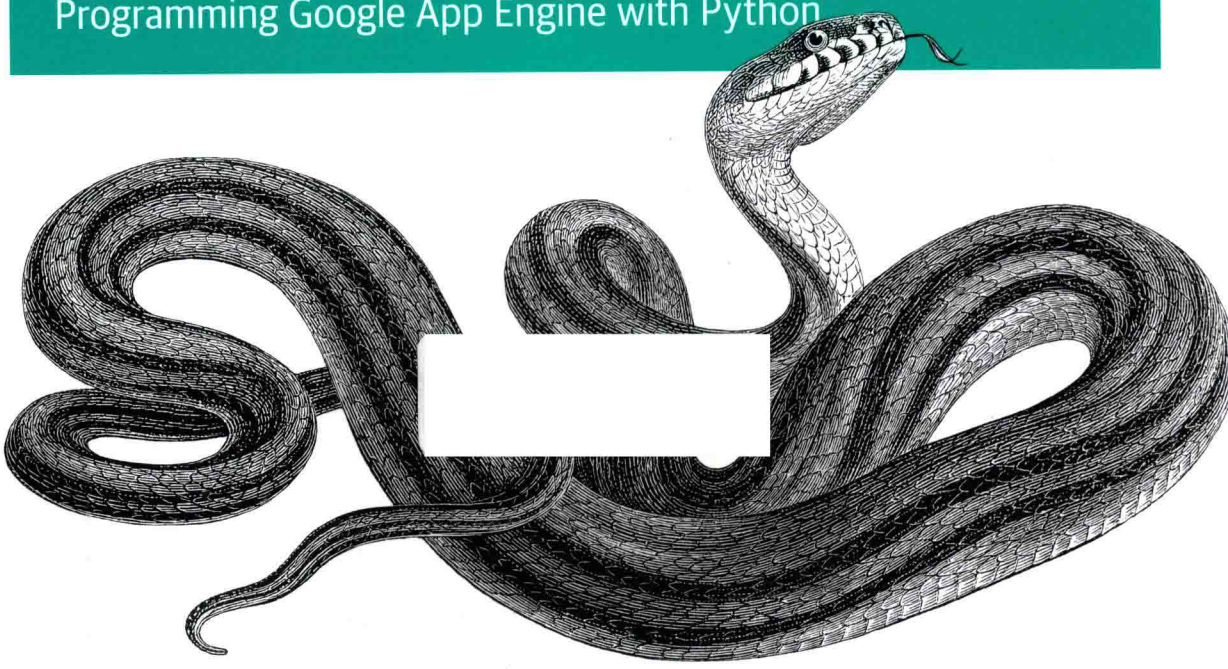
O'REILLY®

基于Python的Google App Engine编程

Programming Google App Engine with Python



Dan Sanderson 著
王晓莉 武凯旋 周勇 译

中国电力出版社

基于Python的Google App Engine编程

Dan Sanderson 著

王晓莉 武凯旋 周勇 译

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo O'REILLY®

O'Reilly Media, Inc. 授权中国电力出版社出版

中国电力出版社

图书在版编目（CIP）数据

基于Python的Google App Engine编程 /（美）丹·桑德森（Dan·Sanderson）著；王晓莉，武凯旋，周勇译. — 北京：中国电力出版社，2017.9

书名原文：Programming Google App Engine with Python

ISBN 978-7-5198-0681-1

I. ①基… II. ①丹… ②王… ③武… ④周… III. ①网页制作工具－程序设计

IV. ①TP393.092.2

中国版本图书馆CIP数据核字(2017)第083709号

北京市版权局著作权合同登记 图字：01-2016-9276号

出版发行：中国电力出版社

地　址：北京市东城区北京站西街19号（邮政编码100005）

网　址：http://www.cepp.sgcc.com.cn

责任编辑：刘 炽（liuchi1030@163.com）

责任校对：王开云

装帧设计：Ellie Volckhausen, 张 健

责任印制：蔺义舟

印　刷：北京天宇星印刷厂

版　次：2017年9月第一版

印　次：2017年9月北京第一次印刷

开　本：787毫米×1092毫米　16开本

印　张：26

字　数：491千字

印　数：0001—3000册

定　价：88.00元

O'Reilly Media, Inc.介绍

O'Reilly Media通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自1978年开始，O'Reilly一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了Make杂志，从而成为DIY革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项O'Reilly的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar博客有口皆碑。”

——Wired

“O'Reilly凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference是聚集关键思想领袖的绝对典范。”

——CRN

“一本O'Reilly的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照Yogi Berra的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去Tim似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

目录

前言

在互联网中，传播是迅速而短暂的。一个热门的新闻网站提到你的网站，立即可以给你带来300000个潜在客户，他们都希望了解你是谁以及你能够提供什么。但是，如果你只是一家刚起步的小公司，你的软硬件不太可能有能力处理这种流量。你明智地建立你的网站，来处理在你的前六个月里真正期望的每小时3万次的访问。在高负载下，该系统甚至无法将公司的logo展示给其他270000个用户。而在访问流量减弱后，这些潜在的客户不太可能再回来访问网站了。

解决这个问题的关键并不是在第一天就花费时间和金钱去建立能服务上百万的访问者的系统，因为该系统在随后的几个月里每天仅仅服务几千人。如果不及时开发一个大系统时，你会错过顾客的反馈来改善你的产品的机会。早期开发大系统的风险就是，开发了一些客户不想要的功能。

从历史上看，小公司从来不会在第一天就访问大的服务器。它们的最佳选择是建立小的系统，并且当它们成长的时候，系统的停用不会损害公司的名誉。幸运的公司能找到融资者，得到了新一轮的投资，并停止新功能开发来重建更大容量的产品。然而，有些公司并不是这么幸运。

现在，公司又有了其他的选择。大型互联网公司（例如亚马逊，Google和微软）会通过采用按次付费的模式来租赁部分大容量系统。公司网站的服务来自于这些有足够的能力处理突发流量的大型系统，且运行非常成功。由于你只要为你所使用的资源付费，所以在流量低的时候就不会有前期投资的浪费。随着用户的增长，成本也会成比例增长。

Google提供的系统统称为Google Cloud Platform，由一组高性能的服务和工具构成，包括各种规模的虚拟机，多种形式的可靠数据存储，可配置的网络，自动缩放架构，甚至是运行Google产品的大数据分析工具。但Google Cloud Platform（谷歌云平台）不仅仅提供Google架构的接入。它封装了应用架构的最好的实际应用，Google的工程师已经在自己的产品中反复使用过这些应用架构。

Google Cloud Platform的核心是Google App Engine，它是一个自动增长的应用托管服务。App Engine运行应用程序使每个访问的用户都可以得到和其他每个用户相同的体验，无论同时存在的用户是几十还是几千。应用程序代码只关注用户个人体验。App Engine负责大规模计算任务，例如负载平衡、数据复制和容错处理。

在传统的系统第一个数据库服务器过度增长时，可扩展的模型发挥了重要的作用。使用此系统，添加负载平衡的Web服务器和高速缓存层可以让应用程序更健壮，但是当你的应用程序需要在不止一个地方写数据的时候，你就面临着难题。当应用开发到依赖数据库软件的功能的阶段，且数据库软件不打算在多台机器上分布数据时，问题更严重。如果事先根据云平台（Cloud Platform）模型考虑数据，你可免于重建整个系统了。

经常被忽视的一个优点是，App Engine的执行模型有助于分布计算以及数据。App Engine擅长于将计算资源快速地分配成小任务。这种方式最初设计是用于优先响应客户的情况下来处理用户的网络请求。将此执行模型与Cloud Platform的任务队列服务相结合，中等至大的计算任务就可以被分解成并行执行的块。如果任务执行失败，系统会不断地重新尝试执行这些任务，直到成功为止，这使得任务弹性面对服务失败。这种执行模型使得设计者积极优化平台的并行性和强壮性。

在Google的框架上运行意味着不需要搭建服务器，更换有问题的硬盘，或解决网卡。你不必在半夜因为一个ISP的小问题发出的警报而被警报器叫醒。使用自动扩展，你也不必因为流量的增加而添加新的硬件。

Google Cloud Platform和App Engine让你专注于应用程序的功能和用户体验。使用云平台，你可以更早地推出系统，得到大量的关注，留住用户，并在用户的帮助下开始改善产品。应用程序的规模与用户规模增长到与谷歌级别成一定比例时，不必重建一个新的系统架构。与此同时，你的竞争对手仍然在绞尽脑汁地补救程序和配置数据库中。

在本书的帮助下，你将学会如何在Google Cloud Platform上开发Web应用程序，以及如何充分利用App Engine的可扩展的执行模型。本书的主要部分讨论谷歌的云数据存储（Google Cloud Datastore），它是一个强大的数据存储服务，与过去数十年在网络

开发中占主导地位的关系型数据库有所区别。应用程序模型和数据存储的结合代表了对于Web应用程序的新的思维方式，尽管几乎和熟知的模型一样简单，但仍旧需要重新思考我们之前忽略的一些原理。

App Engine的简史

如果你读到这里，可能会好奇为什么本书被称为Google App Engine编程，而不是Google Cloud Platform编程。简要的回答是，整个云平台的功能对于一本书来说太广泛了。特别是，计算引擎平台原始虚拟机能力，可以完成各种各样的功能，而不仅是服务于Web应用程序。

根据若干统计（至少是我的），App Engine开始是作为一个早期的云平台理念，后来发展到包括大型且可弹性的计算。当它于2008年第一次推出的时候，App Engine托管Python编写的Web应用程序，它具有可扩展的数据存储、任务队列服务和应用程序代码的API，这些API将在运行应用代码（如网络访问）的“容器”之外。Java“运行时环境”紧随其后，使用相同的可扩展的框架运行基于Java Servlet的Web应用程序。容器化的程序代码、无模式数据存储和面向服务的架构，证明它不仅是一种建立可扩展Web应用程序很好的方式，还是使App Engine产品可靠的一个关键部分（不会再报假警）。

App Engine不断发展，在功能方面有几个主要的里程碑。第一个里程碑是数据存储的大升级，采用了一种新的基于Paxos的复制算法。新算法改变了API保证数据一致性的方式，因此它被作为一个选择迁移（包括自动迁移工具）发布。另一个重要的里程碑是由CPU使用计费的独立请求，转变为通过实例的运行时间计费的长期运行的应用程序实例。使用升级的执行模型，应用程序代码可以推动准备工作发生在用户的请求逻辑之外，并且利用本地存储高速缓存。

Google将计算引擎作为一个单独的产品，是为了通用目的按需访问计算的方式。使用计算引擎虚拟机（Compute Engine VM），你就可以运行任何64位基于Linux的操作系统，并将任何语言编写的执行代码编译（或解释）到该操作系统。应用程序是否在App Engine上运行，都能调用计算引擎（Compute Engine）启动任何数量的虚拟机，完成任务，并且不再需要时关闭机器，或按传统的或自定义配置里让它们运行。

App Engine和Compute Engine采取不同的方法来提供不同的功能。但这些技术已经开始融合。2014年年初，Google宣布托管虚拟机（Managed VMs），以一种类似于App Engine方法的新方式来运行基于VM的代码（当我写这本书的时候这个功能不是完全可用的，但可以检查Google Cloud Platform网站的更新）。总之，你可以尽可能多地