《代码英雄》第二季（7）：无服务器

代码英雄讲述了开发人员、程序员、黑客、极客和开源反叛者如何彻底改变技术前景的真实史诗。

什么是《代码英雄》

Command Line Heroes

代码英雄是世界领先的企业开源软件解决方案供应商红帽（Red Hat）精心制作的原创音频播客，讲述开发人员、程序员、黑客、极客和开源反叛者如何彻底改变技术前景的真实史诗。该音频博客邀请到了谷歌、NASA 等重量级企业的众多技术大牛共同讲述开源、操作系统、容器、DevOps、混合云等发展过程中的动人故事。

本文是《代码英雄》系列播客第二季（7）：无服务器的音频脚本。

导语：无服务器Serverless到底意味着什么？当然，总得有服务器存在 —— 构建网络的基本架构不会改变。不过在将服务器交给一小部分人运维之后，开发者们会发生什么变化呢？

无服务器编程让初学者们可以更加轻松简单地部署自己的应用程序，让工作更有效率，这是它的优点。Andrea Passwater 跟我们分享了抽象底层架构会给我们带来多大的便利。不过凡事必有代价，无服务器化也有很多问题。Rodric Rabbah 解释了无服务器意味着你将部署和回应问题的能力拱手献出 —— 这就是为什么他帮助创建了 Apache OpenWhisk，这是一个开源的无服务环境框架，同时 Himanshu Pant 也来分享了他对于何时应该使用无服务器服务的观点。

empowerment

无服务器编程应该是为开发者们赋能授权的。我们也应该对于全局场景保持关注 —— 尽管我们精简了我们的工具箱。

00:00:03 - Al Gore 档案：

现如今，当然了，在全美乃至全世界，互联网正在彻头彻尾地改变着我们的生活。

00:00:13 - Saron Yitbarek：

那是 1998 年。 Google 才刚刚雇佣了它的第一名员工，副总统 Al Gore 在接受媒体采访。

00:00:22 - Al Gore 档案：

这项技术还处于起步阶段。当我和比尔·克林顿（Bill Clinton）总统入主白宫时，只有 50 个网站。现在看看，我在生日那天收到了一束虚拟鲜花。

00:00:37 - Saron Yitbarek：

好的。我已经感觉到你的眉毛皱起了。为什么我要向你展现某些 20 年前的互联网史？这是因为我想要提醒你，互联网的基础仍然是相同的。

00:00:51：

当然，现在有不止 50 个站点了，我知道。但是，我们仍然在发送虚拟鲜花。从开发人员的角度来看，如果你将我们所有的惊人进步剥离开来，你得到的仍然是相同的“客户端 - 服务器”（client-server）（C/S）模型，这就是一切的开始。一个提供了分布式网络的客户端 - 服务器模型。

00:01:16：

如今，开发人员谈论了很多有关无服务器（Serverless）的问题，这听起来像是 Al Gore 谈到的客户端 - 服务器模型被废弃了。而且，如果我们不小心，我们能够抽象出太多的基础架构，以至于忘记了仍然有服务器在那里做着它们的工作。

00:01:37：

但是，无服务器真的意味着没有服务器吗？真的吗？还是开发人员与服务器之间的关系正在变化？在这一集中，我们将与来自世界各地的人们交谈，以探索这种被称为“无服务器”(serverless)的东西。

00:01:54：

我是 Saron Yitbarek，这里是《代码英雄》，一档来自红帽的原创播客节目。

00:02:03 - Andrea Passwater：

你知道无线网络在某些地方还有线缆吗？

00:02:06 - Saron Yitbarek：

Andrea Passwater 在一家名为 …… 嗯…… “无服务器” 的公司工作。他们创建了一款流行的开源框架来开发无服务器应用程序。Andrea 注意到了各个组织是多么渴望抽象化基础架构的方法，而这正是神奇的“无服务器”一词始终给予人希望的地方。

00:02:28 - Andrea Passwater：

我认为这一术语主要是为了传达这样一个事实，即如果你是从事无服务器应用方面工作的开发人员，你不必考虑那些服务器。你只需要写代码并将代码部署到云提供商即可，而不必担心管理。这就是无服务器的真正含义。

00:02:49 - Saron Yitbarek：

对于 Andrea 来说，无服务器的吸引力很明显。

00:02:53 - Andrea Passwater：

倘若你以无服务器的方式开发应用程序，则可以不必去考虑部署和维护该应用程序的日常工作。这意味着你可以专注于它的商业价值，你可以专注于发挥创造力。

00:03:12 - Saron Yitbarek：

而无服务器的另一大好处是，你不太可能发现自己在重复造轮子。

00:03:18 - Andrea Passwater：

当有像 Auth0 这样可以直接使用的服务存在时，为什么要创建自己的身份验证方法呢？归根结底，无服务器就是为开发人员提供机会，使得他们能够更加轻松快速地构建起能把他们脑子里的所有的主意带到世界上的程序。

00:03:41 - Saron Yitbarek：

我明白了！

00:02:27：

想象一下，你拿了满手的东西，正跌跌撞撞地走向一扇门。这扇门滑开了，以简单、友好……

00:03:50：

（让我来说）

00:03:51：

……的方式。这就是无服务器。它为你打开了大门，使得开发工作不再那么繁琐。事实上，随着各个组织趋向于混合云环境，以及无服务器运动的发展，开发的障碍正在消失。

00:04:09：

Andrea 听说过很多有关非开发人员进行开发的话题。

00:04:15 - Andrea Passwater：

这是传统上认为自己写不了代码，而现如今由于无服务器而得以投身于软件工程的人的故事，并且能够开发这些使得他们自己的工作流程和类似的东西自动化的工具。这与你你做什么工作都没关系。

00:04:31：

你在工作中要做的一些事情是如此的呆板无聊，比如你每天都在做的例行事情，和那些你会想“难道计算机不能为我做这件事吗？”的事情。我开始有了这种感觉的时候，我碰巧在一家名为“无服务器”的公司工作，他们像这样：“你意识到我们制作的产品可以为你提供帮助，对吗？”

00:04:50 - Saron Yitbarek：

Andrea 认为，不久之后，许多从未将自己视为开发人员的人将意识到他们能够自己构建简单的应用程序，基本上免费。

00:05:02 - Andrea Passwater：

借助 Lambda，我从不需要为自己制作的任何小型应用程序付费。我可以让这些机器人为我做一部分工作，是的，我可以提高工作效率。但是，我也不必再做这些无聊的工作了。我可以做些更有趣的事情。

00:05:17 - Saron Yitbarek：

即使是对于专业开发人员来说，这种自动门效果在满手杂物的世界里也是很诱人的。

00:05:25 - Andrea Passwater：

我认为人们对于能够让一两个人的团队，在短时间内就让原型工作起来很感兴趣。在几天时间内，他们就可以启动并运行原型。我认为这使得人们开始意识到，他们可以专注于驱动他们的应用、产品和公司中的商业价值的部分。这非常让人兴奋，他们可以专注于商业价值。

00:05:54 - Saron Yitbarek：

我要再抛出一个术语给你。准备好了吗？功能即服务（Functions-as-a-service FaaS）。就像是 AWS Lambda 或 Apache OpenWhisk 之类的无服务器产品。“功能即服务”意味着，只有在被触发时程序才会执行某个功能，这效率更高。

00:06:15：

此外，这让我对计算能力和运行时间的担心少了很多。最终，无服务器可能会成为一个相当不错的基础配置。事实上，有些人甚至开始怀疑，我们是否要完全使用无服务器？它可以替代容器吗？

00:06:34 - Michael Hausenblas：

我理解这这种想法。

00:06:35 - Saron Yitbarek：

Michael Hausenblas 是 Red Hat OpenShift 团队的开发倡导者。

00:06:41 - Michael Hausenblas：

如果你看一下我们现在拥有的这些东西，包括 OpenShift 和 Cloud Foundry 和一些其他东西，你实质上就拥有了抽象化。基本上，Heroku 或多或少地倾向于向这个想法。对吗？这是非常简单的方式，无需担心程序会如何运行，无需担心它是什么样的。只需要给我们代码，我们来处理剩下的工作。

00:07:03 - Saron Yitbarek：

是的。听起来相当不错。这听起来有点儿像是梦想中的“无运维(no-ops)”环境，一切都自动化并且抽象化了，就像是开发者版本的极简主义室内设计。很棒、很干净的界面。

00:07:21：

但是， Michael 想要让你了解你一些现实情况。

00:07:25 - Michael Hausenblas：

没有运维！是吗？你知道，它只是神奇地以某种方式消失。你可以在 HackerNews 和 Twitter 以及其他任何地方看到这些笑话。无服务器？当然有服务器！我知道，当然有。而且也肯定有运维。

00:07:39：

总得有人去做这些，总得有人去架设服务器、给操作系统打补丁、去创建容器镜像，因为，你猜猜这些功能会在哪里执行？当然是在某种计算机上。

00:07:54 - Saron Yitbarek：

这不是零和博弈。功能即服务无法直接取代容器，而是在工具箱中增加了一个工具。我还有更多的事情要告诉你。通过使用这种新工具，转变成无服务器并不只是意味着运维就是其他人的事情，你仍然需要自己考虑自己的运维。

00:08:14 - Michael Hausenblas：

你会看到在基础架构侧有一点运维工作。但是，也有一点是开发人员的事情。如果你处在一个极端情况之下，比如说使用 Lambda，那么你是没有任何任何类型的管理员权限的，对吧？

00:08:29：

你不能简单地致电或是短信给一名基础架构管理员。显然，你组织之中的某一个人就必须得做这件事。但是，我担心许多组织只看到了它是如此简单而便宜。我们无需动这个，还有这个、这个，然后忘记了谁在待命，谁是真正地在待命？你对此有什么策略吗？

00:08:52：

如果没有的话，那么你可能会想要在进行之前，先制定一个策略。

00:09:00 - Saron Yitbarek：

需要有人处于待命状态。即使选择了“无服务器”，你仍然需要在头脑中萦绕更大的场景，你仍然需要让你的运维有序进行。

00:09:24：

在我早先时候抛出那个“功能即服务”术语时，你是不是有过些许畏缩？过去，基于云的开发为我们带来了大量的 “xx 即服务”的术语。我们有基础架构即服务（IaaS）、平台即服务（PaaS）、软件即服务

（SaaS）、数据即服务（DaaS）、数据库即服务（DBaaS）……等等。

00:09:48：

如果你难以理解它们的不同，那你并不孤单。这就是我们找来了 Himanshu Pant 的原因。他是位于印度德里的苏格兰皇家银行的技术主管。他花了多年时间来分析其中的差异。

00:10:04 - Himanshu Pant：

这些其他的计算范例在名称上和无服务器听起来是如此的相似，以至于人们往往会忘记，或者困惑于为什么没有将它们称之为无服务器，或者为什么这个被称为无服务器。

00:10:20 - Saron Yitbarek：

因此，无服务器与容器不同，无服务器也不是平台即服务。但是 Himanshu 希望将其明确一下。功能即服务能够提供什么？又不能提供什么？

00:10:35：

他与我们分享了两件轶事，有两次他弄清楚了什么时候该使用无服务器，什么时候应该放弃。第一次来自一个 24 小时黑客马拉松。Himanshu 当时正试图开发一个聊天机器人。

00:10:49 - Himanshu Pant：

有各种各样的指标会影响它的选择。例如逻辑覆盖率、可能产生的成本以及可伸缩性。而我选择在无服务器环境下完成这项工作。

00:11:04：

当我在开发它的时候，我意识到成本是一个层面，而这确实是我所青睐的技能。因此，即使其他所有的参与者都有更好的……我想说的是，覆盖率，或者说是逻辑覆盖率。这里讲的是 NLP 语境或其场景。

00:11:19：

但是，就成本和可伸缩性而言，我是手操胜券的，因为借助无服务器，这完全取决于人们在该聊天机器人上所进行调用的次数，并相应的触发该功能。这是一个我十分乐意采用无服务器的用例，因为成本——没有成本。以及更快的开发时间，而且老实说，当时还并不完全是生产规模级别的工作负载。

00:11:45：

我可以使用平台上的某些新兴工具。这对我而言是一次胜利。

00:11:52 - Saron Yitbarek：

很好。那时无服务器才有了意义。但是，在 Himanshu 目前供职的银行里，人们正在将他们的系统从旧版迁移到云端。而这提出了不同的目标。

00:12:07 - Himanshu Pant：

我们正在尝试查看哪些工作负载适用于哪些范例。比如 IaaS、BaaS、FaaS，这显然是属于企业领域的。你要看到这些方面，比如说第一，可靠的供应商难以寻找，以及第二，该技术应该得到广泛的验证。这对于像是银行业这样的规避风险的行业而言更是如此。

00:12:30：

这就是平台即服务（PaaS），但是仍然需要更好的证明、更好的功能，以及它们比传统工具更优越的地方。

00:12:40 - Saron Yitbarek：

Himanshu 正在研究自己的需求以及他自己的舒适区，并且研究哪些工作负载在哪种云计算规范中是有意义的。

00:12:49 - Himanshu Pant：

假设某个听众在一家贸易商店工作，他想构建某种东西，比如说一个入口。对于他或者她来说，无服务器可能并不是真正合适的选择，因为在那种在特定机器的应用程序中，延迟可能是不该出现的。

00:13:05 - Saron Yitbarek：

归根结底，这是一种有节制的做法，而不是将所有东西都丢进一个桶里。当我们思索哪一种基于云的架构是真正我们所想要做的工作时，还有一件事情需要考虑。所有这些抽象的东西，所有解放你双手的东西，最终如何改变的不仅仅是我们的工作方式，还改变了完成工作本身。

00:13:31：

抽象掉一部分工作负载可能意味着更少的自定义选项。想象一下你购买的一辆车。它能工作，它能开。但是接着想象一下你自己组装的一辆车，这辆车会按照你决定的方式工作。

00:13:48 - Rania Khalaf：

这是有代价的。

00:13:50 - Saron Yitbarek：

Rania Khalaf 是 IBM 研究部门的 AI 工程总监。

00:13:56 - Rania Khalaf：

在使用这些无服务器应用程序的过程中，你可能无法完全控制所有正在发生的事情。你无法控制全盘计划，或是程序何时何地运行。

00:14:06 - Saron Yitbarek：

这是一种权衡，对吗？当你使用无服务器时，细粒度控制可能会失误。

00:14:13 - Rania Khalaf：

它对于终端用户而言，抽象化了如此之多的东西，以至于你想要拥有更多的控制权、不同的规划、更多的检查与平衡、功能可以运行多长时间的不同值，等等等等。那么，如果你真的希望能够进入系统并修补，也许你可以创建你自己的部署环境。

00:14:32 - Saron Yitbarek：

不过，这将需要一些新东西，一类新的无服务器环境，开源社区已经在为自己打造了它。Rania 和她的 IBM 团队参与了该运动。

00:14:44 - Rania Khalaf：

我们首先研究是一种语言……它基本上是 JavaScript 的扩展，可以让你创建这些多线程交互服务的组合，以此作为起点，为你提供一种更加轻量级服务的方式。大约在同一时间，云和微服务以及平台即服务开始真正兴起。

00:15:08：

仅仅是将这两种趋势结合起来，就可以用可能来自于你，也可能来自其他人的许多小部件，构建更加高阶的功能。

00:15:18 - Saron Yitbarek：

Rania 和她的团队正在构建 Apache OpenWhisk，一款开源的功能平台。

00:15:23 - Rania Khalaf：

对于 OpenWhisk，我们从一开始就开源了。其中很大的原因是，为了让社区和我们一起参与进来。但是同时也是为了揭掉外包装，将控制权交给想要运行自己的无服务器计算环境的人们，以便他们能够根据自己的需求对其进行自定义，也许将它们置身于自己的控制之中，看看它实际上是如何运行的，以对其进行更好的控制。

00:15:54：

而且，我们还可以提供更加精细的控制，如果仅仅是普通服务，人们就不会有这种控制。

00:16:03 - Saron Yitbarek：

将控制权交还给想要运行自己的无服务器运行环境的人。这是下一阶段的无服务器。加入 OpenWhisk，你将获得像是 Fission 和 Gestalt

之类的其它开源平台。我们开始看到无服务器领域正在演变得比原先更具适应性，而且功能更为强大。

00:16:31：

为了真正了解为什么开源版的无服务器很重要，我与 OpenWhisk 的创始人之一进行了谈话。

00:16:39：

嗨，Rodric。最近好吗？

00:16:40 - Rodric Rabbah：

很好。你好吗？谢谢你邀请我参与节目。

00:16:42 - Saron Yitbarek：

Rodric Rabbah 是构思并创立 OpenWhisk 的三位开发者之一。以下是我们的谈话。

00:16:54 - Rodric Rabbah：

别人可能会很困惑，也可能会窃笑，因为人们可能会想：“倘若没有服务器，你要怎么做计算呢？”

00:17:02 - Saron Yitbarek：

是的。服务器就在某处，只是我不必去费心考虑它。

00:17:05 - Rodric Rabbah：

完全正确。这就是这个模式的真正美妙之处。当你开始以无服务器的方式进行开发时，你就再也不想回到过去了。你知道的，如今我已经置身其中接近 4 年了，并且已经开发了一些达到生产质量的应用程序。

00:17:19：

这是我如今惟一的开发方式。如果你告诉我必须要配置一台计算机并且安装操作系统，这对我而言完完全全是陌生的。我甚至都不确定我是不是还知道该怎么做。

00:17:29 - Saron Yitbarek：

是的。当你这样说的时候，听起来像是减轻了很大的负担。你知道吗？当最初听说无服务器时，至少我会想：“伙计，我必须要去学习的事又多了一件。”

00:17:38：

但是，当你这样说的时候，听起来不错。

00:17:41 - Rodric Rabbah：

这确实听起来很棒。然而，你应该已经意识到你必须要从这幻想的气泡中抽出一点儿空气。它不是万能药。

00:17:50 - Saron Yitbarek：

有哪些令人惊讶的风险或问题是人们在起步时可能没有看到或意识到的呢？

00:17:58 - Rodric Rabbah：

我认为缺乏透明度可能是最大的问题。这有点儿让我想起了新语言问世时出现的，那些提高了计算机抽象水平的技术。在当今的无服务器环境中，这是一种类似的令人震惊的效果。

00:18:16：

在这个过程中，你通常会写一个功能，然后只需部署这个功能即可。它可以立即运行，比如在 web 上作为 APIN 点。它可以大规模伸缩。我的意思是你无需自己做任何工作即可运行数千个实例。

00:18:32：

但是，倘若哪里出了问题，那应该如何调试呢？或者我实际上是想要检查我的功能失败的上下文环境。通常，这些功能在进程内运行，与你隔离 —— 你甚至无法登录计算机查看你的代码在何处运行。它们可能在封闭的容器环境之中运行，你不知道里面有什么。

00:18:53：

获得一点儿透明度对你而言变得很困难。这是工具最终将提供帮助的地方。但是，工具的缺乏某种程度上会让人们陷入一个相当大的陷阱。

00:19:05 - Saron Yitbarek：

这真的很好。那么让我们回到 OpenWhisk。请给我说说关于它的事情。

00:19:11 - Rodric Rabbah：

该项目在 Amazon Lambda 宣布推出产品的那一刻就开始了，这实际上是无服务器开始成为术语，并且开始在该领域获得关注的时刻。当我们看到 Lambda 时，我们开始思索：“这里有许多技术需要开发。不仅仅是在新的云计算的基础层上，而且在置于其上的编程模型之上，这实际上都使得它更易于被程序员访问。”你知道，由于出自 IBM 研究所，我们拥有相当强大的技术，不只是编程语言设计、编译器专业知识以及运行时的专业知识的技能。

00:19:54：

我们的一个小团队，基本上三个人……

00:19:57 - Saron Yitbarek：

哇。

00:19:57 - Rodric Rabbah

……聚集在一起，做了最初的开发和原型，最终成为 OpenWhisk，带有命令行工具，这是现如今无服务器实际上的编程接口。编程模型概

念，然后是它必须支持的实际架构，本质上，是服务器模型的这个功能，提供了无服务器所支持的所有好处。

00:20:22：

请注意，真正的起源是 Amazon Lambda 的出现，并可以说这是一种新的计算模型。

00:20:28 - Saron Yitbarek：

那么花了多长时间？或者说是第一个版本什么时候出现的。

00:20:30 - Rodric Rabbah：

实际上很快。事实上，当 IBM 宣布……好吧，那时还是 IBM OpenWhisk。从我们第一次提交到现在才一年。

00:20:39 - Saron Yitbarek：

哇。我的天哪。

00:20:41 - Rodric Rabbah：

这着实令人激动。

00:20:43 - Saron Yitbarek：

这确实很令人印象深刻。事实上，当它第一次启动时，它不叫 OpenWhisk，而是 Whisk。对吗？

00:20:49 - Rodric Rabbah：

Whisk 是内部名称，没错。我取的这个名字。这个名字背后的意思是迅速而又灵活地行动。

00:21:00 - Saron Yitbarek：

很好。

00:21:01 - Rodric Rabbah：

你“搅拌”了一个功能，就可以了。你可以将其放入烤箱中烘焙。

00:21:07 - Saron Yitbarek：

太好了，因为当我看到它，我肯定想的是鸡蛋。我在想，让我们“whisk搅拌”一些鸡蛋。

00:21:12 - Rodric Rabbah：

对。我们对该名称进行了一些正面和负面的评价。当我们开源一项技术，并将其放到 GitHub 上时，我们会在上面加上 open 前缀，以强调该技术与开源一样开放，可以自由使用、自由下载、自由贡献。

00:21:32 - Saron Yitbarek：

是的。

00:21:33 - Rodric Rabbah：

我们将其开源的目的，实际上是一定程度上提升可以在当今无服务器平台上执行标准。对我们来说，重要的是要建立一个平台，不仅可以用于生产环境，还可以与全世界共享，而且还可用于学术研究或一般性研究。也许因为出自 IBM 的研究机构，我们有点儿在意这个。

00:22:00：

但是，这是有所回报的，我知道一些大学 —— 从 Princeton 到 Cornell —— 在他们的研究中使用 OpenWhisk。我去过 Brown、Williams College、 MIT、 CMU 等几所大学，并且进行了一些讲座，目的是鼓励学生真正地去研究围绕无服务器以及服务器功能的问题、工具、编程模型，并且使他们对技术感兴趣。

00:22:26：

向他们显示，如果他们真的为开源项目做出了贡献，那么有一条路可以让 IBM 的云功能接受它并在生产环境中运行，通常在一周之内。

00:22:34 - Saron Yitbarek：

哇。这么快。

00:22:36 - Rodric Rabbah：

这让一些人感到惊讶。

00:22:38 - Saron Yitbarek：

这是一个非常高效的过程。

00:22:41 - Rodric Rabbah：

这确实证明了我们是如何在开源环境下开发许许多多技术的。这不是一个开放核心模式，有些组件有所保留。实际上在 IBM 云之中所运行的就是 Apache OpenWhisk 项目。

00:22:56 - Saron Yitbarek：

当你思索无服务器的未来，以及我们所选择的前进道路时，你觉得它们将是不可避免地奔向开源吗？

00:23:08：

我认为最近对于开源的价值存在一场激烈的争议，尤其是在云计算领域。

00:23:13 - Saron Yitbarek：

是的，没错。

00:23:15 - Rodric Rabbah：

如果你在思考为什么人们会转向云计算，或者为什么他们可能会厌恶投身于云计算领域，这就是供应商锁定的问题……丧失透明度。开源在一定程度上缓解这些问题，发挥了重要的作用。然后再看看类似 Kubernetes 的服务，它只是在容器和系统管理方面吞噬了云，就那么的成功！

00:23:41 - Rodric Rabbah：

如果你正在做的事情接触到了容器，鉴于它的主导地位，保持闭源与否还值得讨论吗？我倾向于认为开放性是有帮助的，从开发人员的角度来看，这很有吸引力。

00:23:57 - Saron Yitbarek：

当你考虑未来的无服务器生态及其工具、项目以及服务时，你看到了什么呢？对你来说，无服务器的未来是什么样的？

00:24:08 - Rodric Rabbah：

我认为，你会开始越来越少思考底层技术，而变得越来越多地考虑编程体验以及围绕它的工具：用于调试的、用于部署管理的、用于性能分析的、用于安全性。

00:24:26：

我认为，所有这些都非常重要。你如何运行你的功能的底层机制 —— 无论它们是在容器中还是在一些未来的技术下运行，也无论你是将它们运行在一个云上或是多个云上 —— 我认为，它们都不重要。有点儿像是 Kubernetes 在容器以及容器管理方面所做的事情。

00:24:46：

类似的还有一层要放在上面，就是服务分层的功能，可以实现那种无服务器概念。然后，它实际上的表现取决于你在其中安放的中间件。你如何赋能授权给开发者，让他们真正利用这款新的云端计算机，以及你要在它周围投入的辛劳，让他们的体验愉快。

00:25:07 - Saron Yitbarek：

是的。这种赋能授权看起来怎么样？

00:25:13 - Rodric Rabbah：

一言以蔽之，就是高效。这是一种能力，可以只专注于对于我，作为一名开发人员而言有价值的东西，或者如果我在公司工作的话，对于

我公司的价值。这样能够更快地规避问题，因为你释放了你的脑细胞，而不用去考虑基础设施和如何伸缩，以及在硬件层面如何保障事物的安全性。

00:25:38：

现在，你可以真正地创新，将脑力重新投入到更快的创新中去，从而为你的终端用户带来更多的价值。我想把这一切都归结于更高的效率。

00:25:55 - Saron Yitbarek：

Rodric Rabbah 是 OpenWhisk 的一位创始人。还记得我在这一期节目开始的时候所说的吗？互联网所基于的那种老旧的客户端 - 服务器模型并不会消失。改变的是，我是说彻底改变的是我们对服务器的思考方式。

00:26:19：

在所谓的无服务器世界之中，希望我们专注于代码本身，而不用担心基础架构。但是，我们所选择的抽象等级，以及如何保持对于未被抽象的工作的控制，才是使得无服务器世界变得真正有趣的地方。

00:26:40：

无服务器最终应当是为开发人员赋能授权的。免于打补丁、进行扩展和管理基础设施。但是，与此同时，即使我们抽象化了一些任务，依然必须对整体如何工作保持关注。我们会问，我要放弃的是哪些控制权，而我要收回的又是哪些控制权？

00:27:07：

下一集是我们史诗般的第二季的大结局，《代码英雄》将要前往火星。我们将会了解 NASA 的火星探测器如何开始自己的开源革命。并且我们将与 NASA 喷气推进实验室的 CTO 进行交流，了解开源是如何塑造太空探索的未来的。

00:27:39 - Saron Yitbarek：

与此同时，如果你想要更加深入地研究无服务器开发的问题，或是在这一季里我们所探索过的任何主题，请访问 redhat.com/commandlineheroes，查看免费资源。当你在那里时，你甚至可以为我们自己的代码英雄游戏作出贡献。

00:28:00 - Saron Yitbarek：

我是 Saron Yitbarek。感谢收听，编程不已。

> 什么是 **LCTT SIG** 和 **LCTT LCRH SIG**
>
> LCTT SIG 是 LCTT Special Interest Group 特别兴趣小组，LCTT SIG 是针对特定领域、特定内容的翻译小组，翻译组成员将遵循 LCTT 流程和规范，参与翻译，并获得相应的奖励。LCRH SIG 是 LCTT 联合红帽（Red Hat）发起的 SIG，当前专注任务是《代码英雄》系列播客的脚本汉化，已有数十位贡献者加入。敬请每周三、周五期待经过我们精心翻译、校对和发布的译文。
>
> 欢迎加入 LCRH SIG 一同参与贡献，并领取红帽（Red Hat）和我们联合颁发的专属贡献者证书。

---

via: https://www.redhat.com/en/command-line-heroes/season-2/at-your-serverless

作者：Red Hat 选题：bestony 译者：JonnieWayy 校对：acyanbird, wxy

本文由 LCRH 原创编译，Linux中国 荣誉推出