《代码英雄》第三季（6）：Bash Shell 中的英雄

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《代码英雄》第三季（6）：Bash Shell 中的英雄的音频脚本。

导语：Shell 使得大规模 IT 成为可能。它们是现代计算的必要组成部分。但是，如果没有自由软件基金会（Free Software Foundation）一位名叫 Brian Fox 的开发者的辛勤工作，它可能不会变成这样。现在，世界上几乎每台电脑都有 Bash shell。

在上世纪 70 年代，贝尔实验室（Bell Labs）希望将重复的、复杂的命令序列自动化。Chet Ramey 描述了贝尔实验室是如何开发出几个 shell 的 —— 但 UNIX 只能有一个官方支持的 shell。获选的是 Bourne shell。尽管 Bourne shell 是这些之中最好的一个 shell，但它也有其局限性。而且它只有在受到限制的 UNIX 许可证下才能使用。Brian J. Fox 讲述了他在自由软件基金会的工作，他需要创建一个自由的 Bourne shell 版本。它必须兼容但不使用任何原始源代码的元素。这个 Bourne-Again Shell，即 Bash，可能是这个星球

上使用最广泛的软件。而 Taz Brown 描述了它是如何成为一个开发者可以学习使用的最重要的工具之一。

00:00:07 - Saron Yitbarek：

那是 1987 年。里根总统治下的美国正蓬勃发展，一个怀揣远大梦想的人正驱车前往他位于圣巴巴拉（Santa Barbara）的新家。这个人名叫 Brian Fox，27 岁，是高中辍学生。在他车的后备箱里，有两盒巨大的磁带，里面载满了他当时正在编写的代码。

00:00:28：

Fox 多年来一直以程序员的身份工作在所谓的自由软件运动中。他相信他锁在这个后备箱里的代码，可以带来一场革命，这是一种全新的软件范例。他的社区正在一点一点地使之成为现实。

00:00:49：

那年，理查德•斯托曼（Richard Stallman）（RMS）的自由软件基金会（Free Software Foundation）的一组程序员，正在想尽办法给计算机界带来自由。他们想要构建一个 UNIX 的替代品，以取代自从 70 年代以来就主导编程的 UNIX 操作系统。他们的 GNU（表示 GNU's not UNIX）将成为公众的操作系统，任何人都可以使用它，无需担心许可费用或版权问题。

00:01:18：

多年以来，基金会一直在努力制造这个崭新的系统。那么 Brian Fox 汽车后备箱里的那两盒装着代码的巨型磁带是什么？它们存储着这个系统一个至关重要的组成部分。这是一个自由的，而且可更改的 shell，它能够使 GNU 操作系统变得完整。这是 Brian Fox 送给自由软件运动的礼物。他称之为 Bash。

00:01:46：

我是 Saron Yitbarek，这里是代码英雄（Command Line Heroes），一档来自红帽（Red Hat）的原创播客节目。在这一集中，我们将来看看 Bash shell 中的英雄们。我们将探索 shell 的历史，以及它们为什么对我们如今的工作如此重要。大家

可以将 shell 看作要给演员的剧本。它们提供了完整的命令序列，然后 shell 可以快速地运行，就像演员可以一行接一行地读她的台词一样。这是对于实现重复且复杂的代码的，是最终的解决方案，也是自动化的关键。你可能会说，shell 脚本是我们开发的一大助力。但是，是否可以编写一个，能给所有人带来帮助的 shell？这就是挑战所在。

00:02:38 - Ken Thompson：

让我们回到 1969 年。那时候贝尔实验室的几位计算机科学家，正在根据自己的需求开发程序。

00:02:48 - Saron Yitbarek：

这位是代码英雄先驱 Ken Thompson。由贝尔实验室设计的 UNIX 操作系统，在一开始确实是供他们个人使用的。最初，它只是一个内部系统。UNIX 鼓励程序员之间进行密切的交流，不过目的并不是要改变整个世界，而是改变贝尔实验室。

00:03:13 - Ken Thompson：

到现在，几乎整个贝尔实验室都在使用这个系统。我们公司拥有近两万个计算机终端，其中大多数使用 UNIX 系统。

00:03:25 - Saron Yitbarek：

一款由 Ken Thompson 所设计的 UNIX shell 在 1971 年发布。 虽然 Thompson shell 被设计为命令行解释器，但是它却不能很好地支持脚本。所以直到六年后的 1977 年，脚本才开始兴起。

00:03:44 - Chet Ramey：

Shell 参数、特殊参数以及我们如今认为理所当然的变量，起源于 Steve Bourne 和 Bourne shell。

00:03:57 - Saron Yitbarek：

这位是 Chet Ramey，Case Western Reserve 大学的 IT 架构师。Chet 致力于维护 Bash，他也为我们讲述了许多 Bash 的起源故事。他描述了贝尔实验室当时研究 UNIX shell 样子时的情景。

00:04:13 - Chet Ramey：

我们如今使用的编程结构起源于 Steve Bourne，他的 shell 赢得了这场比赛。当时有大量使用 Mashey shell 的用户社区，也有大量用户开始使用 Bourne shell。那时候成立了一个委员会来决定哪一个将会获胜，哪一个将会成为从那时候起得到官方支持的 UNIX shell， Bourne 的 shell 赢了。而其他的 shell，正如他们所说，成为了历史。

00:04:54 - Saron Yitbarek：

不过，这还不是历史的终结。当然，Bourne shell 是一个巨大的飞跃。它打开了一扇通向更高自动化水平的大门。但是尽管有一段时间 Bourne 占据了上风，但是 Bourne shell 并不能解决我们所有的脚本需求。

00:05:14 - Chet Ramey：

Bourne 撰写自己的 shell 时所受到的限制，几乎是现在的你我难以想象的。显然，当你遇到这些限制时，你不得不放弃很多东西，Bourne 就放弃了很多。考虑到他所处理的空间、内存和 CPU 限制，他能够让 Bourne shell 包含那么多东西，这相当了不起。

00:05:42 - Saron Yitbarek：

请记住，Bourne shell 仍然是贝尔实验室 UNIX 系统的一部分。它仍然与 UNIX 许可证绑定。这意味着它不是自由的，不是开放的。这款 shell 是私有的。

00:05:55 - Chet Ramey：

如果你不在大学里，获取 UNIX 源码将会非常困难。显然，这对 Berkeley UNIX 的普及产生了影响。Berkeley UNIX 始于大学中，在大学社区中成长，并走了一条阻力最小的道路。因此，如果你在正确的地方，访问到 Bourne shell 的源码并不困难，但是总的来说，这并不是大众都能够认可的方案。

00:06:36 - Saron Yitbarek：

Chet Ramey 是 Bash shell 的维护者。

00:06:41：

因此，我们有了 shell 的雏形，可以着手写这些关键的组成部分，但是目前为止，最好的 shell 的许可证却有个大大的问题，它是闭源的。对于理查德•斯托曼和他的自由软件基金会而言，这是绝对无法接受的事情。我们所需要的是一个不与任何公司绑定的 shell，一个面向所有人的 shell。

00:07:05：

但这就带来了问题。这意味着我们需要编写某种，能做到 Bourne shell 所能做到的一切，而又不会侵犯到版权的东西。如果逐字复制 Bourne shell 的代码，你会被起诉。

00:07:20：

为了使人们摆脱 Bourne shell 的束缚，你必须找到一位能在没看过 Bourne shell 任何源代码的情况下，编写这款复杂程序的程序员。你必须找到这样的一位局外人天才。而理查德•斯托曼找到了完成这项工作的程序员。

00:07:46：

Brian Fox 是一名 20 来岁的高中辍学生，比贝尔实验室的大多数人更懂代码。他从来没有见过任何 Bourne shell 的源代码，这使得他非常适合手头的任务。

00:08:02 - Brian Fox：

我是 Brian Fox。

00:08:04 - Saron Yitbarek：

为什么不直接问问这个年轻人，这个故事是什么样的呢？现如今，Fox 是一位开源倡导者以及 Opus Logica 的 CEO。但是早在 80 年代后期，他只是一个信仰开源软件运动的年轻人。我们聊了聊过去的日子，以及 Bash 是如何从那时演变过来的。

00:08:23：

所以那时候理查德•斯托曼请你为 UNIX 开发一款 shell。那将会是一款自由的 shell，并且是 Bourne shell 的替代品。你是如何回应的呢？

00:08:38 - Brian Fox：

“我们就不能做个更棒的吗？”

00:08:41 - Saron Yitbarek：

我喜欢这个。再多跟我说说。

00:08:45 - Brian Fox：

我为斯托曼所做的第一件事，其实就是编写个信息技术文档系统。我让理查德惊讶于我做这种编程的速度。他是个优秀的程序员而且工作的很快，但是他不认为其他人也能写得那么快。

00:09:00 - Brian Fox：

因此，在第一周内，我完成了一款名为 GNU Info 的程序的第一版实现，理查德对此有点儿震惊。我说：“我的下一个项目是什么？我的下一个项目是什么？”他说：“好吧，现在给它做个编译器吧。”我就做了，一周时间之内就完成了。然后我说：“我的下一个项目是什么？我的下一个项目是什么？”他说：“好吧，另一个家伙一直在研究那个 shell，但他还没有太多进展。”我说了“好的”，九个月后， Bourne shell 的替代品完成了。

00:09:29 - Saron Yitbarek：

九个月，哇。再多告诉我一些。为什么它如此具有挑战性？

00:09:33 - Brian Fox：

这真是个有趣的问题。它之所以如此具有挑战性，是因为我们必须忠实地模仿 Stephen Bourne 最初的 Bourne shell 的所有行为，同时对其进行扩展，让它成为人们能使用的、更好的工具。

00:09:51：

那时候，我和 Korn shell 的作者 David Korn 私下进行了秘密争论。POSIX 委员会，也就是规定了什么是标准 UNIX 的委员会，他们也参与了进来，并说：“哦，很好，我们需要知道 shell 到底要包含些什么。”而这方面最重要的两个人是我和 David Korn。David Korn 已经写了一个名为 KSH 的 shell。对于他所加入到 KSH 中的每一个功能，他都说：“这应该是一个标准功能。”是这样吗？对他来说这比起拥有最完美的 POSIX shell 要容易得多，如果这仅仅是他的 shell 的话。

00:10:31：

其中的一些功能并不是很好的功能，不是很好的选择，而且使得这款 shell 与 Bourne shell 有些不兼容，或者我觉得缺少功能，对此我们进行了一些讨论和争论，因此构建一个兼容 POSIX 的 shell 与过去为 Bourne shell 所编写的每个 shell 脚本都完全兼容花了超过 3 个月时间。

00:10:54 - Saron Yitbarek：

因此，如果你正在设计的产品不仅可以取代 Bourne shell，而且还试图模仿 Bourne shell 的每个部分，听起来你可能会遇到一些版权问题。你是如何处理的？

00:11:08 - Brian Fox：

为了构建真正开源而自由的软件，你必须得在一个干净的空间里，开始做这项工作。你不能从查看别人的代码开始然后重新实现它。因此，我从未见过与任何贝尔的系统、UNIX 或者甚至 Berkeley UNIX 相关的任何软件，也从未见过这些东西的源代码。

00:11:29：

当我开始构建 Bash shell 时，我使用了一个名为 Bison 的解析器，理查德已经将其整合到自由软件基金会里，并且与之前任何的其他程序完全不同。因此，我已经知道我所要构建的东西，绝对不会侵犯任何先前构建的东西的版权。

00:11:55 - Saron Yitbarek：

创建 Bash 的工作有很多小插曲，对于那些硬核的代码英雄来说，这只是其中一个例子。

00:12:03 - Brian Fox：

有一次，我正致力于在 shell 中实现通配（globbing）扩展。举例来说，这是允许你匹配大量文件的通配符扩展。你可以给出 `*.c`，而这会匹配所有带有 `.c` 扩展名的文件。

00:12:17：

因此我在通配扩展上忙活了几个小时，并且使其生效了，对此我感到很兴奋。这是一个很好的实现。而在创建这一版实现的过程中，我在我的目录里创建了一个名为 `*.c` 的文件，然后我想：“好吧，我应该删掉这个文件”，然后我输入了 RM、空格、引号、星号点 C、闭合引号，在现代 shell 中当你使用了括号，这意味着“不要扩展这个”，然后我按下了回车，提示符过了很长时间才重新出现，因为我们正在使用 Sun 350s，运行缓慢。我意识到，之所以花了很长时间是因为它要删除这个目录里的所有源文件。

00:12:58 - Saron Yitbarek：

哦，不！

00:12:59 - Brian Fox：

是的。所以我当时删掉了 Bash 的源代码。

00:13:01 - Saron Yitbarek：

哦，不要。

00:13:04 - Brian Fox：

这 ——

00:13:05 - Saron Yitbarek：

哦我的天哪，嗯。

00:13:06 - Brian Fox：

这件事让我笑了很久，笑的很大声。我甚至没有感到一丝沮丧。然后在接下来的几天里，我重新输入了全部。这份代码在我脑海里是完全是崭新的。

00:13:20 - Saron Yitbarek：

哇。

00:13:20 - Brian Fox：

问题解决了。只需将其记录到文件中即可。

00:13:25 - Saron Yitbarek：

好的。因此，大多数人会在那一刻完全惊慌失措。而你笑了，只是说：“哦，我想我必须重新做一遍了。”为什么你当时那么冷静呢？

00:13:35 - Brian Fox：

这让我感到疯狂很荒唐，也非常好笑，我正在打造这个工具，而要确保自己能搞好，确保该工具正常工作，你得在构建它的过程中就使用它。但是该工具无法正常工作。我还没有实现引号，并且因为我还没实现引号，所以我输入的命令没有按照我所预期的去执行，我觉得这真的很滑稽。

00:14:06 - Saron Yitbarek：

太神奇了。

00:14:08：

不过，甚至是关于错误的这个故事也能说明 Fox 的才华。他们说莫扎特在头脑中完成了交响曲，然后只需要在完成后写下来即可。Fox 也有类似的天赋。

00:14:23：

因此，当你最终完成并交付 Bash 时，感觉如何呢？

00:14:27 - Brian Fox：

呵，其实感觉很壮观。那么这里有一个故事，其实我一般不讲的。构建这款 shell 花了大约 8 个月的时候，当时我知道，我大概还需要大约一个月时间才能完成工作，然后另一个 shell 发布了 —— ASH，一个开源的 shell 被发布了，我很沮丧，因为我们还没有向任何人发布 Bash shell，所以只有少数人在使用它。我知道这还需一个月的工作量，于是我想："哦，这太糟糕了。我投入的全部能量和精力都不会得到赞许，甚至可能都不会被看见。"所以我非常沮丧。这次我没有笑。

00:15:13 - Saron Yitbarek：

然而，布丁好不好，吃了才知道。GNU 的 Bash 发布于 1989 年并且变成了 Linux 的默认 shell。如今，它是计算机中不可或缺的一部分。

00:15:25：

它无处不在。如此多的人每天都在使用它。它遍布于每一台计算机上。作为 Bash 的作者感觉如何？

00:15:34 - Brian Fox：

大多数时候，我甚至都没有注意到 Bash 是比工具更加重要的东西。我真的没有经常想这件事。每隔一段时间，我会走进一家苹果商店，环顾四周然后想："哇，这里的每台计算机不仅运行着我 27 年前编写的软件，甚至上面还包含有我的名字。"然后我想："互联网上的每台计算机、每台服务器都在运行着 Bash shell，并且其中包含有我的名字。"然后 Windows 在去年还是前年推出了 Power shell，就是 Bash，当时我想："哦，天哪。我的名字遍及地球上的每台计算机了。"

00:16:21 - Saron Yitbarek：

不过，我想让你们能仔细听听 Fox 接下来告诉我的内容，因为它是很重要。他从未想过，它的程序会这样统治全球。他试图提供帮助，试

图帮助他所置身其中的编程文化。

00:16:37 - Brian Fox：

我并没有打算去实现出现在每个人的计算机上这样的宏伟目标。我对此一点都不感兴趣。我想制作一款有用的软件，我希望它有典型的 3 到 5 年软件寿命，而不是像现在这样疯狂的 30 年的寿命。

00:16:58 - Saron Yitbarek：

难道你对于你在计算机领域有如此巨大影响力的事情，一直反应那么平淡吗？

00:17:06 - Brian Fox：

我为自己写了 Bash 而感到骄傲，而且它让我意识到了我的价值，所以有时候我会做一些事情，诸如接受播客邀约谈论 shell 之类的事情。

00:17:14 - Saron Yitbarek：

非常感谢你。

00:17:15 - Brian Fox：

谢谢。但这不是存在于我日常生活中的东西。幸运的是，我只是一个默默无闻的人，对吧？的确，我的软件正运行在每家每户的计算机上，不过也确实没有人知道这一点，对吧？因此我保持了许多个人隐私，而这个 shell 以及某个住在圣芭芭拉的人编写了它这一事实正越来越广为人知，我开始在生活中越来越多地注意到它。人们有时候来看我演奏音乐，然后告诉我说：“你是写了 shell 的那个家伙。”我感觉有点儿像 Keanu Reeves。

00:17:54 - Saron Yitbarek：

很酷。所以你说过你不指望 Bash 出现在每台计算机上。你打算做的是什么呢？你对 Bash 有什么期望？

00:18:04 - Brian Fox：

一个有用的替代工具，成为 GNU 项目的一部分，并帮助创建这个自由的开源操作系统。我实际上以为一旦我们完成了该开源操作系统的创建，该系统上的软件就可以升级，并且我将有机会创建自己想要创建的那种 shell，以帮助人们在某种程度上促进计算机科学的发展。

00:18:35 - Brian Fox：

我最终意识到，Bash 被创建的原因实际上是与已经存在的 UNIX 世界向后兼容，并且这种势头使其保持了活力，这是另一个独一无二的地位，你的工具如此基础，几乎是一副不可或缺的螺母和螺栓。

00:19:01 - Saron Yitbarek：

确实是这样。

00:19:01 - Brian Fox：

知道我创造了世界上某种有价值的、别人仍然还在使用的东西，这真的是一种很棒的感觉。然后当我注意到这是怎么回事时，我意识到，更重要的是，“自由软件”和“开源”这些词存在于日常英语和全世界的日常语言之中了，而最初并不是这样的。这是我和理查德•斯托曼还有其他人所投入努力的产物。作为这一运动的一部分，我很幸运能这么早参与，但让我回过头来看时，也感到非常满意，我想：“哇，开源软件已经存在，而且我就是其中的一部分。”

00:19:50 - Saron Yitbarek：

Brian Fox 是 Bash shell 的创建者和 Opus Logica 的 CEO。

00:20:01 - Steve Bourne：

事实上，我确实听说过 Bash。

00:20:03 - Saron Yitbarek：

这位就是被 Brian Fox 的工作所替代的 Bourne shell 的创建者 Steve Bourne。我们想知道 Bourne 对 Fox 的工作有何看法。他是否将重生的 shell Bash 视为自己作品的开源复制品？我的意思是，他觉得 Bash 怎么样？

00:20:20 - Steve Bourne：

有一天，写了 Bash 的那个人在一次会议上找我，给了我一件 T 恤，前面印着 “Bourne again” 的字样。

00:20:26 - Saron Yitbarek：

那就是 Brian Fox。

00:20:29 - Steve Bourne：

那是一种友好的情绪，当时是：“好吧，希望您不介意，但我只是重写了您的 shell”，而我说：“听起来不错”，然后他给了我一件 T 恤。

00:20:38 - Saron Yitbarek：

如果我在编程领域学到了一件事，那就是每个人都喜欢意外之喜。事实证明，Stephen Bourne 认为 Bash 是他和其他人在贝尔实验室所做工作的必要扩展。一点儿都不为此苦恼。

00:20:52 - Steve Bourne：

曾经有一些人们想要，但是我没做的特性，例如变量替换和字符串管理，但是这些都被加入到了 Bash 中，现如今人们经常会用到。Bash 和原始 shell 之间的关系，我当时的印象是，它只是对语言的重新实现，并且随着时间的推移，它确实添加了功能，因此它确实取得了超越我所写作品的进步，当然是在字符串管理领域。我现在一直在用它。

00:21:21 - Saron Yitbarek：

Steve Bourne 是 Bourne shell 的创建者和 Rally Ventures 的 CTO。

00:21:32 - Saron Yitbarek：

自从 Bash 在前往圣芭芭拉的长途车程中被塞进 Brian Fox 的卡车以来，已经过去了很多年。 2019 年，版本 5.0 被发布，就像 Fox 提到的那样，Bash 现在被内置进了 Linux 中、macOS 中，甚至微软

Windows 中。Bash 已经成了开源世界中脚本编写的基石。这是我们自动化的基础。

00:22:02 - Taz Brown：

随着组织规模的扩大，使用能够使我们更快完成工作的工具变得至关重要。它成为了必需品。

00:22:16 - Saron Yitbarek：

Taz Brown 是 Red Hat 的资深 Ansible 自动化顾问，因此她非常了解 Bash 的价值。

00:22:24 - Taz Brown：

我绝对认为人们在职业生涯初期就应该使用 Bash。与其使用 GUI 或者说是图形用户界面，不如将自己视为管理员或 DevOps 人员。

00:22:39 - Saron Yitbarek：

而这是因为作为一名 Bash 程序员，你将会掌握能让你晋升的核心价值。

00:22:45 - Taz Brown：

学习写脚本有一定的价值，因为这可以让你从自动化的角度，为程序的长期运行做打算。你可以看到脚本的运行方式，然后可以说：“好吧，我可以做到，我可以使这项任务自动化执行。”它开始使你成为与之前不一样的思想家和技术专家。

00:23:09 - Saron Yitbarek：

对于运维而言，自动化已经变得不可或缺。复杂的程序、应用和工具均由优雅的 Bash 代码实现。

00:23:21 - Taz Brown：

如果你愿意的话，你不必重复造轮子。你可以从 GitHub 库或是其他任何你存储这些特定文件的地方拉取它们。Bash 允许你这么做。

Bash 允许你执行这些常见任务，并且可以从 10 台服务器扩展到 1000 台服务器。

00:23:42：

关于自动化的伟大之处在于，一旦你制定了计划，就可以以一种非常有效的方式执行。它允许你执行那些，无法手动执行的操作。

00:23:56 - Saron Yitbarek：

最近 Taz Brown 所从事开发的 Ansible® 这样的最新产品可以始终与 Bash 集成在一起，完成了工作。

00:24:04 - Taz Brown：

虽然时代在不停前进，但是我认为 Bash 永远都会是管理员会去选择使用的工具，特别是他们想要快速自动化的情况下。

00:24:14 - Saron Yitbarek：

最后，这一切的成功，都可以追溯到它是一个自由的、允许所有人加以改进的软件这件事上。它是 Brian Fox 提供给世界的，某种没有许可证和限制的东西。满足了人们一直的需求，所以是 Bash 成功的关键。实际上，他甚至已经不再主管 Bash 开发已经很长一段时间了。这位是 Chet Ramey，他维护了 Bash 数十年。

00:24:38 - Chet Ramey：

我想，Brian 在发布 1.05 版本后就已经决定了他想要继续去从事其他工作。他曾在自由软件基金会负责过其他任务，他想做除了 Bash 以外的事情，而我是 Bash 最活跃的贡献者。他和我一起开发了许多新功能。我们共同努力解决了许多 bug，因此当到了需要其他人接手时，我是最佳人选。

00:25:16 - Saron Yitbarek：

就像 Fox 一样，Ramey 也必须继续努力，因为 Bash 比任何一位维护者都重要。

00:25:25 - Chet Ramey：

我是从 23 岁开始贡献的，有点儿像是我和 Bash 共同成长。在某些时刻，我会需要征集一个团队。我需要征集那些愿意并且有能力投入时间推动 shell 发展向前的人们。

00:25:46 - Saron Yitbarek：

Bash，这款再次降生的 shell 明年将迎来 30 岁（LCTT 译注：Bash 发布于 1989 年，至本译文发表时，已经 31 岁了），并且没有衰老的迹象。Bash 乘着自由软件浪潮，然后是开源浪潮，直到传播至编程世界的每个角落。但曾经，它只是存储在 Brian Fox 汽车后备箱里磁带上的代码。它只是一些程序员，想要带给大家的 shell 语言。几乎偶然的，Brian Fox 在此过程中成为了一名伟大的代码英雄。

00:26:23：

顺便说一句，有些事情始终困扰着我， Brian Fox 驱车将所有 Bash 代码载到了 Santa Barbara。为什么要转移呢？我的意思是，他在某家科技公司找到了新工作吗？

00:26:34 - Brian Fox：

我想要继续我的音乐生涯，而我认为做到这一点的最佳去处就是气温总在 72 华氏度左右、天空没有乌云、海滩很美的地方。

00:26:45 - Saron Yitbarek：

很好，我更喜欢这个理由。

00:26:49：

现在让我们向 Wayne A. Lee 致敬，是他向我们建议了这一集标题《Bash Shell 中的英雄》。干得好，Wayne。

00:26:57：

在下一集中，我们对于自动化的兴趣，将提升到一个全新的高度，并且着眼于 AI 语言，特别是 John McCarthy 创造的 LISP。

00:27:11：

《代码英雄》是 Red Hat 的原创播客节目。如果你访问节目的网站 redhat.com/commandlineheroes，你将更深入了解到有关 Bash 或是我们本季所介绍的任何编程语言的故事。

00:27:28 - Saron Yitbarek：

我是 Saron Yitbarek。下期之前，坚持编程。

> 什么是 **LCTT SIG** 和 **LCTT LCRH SIG**
>
> LCTT SIG 是 LCTT 特别兴趣小组（Special Interest Group），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-3/heroes-in-a-bash-shell

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