# 《代码英雄》第一季（3）：敏捷革命

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》系列播客第一季（3）：敏捷革命 的音频脚本。

现在是 21 世纪之交，开源软件正在改变着科技的格局，现在已经需要一种新的工作模式了。开发者们在寻找一种革命性的方法，让开源开发蓬勃发展。一群开发者在犹他州的一个滑雪场召开了会议，形成的是一份改变一切的宣言。

Manifesto for Agile Software Development Dave Thomas

《敏捷软件开发宣言》的作者之一戴夫·托马斯将我们带回了那个现在著名的静修之地，敏捷革命就是在那里第一次组织起来的。不过，并不是每个人都那么快就签下了这种新方法，在这一集里，我们听听原因。

**Saron Yitbarek：** 有些故事的走向和结局会重新定义一个行业。在这些故事中也传唱着，我们来自哪里，我们是谁，我们正在做什么。上一集中，我们追溯了 Linux^® 开源技术的崛起。但这一集我要讲的，是紧接着发生的故事。操作系统之战结束后，开发者们成为了战争争夺的对象和核心。

**00:00:30**： 在那个新的战场，开发者们将要重塑自己的工作。本集播客，我们将深入了解以开发人员为核心，产生的一种全新的软件开发方法论。这种新颖的工作流程产生了哪些远超我们屏幕上显示的代码所能控制的、意想不到的事情。

我是 Saron Yitbarek，欢迎收听红帽的原创播客《代码英雄 第三集 敏捷革命》。今天的故事始于 2001 年 2 月，发生在美国犹他州的滑雪小屋里。

**00:01:00 - Dave Thomas**： 我们面前有个小屋，眼前是松树梁和壁炉，还有进入屋子的小门。我们前一天晚上到达这里时，然后基本上只是围坐在一起，讨论谈我们准备探讨的内容。紧接着第二天，我们都起床，并来到了预定的会议室。先把桌子移到边上去，然后将椅子摆放成一圈，确切地说是一个椭圆，这样一来我们就可以面对面交流，一定程度上也让人感觉到可以敞开心扉，畅所欲言 。

**00:01:30 - Saron Yitbarek**： 刚才提到的这群人都是开源开发人员，所以保持开放是他们的特点。Dave Thomas 和其他的 16 个人，在那个冬天集聚在雪鸟滑雪场。但是他们的目的并不是滑雪，而是探讨在 90 年代开发者的世界所面临的问题。在这里我用“探讨”，但实际上用“辩论”更准确。他们最初是在
Object-Oriented Programming, Languages and Systems
面向对象编程、语言及系统（OOPSLA）的会议上认识的，这个会议主要议题是面向对象程序设计、编程、语言和系统。

**00:02:00**： 实际上正是那次会议，让他们意识到当前的软件开发方式很混乱，只是对于该怎么办没有达成一致。

所以此次在雪鸟山上的开会，试图寻找解决这个问题的方法。那么究竟是这个问题具体该怎么描述？于是我询问 Dave，开发人员之前的使用方式到底出现了什么问题。

**Dave Thomas**： 所以，我不知道……你有没有装饰过房间？

**Saron Yitbarek**： 我有。

**00:02:30 - Dave Thomas**： ……好。如果我先告诉你，“我想让你坐下来，然后给你一张白纸。接着我希望你能描绘下来这个房间完成后大概的样子。”可以想象吗？

**Saron Yitbarek：** 嗯嗯（肯定的语气）。

**Dave Thomas：** 你能做到吗？

**Saron Yitbarek：** 实际上，我的办公室就是这么布置出来的。首先，我画了一个简单的草图，然后加上一些修饰，最后把所有架子摆放在我觉得合适的位置。不过这种方式没有真正起到作用，我的计划也没有实现。

**Dave Thomas：** 但是，即使你尝试了这种方式，你都做了什么？先把架子放起来，然后说，“哦……这样放不行，因为会挡道。”所以，你又紧接着把架子移到其它地方，或者你会说，“你知道吗，我真的不能把地毯放在那里，因为我的椅子脚会陷进去。”状况频发。

**00:03:00 - Dave Thomas：** 遇到未知的情况，你总需要一种“迭代”的方式去应对。人类的大脑无法准确地对现实世界的发展进行预判，从而提前预知哪里需要改变。所以，软件开发也是一样的，人们不知道他们的需求会怎么改变。对吗？

**Saron Yitbarek：** 嗯嗯（肯定的语气）。

**00:03:30 - Dave Thomas：** 我经历过太多这样的情况，当我从客户那里拿到了详细的要求，然后我已经很好地完成了每一条细则，但却总是吵得不欢而散。“这不是我们想要的。” 但我想说的是，“这就是你要求的啊。”他们说，“是的，但这不是我的意思。”你懂这种情况吗？

**Saron Yitbarek：** 嗯嗯（肯定的语气）。

**Dave Thomas：** 所以说，理想情况是，你可以详细说明流程的每一步，然后通过非常机械的步骤就可以完成一切。

**Saron Yitbarek：** 是啊。

**00:04:00 - Dave Thomas：** 但是在软件行业可行不通。这种方式不适用于有任何模棱两可的情况，也不适用于需要判断的情况。就像任何艺术尝试一样，这种方式就是行不通。总是缺失了关键的一步：反馈。

**Saron Yitbarek：** 也许你已经听说过上世纪 90 年代的软件危机。当时的软件开发一团糟。相比于开发软件的费用，公司在修复软件上的钱花的多得多。与此同时，你我这样的开发人员进退不得。有时候，我们每隔好几年时间才能推出新的软件。

**00:04:30：** 我们疲于应付这些缓慢、陈旧、瀑布式开发的工作流程。从 A 到 B 到 C，完全都是提前确定好的。因此，那时的时间消耗主要在寻找新的流程，寻找更好的软件开发方式上。事实上，每个月似乎都有新的开发者，对如何改善软件开发的过程提出宏伟的设想。

**00:05:00：** 其中就有极限编程、有 Kanban、还有统一软件开发过程等，不胜枚举。在这些方法论的激烈竞争中，也催生出了新的观点和改进方法。那就是 Dave Thomas 和他在雪鸟滑雪场的朋友们迫不及待开始探讨的领域。

值得让这群人齐声欢呼喝彩的就是《敏捷软件开发宣言》。当时的开发速度正在以前所未有的速度保持增长 —— 而开源使开发人员变得更强大。另一方面，开发人员也需要一种新的敏捷的开发模式。

**00:05:30：** 顺便提一下，那些在雪鸟滑雪场会面的人，在经过一番你来我往的争论后，才确定用这个词。敏捷(Agile)，这个词非常切题。这种方式就好像国家地理描述大型猫科动物的方式，一个与瀑布式开发预设路径正好相反的词。随着新的信息层出不穷，这个词让那些愿意改变航向的人看到了一线曙光。请注意这可不是一个名词，而是一个形容词。

**00:06:00：** 敏捷更像是一种习惯，而不是一种具体的说辞。那么，那些采用敏捷的开发者提供了什么呢？他们的总体解决方案是什么？现在很多人都认为敏捷是一个复杂的集合，包括不同的角色和系统。会有一个项目经理(scrum master)，一个项目团队(scrum)，一个产品负责人。同时他们都要进行一到两周的冲刺工作。

**00:06:30：** 与此同时，工作都堆积在“冰盒”和“沙盒”中。好吧，听起来感觉流程很多。但一开始的时候是没有这些流程的。撰写该敏捷宣言的人，目标是简单和清晰。实际上，简单是它制胜的法宝。从那时起，它就具有定义几乎每个开发人员命运之路的能力。

**Dave Thomas**： 我们已经提到了，我们更喜欢某种方式，而不是另一种方式。事实上，在午餐这段时间，我们就写下了几乎所有的观点，现在都是敏捷宣言的一部分。

**00:07:00 - Saron Yitbarek**： 这是可以管理开发的四个奇思妙想。如果你尚且还不熟悉那些敏捷的诫命，他们会这样解释：

> 个体和互动胜过流程和工具；可工作的软件胜过文档；客户协作胜过合同谈判；响应变化胜过遵循计划。

**00:07:30**： 我记得第一次看到这个宣言时的情形。我刚开始学习编程，老实说，当时我并没有觉得这个想法有多棒。一直到我了解到那些支持敏捷开发的工具和平台。对我来说，这只是一些模糊的概念。但是，对于长期以来一直被这些问题纠缠的开发人员来说，这是一个很好的行动方案。

该宣言是一盏灯，可以激发更多奇思妙想。这四点宣言和一些支持材料都发布在 Agilemanifesto.org 网站上，并且呼吁其他开发者签名以表示支持。

**00:08:00 - Dave Thomas**： 很快获得了 1000 个签名，接着 10,000 个，然后签名数一直在增长，我想我们都惊呆了。这基本上变成了一场革新运动。

**Saron Yitbarek**： 他们从来没有计划过把这份敏捷宣言带出滑雪小屋。这只是一群热衷于软件开发的人，并且对帮助他人更好地发展充满热情。但很明显，“敏捷” 本身像长了腿一样。红帽公司首席开发倡导者 Burr Sutter 谈到了“敏捷”对于还困在“瀑布”中的开发人员来说是一种解脱。

**00:08:30 - Burr Sutter**： 因此，敏捷的概念从根本上引起了人们的共鸣，基本上是在说：“看，我们专注于人员而不是流程。我们专注于交互和协作而不是工具和文档。我们认为工作软件高于一切，我们宁愿人们通过小批量的工作，实现高度互动、快速迭代。”

**00:09:00 - Saron Yitbarek**： 而对于一些人来说，这个开发者的革新走得太远。敏捷甚至被视为是给那些不负责任的黑客心态的合理说

辞。早期反对敏捷最重要的声音之一是 Steve Rakitin。他是一名软件工程师，拥有超过 40 年的行业经验。

**00:09:30：** 当他大学毕业时，Rakitin 就开始建造第一个核电站数字控制系统。几十年来，他一直致力于研发电力软件和医疗设备软件。这些都是对安全很注重的软件。没错。你可以预料到，他可不会对这种手忙脚乱的开发方式感兴趣。

因此，在方法论战争的尾声，敏捷横空出世，Rakitin 对此翻了个白眼。

**Steve Rakitin：** 就像是，“好吧，我们换种方式说，如同一群人围坐着喝着啤酒，就想出了开发软件的其他办法。”顺便提一下，敏捷宣言中许多已经得到进一步发展，并应用于早期的开发方法里了。

**00:10:00 - Saron Yitbarek：** 他这么想其实也没有什么错。实际上你可以在 “雪鸟峰会” 前几十年就追溯到敏捷哲学的踪迹。例如，像看板 (Kanban) 这样的精益工作方法可以追溯到 20 世纪 40 年代，当时丰田受到超市货架存货技术的启发……

他们的精益制造理念最终被用于软件开发。不过 Rakitin 有另外一个担忧。

**00:10:30 - Steve Rakitin：** 这篇宣言发表时我非常怀疑它，因为它基本上是为了让软件工程师花更多的时间编写代码，花更少的时间搞清楚需要做什么，同时记录文档的时间少了很多。

**Saron Yitbarek：** 对于 Rakitin 来说，这不仅仅是提出新的工作流程创意。这也关乎到他职业生涯的清白声誉。

**00:11:00 - Steve Rakitin：** 长期以来，相比于电气工程和所有其他工程学科，软件工程并未被视为正规的工程学科。在我看来，部分原因是因为普遍缺乏软件工程师认可的公认实践。

**00:11:30：** 当我们经历了 90 年代的十年，逐渐开始明晰其中的一些流程。似乎其中一些已经在事实上被实施，而且也很合理。

然后敏捷宣言的出现了。如果软件工程将成为正规的工程学科，那么你就需要流程化的东西。其他所有工程学科都有流程，为什么软件工程就没有？

**00:12:00 - Saron Yitbarek：** 我是 Saron Yitbarek，你正在收听的是红帽的原创播客代码英雄。那么，如果我们把在核电站工作的人士的观点放在一边，转而关注更广阔的企业界，我们发现敏捷已经逐渐广受认可。但很自然的，也会有一些公司在抵制。

**Darrell Rigby：** 我倾向于认为我们在采用敏捷开发的过程中，受到的最大阻力来自中高级管理层。

**00:12:30 - Saron Yitbarek：** 这位是 Bain＆Company 的合伙人 Darrell Rigby。他们一直尝试在软件开发公司中推行敏捷开发。不仅如此，还包括产品开发、新闻服务开发、广告计划和忠诚度计划等。不管他们要做什么，项目管理者都会面临点压力。

**Darrell Rigby：** 敏捷改变了他们的价值，因为他们正在逐步退出细节上的管理或干预。现在团队被赋予权力，对他们加以指导。

**00:13:00 - Saron Yitbarek：** 现在，敏捷并不能保证阻止中间轻微的干预。我承认，我第一次看到一个敏捷管理委员会时，我认为这是一个永无止境的待办事项清单，我有了点压迫感。但后来当我开始真正使用敏捷产品管理工具时，我完全变成了它们的粉丝。我是一个编码培训营的新人，我试图弄清楚如何确定功能的优先级并做出产品决策。

**00:13:30：** 那些看起来很可怕的工具让我有了所有这些想法，然后给它们命名、顺序和结构。从而可以帮助我更好地管理我的项目。所以，我确实同意 Rigby 的观点。有些人可能会看到这些工具产生的影响并认为，如果敏捷赋予开发人员权力，那么就会剥夺经理们的管理权。

但是，它的价值比任何一个职位都要大，敏捷开发的发展势如破竹。更重要的是，它正在证明自己。

**00:14:00 - Darrell Rigby：** 目前，成千上万的团队已经采用敏捷开发。因此，我们有大量数据能够说明在哪里可以采用敏捷的方式。答

案是，无论何时你开始创新，相比你现在使用的方式，敏捷团队能更好实现目标。

**00:14:30**： 有许多更大的、知名的公司都在变革自身。亚马逊是敏捷方法的重要用户。奈飞、Facebook 和 Salesforce —— 他们都是敏捷的重度用户，实际上敏捷方法不仅重新定义了工作方式，更是重新定义了行业的运作方式。

**Saron Yitbarek**： 当 Rigby 第一次听说agile敏捷时，他认为这是一种奇怪的语言。他当时正在与许多大型零售商的 IT 部门合作。无意间听到他们谈论 “time boxes”、“sprint” 和 “scrum master” 。起初，他并不懂他们在说什么。他告诉我他实际上是试图忽略任何有关敏捷的字眼，就像这是他不需要学习的另一种语言。毕竟，他本人不是开发人员。

**00:15:00**： 但是如今，他却成为了敏捷信徒，把敏捷带到他的家里，带入他的教堂。

**Darrell Rigby**： 我不一定每天早上都和家人坐在一起，和他们一起参加敏捷开发式的会议。但是，我已经非常擅长为我要做的事情排优先级。

**00:15:30 - Saron Yitbarek**： 十多年来，敏捷已经从边缘走向主流。但是，企业认同还是有代价的。在某些情况下，这种同化甚至会使敏捷宣言的最初意图变得模糊。Dave Thomas 让我想起了这一点。他说，当他和其他 16 位雪鸟会议上的伙伴第一次写下宣言时，根本没有既定的指示。

**00:16:00**： 因此，即使宣言中没有告诉你如何应用这些条例，我猜想你已经对大概会发生什么，还有人们会怎么做，有一些大概的思路了吧？

**Dave Thomas**： 老实说啊，我还真没有。

**Saron Yitbarek**： 听到这里，你可能会感到惊讶。因为敏捷现在看起来很有说服力。有书籍、认证、工具、课程和产品的整个市场，向你展示如何“实现敏捷”。

Dave Thomas 表示，尽管有成千上万的手册和专业人士想要向你展示“唯一真理”，他们却错过了重点。

**Dave Thomas**： 实际上它是一组价值观。

**00:16:30 - Saron Yitbarek**： 嗯嗯（肯定的语气）。

**Dave Thomas**： 我想这就像黄金法则。你知道，如果你要做一些邪恶恶毒的事情，你会想，“好吧，如果有人这样做，我又怎么会喜欢。”你知道吗，这种场合也适合用黄金法则。

好吧，敏捷宣言也是如此。它并没有告诉你该做什么，不该做什么，它只是告诉你，你做的一切是否符合这个价值观。

**00:17:00 - Saron Yitbarek**： 是的。我想只要回到敏捷软件开发宣言的名称、真正脱颖而出并且经久不衰的一个词，也是人们真正关注的就是“敏捷”。那么现在使用“敏捷”这个词又出了什么问题呢？

**00:17:30 - Dave Thomas**： “敏捷”这个词的问题在于，在我们刚开始提出的时候，它是描述软件开发的形容词。但接下来人们就产生了疑问：“我该怎么着手实施敏捷呢？”

**00:18:00**： 突然之间，涌出了一大批咨询顾问，他们看到了Extreme Programming极限编程（XP）的成功，看到了宣言的成功，“嘿，那里有座金山。”然后就开始告诉人们如何“做敏捷”。这是一个问题，因为你不能“做”敏捷。敏捷不是你要“做”的事情，而是你如何做事情的方式。

然而，有些公司会乐意卖给你敏捷相关的套装。我觉得这很讽刺。这里的咨询就好像是进入一家财富 1000 强企业，然后帮助他们设定“敏捷”。然后带走了 500 万美元。你懂吗？太棒了，钱真好赚。

**00:18:30**： 但是，现实情况是，这就像告诉要老虎如何变得敏捷一样，说：“先走七步，然后左脚迈出来，然后再走两步，然后迈出右脚。”嗯，实际上只有瞪羚做同样的事情，这才会是有用的。你猜怎么着？没有人告诉瞪羚这样做。瞪羚基本都会跑到地平线尽头上大笑起来，因为老虎在“邯郸学步”。

**00:19:00**： 当你告诉团队如何敏捷时，会发生同样的事情。如果你对他们说，“这是你必须遵循的规则，这是你必须遵循的流程”，然后他们唯一能做的就是跟随职责，因为他们已被设定好该执行的程序。管理层将根据他们服从原则或程序的程度，而不是开发软件的水平来判断表现如何。

**00:19:30 - Saron Yitbarek**： 所以，回顾一下，宣言发布之前和之后的开发者的角色，是如何因为你的宣言本身改变或扩展的呢？

**00:20:00 - Dave Thomas**：我认为大多数程序员都能理解到关键点，这值得肯定。我觉得敏捷宣言给了许多开发人员按照这种方法去做的授权，某种程度上是他们以前就知道该如何，但从来没有权利这样做。像测试收集反馈，缩短迭代周期之类的事情。因此，在许多方面，工作变得更有趣，更充实。

同时我认为，程序员也可能会感到有点害怕，因为现在他们有了责任。过去，他们只是遵循命令。这个程序不起作用？好吧，但我遵循了规范。而如今，程序员肩负着责任。

**00:20:30**： 所以，我觉得这个职业因敏捷宣言而有所成长。我认为人们开始意识到，他们对自己所开发东西负有点对点的责任。

**00:21:00 - Saron Yitbarek**：敏捷取得了如此广泛得成功，改变了工作流程和态度，远远超出了开发者世界的范畴 —— 当然也超越了雪鸟会议召开的小木屋。我们不禁要问，“相比于 2001 年撰写宣言时，今天成为敏捷开发人员意味着什么？”

最初的敏捷精神是否仍然存在？如果确实发生了变化，这是一件坏事吗？对于谷歌的多元化业务合作伙伴 Ruha Devanesan 来说，敏捷的思维方式，可能已经发展到影响公平性和在工作场所中的平等性了。

**00:21:30 - Ruha Devanesan**：让团队具有包容性的部分原因，是他们在进行非常基础的工作时，可以评价和反思自己。当大多数团队一起工作时，他们没有足够的时间这么做。没有足够的动力停下来思考他们团队宗旨，是否每个人都在能桌上发表意见，关于是否有人在推动其他人，或者是否有人在一直都保持沉默。如果他们保持沉默，为什么他们保持沉默？

**00:22:00**： 因此，在考虑包容性时，我认为敏捷团队使用的一些工具在为团队创建架构，或更具包容性的框架方面非常有用。所以多样性包括性别、种族，还有功能多样性。功能多样性为团队带来了复杂性。

**00:22:30 - Saron Yitbarek**： 但是，我们在这里要声明他们的不同。Ruha 并不是说敏捷就等于多样性。她的意思是，“敏捷加多样性等于更好的团队。”Ruha 的想法在她写的一篇名为《论通过敏捷方法解锁多样性》的文章中得到了体现。我们将在演示笔记中添加一个链接 —— 这可是值得一读的文章。

在这篇文章中，她会引导你去了解，多元化不仅仅是人力资源部门一直在谈论的模糊概念。这里提供了一个强有力的商业案例。利用敏捷工具，可以创建一个包容性的工作场所，和创新效率提升。多样性可以与敏捷相辅相成。

**00:23:00 - Ruha Devanesan**： 这篇介绍复杂性的文章，最终目的是让大家从不同的角度看待你的结果或产品。当我们说为团队增加多样性可以带来更好的结果，带来更多的创新和创造力时，我们持有的是基本同样的观点。因为当你从多个角度去看待并协作解决工作中的问题时，你更有可能得出一个更好的结果。

**00:23:30 - Saron Yitbarek**： 团队中的每个人，甚至可以对日常会议这样简单的事情提出反馈，这会让内向的人或其他不爱说话的人发表自己的见解。

**Ruha Devanesan**： 我真正喜欢敏捷的原因是，有一些内置的机制来帮助团队停下来进行思考。这可能是因为敏捷开发是如此之快，并且有为时两周的冲刺任务。如果你没有建立这些机制，你可能会偏离轨道，没法再回到正轨。

**00:24:00**： 但是，我觉得，“停止并反馈”这种价值观非常重要。这对于团队的包容性增加也非常重要，因为让大家都能提出工作反馈，并借此不断改善，这是团队能够包容的基本表现。

**Saron Yitbarek**： 既然我们谈论的是包容性，现在可能是指出那些敏捷宣言的 17 位创始人的好时机，是的……他们都是白人。

**00:24:30 - Dave Thomas**： 实际上那个房间没有多样性。这是对该组织的一种非常普遍的批评，而且我对此深表同情。

**Saron Yitbarek**： 如果敏捷宣言创始人采用了这些敏捷原则，并将其应用于他们自己的会议，那么他们可能在完成部分工作后，会问他们自己……“嘿，你注意到我们没有邀请任何女性参加这次会议吗？”我在想会不会有一个有色人种会持有不同意见。

**00:25:00 - Ruha Devanesan**： 物以类聚，人以群分。所以，如果考虑敏捷宣言的第一个人是白人，他邀请到桌上的人也是白人也就不足为奇了。但是，我们有机会在那方面做得更好，我们有机会停下来说：“让我们退后一步，让我们扩大我们的视野，寻找我们现在拥有的关系网络之外的人。谁可以带来不同的视角并帮助我们更好地改进这种开发方式。”

**00:25:30 - Saron Yitbarek**： 对我来说这很有道理，因为敏捷开发正是如此……好吧，敏捷，我们可以将它应用于不同的问题和行业。敏捷的应用方面，以及其在现实生活中出现时候的样子，是不断变化、不断扩展的。我想它正在将宣扬的内容付诸实践。没有最正确的答案，没有最后的终点。这是我们有时会忘记的事情：硬规则是敏捷的敌人。

**00:26:00**： 因此，如果一个敏捷团队告诉你敏捷的一部分意味着你必须每两周开发一个新的版本，或者你必须做什么事，那么，根据定义，这可不是敏捷。你老是说“总是”，你也不再是敏捷了。

**00:26:30**： 那些在犹他州雪鸟会议碰面的 17 名男子，最后宣称成立敏捷联盟。该联盟成为一个非营利组织，每年都举办一次会议。这个联盟的成长和发展，催生了更多全新的理论和方法。

这正是我感觉非常有趣的东西。在 2008 年的会议上，比利时开发人员 Patrick Debois 参加了并开始引领一条道路，他发明了一种全新的软件开发实践 DevOps。我从未想到与敏捷的一系列原则与 DevOps 和整个行业是都紧密相关。

**00:27:00**： 但是，现在我在想，“敏捷的兴起与 DevOps 的发明之间有多少关联？一个突破是否孕育了另一个突破？”我们会一起去探索，因为我们的下一集是正是 DevOps，对！一整集的内容。

**00:27:30**：《代码英雄》是红帽的原创播客。有关我们的播客和其他更多信息，请访问 Redhat.com/commandlineheroes 。在那里，你也可以关注我们的消息订阅。想要免费听取最新内容，请订阅我们的节目。也可以在 Apple Podcast、Spotify、 Google Play、CastBox 中搜索 “Command Line Heroes”，然后点击“订阅”，你将在第一时间获得我们的内容更新。

我是 Saron Yitbarek，感谢收听，请坚持编程。

### 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。

### 关于重制版

本系列第一季的前三篇我们已经发布过，这次根据新的 SIG 规范重新修订发布。

---

via: https://www.redhat.com/en/command-line-heroes/season-1/agile-revolution

作者：RedHat 选题：bestony 译者：redhat 校对：acyanbird,FineFan

本文由 LCRH 原创编译，Linux中国 荣誉推出