Python + Django 的最佳组合开发、设计和架设自己的网站，并部署到真实世界的因特网主机上，尽快投入实际运营。

本书跳过了一般 Python 程序设计语言教科书“事无巨细”的烦琐，也摒弃了普通 Django 参考书“细枝末节”的繁复，而是直截了当地教授读者逐步架设和部署一些实用的范例网站，如个人博客、投票网站、子域管理网站、名言佳句网站、电子商店网站等。读者可以在本书的指导下让这些实际可以投入使用的网站“鲜活”地出现在因特网上。

这些范例网站的源码和网站文件夹结构及其文件都打包在一个压缩文件中，下载网址为 <https://pan.baidu.com/s/1ma5k98tdK7m8uUi73SQ3kg>（注意区分数字和英文字母大小写）或者扫描右侧的二维码。如果下载有问题，请发送电子邮件至 booksaga@126.com，邮件主题设置为“求 Python 新手学 Django 架站的 16 堂课第 2 版代码”。



读者可以参照这些范例网站，按照本书各堂课的内容直接使用或者以它们为蓝本进行扩展设计和开发，最后将自己心仪的网站架设和部署到因特网上去。

因为涉及网站的部署，所以读者需要用自己的电子邮箱或者其他知名网站的 ID 去注册或申请网络域名以及网址，在实际部署本书的范例网站时替换掉范例中的网络域名或网址，这样才能让这些网站真正部署成功并且属于读者自己。具体步骤可以参考书中各堂课的相关内容。

最后祝大家学习顺利，早日成为使用 Python + Django 领域的“大师”！

资深架构师 赵军

2019 年 1 月

目 录

第 1 堂 网站开发环境的建立	1
1.1 网站的基础知识.....	1
1.1.1 网站的运行流程	1
1.1.2 Python/Django 扮演的角色	3
1.1.3 使用 Python/Django 建立网站的优势	4
1.2 建立网站开发流程.....	4
1.2.1 开发流程简介	4
1.2.2 在 Windows 建立 Linux 虚拟机.....	5
1.2.3 在 Mac OS 安装 Linux 虚拟机.....	12
1.2.4 在 Linux 虚拟机中创建 Python Django 开发环境.....	18
1.2.5 设置 SSH、PuTTY 以及 FTP 服务器	19
1.2.6 安装 Notepad++ 程序编辑器	23
1.3 活用版本控制系统.....	27
1.3.1 版本控制系统 Git 简介	27
1.3.2 申请 Bitbucket 账号	28
1.3.3 在虚拟机中连接 Bitbucket.....	30
1.3.4 在不同的计算机之间开发同一个网站.....	32
1.4 其他网站项目开发环境的安装建议.....	33
1.4.1 在 Windows 10 创建开发环境	33
1.4.2 在 MacOS 中创建开发环境	35
1.4.3 在 Cloud9 中创建开发环境.....	37
1.4.4 在 DigitalOcean VPS 中创建开发环境.....	39
1.5 习题.....	40
第 2 堂 Django 网站快速入门.....	41
2.1 个人博客网站规划.....	41
2.1.1 博客网站的需求与规划	41
2.1.2 产生第一个网站框架	42
2.1.3 Django 文件夹与文件解析.....	44
2.2 创建博客数据表.....	46
2.2.1 数据库与 Django 的关系	46

2.2.2	定义数据模型	46
2.2.3	启动 admin 管理界面	47
2.2.4	读取数据库中的内容	52
2.3	网址对应与页面输出	54
2.3.1	创建网页输出模板 Template	54
2.3.2	网址对应 urls.py	58
2.3.3	共享模板的使用	60
2.4	高级网站功能的运用	63
2.4.1	JavaScript 以及 CSS 文件的引用	63
2.4.2	图像文件的应用	67
2.4.3	在主网页显示文章摘要	68
2.4.4	博客文章的 HTML 内容处理	70
2.4.5	Markdown 语句解析与应用	73
2.5	习题	75
第 3 堂	让网站上线	76
3.1	DigitalOcean 部署	76
3.1.1	申请账号与创建虚拟主机	76
3.1.2	安装 Apache 网页服务器及 Django 执行环境	80
3.1.3	修改 settings.py、000-default.conf 等相关设置	82
3.1.4	创建域名以及多平台设置	84
3.2	在 Heroku 上部署	87
3.2.1	Heroku 账号申请与环境设置	87
3.2.2	修改网站的相关设置	89
3.2.3	上传网站到 Heroku 主机	90
3.2.4	Heroku 主机的操作	93
3.3	在 Google Cloud Platform 上部署	94
3.3.1	Google Cloud Platform 的介绍	94
3.3.2	Google Computing 启用与设置	98
3.3.3	Google App Engine 的说明与设置	101
3.4	习题	110
第 4 堂	深入了解 Django 的 MVC 架构	111
4.1	Django 的 MVC 架构简介	111
4.1.1	MVC 架构简介	111
4.1.2	Django 的 MTV 架构	112
4.1.3	Django 网站的构成以及配合	113
4.1.4	在 Django MTV 架构下的网站开发步骤	114
4.2	Model 简介	115

4.2.1 在 models.py 中创建数据表	116
4.2.2 在 admin.py 中创建数据表管理界面	118
4.2.3 在 Python Shell 中操作数据表	121
4.2.4 数据的查询与编辑	123
4.3 View 简介	125
4.3.1 建立简易的 HttpResponse 网页	126
4.3.2 在 views.py 中显示查询数据列表	127
4.3.3 网址栏参数处理的方式	128
4.4 Template 简介	131
4.4.1 创建 Template 文件夹与文件	131
4.4.2 传送变量到 Template 文件中	132
4.4.3 在 Template 中处理列表变量	135
4.5 最终版本摘要	135
4.6 习题	138
第 5 堂 网址的对应与委派	139
5.1 Django 网址架构	139
5.1.1 URLconf 简介	139
5.1.2 委派各个的网址到处理函数	141
5.1.3 urlpatterns 的正则表达式语法说明 (适用于 Django 2.0 以前的版本)	144
5.1.4 验证正则表达式设计 URL 的正确性	147
5.2 高级设置技巧	148
5.2.1 参数的传送	148
5.2.2 include 其他整组的 urlpatterns 设置	149
5.2.3 URLconf 的反解功能	149
5.3 习题	150
第 6 堂 Template 深入探讨	151
6.1 Template 的设置与运行	151
6.1.1 settings.py 设置	151
6.1.2 创建 Template 文件	153
6.1.3 在 Template 文件中使用现有的网页框架	154
6.1.4 直播电视网站应用范例	155
6.1.5 在 Template 中使用 static 文件	158
6.2 高级 Template 技巧	160
6.2.1 Template 模板的继承	160
6.2.2 共享模板的使用范例	162
6.3 Template 语言	163
6.3.1 判断指令	163

6.3.2 循环指令	164
6.3.3 过滤器与其他的语法标记	168
6.4 习题	171
第 7 堂 Models 与数据库	172
7.1 网站与数据库	172
7.1.1 数据库简介	172
7.1.2 规划网站需要的数据库	173
7.1.3 数据表内容设计	176
7.1.4 models.py 设计	177
7.2 活用 Model 制作网站	178
7.2.1 建立网站	178
7.2.2 制作网站模板	180
7.2.3 制作多数据表整合查询网页	182
7.2.4 调整 admin 管理网页的外观	186
7.3 在 Django 使用 MySQL 数据库系统	188
7.3.1 安装开发环境中的 MySQL 连接环境 (Ubuntu)	188
7.3.2 安装开发环境中的 MySQL 连接环境 (Windows)	189
7.3.3 使用 Google 云端主机的商用 SQL 服务器	192
7.4 习题	196
第 8 堂 网站窗体的应用	197
8.1 网站与窗体	197
8.1.1 HTML <form>窗体简介	197
8.1.2 活用窗体的标签	200
8.1.3 建立本堂课范例网站的数据模型	202
8.1.4 网站窗体的建立与数据显示	204
8.1.5 接收窗体数据存储于数据库中	206
8.1.6 加上删除帖文的功能	207
8.2 基础窗体类的应用	209
8.2.1 使用 POST 传送窗体数据	209
8.2.2 结合窗体和数据库	213
8.2.3 数据接收与字段的验证方法	216
8.2.4 使用第三方服务发送电子邮件	219
8.3 模型窗体类 ModelForm 的应用	223
8.3.1 ModelForm 的使用	223
8.3.2 通过 ModelForm 产生的窗体存储数据	225
8.3.3 为窗体加上防机器人验证机制	226
8.4 习题	231

第 9 堂 网站的 Session 功能	232
9.1 Session 简介	232
9.2.1 复制 Django 网站	232
9.1.2 Cookie 简介	233
9.1.3 建立网站登录功能	234
9.1.4 Session 的相关函数介绍	239
9.2 活用 Session	240
9.2.1 建立用户数据表	240
9.2.3 整合 Django 的信息显示框架 Messages Framework	246
9.3 Django auth 用户验证	249
9.3.1 使用 Django 的用户验证系统	249
9.3.2 增加 User 的字段	252
9.3.3 显示新增加的 User 字段	254
9.3.4 应用 auth 用户验证存取数据库	256
9.4 习题	261
第 10 堂 网站用户的注册与管理	262
10.1 建立网站用户的自动化注册功能	262
10.1.1 django-registration-redux 安装与设置	262
10.1.2 创建 django-registration-redux 所需的模板	263
10.1.3 整合用户注册功能到分享日记网站	267
10.2 Pythonanywhere.com 免费 Python 网站开发环境	271
10.2.1 注册 Pythonanywhere.com 账号	271
10.2.2 在 Pythonanywhere 免费网站中建立虚拟环境以及 Django 网站	278
10.2.3 建立投票网站的基本架构	283
10.3 使用 Facebook 验证账号操作实践	290
10.3.1 在 Pythonanywhere 中安装 django-allauth 与设置	290
10.3.2 到 Facebook 开发者网页申请验证机制	293
10.3.3 在网站中识别用户的登录状态	298
10.3.4 客户化 django-allauth 页面	301
10.4 习题	303
第 11 堂 社交网站应用实践	304
11.1 投票网站的规划与调整	304
11.1.1 网站功能与需求	304
11.1.2 数据表与页面设计	306
11.1.3 网站的转移	309
11.2.4 移动设备的考虑	311

11.2	深入探讨 django-allauth.....	312
11.2.1	django-allauth 的 Template 标签	313
11.2.2	django-allauth 的 Template 页面	314
11.2.3	获取 Facebook 用户的信息	316
11.3	投票网站功能解析.....	317
11.3.1	首页的分页显示功能	318
11.3.2	自定义标签并在首页显示目前的投票数.....	319
11.3.3	使用 AJAX 和 jQuery 改进投票的效果	322
11.3.4	避免重复投票的方法	327
11.3.6	新建 Twitter 账号链接.....	329
11.4	习题.....	334
第 12 堂	电子商店网站实践	335
12.1	打造迷你电商网站.....	335
12.1.1	复制网站, 不要从零开始	335
12.1.2	建立网站所需要的数据表	337
12.1.3	上传照片的方法 django-filer.....	341
12.1.4	把 django-filer 的图像文件加到数据表中	345
12.2	增加网站功能.....	348
12.2.1	分类查看产品	348
12.2.2	显示详细的产品内容	352
12.2.3	购物车功能	353
12.2.4	建立订单功能	357
12.3	电子支付功能.....	365
12.3.1	建立付款流程	366
12.3.2	建立 PayPal 付款链接	368
12.3.3	接收 PayPal 付款完成通知	374
12.3.4	测试 PayPal 付款功能	375
12.4	习题.....	381
第 13 堂	全功能电子商店网站 django-oscar 实践	382
13.1	Django 购物网站 Oscar 的安装与使用.....	382
13.1.1	电子购物网站模板	382
13.1.2	Django Oscar 购物车系统测试网站安装	383
13.2	建立 Oscar 的应用网站	386
13.2.1	安装前的准备	386
13.2.2	建立网站的域名	387
13.2.3	调整 Apache2 配置文件	388
13.2.4	建立 Django Oscar 购物网站项目	389

13.2.5	加上电子邮件的发送功能	397
13.2.6	简单地修改 Oscar 网站的设置	398
13.2.7	增加 PayPal 在线付款功能	401
13.3	自定义 Oscar 网站	406
13.3.1	建立自己的 templates, 打造客户化的外观	407
13.3.2	网站的中文翻译	416
13.4	习题	417
第 14 堂	使用 Mezzanine 快速打造 CMS 网站	418
14.1	快速安装 Mezzanine CMS 网站	418
14.1.1	什么是 Mezzanine	418
14.1.2	安装 Mezzanine	419
14.1.3	安装 Mezzanine 主题	425
14.1.4	Mezzanine 网站的设置与调整	427
14.2	使用 Mezzanine 建立电子商店网站	429
14.2.1	安装电子购物车套件与建立网站	429
14.2.2	自定义 Mezzanine 网站的外观	431
14.3	在 Heroku 部署 Mezzanine 网站	435
14.4	习题	440
第 15 堂	名言佳句产生器网站实践	441
15.1	建立网站前的准备	441
15.1.1	准备网站所需的素材	441
15.1.2	图文整合练习	442
15.1.3	建立可随机显示图像的网站	444
15.2	产生器功能的实现	446
15.2.1	建立产生器界面	447
15.2.2	产生唯一的文件名	449
15.2.3	开始合并随后产生图像文件	449
15.2.4	准备多个背景图像文件以供选择	453
15.3	自定义图像文件功能	458
15.3.1	加入会员注册功能	458
15.3.2	建立上传文件的界面	458
15.3.3	上传文件的方法	462
15.3.4	实时产生结果	464
15.4	习题	466
第 16 堂	课程回顾与你的下一步	467
16.1	善加运用网站资源	467

16.2	部署上线的注意事项.....	470
16.3	SSL 设置实践.....	472
16.4	程序代码和网站测试的重要性.....	483
16.5	其他 Python 框架	486
16.6	你的下一步.....	486

第 1 堂

网站开发环境的建立

本堂课将介绍如何通过版本控制系统 Git 以及虚拟机 (Virtual Machine) 来建立一个可以随时开发 Python Django 网站的环境。不管个人计算机的操作系统是 Windows、Mac OS 还是 Linux，也不管有几台计算机，只要掌握本堂课的教学内容，并充分加以练习，熟悉开发环境的建立以及流程，就会对日后开发和设计网站项目有很大帮助。

在本书的大部分内容中，我们都是以新版的 Django 2.0 进行示范，然而由于一些 Django 的第三方模块的适用版本还没有跟上新版的 Django 2.0，如果在相关章节中使用的是 1.x 版的 Django 框架时，作者会在适当的地方加以说明。

1.1 网站的基础知识

Web 网站可以简单到只要找到主机空间，把.html 和.jpg 文件放在适当的位置即可，也可以使用类似 WordPress 的 Open Source CMS 系统，通过安装以及设置的方式，全部以浏览器所提供的界面管理其内容，完全不用担心系统实际执行的程序是哪些，甚至可以使用 Python 这类程序设计语言自己设计一个完全客户化的动态网站内容。不管使用哪一种方式，只有了解 Web 网站运行的流程，才能够知道如何决定适合自己的网站构建方式。

1.1.1 网站的运行流程

架设网站的方式有许多种，最简单的方式就是把文件放在网络主机上，当浏览者通过浏览器连接进来的时候，网络服务器把这个文件提供给浏览器加以解析，再呈现或显示给浏览者，如图 1-1 所示。

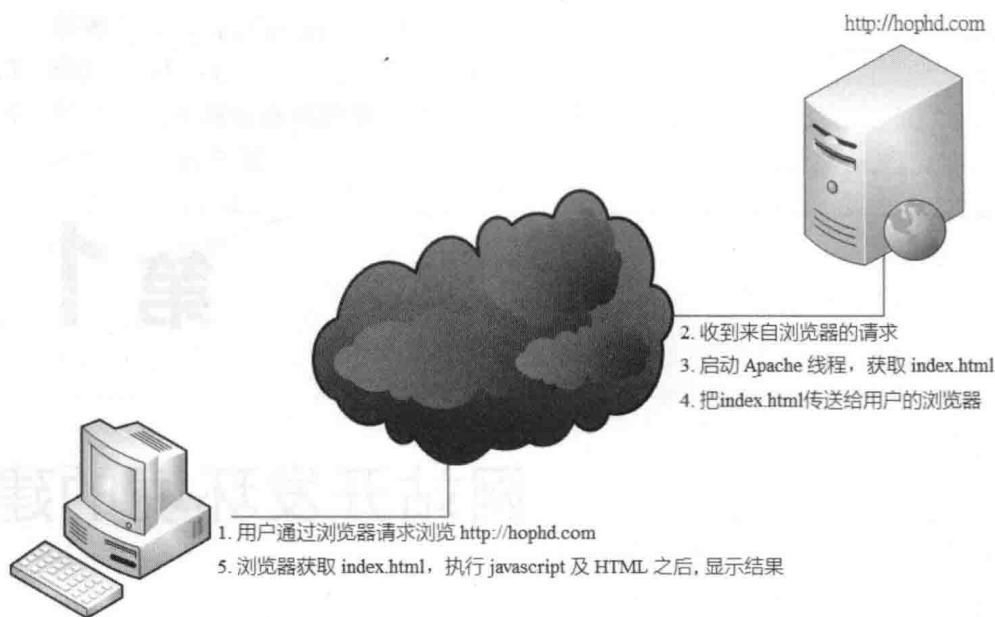


图 1-1 用户浏览网站的基本流程

图 1-1 除了给出硬件的网站主机外, 另一个重点是在主机上运行的 Web 网站服务器系统软件。常见的 Web 网站服务软件有 Apache、Nginx 以及 Windows Server 上的 IIS。也就是大家所知道的: 要让网站得以正常运行需要有一台网络主机, 而真正响应远程浏览器访问请求的是在主机中运行的服务软件。幸运的是, Apache 和 Nginx 在许多地方是兼容的, 而可供部署的主机操作系统还是以 Linux 为主, 因此本书之后的所有说明都是以 Linux 加上 Apache 环境为主。

前面所说的网站被称为静态网站。所有文件以及数据都不会针对用户的需求而临时产生, 都是事先准备好的。如果需要应对不同的访问而产生不同的数据, 甚至在显示数据前还要从数据库或其他数据源获取数据, 加以整合、计算、分析后再显示给浏览者, 这种情况下就有一些工作必须在网站主机上先执行, 这就是所谓的动态网站, 如图 1-2 所示。

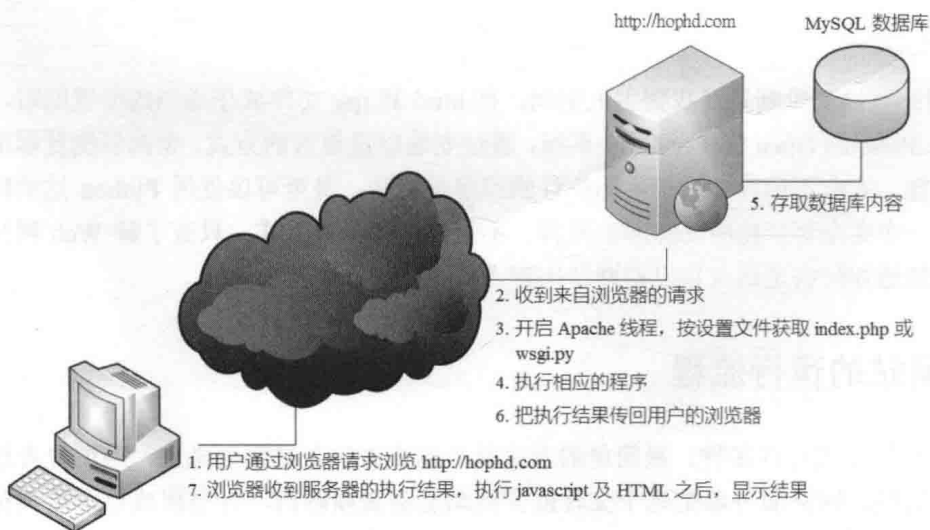


图 1-2 动态网站的运行流程

一个好的动态网站会通过浏览器给浏览者提供一个完整的输入界面，通常都是以窗体的方式呈现，并在浏览者设置数据、选择内容、按下提交（Submit）按钮之后，将得到的数据在网站主机上进行处理，再把筛选过的数据提供给浏览者查看。对于那些在主机上执行的程序，我们一般把它们称为后端程序或后端服务程序。

能在后端执行的程序，它们可以用多种程序设计语言来编写，常见的有 PHP、Java、JavaScript、Perl、Ruby 以及本书的主角 Python。

1.1.2 Python/Django 扮演的角色

从 1.1.1 小节的介绍可以了解到，浏览者输入一个网址，网站主机的软件服务器（下面都以 Apache 为例）在收到此请求后会先按照配置文件的内容决定把此网址请求交由哪一个文件来处理。如果没有特别的设置，一般的 Linux 虚拟主机环境都会以执行 PHP 程序为主。也就是说，如果想要以别的程序设计语言来执行网页的请求，就必须做好 Apache 的设置，安装所需的接口程序，在设置完毕后，理论上所有可以在服务器上执行的程序设计语言都可以当作后端程序的设计语言。只是在开始处理浏览器的请求时，表示处理的程序也要自行处理 HTTP（Hyper Text Transfer Protocol，WWW 网站的专用通信协议）的相关细节，对于没有相关模块可用的程序设计语言来说，其实是非常麻烦的。

所幸的是，这些常被用来处理网站的程序设计语言（如 Python、Ruby、Perl 等）已经有了许多相关模块可以使用，因而在设置这些网站后端程序设计语言的服务环境时就容易多了。进一步讲，针对许多现代网站的必备功能（如用户账号管理、窗体输入、网页输出样式、网站和数据库链接等），出现了所谓的 Web Framework（网站框架），让这些程序设计语言的网站开发者可以使用一些简单的命令生成一个网站的基本架构（或框架），然后遵循其框架的设计，系统且结构化地设计出正式的、可商用的多功能网站。其中，Ruby 程序设计语言有名气的当属 Ruby on Rails，Python 程序设计语言以 Flask 和 Django 的使用量多一些。

Django 是为网站开发人员设计并使用 Python 语言编写的网站框架。简单地说，它是可以协助程序设计人员迅速建立全功能网站的一组 Python 程序，通过 MVC（Model View Controller）概念把视图和控制逻辑分割开来，让程序设计人员可以尽量不用担心网站通信协议的琐碎细节而专心于要建立的网站功能。此组程序放在主机某一个特定的文件夹下，通过 Apache 的配置文件指定此组程序所在的位置，当有网页被存取时再次执行并返回结果给 Apache，最后传送到用户的浏览器中。

也就是说，每当主机接收到来自浏览器的连接请求时，Django 中的某一个程序文件就可以得到被执行的机会，我们就可以在这个程序文件中以 Python 语言来编写需要处理、运算、存取数据库等的程序代码，让网页的请求可以更加客户化，实时响应用户的需求，提供更多网站的服务。

因此，对于想要使用 Python 来建立网站服务的程序设计人员来说，只要学会了 Django 的架构内容以及运行原理，就可以充分运用 Python 实现字符串处理、数据库、图像绘图、商业统计、科学运算、数据分析、数据可视化、网页提取等相关功能，以及处理网站的相关细节，并提供更多网站服务功能。