DHANJI R PRASANNA

LENNY'S PODCAST

BILINGUAL TRANSCRIPT

ORIGINAL BY

Lenny Rachitsky

@lennysan • x.com/lennysan

ANALYSIS BY

@Penny777 • x.com/penny777

Generated: 2026-01-16

Dhanji R. Prasanna - 双语对照

This is the complete bilingual (English-Chinese) transcript for Lenny's Podcast featuring Dhanji R. Prasanna, CTO at Block.

(00:00:00) Lenny Rachitsky

English:

There's a lot of talk about productivity gains through AI. There's this camp of people that are so overhyped, nothing's working, nobody's actually adopting this at scale.

中文翻译:

现在有很多关于通过 AI 提高生产力的讨论。有一派人认为这完全是过度炒作，觉得没什么真正起作用的东西，也没有人真正大规模采用它。

(00:00:07) Dhanji R. Prasanna

English:

We see a significant amount of games. We find engineering teams that are very, very AI forward are reporting about eight to 10 hours save per week. Whenever I hear a stat like this, I think an important element is this is the worst it will ever be. This is now the baseline. The truth is the value is changing every day, so you need to ride that wave along with it.

中文翻译:

我们看到了显著的进展。我们发现那些非常超前使用 AI 的工程团队报告称，每周可以节省大约 8 到 10 小时。每当我听到这样的统计数据时，我认为一个重要的点是：这已经是它表现最差的时候了。现在这只是基准线。事实是，AI 的价值每天都在变化，所以你需要顺应这股浪潮共同成长。

(00:00:27) Lenny Rachitsky

English:

There's a story I heard you share on a different podcast where there's an engineer who has Goose watching.

中文翻译:

我在另一个播客中听你分享过一个故事，说有一个工程师让 Goose（Block 内部的 AI 智能体）一直观察他的工作。

(00:00:31) Dhanji R. Prasanna

English:

You'll be talking to a colleague on Slack or an email, and they'll be discussing some feature that they think is useful to implement. Now a few hours later, he'll find that Goose has already tried to build that feature and opened a PR for it on Git.

中文翻译:

你可能正在 Slack 或邮件里和同事聊天，讨论某个觉得值得实现的特性。几个小时后，他会发现 Goose 已经尝试构建了那个特性，并在 Git 上提交了一个 PR（拉取请求）。

(00:00:43) Lenny Rachitsky

English:

What level of engineer is most benefiting from these tools?

中文翻译:

什么级别的工程师从这些工具中获益最多?

(00:00:47) Dhanji R. Prasanna

English:

What's been surprising and really amazing, the non-technical people using AI agents and programming tools to build things, the people that are able to embrace it to optimize for their particular workday and their particular set of tasks are really showing the most impact from these tools.

中文翻译:

令人惊讶且真正了不起的是，那些非技术人员正在使用 AI 智能体和编程工具来构建东西。那些能够拥抱 AI 来优化自己特定工作日和特定任务集的人，实际上从这些工具中展现出了最大的影响力。

(00:01:07) Lenny Rachitsky

English:

How do you think things will look in a couple of years in terms of how engineers work that's different from today?

中文翻译:

你认为几年后工程师的工作方式会与今天有什么不同?

(00:01:12) Dhanji R. Prasanna

English:

All these LLMs are sitting idle overnight and on weekends, while humans aren't there. There's no need for that. They should be working all the time. They should be trying to build in anticipation of what we want.

中文翻译:

所有这些大语言模型（LLM）在深夜和周末人类不在的时候都处于闲置状态。这完全没必要。它们应该一直工作，应该尝试根据我们的需求预判并进行构建。

(00:01:24) Lenny Rachitsky

English:

What's maybe the most counterintuitive lesson you've learned about building products or building teams?

中文翻译:

在构建产品或团队方面，你学到的最反直觉的教训是什么？

(00:01:29) Dhanji R. Prasanna

English:

A lot of engineers think that code quality is important to building a successful product. The two have nothing to do with each other.

中文翻译:

很多工程师认为代码质量对于构建成功的产品至关重要。实际上，这两者之间没有任何关系。

(00:01:37) Lenny Rachitsky

English:

Today my guest is Dhanji Prasanna. Dhanji is Chief Technology Officer at Block, where he oversees a team of over 3,500 people. With Dhanji's leadership, Block has become one of the most AI-native large companies in the world and has basically achieved what many eng and product leaders are trying to achieve within their companies.

中文翻译:

今天的嘉宾是 Dhanji Prasanna。Dhanji 是 Block 的首席技术官（CTO），负责领导一支超过 3500 人的团队。在 Dhanji 的领导下，Block 已成为全球最具备“AI 原生”特质的大型公司之一，基本上实现了许多工程和产品领导者试图在自己公司内部实现的目标。

(00:01:55) Lenny Rachitsky

English:

In our conversation, we chat about their internal open source agent called Goose, that by their measure is saving employees on average eight to 10 hours a week of work time, and that number is going up, how AI specifically making their teams more productive and the teams that are benefiting most. Interestingly, it's not the engineering team, what it took to shift the culture to be very AI-oriented, the very boring change they made internally that boosted productivity even more than any AI tool.

中文翻译:

在我们的对话中，我们聊到了他们内部的开源智能体 Goose。据他们测算，Goose 平均每周为员工节省 8 到 10 小时的工作时间，而且这个数字还在上升。我们还聊到了 AI 具体是如何让他们的团队更高效的，以及哪些团队受益最大（有趣的是，受益最大的并不是工程团队）。此外，我们还讨论了将文化转向 AI 导向需要付出什么，以及他们在内部做出的一个非常“无聊”的改变，而这个改变对生产力的提升甚至超过了任何 AI 工具。

(00:02:24) Lenny Rachitsky

English:

Also, lessons from building Google Wave and Google Plus and Cash app and so much more. This episode is for anyone curious to see what a highly AI-forward technology-driven large company looks like and can act like. If you enjoy this podcast, don't forget to subscribe and follow it in your favorite podcasting app or YouTube. It helps tremendously.

中文翻译:

此外，还有来自构建 Google Wave、Google Plus、Cash App 等产品的经验教训。这一集适合任何想了解一家高度 AI 超前、技术驱动的大型公司是什么样子、以及如何运作的人。如果你喜欢这个播客，别忘了在你不常用的播客应用或 YouTube 上订阅和关注。这对我们帮助很大。

(00:02:45) Lenny Rachitsky

English:

Also, if you become an annual subscriber of my newsletter, you get a year free of 16 incredible products including Devin, Replit, Lovable, Bolt, n8n, Linear, Superhuman, Descript, Wispr Flow, Gamma, Perplexity, Warp, Granola, Magic Patterns, Raycast, ChatPRD and Mobbin. Head on over to Lennysnewsletter.com and click Product Pass. With that, I bring you Dhanji Prasanna after a short word from our sponsors.

中文翻译:

另外，如果你成为我时事通讯（Newsletter）的年度订阅者，你将免费获得 16 款不可思议的产品的一年使用权，包括 Devin, Replit, Lovable, Bolt, n8n, Linear, Superhuman, Descript, Wispr Flow, Gamma, Perplexity, Warp, Granola, Magic Patterns, Raycast, ChatPRD 和 Mobbin。请访问 Lennysnewsletter.com 并点击“Product Pass”。在听完赞助商的简短介绍后，我们将开始与 Dhanji Prasanna 的对话。

(00:03:12) Lenny Rachitsky

English:

This episode is brought to you by Sinch the Customer Communications Cloud. Here's the thing about digital customer communications. Whether you're sending marketing campaigns, verification codes, or account alerts, you need them to reach users reliably, that's where Sinch comes in. Over 150,000 businesses, including eight of the top 10 largest tech companies globally use Sinch's API to build messaging, email and calling into their products.

中文翻译:

本集由客户沟通云平台 Sinch 赞助。关于数字客户沟通，情况是这样的：无论你是在发送营销活动、验证码还是账户提醒，你都需要它们可靠地送达用户，这就是 Sinch 的用武之地。超过 15 万家企业，包括全球前十大科技公司中的八家，都在使用 Sinch 的 API 将短信、邮件和通话功能集成到他们的产品中。

(00:03:38) Lenny Rachitsky

English:

And there's something big happening in messaging that product teams need to know about. Rich Communication Services or RCS. Think of RCS as SMS 2.0. Instead of getting texts from a random number, your users will see your verified company name and logo without needing to download anything new. It's a more secure and branded experience, plus you get features like interactive carousels and suggested replies, and here's why this matters. US carriers are starting to adopt RCS.

中文翻译:

在消息传递领域正在发生一件产品团队需要了解的大事：富通信服务（RCS）。你可以把 RCS 想象成短信 2.0。用户收到的不再是来自随机号码的文本，而是能看到你经过验证的公司名称和 Logo，且无需下载任何新应用。这是一种更安全、更具品牌感的体验，此外你还可以获得交互式轮播图和建议回复等功能。这之所以重要，是因为美国运营商正开始采用 RCS。

(00:04:06) Lenny Rachitsky

English:

Sinch is already helping major bands send RCS messages around the world and they're helping Lenny's podcast listeners get registered first before the rush hits the US market. Learn more and get started at sinch.com/lenny. That's S-I-N-C-H .com/lenny. This episode is brought to you by Figma, makers of Figma Make. When I was a PM at Airbnb, I still remember when Figma came out and how much it improved how we operated as a team.

中文翻译:

Sinch 已经在帮助各大品牌在全球范围内发送 RCS 消息，他们正在帮助 Lenny 播客的听众在美国市场爆发前抢先注册。访问 sinch.com/lenny 了解更多并开始使用。本集还由 Figma 赞助，他们是 Figma Make 的创造者。当我在 Airbnb 担任产品经理时，我仍然记得 Figma 问世的情景，以及它如何极大地改善了我们团队的运作方式。

(00:04:35) Lenny Rachitsky

English:

Suddenly, I could involve my whole team in the design process, give feedback on design concepts really quickly, and it just made the whole product development process so much more fun. But Figma never felt like it was for me. It was great for giving feedback and designs, but as a builder, I wanted to make stuff. That's why Figma built Figma Make. With just a few prompts, you can make any idea or design into a fully functional prototype or app that anyone can iterate on and validate with customers.

中文翻译:

突然之间，我可以让整个团队都参与到设计过程中，快速对设计概念给出反馈，这让整个产品开发过程变得有趣得多。但 Figma 以前给我的感觉并不完全是为我准备的。它在反馈和设计方面很棒，但作为一个构建者，我想亲手做出东西。这就是为什么 Figma 开发了 Figma Make。只需几个提示词，你就可以将任何想法或设计变成一个功能齐全的原型或应用，任何人都可以对其进行迭代并与客户进行验证。

(00:05:02) Lenny Rachitsky

English:

Figma Make is a different kind of vibe coding tool, because it's all in Figma. You can use your team's existing design building blocks, making it easy to create outputs that look good and feel real and are connected to how your team builds. Stop spending so much time telling people about your product vision and instead show it to them. Make code back prototypes and apps fast with Figma Make. Check it out at figma.com/lenny. Dhanji, thank you so much for being here and welcome to the podcast.

中文翻译:

Figma Make 是一种不同寻常的“氛围编程”（vibe coding）工具，因为它完全集成在 Figma 中。你可以使用团队现有的设计组件，轻松创建出看起来美观、感觉真实且与团队构建方式紧密相连的产出。别再花那么多时间向人们描述你的产品愿景了，直接展示给他们看。使用 Figma Make 快速制作带有代码支持的原型和应用。请访问 figma.com/lenny 查看。Dhanji，非常感谢你能来，欢迎来到本播客。

(00:05:34) Dhanji R. Prasanna

English:

Thank you Lenny. It's a great pleasure to be here.

中文翻译:

谢谢 Lenny。很高兴来到这里。

(00:05:37) Lenny Rachitsky

English:

I want to start with a letter that I hear you wrote to Jack Dorsey to convince him that he and that Block needed to take AI a lot more seriously. I think you called it your AI manifesto and it seems like it really worked. We're going to talk a lot about the changes that came as a result of that. So let me just ask, what did you say in this letter and what happened right after you sent that letter to him?

中文翻译:

我想从一封信开始。我听说你写了一封信给 Jack Dorsey（Block 创始人），说服他和 Block 需要更加严肃地对待 AI。我想你把它称作你的“AI 宣言”，而且看起来它确实起作用了。我们将深入探讨由此产生的一系列变化。所以我想问，你在信里说了什么？发给他之后发生了什么？

(00:06:00) Dhanji R. Prasanna

English:

So about two and a half years ago or so, Jack really felt like things needed to change. I think he had a sense that the industry was going in a different direction. So he got about 40 of the company's top executives into a room on a weekly basis, and they all used to sort of talk everything through that was going on and he added me to that group. At some point, I observed that we were talking about lots of deep things, lots of relevant things, but no one was really paying attention to AI, and so that's when I wrote that letter. And to be honest, it's I think taken on a life of its own, but there wasn't much to the letter other than I think we should do this. I think we should do it centrally and it's important for us to be ahead of the game and be an AI native company because that's where the industry is heading.

中文翻译:

大约两年半前，Jack 确实觉得事情需要改变。我认为他预感到行业正朝着不同的方向发展。于是他每周召集公司约 40 名高管在一个房间里讨论正在发生的一切，他也把我拉进了那个小组。在某个时刻，我观察到我们讨论了很多深刻且相关的事情，但没有人真正关注 AI。于是我写了那封信。老实说，这封信后来被传得神乎其神，但信的内容其实很简单，就是：我认为我们应该做这件事，应该集中力量去做，而且保持领先并成为一家“AI 原生”公司对我们至关重要，因为那是行业的未来。

(00:06:55) Lenny Rachitsky

English:

Let me just say it's important to note you were not CTO at this point. You were just a senior engineer kind of person?

中文翻译:

我想特别指出一点，当时你还不是 CTO。你当时只是一个类似资深工程师的角色?

(00:07:00) Dhanji R. Prasanna

English:

No, yeah, in fact, I was part-time at the time because I had just had a kid and I was coming back in and I was helping out one of the engineering teams and then Jack came over to Sydney and spent two days with me and both of us like long walks. So we walked all around Sydney and talked it through up and down, and then yeah, he offered me the job and I thought it was a great opportunity once in a lifetime, so I took it.

中文翻译:

是的，事实上我当时是兼职。因为我刚有了孩子，刚回到公司帮一个工程团队做事。后来 Jack 来到悉尼，和我待了两天。我们俩都喜欢长距离散步，所以我们走遍了悉尼的大街小巷，深入探讨了这件事。然后，他把这份工作交给了我，我觉得这是一个千载难逢的机会，于是就接受了。

(00:07:30) Lenny Rachitsky

English:

It's like be careful what you're good at sort of situation. Okay. So what were some of the bigger changes that you made after Jack is on board and Block execs are on board of are, "Cool, this is completely right. We need to go much bigger and think much more deeply about how AI is changing, how we build and how we should build." Or some of the bigger changes that you made from a perspective of other companies listening to this, trying to think about what they should be doing?

中文翻译:

这真是一个“小心你擅长的事”的典型案例。好，那么在 Jack 和 Block 的高管们达成共识，认为“没错，我们需要玩得更大，更深入地思考 AI 如何改变我们的构建方式”之后，你做出了哪些重大改变？对于那些正在收听节目并思考自己该怎么做的公司来说，有哪些值得借鉴的重大举措?

(00:07:53) Dhanji R. Prasanna

English:

At the start, my main focus was to get block to think like a technology company. And for a long time we had had a little bit of, I'm going to call it identity drift, maybe. We were talking about ourselves as a financial services company. Some people called us FinTech, all of this stuff. But when I started working at what was then known as Square, we were always thought of as a technology company just like Google or Facebook or any of the others. And so I wanted to get us back to that. And so the first thing I did was to try and institute a number of programs that focused on that. So everything from getting the top ICs in the company together to talk to each other, to starting a whole bunch of special projects. So we got about two to five engineers per project. There were about eight or nine different projects and we had reinstituted, the company-wide hack week. And so all of this just kind of created a little bit of a spark of, "Hey, we're building technology again, we're trying to push the frontier again." And that's how it started, and then there were a whole number of steps after that where we went from a GM structure to a functional org structure, which was I think the key to making our transformation into being more of an AI-native company.

中文翻译:

刚开始，我的主要重点是让 Block 像一家科技公司一样思考。很长一段时间以来，我们有一点……我称之为“身份漂移”。我们把自己定位为一家金融服务公司，有人称我们为金融科技（FinTech）等等。但当我开始在当时名为 Square 的公司工作时，我们一直被认为是一家科技公司，就像 Google 或 Facebook 一样。我想让我们回归初心。所以我做的第一件事就是建立一系列专注于此的项目。从召集公司顶尖的 IC（独立贡献者）交流，到启动一系列特别项目。每个项目大约有 2 到 5 名工程师，总共有 8 到 9 个不同的项目，我们还恢复了全公司范围的黑客周（Hack Week）。所有这些都创造了一种火花：“嘿，我们又在构建技术了，我们又在尝试突破前沿了。”这就是开始。之后还有一系列步骤，我们从 GM（总经理制/事业部制）架构转变为职能型组织架构，我认为这是我们转型为 AI 原生公司的关键。

(00:09:19) Lenny Rachitsky

English:

Okay, talk more about that. What does that mean? What does that look like? Why is that so important?

中文翻译:

好，请多谈谈这个。那意味着什么？具体是什么样的？为什么它如此重要？

(00:09:23) Dhanji R. Prasanna

English:

Absolutely. So when we were in our mature phase, so when Square was working quite well, it was a very large business, and then we had started Cash App and that also followed suit. We had spun them out almost as what we call a GM structure. So they were effectively run as a portfolio of independent companies and they had their own CEOs who all reported to Jack and it was still one single executive team, but they had separate engineering practices, they had separate design teams. They were kind of separate in almost every way except for some shared resources like our foundational resources like legal and some platforms and things like that. So I think that that was very useful for us for the stage of company that we were in, but when you really want to go deep in technology, when you really want to connect with these things that are industry changing events that are happening, you need a singular focus, and we changed the organization. So all engineers report into one single team now, all designers report into one single team and there's single head of engineering, single head of design, et cetera. And so that was the big transformation that we made, and that meant we could really drive forward AI, we could drive forward platform and just technical depth generally.

中文翻译:

当然。当我们处于成熟阶段时，比如 Square 运作得非常好，业务规模巨大，然后我们启动了 Cash App，它也紧随其后。我们几乎是以所谓的 GM 架构将它们拆分出来的。也就是说，它们实际上是作为一组独立公司的组合来运行的，它们有自己的 CEO，都向 Jack 汇报。虽然仍是一个统一的高管团队，但它们有独立的工程实践和独立的设计团队。除了法律、某些平台等基础共享资源外，它们在几乎所有方面都是独立的。我认为这对于我们当时的阶段非常有用，但当你真正想要深入技术，真正想要与那些正在发生的改变行业的事件接轨时，你需

要一个统一的焦点。于是我们改变了组织架构。现在所有工程师都向一个统一的团队汇报，所有设计师也向一个统一的团队汇报，有统一的工程负责人、统一的设计负责人等等。这就是我们做出的重大转型，这意味着我们可以真正推动 AI、推动平台化以及整体的技术深度。

(00:10:51) Lenny Rachitsky

English:

For companies that are struggling with this potentially or trying to figure out how to do this, two things I'm hearing here is start to see yourself as a technology company. It doesn't necessarily apply to every company, but seems like an important element is like we're building technology, we're not a financial company, we're not a real estate company, we're not a technology company. And then two is organize the team such that say engineers report up to an engineering leader versus a GM who maybe doesn't understand engineering as well or doesn't take it as seriously as they should.

中文翻译:

对于那些可能正在为此挣扎或试图弄清楚该怎么做的公司，我听到了两点：第一，开始把自己看作一家科技公司。这不一定适用于每家公司，但一个重要的元素似乎是——我们是在构建技术，我们不是金融公司，不是房地产公司，我们是科技公司。第二，组织团队时，让工程师向工程领导汇报，而不是向一个可能不太懂工程、或者没那么重视工程的 GM 汇报。

(00:11:19) Dhanji R. Prasanna

English:

Yeah, I think that's pretty much what we did. And not to lean too heavily on this, but this is what jobs did when he came back to Apple as well. He reorganized Apple to be functional, and it wasn't like we were following a playbook. We discovered this as we were investigating what it's going to take to make these teams more tech-focused and to bring our DNA back to our roots, which really was putting engineering and design first, which is what technology first means to me. So yeah, I would say to companies, find your DNA and really try to optimize for what that is in a very simple and clear way.

中文翻译:

是的，我想这基本上就是我们所做的。不想说得太绝对，但这正是乔布斯回到苹果时所做的。他将苹果重新组织为职能型架构。我们并不是在照搬剧本，而是在研究如何让团队更专注于技术、如何让我们的 DNA 回归本源（即工程和设计优先，这对我来说就是“技术优先”的含义）的过程中发现这一点的。所以，我会对其他公司说：找到你们的 DNA，并尝试以一种非常简单明了的方式对其进行优化。

(00:12:00) Lenny Rachitsky

English:

Okay, so you made a bunch of changes, you had this manifesto, everyone's on board, you made a bunch of changes. Functional technology first, comparing the way that your say engineering team works today versus two or three years ago, what is most different?

中文翻译:

好，你做了一系列改变，有了宣言，大家也都支持。职能型架构、技术优先。对比你们现在的工程团队和两三年前的工作方式，最大的不同是什么？

(00:12:13) Dhanji R. Prasanna

English:

Not everyone was on board, I'll tell you that. It was quite a painful transformation. I think that one of the things that I learned the most throughout this process is that Conway's Law can be really, really powerful. So it's the law that basically says you ship your org structure. So what you're organized as in terms of teams, in terms of collaborating groups and your operating model matters a lot to what you build. And so I think that that was essentially the biggest change is we had a lot of momentum in each of these silos, be it Cash App, be it Afterpay, be it Square or even TIDAL or music streaming service. And no one was really talking to each other, no one was really aligned on technical strategy on what we even wanted to be five years from now as a collective team. And so all those things are different now. I'm not saying it's perfect, there's still a long road ahead of us, but we at least speak the same language. We're all have access to the same tools, we share the same policies. So a certain level of senior engineer means the same thing across the whole company. People can move from one team to another's into an area of need. All of these things are very different. But to sum it up, I would say we're technically focused and we're focused on advancing technical excellence as a goal. And that just really wasn't that true two to three years ago. There were other things we were optimizing for then.

中文翻译:

我得告诉你，并不是每个人都支持。这是一个相当痛苦的转型过程。在这个过程中我学到最深的一点是，康威定律（Conway's Law）真的非常强大。这条定律基本上是说：你交付的产品就是你组织架构的反映。你的团队组织方式、协作小组和运营模式对你构建的东西影响巨大。我认为本质上最大的改变是，以前我们在每个孤岛（无论是 Cash App、Afterpay、Square 还是 TIDAL 音乐流媒体服务）中都有很大的惯性。大家互不交流，在技术战略上也没有达成一致，甚至作为一个整体团队，我们五年后想成为什么样都不清楚。现在这些都不同了。我不是说现在很完美，前面还有很长的路要走，但至少我们现在说的是同一种语言。我们使用相同的工具，共享相同的政策。在全公司范围内，某个级别的资深工程师意味着同样的水平。人们可以根据需要从一个团队调动到另一个团队。所有这些都非常不同。总而言之，我们现在以技术为中心，并将提升技术卓越性作为目标。而两三年前，这并不是事实，那时我们在优化其他东西。

(00:13:58) Lenny Rachitsky

English:

Maybe going one level deeper in terms of how people actually work at a day. So if you're looking at an engineering team, say the average engineering team and maybe also the top most optimal engineering team, how is the way they work today different from a couple of years ago?

中文翻译:

也许可以再深入一层，看看人们日常具体是如何工作的。如果你观察一个工程团队，比如一个普通的工程团队，或者一个最顶尖、最优化的工程团队，他们今天的工作方式与几年前有什么不同?

(00:14:12) Dhanji R. Prasanna

English:

In the small, certain teams that are very, very AI natives or teams that are building AI first everywhere are working much differently than before because they're using vibe code tools and they're essentially building without writing lines of code by hand, and that just wasn't true through the three years ago. I don't think it was true anywhere in the world. So that's dramatically different in teams that are still working with very heavy legacy code bases. It's less true, but they're also encountering these background

AI processes. So we have these tools that run 24/7 or run in the CI pipeline and they're analyzing vulnerabilities. They're looking at even bugs filed on tickets and trying to build patches while engineers are asleep. So they come in the next day and look at it. So I would say there are a number of ways in which they're different, but different teams have adapted in different ways depending on how close they are to the tools.

中文翻译:

从微观上看，某些非常“AI 原生”或者处处坚持“AI 优先”的团队，工作方式与以前大不相同。因为他们在使用“氛围编程”（vibe code）工具，基本上不需要亲手写代码就能进行构建，这在三年前是不可想象的，我想当时全世界都没有这种方式。对于那些仍在处理沉重遗留代码库的团队来说，这种变化没那么剧烈，但他们也会遇到后台 AI 进程。我们有一些 24/7 运行或在 CI（持续集成）流水线中运行的工具，它们在分析漏洞，甚至在查看工单中提交的 Bug，并在工程师睡觉时尝试构建补丁。工程师第二天上班就能看到结果。所以，不同团队根据他们与工具的接近程度，以不同的方式进行了适应。

(00:15:25) Lenny Rachitsky

English:

Okay, so let me lean into that AI piece, which is I think where you guys are most ahead of a lot of other companies. You guys built your own agent I think is how you describe Goose. So there's a lot of talk about productivity gains through AI. There's this camp of people are like, you don't understand how much productivity there is to gain from AI. It's the future, this is the way it's all going work. We're all accelerating 10X. There's also this camp where people are like, I'm so overhyped, nothing's working. People talk about it. All these pilots are failing. Nobody's actually adopting this at scale. I feel like you're probably in that first camp. What sort of gains have you seen practically from AI tools on your teams?

中文翻译:

好，让我们深入探讨 AI 这一块，我认为这是你们领先于许多其他公司的地方。你们构建了自己的智能体，也就是你描述的 Goose。现在关于 AI 提高生产力有很多讨论。有一派人认为：你根本不明白 AI 能带来多大的生产力提升，它是未来，一切都会这样运作，我们都会加速 10 倍。还有另一派人觉得：这全是过度炒作，没一样东西好使，大家只是在空谈，所有的试点项目都在失败，没人真正大规模采用。我觉得你可能属于第一派。在你的团队中，你实际上看到了什么样的收益？

(00:16:03) Dhanji R. Prasanna

English:

Our number one priority is through automate Block, which means getting AI and getting AI forms of automation through our entire company. And we feel that that's just at the beginning of where the utility is with all these large language models, and I think we're going to continue to see that improve. But even now, we find engineering teams that are very, very AI forward that are using Goose every day are reporting about eight to 10 hours saved per week, and this is self-reported. And then we also have a number of check metrics to try and validate that. So we look at PRs, we look at throughput of features, we look at a whole bunch of things and we have our data scientists come up with a complicated formula that tries to distill it all into something meaningful. And we feel across the whole company, we're probably trending towards 20 to 25% of manual hours saved. And I think that's just the start of all of this. I do feel that the more AI-native companies are doing a better job of realizing this. So companies that started just with AI startups mostly, but there is some truth to this notion that AI isn't a panacea and it's growing as well in capability. So you need to ride that wave along with it. And I think a lot of the companies aren't realizing this. They're like, "Well, where's the value?" And the truth is the value is changing every day. And so you need to be adaptable and look at what the value is today and plan for what the value will be tomorrow and then slowly expand to the areas where it's most efficacious.

中文翻译:

我们的首要任务是“自动化 Block”，这意味着在整个公司范围内引入 AI 和各种形式的 AI 自动化。我们觉得大语言模型的效用才刚刚开始展现，未来还会持续改进。但即便现在，我们发现那些非常超前使用 AI、每天都在使用 Goose 的工程团队报告称，每周可以节省大约 8 到 10 小时，这是员工自报的数据。我们还有一系列校验指标来验证这一点：我们会查看 PR 数量、功能产出量等一系列数据，我们的数据科学家会用一个复杂的公式将这些数据提炼成有意义的结论。我们觉得在全公司范围内，手动工时的节省比例可能正趋向于 20% 到 25%。我认为这仅仅是个开始。我确实觉得那些更具“AI 原生”特质的公司在实现这一价值方面做得更好。虽然 AI 不是万灵药，而且它的能力也在不断增长，所以你需要顺应这股浪潮。很多公司没有意识到这一点，他们会问：“价值在哪儿？”事实是价值每天都在变化。因此，你需要具备适应性，看清今天的价值，规划明天的价值，然后慢慢扩展到最有效的领域。

(00:17:54) Dhanji R. Prasanna

English:

I'll give you an example. One area in which we find that it's really good is for non-technical teams to be able to build little software tools for themselves. So this has been one of the most surprising and energizing uses of Goose within Block is we'll have our enterprise risk management team build a whole system for self-servicing enterprise risk, and this is compressing weeks of work into hours, or ordinarily, they would be waiting for an internal apps team or something to go and build that and they would put that on their Q2 roadmap and everyone would be twiddling their thumbs until it all clicked into place, but now you can just go and do it. And so a lot of these kinds of use cases we're seeing an enormous amount of productivity gain in the other area, which I'm really excited about is we have this other tool called Gosling, which is a goose for mobile effectively. So it operates your Android OS at a native level using the accessibility API. And we use that for automating UI tests. So before, you would have to hire an army of contractors or QAs who would go and click through every screen, but now we can just bake those into automated tests and then give you a report at the end. So we're seeing a lot of advantages in those types of areas, but where you have a lot of depth and a lot of really strong people come together is where AI, I think still underperforms humans. And that's something that's probably going to get better over time, but it's also something where we should lean into as humans. So when you have some very senior engineers and they're thinking about things like architecture and design and race conditions, orchestration, things like this, that's still an area where AI isn't quite there. And so I think the companies that aren't feeling the success in AI are trying to just throw these tools at their giant code bases and hoping good things will happen, and that's not how it's playing out. Eventually, I do think it'll get there, but right now we're still in the early utility phase.

中文翻译:

我给你举个例子。我们发现一个效果非常好的领域是让非技术团队能够为自己构建小型软件工具。这是 Goose 在 Block 内部最令人惊讶且振奋的用途之一：我们的企业风险管理团队构建了一整套企业风险自服务系统，这把几周的工作压缩到了几小时。通常情况下，他们得等内部应用团队去开发，对方会把它排进第二季度的路线图，然后大家就只能无所事事地干等着。但现在，你可以直接动手做。在这些用例中，我们看到了巨大的生产力提升。另一个让我兴奋的领域是我们有一个叫 Gosling 的工具，它实际上是移动版的 Goose。它通过辅助功能 API 在原生层面操作 Android 系统。我们用它来自动化 UI 测试。以前，你需要雇佣大量的合同工或 QA（测试人员）去点击每一个屏幕，但现在我们可以直接把这些做成自动化测试，最后给你一份报告。我们在这些领域看到了很多优势。但在需要极深造诣和顶尖人才协作的地方，我认为 AI 的表现仍然不如人类。这可能会随时间而改善，但也是我们作为人类应该深耕的地方。比如资深工程师思考架构、设计、竞态条件、编排等问题

时，AI 还没达到那个水平。所以我觉得那些没在 AI 上看到成功的公司，只是把这些工具扔进庞大的代码库，然后指望奇迹发生，但这行不通。最终我认为它会实现的，但现在我们还处于早期实用阶段。

(00:20:18) Lenny Rachitsky

English:

Holy moly, there's so much there in what you just shared. There's like five things I want to follow up on. Okay, so one is this metric you kind of alluded to, which is how you measure the impact of AI in your team. So it was human manual hours saved, is that how you describe it?

中文翻译:

天哪，你刚才分享的内容信息量太大了。我有大约五件事想追问。好，第一点是你提到的那个指标，也就是你如何衡量 AI 对团队的影响。你把它描述为“节省的人工手动工时”，对吗?

(00:20:35) Dhanji R. Prasanna

English:

That's correct. Yeah.

中文翻译:

没错。是的。

(00:20:36) Lenny Rachitsky

English:

So it's roughly a fourth of an engineer's time currently is being saved by AI tooling.

中文翻译:

所以目前工程师大约四分之一的时间是被 AI 工具节省下来的。

(00:20:43) Dhanji R. Prasanna

English:

That metric is across all teams. So that would be our support teams, our legal teams, our risk teams, all of them together.

中文翻译:

那个指标是跨所有团队的。包括我们的支持团队、法律团队、风险团队，所有团队加在一起。

(00:20:51) Lenny Rachitsky

English:

Wow.

中文翻译:

哇。

(00:20:52) Dhanji R. Prasanna

English:

And then on the engineering side, it's very variable because like I said before, it matters how big and how complex the code base is. And so if you're building a totally new Greenfields code base or you're building an app for a new platform, then we're seeing those pretty aggressive gains, but in very complex code bases that already exist, those gains are not quite there yet.

中文翻译:

在工程方面，这个数值差异很大。正如我之前说的，这取决于代码库的规模和复杂程度。如果你是在构建一个全新的“绿地”项目（Greenfields，指从零开始的项目），或者为新平台开发应用，我们能看到非常显著的收益；但在已经存在的非常复杂的代码库中，这些收益还没那么明显。

(00:21:18) Lenny Rachitsky

English:

That's amazing. And whenever I hear a stat like this, I think an important element that people need to think about is this is the worst it will ever be. This is the lowest, this is now the baseline. And so it may not sound that crazy yet, but it's going to get crazy. Okay, the other thing that you talked about is Goose, you haven't explained what Goose is. This is a huge deal. Explain what Goose is and how important this has become to you guys.

中文翻译:

太神奇了。每当我听到这样的统计数据，我认为人们需要记住的一点是：这已经是它表现最差的时候了。这是最低点，是现在的基准线。所以虽然现在听起来还没那么疯狂，但以后会变得非常疯狂。好，你提到的另一件事是 Goose，你还没解释 Goose 到底是什么。这可是件大事。请解释一下 Goose 是什么，以及它对你们有多重要。

(00:21:49) Dhanji R. Prasanna

English:

So Goose is a general purpose AI agent. So you can think of it as a desktop tool or a program that you can download and install on your computer and then it has a UI. You can talk to it just like a chatbot and you can say anything from, "Hey Goose, organize my photos by category, and it has the ability to look within your photos and if there are a lot of trees, it'll organize them as nature photos. And there are a lot of people, it'll organize them as portraiture." All of this sort of stuff to writing software for you. So it can do all of these tasks, and the way we've been able to do this is through something called a model context protocol or the MCP, which a lot of your listeners might've heard. And this is something that Anthropic came up with that we were a very early contributor to. And the model context protocol is very simply just a set of formalized wrappers around existing tools or existing capabilities. So if you have tools that you use in the enterprise, be it Salesforce or be it Snowflake or SQL, any of these things, you can wrap them in the MCP and then it exposes them to your LLM to be able to manipulate. So until that point, the LLMs were not really able to do much other than chat, but Goose gives these brains arms and legs to go out and act in our digital world, and that's where we find it's had most impact and it's built on this fairly open protocol that anyone can implement. There have been an explosion of MCPs. Goose is entirely open source, by the way, so any of you can download it and extend it, write your own MCPs, and that's been our core successes through Goose.

中文翻译:

Goose 是一个通用 AI 智能体。你可以把它看作一个桌面工具或程序，下载安装到电脑上，它有一个 UI 界面。你可以像和聊天机器人说话一样和它交流，你可以说任何话，比如："嘿 Goose，按类别整理我的照片。"它有能力查看你的照片，如果照片里有很多树，它会把它们归类为自然照片；如果有很多面孔，它会归类为肖像。从这类琐事到为你编写软件，它都能胜任。我们之所以能做到这一点，是通过一种叫做"模型上下文协议"（Model Context Protocol，简称 MCP）的东西，很多听众可能听说过。这是 Anthropic 提出的，我们是非常早期的贡献者。MCP 简单来说就是一套围绕现有工具或能力的正式封装。如果你在企业中使用 Salesforce、Snowflake 或 SQL 等工具，你可以用 MCP 封装它们，然后将它们暴露给 LLM 进行操作。在此之前，LLM 除了聊天基本干不了别的，但 Goose 给这些"大脑"装上了"手脚"，让它们能在数字世界中采取行动。这就是我们发现它影响力最大的地方。它建立在这个任何人都可以实现的开放协议之上。现在 MCP 已经迎来了爆发式增长。顺便说一下，Goose 是完全开源的，所以你们任何人都可以下载、扩展它，编写自己的 MCP。这就是我们通过 Goose 取得的核心成功。

(00:24:06) Lenny Rachitsky

English:

Okay. So essentially like Claude code with a UI, desktop app sort of thing built on top of Claude and OpenAI ChatGPT and a bunch of open source models. Is that right?

中文翻译:

好。所以本质上它就像是带 UI 的 Claude Code，是一个构建在 Claude、OpenAI ChatGPT 和一堆开源模型之上的桌面应用。对吗?

(00:24:06) Dhanji R. Prasanna

English:

Yeah, it can use any model. So we have a pluggable provider system and you can either bring your own API keys and use the Claude family models or OpenAI's family models, or you can use open source models and you can download them and use them directly or via Ollama and other, there are several tools that help you do that, but essentially it's taking the capability of these models to generate text and to interpret text and applying them to real world situations. So one example that I really like is you can ask Goose to go and build your marketing report and it has MCPs to connect to Snowflake and Tableau and Looker. So it'll write SQL to pull out data from there, it'll do some analysis and a CSV so it can write Python code on your desktop to do all that. It will generate some graphs using some JavaScript charting library that it knows about. And then finally, it'll put this all into a PDF or Google Doc or whatever and it can even email it for you or upload it somewhere. And it's doing all of this on its own, by the way. No one's sitting here telling it that, you're just saying, "Hey, I want this report, I want this emailed here, I want these pretty charts." And it's orchestrating across all these systems.

中文翻译:

是的，它可以使用任何模型。我们有一个可插拔的提供商系统，你可以带上自己的 API Key 使用 Claude 系列或 OpenAI 系列模型，也可以下载开源模型通过 Ollama 等工具直接使用。本质上，它是利用这些模型生成和理解文本的能力，并将其应用于现实世界。我非常喜欢的一个例子是：你可以让 Goose 去生成一份营销报告，它有 MCP 可以连接到 Snowflake、Tableau 和 Looker。它会编写 SQL 从中提取数据，在 CSV 中进行分析（它可以在你的桌面上编写 Python 代码来完成这些），它还会利用它知道的 JavaScript 图表库生成图表。最后，它会把这些放入 PDF 或 Google Doc 中，甚至帮你发送邮件或上传到某处。顺便说一下，这一切都是它自主完成的。

没人坐在那儿一步步教它，你只需说：“嘿，我想要这份报告，发到这个邮箱，我想要这些漂亮的图表。”它就会在所有这些系统之间进行编排。

(00:25:31) Lenny Rachitsky

English:

So essentially at Block, instead of using Claude or ChatGPT directly or even Cursor and all these apps, they use Goose?

中文翻译:

所以本质上在 Block，员工们不再直接使用 Claude 或 ChatGPT，甚至也不用 Cursor 这些应用，而是使用 Goose?

(00:25:40) Dhanji R. Prasanna

English:

Yeah, we allow our engineers and our general employee population to use any tools that they want. Goose is the one that's most well-integrated into all of our systems because it's built on the MCP and it's so easy to create an MCP for an existing system. So for example, if you're using a issue tracking tool and you want some AI automation added to it, before Goose, our teams would have to wait for the vendor to build that AI capability in there, or maybe there's some way in which OpenAI or Anthropic or Google would provide a general purpose capability where we could plug those in. But with Goose, that's no longer necessary with a few lines of code that an MCP represents. All these systems are orchestratable with AI basically overnight, and Goose can write its own MCPs. So it's pretty bootstrappable as well.

中文翻译:

是的，我们允许工程师和普通员工使用任何他们想用的工具。Goose 是与我们所有系统集成得最好的，因为它基于 MCP，而为现有系统创建 MCP 非常容易。举个例子，如果你在使用一个工单追踪工具，想加入 AI 自动化功能，在有 Goose 之前，我们的团队必须等待供应商开发 AI 功能，或者寄希望于 OpenAI、Anthropic 或 Google 提供某种通用的插件接口。但有了 Goose，这就不再必要了，只需几行 MCP 代码，所有这些系统基本上在一夜之间就能通过 AI 进行编排。而且 Goose 还能编写自己的 MCP，所以它的自我引导能力也很强。

(00:26:43) Lenny Rachitsky

English:

And this is open source and basically you've spent all this time building this thing, any other company can now implement it and build on all the work you've done?

中文翻译:

而且这是开源的，基本上你们花了这么多时间构建的东西，任何其他公司现在都可以直接采用，并在你们的工作基础上继续构建?

(00:26:51) Dhanji R. Prasanna

English:

Yeah, and we have a lot of companies using Goose pretty actively. I don't want to name too many names, but from our competitors to our close partners, a lot of them are using Goose pretty regularly on their teams. I know Databricks talks about it a lot, but everyone you can think of in this mid-tech tier is using Goose in some form.

中文翻译:

是的，有很多公司在积极使用 Goose。我不便透露太多名字，但从我们的竞争对手到紧密合作伙伴，很多团队都在定期使用 Goose。我知道 Databricks 经常提到它，基本上你能想到的中型科技公司都在以某种形式使用 Goose。

(00:27:23) Lenny Rachitsky

English:

That's insane. This feels like it could've been a massive business of its own, some of the fastest growing companies in the world, basically this is their product and you've built it and given away.

中文翻译:

这太疯狂了。这感觉本身就可以成为一项巨大的业务，世界上一些增长最快的公司，其核心产品基本上就是这个，而你们把它做出来并免费送人了。

(00:27:23) Dhanji R. Prasanna

English:

Yeah, we believe in the power of open source and one of our core missions is to increase openness, and that means contributing to open protocols and contributing to open source. And as a tech company, we're built on a lot of open source software. I think pretty much every tech company is whether you're talking about Linux or Java or MySQL or any of these essential components, and so we feel like we have a strong imperative to give back. We want to build things that not only are good for us and our customers, but that outlast Block and outgrow Block, that's certainly a core value for us and has been from the beginning even long before this whole AI phase. So yeah, Goose follows in that proud tradition and yeah, we're very excited that its had the success it's had.

中文翻译:

是的，我们相信开源的力量，我们的核心使命之一就是增加开放性，这意味着为开放协议和开源做贡献。作为一家科技公司，我们是建立在大量开源软件之上的。我想几乎每家科技公司都是如此，无论是 Linux、Java、MySQL 还是任何这些核心组件。因此，我们觉得有强烈的责任去回馈。我们想要构建的东西，不仅要对我们和我们的客户有好处，还要能比 Block 存在得更久、发展得更广。这确实是我们的核心价值观，从公司创立之初就是如此，远早于现在的 AI 浪潮。所以，Goose 继承了这一优良传统，我们对它取得的成功感到非常兴奋。

(00:28:18) Lenny Rachitsky

English:

What's the story with the name Goose, by the way? Can't help but ask.

中文翻译:

顺便问一下，Goose 这个名字是怎么来的？忍不住想问。

(00:28:21) Dhanji R. Prasanna

English:

Goose is a Top Gun reference. So our engineer that came up with it. He also looks exactly like Goose, so it's kind of crazy if you put them side to side, he's going to be really embarrassed with my sharing this, but that's the reason why they call it Goose, and then we lent into the whole bird theme after that.

中文翻译:

Goose 是致敬电影《壮志凌云》（Top Gun）。想出这个名字的工程师长得简直和电影里的 Goose 一模一样，如果你把他们放在一起看，真的挺疯狂的。他要是知道我分享这个肯定会很尴尬，但这就是叫它 Goose 的原因。之后我们就顺着这个思路，采用了整套鸟类主题。

(00:28:41) Lenny Rachitsky

English:

That's incredible. There's a story I heard you share on a different podcast where there's an engineer who takes this to the extreme and has Goose watch him. Talk about that, share that story.

中文翻译:

太有意思了。我在另一个播客里听你分享过一个故事，说有一个工程师把这做到了极致，让 Goose 盯着他看。能聊聊那个故事吗?

(00:28:50) Dhanji R. Prasanna

English:

Yeah, absolutely. So he is very, very AI-focused and he's trying to extract all these crazy ideas from Goose and Goose can do all of the things that I described through specific interactions with tools, but it can also just watch your screen so it understands how to process images and process the things that it's looking at through screenshots. And so he built this system where it's essentially just watching everything he does all the time and he'll be talking to a colleague on Slack or an email and they'll be discussing some feature that they think is useful to implement. And then a few hours later he'll find that Goose has already tried to build that feature and opened a PR for it on Git and all sorts of other wacky things like that. So it'll try to nudge him out of a workflow. If he's running over on a meeting and he's late for something else, it comes up with these creative things that he didn't program or he didn't write prompts for, but that it thinks will help him improve his productivity or improve his work day. So yeah, it's pretty crazy. You have to have the stomach for it to be that level of tied into your working tools, but it kind of shows you what's possible with tools like this.

中文翻译:

当然。他非常专注于 AI，试图从 Goose 身上挖掘各种疯狂的想法。Goose 除了能通过工具交互完成我之前说的那些事，还能直接“看”你的屏幕，通过截图理解图像和内容。于是他构建了一个系统，基本上让 Goose 一直盯着他做的一切。比如他在 Slack 或邮件里和同事讨论某个觉得有用的功能，几小时后，他会发现 Goose 已经尝试写好了代码并在 Git 上开了 PR，还有各种其他稀奇古怪的事。比如它会尝试把他从某个工作流中拉出来；如果他开会超时了，导致另一件事迟到，它会想出一些他没编程过、也没写过提示词的创意点子，因为它觉得这能帮他提高效率或改善工作日体验。这确实挺疯狂的。你得有足够强大的心理承受能力才能让 AI 如此深度地介入你的工作，但这确实展示了这类工具的可能性。

(00:30:14) Lenny Rachitsky

English:

Clearly this is where things are going. Once this gets good enough, I love this guy is just trying it. So it's basically watching him work and anticipating what he should be doing and does the work for him as a first draft so that he's like, "Oh, the PR is already done on this thing. We were just talking about it at this meeting." That's incredible.

中文翻译:

显然这就是未来的方向。一旦这变得足够好……我很喜欢这家伙勇于尝试的精神。所以它基本上是在观察他的工作，预判他该做什么，并为他完成初稿，以至于他会惊叹："噢，这事的 PR 已经做好了，我们刚才开会才聊到它。"太不可思议了。

(00:30:31) Dhanji R. Prasanna

English:

Exactly.

中文翻译:

没错。

(00:30:32) Lenny Rachitsky

English:

How good is it? Where's it at? If you had to go zero to a hundred of like, "Okay, going to, all you have to do is now think and talk and that'll just do your job."

中文翻译:

它现在有多好？处于什么水平？如果 0 到 100 分，100 分代表"你只需要思考和说话，它就能完成你的工作"，它现在能打多少分?

(00:30:41) Dhanji R. Prasanna

English:

Yeah, so voice is the other big part of it. It has voice processing capability, so it's always listening to what he's saying as well and trying to interpret that. I would say that this is mostly an experiment, given that he's on our core Goose team and he contributes to Goose, so he has a day job. This is a kind of thing on the side that he was developing. So once this evolves into more of a native feature of Goose itself or other tools that we use in the enterprise, I think it can have a lot of legs, but it's already pretty good. It's probably cutting down enormous amounts of busy work that he has to do. So for example, one thing he'll do is he'll say, "Oh, I have a meeting conflict. I can't make it that time, or I have to go pick up my kid." And Goose will automatically reschedule that meeting without him ever sitting in front of his calendar and clicking through 10 times. Yeah, so these are things that I think we were waiting for the calendar vendor to build as features into calendar, but we don't need to do that anymore because AI is able to orchestrate this for us.

中文翻译:

是的，语音是另一个重要部分。它具有语音处理能力，所以它也一直在听他在说什么并尝试解读。我想说这目前主要还是个实验，因为他在我们的 Goose 核心团队工作，他有自己的本职工作，这是他业余开发的东西。一旦这演变成 Goose 本身或我们在企业中使用的其他工具的原生功能，我认为它会有很大的发展空间。但它现在已经很不错了，可能帮他省去了大量的琐碎工作。比如，他会说："噢，我会议冲突了，那个时间去不了，或者我得去接孩子。" Goose 就会自动重新安排会议，而不需要他坐在日历前点来点去。这些功能我们以前可能在等日历供应商去开发，但现在不需要了，因为 AI 能够为我们编排这一切。

(00:31:53) Lenny Rachitsky

English:

This isn't that guy that had four jobs at four different startups that he was able to paralyze all his work and hire people.

中文翻译:

这家伙不是那种在四家不同初创公司打四份工，然后把工作全部并行化并雇人来干的那种人吧?

(00:31:58) Dhanji R. Prasanna

English:

No, it's not. He's someone that I've worked with for a long time and he's been at Block for a long time. He just loves experimenting and he embodies that culture of experimentation just like our creator of Goose who did the same thing.

中文翻译:

不，不是。他是我共事很久的人，在 Block 待了很多年。他只是热爱实验，他体现了那种实验文化，就像我们 Goose 的创造者一样。

(00:32:16) Lenny Rachitsky

English:

So let me pull on that thread a little bit. You're kind of seeing a glimpse of where things are going. You're very ahead of the curve in a lot of ways at Block. How do you think things will look in a couple of years in terms of how engineers work, how product teams work that's different from today?

中文翻译:

那让我顺着这个话题再深入一点。你已经看到了未来的一瞥。在 Block，你们在很多方面都处于领先地位。你认为几年后，工程师和产品团队的工作方式会与今天有什么不同?

(00:32:32) Dhanji R. Prasanna

English:

I think a lot of it is dependent on the improvement of LLM performance, but I can tell you the way I'm trying to change how I work and how I'm trying to change our immediate team's way of working. So I think vibe coding has been an interesting, exciting thing, which is you talk to a chatbot essentially and it goes and builds software for you, but I think this is highly limiting. It's very ping pong. You do something, you wait for three or four minutes and it comes back with something sort of half-baked and you have to nudge it and guide it and massage it to get where it needs to be. I think that we're going to see much more autonomy. So where we're working on a couple of experiments with Goose, with the next version of Goose where we're really trying to push it to work not just for two or three or five minutes at a time, our median session length is five minutes and on average, seven, but we're trying to push it to hours. We're trying to say, "Hey, all these LLMs are sitting idle overnight and on weekends while humans aren't there, there's no need for that." They should be working all the time. They should be trying to build in anticipation of what we want if we go back to the earlier part of the conversation. But also I think that they should be able to build in ways that were never possible before. Before as humans, we had limited resources, limited bandwidth, and a lot of coordination overhead. So we would have to choose the best path to try in an experiment, and I don't think we need that anymore. We need instead to be able to describe multiple different experiments in a great amount of detail. And then maybe we go to sleep and then in the morning, all those experiments are built and we can sort of throw away five or six of them.

中文翻译:

我认为很大程度上取决于 LLM 性能的提升，但我可以告诉你我正在尝试如何改变自己的工作方式，以及如何改变我们核心团队的工作方式。我觉得“氛围编程”（vibe coding）是一件有趣且令人兴奋的事，本质上就是你和聊天机器人说话，它帮你构建软件。但我认为这有很大的局限性，非常像打乒乓球：你做点什么，等三四分钟，它带回一个半成品，你得不断微调、引导、修饰才能达到要求。我认为我们将看到更多的自主性。我们正在对下一代 Goose 进行一些实验，尝试让它不仅能连续工作两三分钟或五分钟（我们目前的中位会话长度是 5 分钟，平均 7 分钟），而是能连续工作数小时。我们想说：“嘿，所有这些 LLM 在深夜和周末人类不在时都闲着，这没必要。”它们应该一直工作，应该尝试预判我们的需求进行构建。而且我认为它们应该能以以前不可能的方式进行构建。以前作为人类，我们的资源、带宽有限，协调成本很高，所以我们必须在实验中选择一条最佳路径。但现在我觉得不需要了，我们可以详细描述多个不同的实验，然后去睡觉，第二天早上所有实验都做好了，我们可以直接扔掉其中五六个不行的。

(00:34:04) Dhanji R. Prasanna

English:

So one of the things that I do regularly, so I write code every day, but one of the things that I do regularly is just throw away huge, huge amounts of code, and it's kind of hard for me because I've never done that before. I mean obviously engineers love deleting code, but this is different. You build a whole new system or a whole new feature and you're like, "Ah, it doesn't feel exactly right. I'm just going to delete and start over." So I think you're going to see a lot more of that way of working. And I think that you're going to see instead of us, for example, refactoring an app to have a different UI or to evolve into its new version, we're just going to rewrite that app from scratch. And one of the things I'm really pushing our teams to think about is what would our world look like if every single release, RM minus RF deleted the entire app and rebuilt it from scratch? And so we can't really do that today, but I think this shows you some of the direction of what's possible and where these tools are taking us.

中文翻译:

我每天都写代码，但我现在经常做的一件事就是扔掉大量的代码。这对我来说有点难，因为我以前从未这样做过。虽然工程师都喜欢删代码，但这不一样。你构建了一个全新的系统或功能，然后觉得“嗯，感觉不太对，我直接删了重来”。我认为你会看到更多这样的工作方式。而且我认为，未来我们不再是重构一个应用来更换 UI 或升级版本，而是直接从头重写。我一直在推动团队思考：如果每一次发布，我们都用 `rm -rf` 删掉整个应用并从头重建，我们的世界会变成什么样？虽然今天还做不到，但这展示了未来的可能性以及这些工具将带我们走向何方。

(00:35:42) Lenny Rachitsky

English:

What's interesting about that is that there's this common rule in software engineering and just product, don't ever just rewrite. Don't try to rewrite your thing. You're going to forget all of the small improvements and tweaks and bug fixes people have made over the years, and you think it's going to be the simple straightforward thing. It ends up being now it's like a year or more of just getting it back to where it was. And so interesting that AI now can make that possible, and what you're saying is that's actually maybe the way you should be working.

中文翻译:

有趣的是，软件工程和产品界有一条通用准则：永远不要轻易重写。不要尝试重写你的东西，因为你会忘记人们多年来做出的所有微小改进、调整和 Bug 修复。你以为重写很简单直接，结果最后花了一年甚至更久才让它回到原来的水平。所以 AI 让这变得可能确实很有趣，而你认为这也许正是我们应该工作的方式。

(00:36:09) Dhanji R. Prasanna

English:

I think so. And I think that the trick is getting the AI to respect all of those incremental improvements, yeah, and sort of bake those in as a part of the specification, if you will. Yeah.

中文翻译:

我是这么认为的。关键在于让 AI 尊重所有这些增量改进，并将它们作为规范（specification）的一部分融入其中。

(00:36:25) Lenny Rachitsky

English:

Also, the point you made about this agent, just you give it a bunch of ideas that builds them overnight and then you could see, I imagine it goes even further up the stack and comes up with the ideas and then starts building them and then you're like, "Okay, oh, that was a great idea. Now I can see it immediately in the same workflow."

中文翻译:

还有你提到的关于智能体的点，你给它一堆想法，它一夜之间就建好了。我能想象它甚至会进一步向上发展，自己想出点子然后开始构建，然后你会说：“噢，这个主意太棒了，我现在就能在同一个工作流里看到它。”

(00:36:39) Dhanji R. Prasanna

English:

Yeah, that's true. I was actually literally trying what you're saying just last week. And so I have this new version of Goose that we're working on and I was asking it to come up with ideas to improve itself and implement it overnight. And sometimes-

中文翻译:

是的，没错。上周我真的尝试了你说的这种方式。我正在开发新版本的 Goose，我让它想出改进自己的点子，并在一夜之间实现。有时候——

(00:36:57) Lenny Rachitsky

English:

Slip problem.

中文翻译:

（笑）这可能会出问题。

(00:36:59) Dhanji R. Prasanna

English:

... Sometimes it kind of goes off the script entirely and you have to sort of pull it back a bit. So I think we're not quite at that era where it's completely self-improving and completely autonomous, but I do think we're in a transition phase where we can give it that nudge and say, "Hey, here's my wishlist of 10 things that I wish you could do. Go and figure out the best way to do them." And it's successful I would say on 60% of those things, if the features are well enough described and it struggles on the remaining 40 where you have to kind of intervene and massage it. Yeah.

中文翻译:

……有时候它会完全脱离剧本，你得把它拉回来一点。所以我认为我们还没到那个完全自我改进、完全自主的时代，但我确实认为我们正处于过渡阶段。我们可以推它一把，说："嘿，这是我希望你能做的 10 件事。去想出实现它们的最佳方案。"如果功能描述得足够清楚，我认为它在 60% 的事情上是成功的，但在剩下的 40% 上会比较吃力，需要你介入和调整。

(00:37:43) Lenny Rachitsky

English:

Oh man, I'm just imagining this feature where you give it the goal of drive revenue and growth and then it's just like, "Okay, everyone's fired. Here's your paychecks. I'll take it from here."

中文翻译:

天哪，我正在想象这样一个功能：你给它设定一个"驱动收入和增长"的目标，然后它说："好，大家都解雇了。这是你们的工资单。接下来的事交给我。"

(00:37:56) Dhanji R. Prasanna

English:

I don't think we're going to be there. I do think we're going to need a lot of human taste to anchor these AIs so they don't go off script to be honest. And that's really where our design lead and our design teams are pushing us to think, and that's a differentiator that I think will push us beyond this era of AI slop that everyone's talking about. So yeah, it's very much anchoring it into a thing that matters to people and the thing that's tasteful and useful and has value.

中文翻译:

我不认为我们会走到那一步。老实说，我认为我们需要大量的人类品味来锚定这些 AI，让它们不脱离轨道。这正是我们的设计负责人和设计团队在推动我们思考的地方。我认为这是一个差异化因素，能让我们超越大家都

在谈论的“AI 垃圾内容”（AI slop）时代。所以，这更多是将其锚定在对人们重要、有品味、有用且有价值的事情上。

(00:38:30) Lenny Rachitsky

English:

To make that even more concrete, is there an example of something maybe AI was trying to, or a team was trying to pitch where you had to just know this is where humans are going to step in and keep things on track?

中文翻译:

为了更具体一点，有没有什么例子是 AI 尝试去做，或者团队尝试推销某个方案，而你必须明确知道“这里需要人类介入来保持方向正确”？

(00:38:43) Dhanji R. Prasanna

English:

I'd say it was more around things like process automation or a lot of times I'll get this sort of request where a team will say, "We need to buy this new tool from this vendor because our current tool is entering X, Y and Z." Another team will say, "No, no, no, we can just use Goose to build an app that will do the same thing for us in half the time or less." And then as a human, you're sitting there thinking, "Is any of this necessary? If we just change the process, do we even need to think about building tools?" And this is the thing that AI isn't good at, it's not able to have this portfolio judgment or judgment across a global sense of what's important and what matters. So a lot of times, I tell teams just question the base assumption, particularly our InfoSec teams because they'll twist themselves into knot sometimes trying to secure something and you'll be like, "We'll just ask the team that's building it to do it differently or to not build that at all if it doesn't matter, and then you won't have to increase your surface area of securing it." So I think those are the areas where it's better for a human to use judgment and AI has not done a great job.

中文翻译:

我想说这更多是关于流程自动化。很多时候我会收到这样的请求，一个团队说：“我们需要从这个供应商那里买个新工具，因为我们现在的工具遇到了 X、Y、Z 问题。”另一个团队会说：“不不不，我们可以直接用 Goose 开发一个应用，用一半甚至更少的时间就能实现同样的功能。”然后作为人类，你坐在那里会想：“这些真的有必要吗？如果我们直接改变流程，我们还需要考虑构建工具吗？”这就是 AI 不擅长的地方——它无法具备这种全局性的判断力，无法从整体上判断什么是重要的、什么是关键的。所以很多时候，我告诉团队要质疑基本假设。特别是我们的信息安全团队，有时他们为了保护某样东西会把自己搞得焦头烂额，而你会说：“直接让开发团队换个方式做，或者如果那东西不重要就干脆别做了，这样你就不需要增加安全防御面了。”我认为在这些需要判断力的领域，人类表现更好，而 AI 做得不够好。

(00:40:11) Lenny Rachitsky

English:

You make this point about building your own software, your own tools instead of buying stuff. This is a big question with AI, is it's going to replace all these SaaS apps to Salesforce over. Is there a sense of just either how much money you guys have maybe saved building your own stuff, or have you built a new-found respect for the existing SaaS software that everyone's using and pays lots of money for?

中文翻译:

你提到了构建自己的软件和工具而不是购买。这是关于 AI 的一个大问题：它是否会取代所有的 SaaS 应用，比如 Salesforce 之类的。你们有没有算过自己构建工具省了多少钱？或者说，你们是否对大家都在用且支付巨额费用的现有 SaaS 软件产生了新的敬意？

(00:40:31) Dhanji R. Prasanna

English:

I think there's a trap in getting away from your core purpose as a company. And our core purpose is economic empowerment. So getting customers or merchants or artists the ability to make a sale or pay their rent or upload their latest creation to TIDAL. And I think that anything that serves that purpose, we should encourage and we should invest in, but if we're just purely looking at dollars versus dollars, then that's pulling us off that purpose. The savings and costs that there might be in replacing a vendor tool by something you build in-house is probably not worth it in the mental bandwidth that you've lost and the amount of the team's technical focus that's being taken away. So yeah, I would say it just keep coming to the thing that matters to you as a company and then the rest will follow from that.

中文翻译:

我认为脱离公司的核心宗旨是一个陷阱。我们的核心宗旨是“经济赋能”——让客户、商家或艺术家有能力完成一笔销售、支付房租或将最新的创作上传到 TIDAL。我认为任何服务于这个宗旨的事情，我们都应该鼓励和投资。但如果我们纯粹只看省了多少钱，那就会让我们偏离宗旨。通过自研工具取代供应商工具所节省的成本，可能抵不上你损失的精力带宽，以及团队技术焦点被分散所带来的损失。所以，我会说，始终回到对公司真正重要的事情上，其他的自然会水到渠成。

(00:41:38) Lenny Rachitsky

English:

Yeah, I think people forget just how much maintenance it takes to keep something you've built. Like, "Okay, cool, we built it in a weekend and now it's years of endless maintenance and requests and support." And also to your point, it feels like it comes back to the always motto of just focus on your core competencies and then buy everything else.

中文翻译:

是的，我想人们常忘了维护自研产品需要投入多少精力。就像，“太棒了，我们一个周末就把它做出来了”，结果接下来是数年无休止的维护、需求和支持。而且正如你所说，这似乎又回到了那个永恒的座右铭：专注于你的核心竞争力，其他的都用买的。

(00:41:57) Dhanji R. Prasanna

English:

Yeah, it's the classic 80/20 problem, and we have that enough with the apps that we build for our customers. We'll build some great experiments that really resonate, and then we have to spend a lot time ironing out the long tail of problems. So in Cash Card, for example, we built the entire functionality of Cash Card, I would say pretty much in a weekend or maybe a week of integration and work. And then it took a really long time to iron out all these edge cases where someone would tip twice the value of the bill and then it would completely break something in the back end, or people would use it as a gas station and they have a different way of billing your card. So yeah, it's very much that. And to your point, I would always come back to what is the reason we're doing this? Why does it matter to us and to our customers? And if it doesn't clearly satisfy that, I would just push it off as a not interesting thing.

中文翻译:

是的，这是经典的 80/20 问题。我们为客户开发应用时也经常遇到。我们会做一些反响很好的实验，然后得花大量时间去解决长尾问题。比如 Cash Card，我想说我们大概一个周末或者一周的集成工作就完成了核心功能。但后来花了很长时间去解决各种边缘情况：比如有人给的小费是账单金额的两倍，结果导致后端崩溃；或者人们在加油站使用它，而加油站扣费的方式很特别。所以确实如此。回到你的观点，我总是会问：我们做这件事的原因是什么？它对我们和客户为什么重要？如果它不能明确满足这一点，我就会把它当作不感兴趣的事情推掉。

(00:43:04) Lenny Rachitsky

English:

This episode is brought to you by Persona, the verified identity platform, helping organizations onboard users fight fraud and build trust. We talk a lot on this podcast about the amazing advances in ai, but this can be a double-edged sword. For every wow moment, there are fraudsters using the same tech to wreak havoc, laundering money, taking over employee identities and impersonating businesses. Persona helps combat these threats with automated user business and employee verification. Whether you're looking to catch candidate fraud, meet age restrictions or keep your platform safe, Persona helps you verify users in a way that's tailored to your specific needs. Best of all, Persona makes it easy to know who you're dealing with without adding friction for good users. This is why leading platforms like Etsy, LinkedIn, Square and Lyft trust Persona to secure their platform persona is also offering my listeners 500 free services per month for one full year, just head to withpersona.com/lenny to get started, that's withpersona.com/lenny.

中文翻译:

本集由身份验证平台 Persona 赞助。Persona 帮助组织入驻用户、打击欺诈并建立信任。我们在播客中经常谈论 AI 的惊人进步，但这可能是一把双刃剑。在每一个令人惊叹的时刻背后，都有诈骗者利用同样的技术制造混乱、洗钱、盗取员工身份或冒充企业。Persona 通过自动化的用户、企业和员工验证来应对这些威胁。无论你是想抓捕应聘欺诈、满足年龄限制还是保护平台安全，Persona 都能根据你的特定需求提供定制化的验证方式。最重要的是，Persona 让你在不增加正常用户负担的情况下，轻松了解你的交易对象。这就是为什么 Etsy、LinkedIn、Square 和 Lyft 等领先平台都信任 Persona。Persona 还为我的听众提供为期一年的每月 500 次免费服务，请访问 withpersona.com/lenny 开始使用。

(00:44:06) Lenny Rachitsky

English:

Thanks again to Persona for sponsoring this episode. One of the biggest parts of the conversation around AI is hiring jobs, things like that. So I have two kind of this two-part question. One is just how has the rise of all these AI tools, this increased productivity impacted the way you plan head counts and hire? And then what do you look for that's different in people you're hiring now that AI is such a big part of the way you guys work?

中文翻译:

再次感谢 Persona 的赞助。关于 AI 的讨论中，很大一部分是关于招聘和工作。我有两个问题：第一，这些 AI 工具的兴起和生产力的提高，如何影响了你们规划员额（headcount）和招聘的方式？第二，既然 AI 已经成为

你们工作的重要组成部分，你们现在招聘人才时，会看重哪些与以往不同的特质？

(00:44:34) Dhanji R. Prasanna

English:

I don't think that things have progressed far enough that it's really impacted in a fundamental way how many people you would need to build an app of the scale of Cash App, for example. I think what's changed for us is much different and it has nothing to do with AI, it's what we talked about earlier is moving from our GM structure to a functional structure. And in our GM structure, our incentives were always to think of engineering headcount as a commodity. And so we would just add more engineers if we wanted to build more features and the classic mythical man person month trap or whatever it's called. And I think that moving to a functional structure completely changes that and you're like, "Well, we can leverage common platforms, common modules, we can bring in experts from across the company to advise us on how better to do this." And so those kinds of things I think have made it much different and how we hire and we no longer see engineers as a commodity to just add 100 people to go and build the next product in Cash App. But on the AI side, we're very much looking for people that are embracing these tools and that are eager to try and learn from it. We're not looking for people who are amazing AI practitioners on the get-go. I think we have those people and we're interested in those people if they ever want to work with us. But I'm much more keen on looking for that college grad who just really is eager to learn about these tools and open to it, or even the veteran who has embraced these tools and figured it out. And that's kind of where we're optimizing for who we look for rather than a specific set of skills.

中文翻译:

我不认为目前的进展已经到了能从根本上影响构建像 Cash App 这种规模的应用所需人数的程度。我认为对我们来说，真正的改变与 AI 无关，而是我们之前谈到的从 GM 架构转向职能型架构。在 GM 架构下，我们的激励机制总是把工程员额看作一种“商品”。如果我们想开发更多功能，就直接增加更多工程师，陷入了经典的“人月神话”陷阱。转向职能型架构完全改变了这一点，你会觉得：“我们可以利用通用平台、通用模块，可以从全公司请来专家指导我们如何做得更好。”这些因素改变了我们的招聘方式，我们不再把工程师看作可以随手增加 100 人去开发下一个产品的“商品”。但在 AI 方面，我们确实在寻找那些拥抱这些工具并渴望从中学习的人。我们不一定要求对方一上来就是顶尖的 AI 专家，虽然我们对这类人才也很感兴趣。但我更倾向于寻找那些对这些工具充满好奇、心态开放的应届生，或者是已经拥抱并掌握了这些工具的资深人士。这是我们在人才筛选上的优化方向，而不是针对某项特定的技能。

(00:46:44) Lenny Rachitsky

English:

So essentially the biggest change is just looking for people that are embracing AI, not being like, "No, I don't need this stuff. I'm an amazing engineer. I don't need to use Cursor or Goose or all these things."

中文翻译:

所以本质上最大的改变就是寻找那些拥抱 AI 的人，而不是那种说“不，我不需要这玩意儿，我是个天才工程师，我不需要用 Cursor 或 Goose”的人。

(00:46:55) Dhanji R. Prasanna

English:

Yeah, a learning mindset is how I would put it. This is something that Jack our CEO talks about a lot is he wants us to be a learning first company. So everything we do, every experiment that we ship, what can we learn from it and did we feel that we gave it our best shot? And I think that that's more important to him than even sort of coming up with the right business answer every time.

中文翻译:

是的，我会称之为“学习心态”。我们的 CEO Jack 经常谈到这一点，他希望我们成为一家“学习优先”的公司。我们做的每一件事，发布的每一个实验，我们能从中学到什么？我们是否觉得自己尽力了？我认为对他来说，这甚至比每次都给出正确的商业答案更重要。

(00:47:25) Lenny Rachitsky

English:

What about when you're interviewing? Are you encouraging engineers to use AI tools as they're doing exercises? How did that change over the past year or two?

中文翻译:

那面试的时候呢？你会鼓励工程师在做题时使用 AI 工具吗？过去一两年这方面有什么变化？

(00:47:33) Dhanji R. Prasanna

English:

Yeah, we're starting to do that now. So traditionally we would just use CoderPad or something like that to wipe boards or a problem or even program it in Pseudocode or near Pseudocode. But now we're looking at can you use Vibe code to build something? How comfortable are you with these tools or how are you thinking about evolving with them as well? But it's early days yet I would say that it's not clear to me that necessarily how someone knows how to use, be it Goose or Cursor or any of these other tools matters that much to whether they're a good engineer. I still think that things that we interviewed for in the past, a critical mindset, the ability to really understand deeply the technical nature of a problem is still much more important than whether you're a fully AI native programmer or not.

中文翻译:

是的，我们现在开始这么做了。传统上，我们会使用 CoderPad 之类的工具进行白板面试，或者写伪代码。但现在我们会看：你能用“氛围编程”构建东西吗？你对这些工具的熟悉程度如何？你如何思考与它们共同进化？但这还处于早期阶段。我想说，目前还不清楚一个人是否擅长使用 Goose 或 Cursor 对他是否是一个优秀的工程师有多大影响。我仍然认为我们过去面试看重的东西——批判性思维、深入理解问题技术本质的能力——依然比你是否是一个完全的“AI 原生”程序员重要得多。

(00:48:37) Lenny Rachitsky

English:

Another question that I've always been thinking about a lot of people wonder is what level of engineer is most benefiting from these tools? You could argue it's the junior engineers now, they could just get all this work done. You could argue it's senior engineers because they know so much more about how things work and now they could just orchestrate thousands of agents doing their bidding. What have you seen in terms of which level is benefiting most?

中文翻译:

另一个我一直在思考、很多人也好奇的问题是：什么级别的工程师从这些工具中获益最多？你可以说是初级工程师，因为他们现在能搞定所有工作了；也可以说是资深工程师，因为他们更懂原理，现在可以指挥成千上万个智能体为他们效劳。根据你的观察，哪个级别受益最大？

(00:48:56) Dhanji R. Prasanna

English:

Yeah, so two answers to that. One is you're definitely right that the more senior and the more junior they are, the more comfortable or the more eager they are to adopt these AI tools. And I think that's for a variety of reasons, including some of them that you named. I think the senior people really understand in great depth how everything works. And so they're almost relieved that this tool exists that can go and do all these things that they've done a million times before and couldn't be bothered. And then the junior people are like my niece and nephew on a BlackBerry or something, they're just blitzing through things, not BlackBerry in the early days and iPhones now, they're blitzing through a text message when I'm still seek and destroying through my keyboard, shows you how old I am. So I think there's that, but I think the non-technical people using AI agents and programming tools to build things is really what's been surprising and really amazing. And I think that speaks to how these roles are going to evolve in the future. The lines are going to be blurred between whether you're in legal or in risk or in engineering and design even. And so I think that the people that are able to embrace it to optimize for their particular work day and their particular set of tasks are really who are showing the most impact from these tools.

中文翻译:

关于这个问题有两个答案。第一，你绝对是对的，越资深和越初级的人，往往越容易接受或渴望采用这些 AI 工具。原因有很多，包括你提到的那些。资深工程师深谙一切运作原理，所以他们几乎是如释重负：终于有工具能帮他们做那些已经做过一百万次、不想再亲自动手的琐事了。而初级工程师就像我那些用 iPhone 的侄子侄女，他们发信息的速度极快，而我还在键盘上费劲地找字母，这说明我老了。第二，我认为非技术人员使用 AI 智能体和编程工具来构建东西，才是真正令人惊讶和了不起的。这预示了未来角色的演变：法律、风险、工程甚至设计之间的界限将变得模糊。因此，我认为那些能够拥抱 AI 来优化自己特定工作日和特定任务的人，才是真正展现出这些工具最大影响力的人。

(00:50:36) Lenny Rachitsky

English:

It's interesting. No one talks about that element of engineering productivity, which is the reduction of asks from all the other parts of the company to build random one-off things. That feels like a huge productivity gain for engineers.

中文翻译:

很有趣。没人谈论工程生产力的这一个维度，即：公司其他部门要求构建随机的一次性工具的需求减少了。这对工程师来说感觉是一个巨大的生产力提升。

(00:50:47) Dhanji R. Prasanna

English:

It is massive, although I think that it's a little bit like the analogy of if you build a bigger highway, you'll just get more cars on the road. So I think the fact that everyone's building software means that there's more software to be built, more coordination to happen, and everyone's more eager to ship things faster and with greater results. And so we're just seeing an overall uptake in velocity and the ask for more features, if that makes sense. Yeah.

中文翻译:

确实非常巨大。不过我觉得这有点像修高速公路：路修得越宽，路上的车就越多。由于现在每个人都能构建软件，这意味着有更多的软件需要被构建，需要更多的协调，而且每个人都更渴望更快地发布产品并获得更好的结果。所以我们看到的是整体速度的提升和对更多功能的需求。

(00:51:22) Lenny Rachitsky

English:

Absolutely. And it connects to your point about you're not slowing hiring. What I'm hearing is just headcount, hiring desires for more engineers, more product people is not slowing at all. You're basically, it's as if AI wasn't really there.

中文翻译:

完全正确。这和你提到的不放缓招聘相呼应。我听到的是，员额需求、对更多工程师和产品人员的渴望完全没有放缓。基本上，就好像 AI 并不存在一样（需求依然旺盛）。

(00:51:37) Dhanji R. Prasanna

English:

We're being more thoughtful about it. So like I said, we were looking at as a commodity in the GM era, and now that we're functional, it's much less about how many engineers we need as a function of the number of features we have in Square or Cash App and in the functional org structure, we think of it much more as what are the areas of optimization? Where can we build depth and what really accelerates our priorities through things like modularization reuse and going deep into platforms.

中文翻译:

我们现在对此更加深思熟虑。正如我所说，在 GM 时代我们把人看作商品。现在在职能型架构下，我们不再简单地根据 Square 或 Cash App 的功能数量来决定需要多少工程师。我们更多地思考：哪些领域可以优化？哪里可以建立深度？什么能通过模块化复用和深耕平台来真正加速我们的优先事项？

(00:52:12) Lenny Rachitsky

English:

I love this hot take of if you're trying to be more productive, forget AI, just re-org into a functional structure.

中文翻译:

我喜欢这个“暴论”：如果你想提高生产力，忘掉 AI 吧，直接重组为职能型架构。

(00:52:21) Dhanji R. Prasanna

English:

It's not wrong in some ways. So here's another really interesting example where we are trying to improve our build times and you were using Goose and a lot of other tools to help us with this too, and they've done remarkable things. So we have this really cool tool that analyzes our test suites and selects the right test to run for changes that were made. So we cut down basically 50% of test runs this way, which is pretty great, and we're not warming the planet as much with all these unnecessary CPU cycles being wasted on tests. But then things like offloading tests to the cloud or simply just deleting tests that don't make sense anymore, probably save you two to three times that. So there is still a portfolio approach that you need to take for lack of a better term. It's like that example I told you earlier about should we buy a vendor tool? Should we build this in-house? It's like, "Well, do we even need to do this process at all?" So in some ways, structure matters more than the efficacy of the tools you have.

中文翻译:

在某种程度上这并没错。这里还有一个有趣的例子：我们尝试缩短构建时间，我们使用了 Goose 和很多其他工具，效果显著。我们有一个很酷的工具可以分析测试套件，并根据代码更改选择正确的测试来运行。通过这种方式，我们减少了大约 50% 的测试运行，这很棒，也减少了不必要的 CPU 消耗。但是，像把测试卸载到云端，或者干脆删掉那些不再有意义的测试，可能能帮你节省两到三倍的时间。所以你仍然需要采取一种“组合拳”式的方法。就像我之前说的：我们要买工具还是自研？其实应该问：“我们到底需不需要这个流程？”所以从某种意义上说，组织结构比工具的效能更重要。

(00:53:34) Lenny Rachitsky

English:

Wise words makes me think about Elon has this whole process for optimize stuff and one of the steps is like, "Do we even need this thing before we start out optimizing and automating it?" Before I zoom out and ask about just general lessons that you've learned over the course of your career, is there anything else that you think might be really valuable or useful to folks that are trying to lean in further into AI or just help their teams think a little bit more forward thinking?

中文翻译:

富有哲理。这让我想起马斯克有一套优化流程，其中一步就是：“在我们开始优化和自动化之前，我们真的需要这玩意儿吗？”在我把话题放大，询问你职业生涯中的通用教训之前，对于那些想要进一步拥抱 AI，或者想让团队更具前瞻性思维的人，你还有什么觉得有价值的建议吗？

(00:54:04) Dhanji R. Prasanna

English:

I would say really try and use these tools yourself. So the way in which I think we've been able to drive most of the adoption is Jack uses Goose, I use Goose, our executive team all have used Goose and use it regularly and use other AI programming tools and assistance as well, and we do it every single day. And so we learn a lot about how our own workflow can change, and that's going to tell you so much more about how are you going to change your organization's workflow than if you're reading a bunch of think pieces on LinkedIn or Harvard Business Review or whatever it is, and then trying to get your teams to follow suit. So I think we do this with everything. It's feel it, use the product yourself, feel it, understand its strengths and weaknesses and its ergonomics, and then figure out how to apply it to your teams.

中文翻译:

我会说，一定要亲自尝试使用这些工具。我认为我们之所以能推动如此大规模的采用，是因为 Jack 在用 Goose，我也在用，我们的高管团队都在用，而且是每天都在用，同时也用其他的 AI 编程工具和助手。通过这

种方式，我们了解了自己的工作流可以如何改变。这比你在 LinkedIn 或《哈佛商业评论》上看一堆深度文章，然后强迫团队效仿要有效得多。我认为我们对待任何事情都应该这样：去感受它，亲自使用产品，理解它的优缺点和易用性，然后再想办法应用到团队中。

(00:55:06) Lenny Rachitsky

English:

Something I've found helpful in doing that, which I completely agree with, which is stop reading about it, stop listening to us talking about it, just build some stuff. The thing that I found really helpful there is have a specific task or problem you want to solve for yourself because that really motivates you and makes it very real. For example, just the other day, I was trying to pull images out of a Google Doc. Google Doc, it's like I think of it as Hotel California. You put images in there, but there's no way to get them back out unless you do some crazy stuff. So I just went to Lovable and like Bill an app, or I can give you a Google Doc URL and let me download the images real easily and bam. Perfect.

中文翻译:

我发现有一点很有帮助，我也完全同意你的看法：别再光读文章了，别再光听我们在这儿聊了，动手做点东西。我发现最有效的方法是找一个你想为自己解决的具体任务或问题，因为这会让你动力十足，而且非常真实。比如前几天，我想从 Google Doc 里提取图片。Google Doc 就像“加州旅馆”，图片放进去容易，想拿出来除非你搞点骚操作。于是我去了 Lovable，做了一个应用，只要给它 Google Doc 的 URL，就能轻松下载图片。搞定，完美。

(00:55:41) Dhanji R. Prasanna

English:

Yeah, great example. I did something like this a couple months ago as well, where my son has a whole bunch of therapies, he has additional needs, and so I was trying to gather the receipts for all these therapies and share them with my wife and she will claim it from our insurer, and I was really struggling to do this because they're in various forms. There are screenshots in some cases or PDFs or whatever. So I asked Goose to do this and it was all sitting on my laptop and Goose figured out that it could put all of these receipts into my Apple Notes app into a single note. It converted it to HTML so it would sync seamlessly to my phone and then I could email it or share it with her from there. And that's just something I just never would've thought of. And it did this using Apple Script, so it just controlled my computer for me in the background. Yeah, so these are surprising ways in which these tools help us, and the more you use them to solve real problems to your point, the more you understand what their strengths are and where you can deploy them.

中文翻译:

是的，好例子。几个月前我也做过类似的事。我儿子需要接受很多治疗，他有一些特殊需求。我尝试收集所有这些治疗的收据并分享给我妻子，好让她向保险公司理赔。我当时很头疼，因为收据格式五花八门，有截图、PDF 等等。于是我让 Goose 来处理。Goose 发现它可以把所有这些收据放入我的 Apple Notes 应用里的一个便签中。它把内容转换成 HTML，这样就能无缝同步到我的手机上，然后我就可以直接发邮件或分享给她。这事儿我压根儿没想过能这么干。它是通过 Apple Script 实现的，直接在后台帮我控制电脑。所以，这些工具帮助我们的方式往往出人意料。正如你所说，你越是用它们解决现实问题，你就越能理解它们的优势以及该如何部署它们。

(00:56:51) Lenny Rachitsky

English:

I love this example. So did you just go to Goose and be like, "Here's the problem I have, how would you solve it?"

中文翻译:

我喜欢这个例子。所以你只是去找 Goose 说：“这是我遇到的问题，你会怎么解决？”

(00:56:56) Dhanji R. Prasanna

English:

Yeah, pretty much. I said, "I have all these receipts there in Google Drive, so we have similar origin problem there and I need to get them into a single form and I need to collate the totals and do all this." So it tried a few approaches first. It tried to download them and it tried to read them using a PDF reader and this and that. And then the thing about Goose that I think a lot of the other AI agents learn from us as well is if it tries a few things and fails, it'll back up and it'll try a different route and it'll just keep going until it makes some progress. And that's what it did. Then it picked Apple Script as a way to do it because it had the MCP extension to control my computer, and this is the same thing that our engineer, we were talking about the other day uses to watch his screen and things like that, but this was a very focused problem and it managed to do that. So yeah, it's surprising what these tools can do and allowing them the flexibility to do that is a big part of learning how to use them.

中文翻译:

是的，差不多。我说：“我所有的收据都在 Google Drive 里（看来我们遇到了类似的源头问题），我需要把它们汇总成一个表格，核对总额等等。”它先尝试了几种方法：尝试下载、尝试用 PDF 阅读器读取等等。Goose 有一点我觉得很多其他 AI 智能体都在向我们学习，那就是：如果它尝试了几次都失败了，它会退后一步，尝试另一条路线，一直尝试直到取得进展。它就是这么做的。最后它选择了 Apple Script，因为它有控制我电脑的 MCP 扩展。这和我们之前聊到的那个让 Goose 盯着屏幕看的工程师用的是同一种能力。这是一个非常具体的问题，它成功解决了。所以，这些工具能做的事情令人惊讶，给它们灵活性是学习如何使用它们的重要部分。

(00:58:02) Lenny Rachitsky

English:

That's cool. I love the, by the way, can you run Goose as a regular person? Can you just download Goose and use that instead of Claude?

中文翻译:

太酷了。顺便问下，普通人能运行 Goose 吗？能直接下载 Goose 用它来代替 Claude 吗？

(00:58:08) Dhanji R. Prasanna

English:

Yeah, absolutely. Yeah. You can just download it from our URL. We can share it in the show notes for you and yeah, you can install it. It comes for Mac and Windows and Linux I believe. It's an electron app, so it'll work on all of them. It also has a command line, so for people who are more comfortable using that, we have that UI as well.

中文翻译:

当然可以。你可以直接从我们的 URL 下载。我们可以在节目介绍里分享链接。你可以安装它，它支持 Mac、Windows 和 Linux。它是一个 Electron 应用，所以全平台通用。它也有命令行界面，适合那些更喜欢用终端的人。

(00:58:32) Lenny Rachitsky

English:

Wow, you really are competing with these massive foundational model companies building. What's the simplest way to compare Goose to something else? Is it like this Claude code, this simplest comparison or something else?

中文翻译:

哇，你们真的在和那些构建基础模型的大型公司竞争。对比 Goose 和其他东西最简单的方法是什么？是把它比作 Claude Code 吗？还是别的什么？

(00:58:43) Dhanji R. Prasanna

English:

I think it's a bit different than Claude Code because at its core, Goose is a platform that implements MCPs. So MCPs give it this dynamically extensible nature so it can do all of these things for you, whether it's automating things like we were talking about with Google Docs and notes and things like that, or it can do straight up programming tasks for you using other MCPs. It can index code and do it that way. So it's really more of an extensible platform. So I would say it sits somewhere between your classic AI assistant where you just ask it, "What's the weather today? Can you calculate how many months it's been since this date?" Or whatever it is, to the more focused cursors and Claude codes of the world.

中文翻译:

我认为它和 Claude Code 有点不同，因为 Goose 的核心是一个实现 MCP 的平台。MCP 赋予了它动态扩展的能力，所以它能为你做所有这些事，无论是像我们刚才聊的 Google Docs 和便签自动化，还是利用其他 MCP 完成纯编程任务。它可以索引代码并以此工作。所以它更像是一个可扩展平台。我会说它介于传统的 AI 助手（你问它天气、算日期差）和像 Cursor、Claude Code 这种更专注的工具之间。

(00:59:39) Lenny Rachitsky

English:

Basically, it's everything combined wholly and free. You pay for the LM tokens, but yeah.

中文翻译:

基本上，它是所有功能的集合体，而且是免费的。你只需要支付 LLM 的 Token 费用，对吧。

(00:59:46) Dhanji R. Prasanna

English:

Yeah, there's not like an open source models which-

中文翻译:

是的，如果你用开源模型甚至连 Token 费都不用……

(00:59:50) Lenny Rachitsky

English:

Oh my God, this is crazy. What a cool team to be on building Goose at Block. Must having must be having so much fun. Oh man. Okay. Let me zoom out a little bit. So you've been CTO in LinkedIn right now for just about two years. What's something that you wish you knew before you stepped in this role? If you could go back a couple years and just whisper a few tips and tricks or lessons into your ear, what would they be?

中文翻译:

天哪，这太疯狂了。在 Block 开发 Goose 的团队一定很有趣。好，让我把话题放大一点。你担任 CTO（注：Lenny 此处口误说成了 LinkedIn，实际应为 Block）大约两年了。有什么是你希望在接手这个职位之前就知道的？如果你能回到两年前，在自己耳边低声说几个技巧或教训，会是什么？

(01:00:13) Dhanji R. Prasanna

English:

I think maybe two different things. One is just the power of Conway's Law, like we talked about before. It's like how difficult it is to change outcomes without changing the structure of relationships between people in an organization. And I think I always kind of knew that at some level, but really appreciating it in a visceral way is big. The other thing that I really learned the hard way maybe is you only hear about it when things are going wrong. So when things are going well, you kind of have this eerie silence and you're like, "Well, am I doing the right things here? Am I focusing on the right problems?" So having a bit of judgment, having a bit of time to step back and look at things holistically, those are things that you really need to make time for and do on a regular basis, which I wish I had known when I took up the role.

中文翻译:

我想可能有两点。第一是康威定律的力量，就像我们之前聊的。如果不改变组织中人与人之间的关系结构，想要改变产出结果是极其困难的。我以前在某种程度上知道这一点，但现在有了切肤之痛。第二点是我通过挫折学到的：你只有在出问题的时候才会听到反馈。当一切顺利时，你会感受到一种诡异的沉默，你会想："我做的事情对吗？我关注的问题对吗？"所以，保持判断力，花时间退后一步从全局审视，这些是你必须定期腾出时间去做的事。我希望我刚上任时就知道这一点。

(01:01:15) Lenny Rachitsky

English:

Looking back at your time at Block, I keep trying to, I almost say Square because I'm so used to that over the air, but I know Block is the name of the broader company and Square is just one. Just so people understand, Square is one business unit, one product within Block.

中文翻译:

回顾你在 Block 的时光——我总是忍不住想说 Square，因为我太习惯那个名字了，但我知道 Block 是母公司的名字，Square 只是其中之一。为了让听众明白：Square 是 Block 旗下的一个业务单元、一个产品。

(01:01:28) Dhanji R. Prasanna

English:

Correct, yeah, we have Square, Afterpay, Cash App and TIDAL are four major brands, and then we also have Bitkey and Proto that are focused on Bitcoin for us and we chip hardware in those two brands.

中文翻译:

没错。我们有 Square、Afterpay、Cash App 和 TIDAL 四大品牌。此外我们还有专注于比特币的 Bitkey 和 Proto，这两个品牌还涉及硬件业务。

(01:01:44) Lenny Rachitsky

English:

Okay, great. I think that some people are like, what are you guys talking about? Okay, cool. So reflecting back on your time at Block, what's maybe the most counterintuitive lesson you've learned about building products or building teams that goes against what most people believe, say common startup wisdom?

中文翻译:

太好了，我想有些听众刚才可能还在纳闷你们在聊什么。好，回顾你在 Block 的经历，关于构建产品或团队，你学到的最反直觉的教训是什么？也就是那些违背大多数人认知或初创公司常识的东西。

(01:02:02) Dhanji R. Prasanna

English:

I think code quality is one. Being an engineer. I learned this very early on and it keeps coming true over and over and over again. A lot of engineers think that code quality is important to building a successful product. The two have nothing to do with each other, but my favorite example is YouTube. I was working at Google around the time YouTube was acquired and I just remember there was this whole wash of angst about how horrible the YouTube code base is and how terrible their architecture is, and they're storing videos as blobs in MySQL and whatnot. And you could argue that YouTube is the most successful product at Google by a long way, maybe more successful than many of their others combined. And so it really has very little to do with how well it was architected because the flip side of that Google video, which is product that I don't know if people remember, it existed before YouTube. It supported more formats, it supported higher resolution. You could upload hour long videos, YouTube had none of this. It just had the one or two minute quick video thing and it's far and away, blown away its competition. And so I think just keeping that front and center is why are we building these tools or these apps or these products? They're for people to solve a specific problem. So in our case, it's for a square merchant to make a sale, to sell coffee to you or to sell something they've made. And that's really what's important. It's not really important how well our Android platform performs unless it's serving that need. And so I think that's been a really hard one for me over my career. And I continually encounter engineers who think we need to refactor, we need to do this in a better way. And then I'm like, "No, all this code could be thrown away tomorrow. So just focus on what we're trying to build and whom we're trying to build for."

中文翻译:

我认为代码质量就是其中之一。作为一个工程师，我早年就学到了这一点，而且它一次又一次地被验证。很多工程师认为代码质量对构建成功产品至关重要，其实两者毫无关系。我最喜欢的例子是 YouTube。YouTube 被收购时我正在 Google 工作，我记得当时大家都在焦虑 YouTube 的代码库有多烂、架构有多糟糕，他们居然把视频作为 Blob 存储在 MySQL 里等等。但你可以说 YouTube 是 Google 迄今为止最成功的产品，甚至比其他很

多产品加起来都成功。这和架构好坏几乎没关系。反观 Google Video（不知道还有没有人记得，它在 YouTube 之前就存在了），它支持更多格式、更高分辨率，可以上传一小时长的视频。而当时的 YouTube 啥都没有，只能传一两分钟的短视频，但它却远远甩开了竞争对手。所以，核心问题是：我们为什么要构建这些工具或应用？是为了帮人们解决具体问题。对我们来说，是帮 Square 商家完成一笔交易，卖出一杯咖啡。这才是最重要的。除非能服务于这个需求，否则我们的 Android 平台表现多好并不重要。这对我职业生涯来说是个很难接受的教训。我不断遇到想要重构、想要用"更好方式"去做的工程师，而我会说："不，所有这些代码明天都可能被扔掉。所以，专注于我们要构建什么，以及为谁构建。"

(01:04:19) Lenny Rachitsky

English:

That is an incredible insight and lesson. This YouTube story is so fun and such a good example. You're saying they were storing the video content in a MySQL set like row and column as a blob data.

中文翻译:

这是一个非常深刻的见解。YouTube 的故事很有趣，也是个极好的例子。你是说他们当时把视频内容直接存放在 MySQL 的行和列里，作为 Blob 数据?

(01:04:34) Dhanji R. Prasanna

English:

Yeah, this is what, I didn't actually look the code so I couldn't verify it, but this was the common wisdom. And then they had an entirely Python stack that was incredibly slow compared to the state-of-the-art C++ and Java servers that we had hyper-optimized at Google back in those days.

中文翻译:

是的，虽然我没亲眼看过代码无法证实，但当时大家都是这么传的。而且他们用的是一整套 Python 技术栈，比起我们当时在 Google 深度优化的 C++ 和 Java 服务器，那速度简直慢得惊人。

(01:04:57) Lenny Rachitsky

English:

That is hilarious. It makes me think about also companies when you look inside a company, if you work at a company, you're just like, "This is just pure chaos. No one knows what's going on. This is just about to all fall apart." And that's basically what it's like at every successful hyper-growth company.

中文翻译:

太好笑了。这让我想起，当你身处一家公司内部时，你往往会觉得："这简直是一团糟，没人知道发生了什么，公司马上就要垮了。"而事实上，每一家成功的超高速增长公司内部基本上都是这样的。

(01:05:14) Dhanji R. Prasanna

English:

Yeah, there's some truth to that for sure. Yeah.

中文翻译:

没错，确实如此。

(01:05:17) Lenny Rachitsky

English:

And so I think again, it's just there's so much more that is more important to the success of a business. And it's what you said is are you solving a real problem for people? Can you get in their hands? Can you continue solving real problems for them? It's not about the quality of the code, it's not how well you operate internally.

中文翻译:

所以我觉得，对于商业成功来说，有太多东西比代码质量更重要。就像你说的：你是否在为人们解决真实问题？你能不能把产品送到他们手中？你能不能持续解决问题？这与代码质量无关，也与内部运作是否完美无关。

(01:05:32) Dhanji R. Prasanna

English:

Absolutely. I think on Cash App we had that as well. So in the early days of Cash App, I was head of engineering from when we were about 10 engineers to 200 plus and took us to about 10 plus or 20 million users thereabouts. And there was a very similar thing there. From the outside it looked like everything was really chaotic. It's like people would build random experiments and ship them and it just didn't look like we were following strict policies on things like software life cycle and stuff like that, and it was kind of true. And my philosophy was always, we have all these brilliant engineers and I'm going to do more harm than good by trying to harness them into very strict blinkered areas. If they want to spin their wheels building something that is a complete waste of time for a little bit. But at the same time, if they're delivering these amazing things on the flip side, then I'll almost allow that. I'll be okay with that. And it's a fine balance because engineers can really go off and into rabbit holes if you let them. But yeah, there's a certain amount of creativity that chaos breeds and you have to know how to build controlled chaos in some ways. So you have to create a foundation that isn't liable to rupture. You have major liability problems or something like that, or you're going to lose money in our case. And so as long as those things are bedded down and you allow your engineers to have the freedom to experiment and iterate and do the things that energizes them, that's the ideal.

中文翻译:

绝对正确。我想 Cash App 早期也是这样。我是 Cash App 的工程负责人，见证了它从 10 名工程师发展到 200 多名，用户从零增长到一两千万。当时情况非常相似：从外面看，一切都很混乱。人们构建随机的实验并发布，看起来完全没有遵守严格的软件生命周期政策。事实也确实如此。我的哲学一直是：我们有这么多才华横溢的工程师，如果我强行把他们限制在极其严格的条框里，往往适得其反。如果他们想花点时间折腾一些看似浪费时间的东西，但同时又能交付令人惊叹的成果，我基本上是允许的。这是一个微妙的平衡，因为如果你放任不管，工程师真的会钻进死胡同。但是，混乱确实能孕育创造力，你必须学会如何构建某种“受控的混乱”。你必须建立一个稳固的基础，确保不会出现重大责任事故或资金损失。只要这些底线守住了，给工程师实验、迭代和做让他们兴奋的事情的自由，就是最理想的状态。

(01:07:26) Lenny Rachitsky

English:

Speaking of controlled chaos, one of your titles during your time at Block, I guess this was while you were actually at Square, was Mad Scientist for four and a half years.

中文翻译:

说到“受控的混乱”，你在 Block（当时还是 Square）期间的一个头衔居然是“疯狂科学家”（Mad Scientist），而且干了四年半。

(01:07:38) Dhanji R. Prasanna

English:

Yeah, that was a time when I was working part-time, mostly because I had very young kids with lots of additional needs and I was a consultant on various different projects and I was trying to help some wacky things get off the ground. And yeah, I'm really grateful to Block that they afforded me the freedom to have that role in my career as well.

中文翻译:

是的，那是我兼职工作的一段时间。主要是因为当时孩子还小，有很多特殊需求。我当时担任各种不同项目的顾问，尝试帮助一些古怪的想法落地。我很感激 Block 在我的职业生涯中给了我担任那个角色的自由。

(01:08:08) Lenny Rachitsky

English:

Maybe one more question before I take us to Fail Corner, which I'll explain. So you've shared a few lessons of things you've learned over the course of your career. Are there any other, just let's say core leadership lessons that you've learned that you think have been important to you being successful at the work that you've done?

中文翻译:

在进入“失败角落”（Fail Corner）环节之前，我再问一个问题。你已经分享了一些职业生涯中的教训。还有没有其他的核心领导力经验，是你认为对你取得成功至关重要的？

(01:08:26) Dhanji R. Prasanna

English:

I think start small with everything. If you try to boil the ocean to make a cup of tea, I don't know who said that, but it's a really a useful phrase that I keep coming back to. You'll never get there. So if you're making a cup of tea, just make the cup of tea. You don't need to boil all the water that there is.

中文翻译:

我认为是“凡事从小处着手”。如果你想通过煮干整个海洋来泡一杯茶——我不知道是谁说的，但这句名言我经常想起。你永远无法实现目标。所以如果你想泡茶，就只煮那一壶水。你不需要煮干所有的海水。

(01:08:47) Lenny Rachitsky

English:

That sounds like really not delicious tea. Ocean water.

中文翻译:

那茶听起来可不好喝。海水泡的茶。

(01:08:52) Dhanji R. Prasanna

English:

Yeah, I think there's another one of, I think Carl Sagan said, "If you want to make an apple pie from scratch, you have to first invent the universe." So it's like narrow your scope to the thing that's in front of you and that's achievable. And so that I think is really important and that's one of our core tenets and always has been even when we were just Square in the early days, start small.

中文翻译:

是的。卡尔・萨根也说过："如果你想从零开始做一个苹果派，你必须先创造宇宙。"所以，要把你的范围缩小到眼前且可实现的事情上。我认为这非常重要，这也是我们的核心原则之一，从 Square 早期开始就是如此：从小处着手。

(01:09:19) Lenny Rachitsky

English:

Is there an example that maybe worked really well or maybe didn't work?

中文翻译:

有没有什么例子是这种做法非常成功，或者反之失败了的?

(01:09:23) Dhanji R. Prasanna

English:

Yeah, Goose started small. It was just an engineer working on their own time trying to build something that was useful and that satisfied a thesis that they had. So Brad, our creator of Goose, believed very early on, I think long before we heard the buzzword going around that agents would be how we unlock value from LLMs. And he built a proof concept and he shared it with a bunch of people. He shared it with Databricks and Anthropic, got them excited and learned a lot from them. And so it just sort of built momentum from there. And even internally, it was quite a similar thing. Cash App itself was like that and Cash App started more or less as a hack week sort of idea and grew into a bigger and bigger and bigger thing. So a lot of our projects start with these small experiments that we try to then build on top of. We became the very first company that was a public company to launch a Bitcoin product. And that was again a hack week idea that actually Jack and me and another engineer worked on.

中文翻译:

有的，Goose 就是从小处开始的。最初只是一个工程师利用业余时间尝试构建一些有用的东西，来验证他的一个设想。Brad（Goose 的创造者）很早就相信——远在"智能体"这个词流行之前——智能体将是我们释放 LLM 价值的方式。他做了一个原型，分享给了一些人，包括 Databricks 和 Anthropic，让他们感到兴奋并从中学习。势头就这样建立起来了。内部项目也是如此，Cash App 本身最初也只是一个黑客周的想法，然后越滚越大。我们的很多项目都是从这些小实验开始，然后在其基础上构建。我们是第一家推出比特币产品的上市公司，而那也是一个黑客周的想法，当时是 Jack、我和另一名工程师一起做的。

(01:10:40) Lenny Rachitsky

English:

That was the hackathon team? You and Jack Dorsey and an engineer?

中文翻译:

那是当时的黑客松团队？你、Jack Dorsey 还有一名工程师？

(01:10:44) Dhanji R. Prasanna

English:

Yeah, it was the three of us.

中文翻译:

是的，就我们三个。

(01:10:46) Lenny Rachitsky

English:

Unreal.

中文翻译:

不可思议。

(01:10:48) Dhanji R. Prasanna

English:

Yeah, and it was great. We went and bought a cup of coffee, a blue bottle, and it was bought using Bitcoin over cash card. And I'll tell you those in hindsight, probably the most expensive cup of coffee.

中文翻译:

是的，那很棒。我们去 Blue Bottle 买了一杯咖啡，是用 Cash Card 通过比特币支付的。回想起来，那可能是史上最贵的一杯咖啡。

(01:11:02) Lenny Rachitsky

English:

What was Bitcoin at? 20,000?

中文翻译:

当时比特币多少钱？两万？

(01:11:02) Dhanji R. Prasanna

English:

I think it was 6,000 or 7,000 back then. I don't know.

中文翻译:

我想当时大概是六七千美元吧。

(01:11:07) Lenny Rachitsky

English:

It's like 120,000 now. Great.

中文翻译:

现在都快 12 万了。厉害。

(01:11:12) Dhanji R. Prasanna

English:

But yeah, it's an example of how you get to a working useful product to people if you focus on a small thing first in a build.

中文翻译:

但这就是一个例子：如果你在构建时先专注于一件小事，你就能做出对人们有用的产品。

(01:11:22) Lenny Rachitsky

English:

And just to double down on this counter too. "Okay, we have a big idea, we're just going to put a bunch of resources on it and go big immediately."

中文翻译:

这又是对那种“我们有个大主意，我们要投入大量资源，立刻做大”想法的反击。

(01:11:29) Dhanji R. Prasanna

English:

Yeah, absolutely. And I've been part of teams like that too. So in my career, I worked at Google on this product called Google Wave, which was trying to be everything to everyone and we were 70, 80 engineers building this thing before it even really had any users outside Google. And so I think that's an example of something that started big, tried to go big on day one and probably lacked some of that meeting the earth where reality lies and adapting accordingly.

中文翻译:

是的，绝对是。我也待过那样的团队。在我的职业生涯中，我在 Google 参与过 Google Wave 项目。它试图成为满足所有人一切需求的东西，在它甚至还没有 Google 以外的用户之前，我们就投入了七八十名工程师。这就是一个“起步就想做大”的例子，它缺乏与现实接轨的过程，也缺乏相应的适应性。

(01:12:08) Lenny Rachitsky

English:

I remember Google Wave. Absolutely. It was beautiful. A lot of hype. I don't remember what it was for specifically, but it looked really nice.

中文翻译:

我记得 Google Wave。确实，它很漂亮，炒作得很厉害。我不记得它具体是干嘛的了，但看起来确实很高级。

(01:12:15) Dhanji R. Prasanna

English:

Yeah, a lot of learnings from that one for me. Yeah.

中文翻译:

是的，我从那个项目中学到了很多。

(01:12:19) Lenny Rachitsky

English:

What else? Any other big lessons?

中文翻译:

还有吗？还有什么重大的教训?

(01:12:21) Dhanji R. Prasanna

English:

Those two are the big ones, but I would also say question base assumptions on everything. Sometimes we get into traps where we are as professionals, hyper focused on what we're building that day, that week, that month. And we don't stop to think should we even build this at all? Or what's the purpose of building this? Could we build something completely different that would matter more to our core reason for being? So I would say, yeah, question the sort of base assumptions. It's somewhat of a cliche, but you really need to remind yourself to apply it over and over and over again.

中文翻译:

那两点是核心，但我还会说：质疑一切基本假设。有时作为专业人士，我们会陷入陷阱，过度专注于当天、当周或当月正在构建的东西，而没有停下来思考：我们到底应不应该构建这个？构建它的目的是什么？我们能不能构建一些完全不同的、对我们的核心宗旨更有意义的东西？所以，质疑基本假设虽然听起来像陈词滥调，但你真的需要不断提醒自己去实践它。

(01:13:03) Lenny Rachitsky

English:

I had a colleague of yours on the podcast back in the day, IO, who worked with you on Cash App.

中文翻译:

我以前邀请过你的一位同事上播客，IO（Ayo），他曾和你一起开发 Cash App。

(01:13:08) Dhanji R. Prasanna

English:

Yeah.

中文翻译:

是的。

(01:13:09) Lenny Rachitsky

English:

He's a friend of mine, he's amazing. He had a quote along those lines of just like, I forget exactly what it was, but it was just get to the bare metal of the thing that you're working on, just touch the thing that you're building and go to the base of it to really understand what's going on. And I imagine that was really important with Building Cash App and Cash Card.

中文翻译:

他是我的朋友，非常出色。他有一句类似的话，我记不清原话了，大概是：深入到你所做事情的“裸机”（bare metal）层面，亲手触摸你正在构建的东西，深入底层去真正理解发生了什么。我想这在构建 Cash App 和 Cash Card 时非常重要。

(01:13:26) Dhanji R. Prasanna

English:

Yeah, IO is one of the best product people I've ever worked with and one of my closest friends actually. So absolutely with him, and you on that one, yeah.

中文翻译:

是的，IO 是我合作过的最优秀的产品人之一，也是我最亲密的朋友。我完全同意他（和你）的观点。

(01:13:38) Lenny Rachitsky

English:

Okay. I'm going to take us to a recurring segment on the podcast I call Fail Corner. You already shared one example of a product that failed that you worked on. I'm curious if there's another, and the question is just what's the product you worked on that did not work out? Because people listening to this hear all these amazing successful people come on the podcast, share all these stories of success, endless success, but they don't hear the stories when things don't work out. And so this question is just, "What's a product you worked on that didn't work out and what did that teach you?"

中文翻译:

好。现在进入播客的一个固定环节，我称之为“失败角落”（Fail Corner）。你已经分享了一个失败产品的例子。我很好奇还有没有其他的？问题是：你参与过的哪个产品没能成功？因为听众听到的往往是成功人士分享的无尽成功故事，却听不到失败的故事。所以这个问题是：“哪个你参与的产品失败了，它教会了你什么？”

(01:14:08) Dhanji R. Prasanna

English:

It's a very valuable point. My career has basically been a string of failed product on top of failed product. And I think that, "Yeah, the Google wave example's there." I worked for Hot Minute on Google+, which was another epic failure.

中文翻译:

这是一个非常有价值的点。我的职业生涯基本上就是一连串失败产品堆叠起来的。Google Wave 是一个，我还参与过一小段时间的 Google+，那也是一个史诗级的失败。

(01:14:23) Lenny Rachitsky

English:

Good one.

中文翻译:

经典案例。

(01:14:23) Dhanji R. Prasanna

English:

I worked at this social networking startup called Secret, which burned hot for a bright minute and then blew up. And then there was an email startup that we did, and that was, again, very promising, and then that fizzled. So the co-founder of Canva and I worked on that one. So there's been a whole string of failures, but at each point, I think I learned something and I learned that I need to never make that class of failures or errors again. And so Cash App was probably the big success for me that a product that I worked on that was very early on and grew to be this giant business and product that people love. So yeah, been my career is essentially taking the learnings from all these failures, getting some humility out of it in the process too, coming into things, willing to listen to other people's points of view, critical points of view, and not just thinking that I have all the answers, yeah.

中文翻译:

我还曾在一家叫 Secret 的社交网络初创公司工作过，它火爆了一阵子然后就爆炸了。后来我们还做过一个邮件初创公司，当时看起来很有前途，结果也无疾而终（那是和 Canva 的联合创始人一起做的）。所以，我经历了一连串的失败，但每一步我都学到了一些东西，学到了永远不要再犯那一类错误。Cash App 对我来说是巨大的成功，我参与了它的早期阶段，见证它成长为人们喜爱的庞大业务。所以，我的职业生涯本质上就是从失败中汲取教训，在这个过程中也变得更加谦逊，愿意倾听他人的观点和批评，而不是总觉得自己掌握了所有答案。

(01:15:36) Lenny Rachitsky

English:

And I bet all these products that failed had really beautiful code. A lot of really good architecture decisions were made. Some of them, some of them were awful in every way. So many reasons for it to fail. Incredible. Dhanji, is there anything else that you wanted to share or I don't know, double down on before we get to our very exciting lightning round?

中文翻译:

我敢打赌，那些失败的产品代码一定写得很漂亮，架构决策也做得很好。当然，有些可能在各方面都很糟糕。失败的原因有很多。太不可思议了。Dhanji，在进入激动人心的闪电轮提问之前，你还有什么想分享或强调的吗？

(01:15:59) Dhanji R. Prasanna

English:

I would say I think that we're in this era of a lot of change and people are scared or reticent or uncertain about where things are going. And I think that look at the things that matter to you. For us, it's open source, open protocols, improving access for everyone. I've been very lucky in my career to only work on products that are either free or almost free to anyone or they have a free tier and then you pay for some premium services and that are usable by everyone. So anyone can become a Square seller. I remember even in the early days, people used it to pay each other as a peer-to-peer money transfer system and that's why we built Cash App and that was really successful on the back of that. So I think it's really look at the things that are important to you and optimize for them. It's not really that important that the technology trends are growing in a certain way because technology is here to serve us, and if we have an important reason for being and an important purpose, then we can make that technology serve us. And that's much more important than being deep with the technology or being at the forefront of every trend.

中文翻译:

我想说，我们正处于一个剧烈变革的时代，人们对未来的走向感到恐惧、沉默或不确定。我认为你应该关注那些对你真正重要的事情。对我们来说，是开源、开放协议、提高每个人的准入门槛。在我的职业生涯中，我很幸运能参与那些对所有人免费或几乎免费的产品，或者有免费层级的产品。任何人都可以成为 Square 卖家。我记得早期人们甚至用它来进行 P2P 转账，这就是我们构建 Cash App 的原因，并因此取得了成功。所以，看清对你重要的事情并为此优化。技术趋势如何发展其实没那么重要，因为技术是为我们服务的。如果我们有明确的存在意义和目标，我们就能让技术为我们所用。这比钻研技术本身或站在每个潮流的最前沿要重要得多。

(01:17:27) Lenny Rachitsky

English:

Such great advice when there's so much to pay attention to and so much happening. So stressful to feel like I'm just not aware of all the things. I'm not as good as all these people I'm seeing on social media, but what's happening with AI, I'm just so behind. What I'm hearing from you is just like what is actually important to you? And just do that. Don't feel like you need to be the best at everything that's happening on top of all the latest AI news.

中文翻译:

在信息爆炸、变幻莫测的当下，这真是极好的建议。感到自己“跟不上节奏”确实很有压力，看着社交媒体上的人总觉得自己不够优秀，觉得自己在 AI 浪潮中落后了。我从你这里听到的是：找到对你真正重要的东西，然后去做。不要觉得你必须精通每一条最新的 AI 新闻或掌握每一项新技术。

(01:17:51) Dhanji R. Prasanna

English:

Yeah, exactly. And if it's not meaningful and fun, then you shouldn't be doing it probably.

中文翻译:

没错。如果一件事既没有意义也不好玩，那你可能根本就不该去做它。

(01:17:58) Lenny Rachitsky

English:

With that Dhanji, we've reached our very exciting lightning round. I've got five questions for you. Are you ready?

中文翻译:

好，Dhanji，现在进入激动人心的闪电轮。我有五个问题，准备好了吗?

(01:18:03) Dhanji R. Prasanna

English:

Okay. Shoot.

中文翻译:

好，开始吧。

(01:18:04) Lenny Rachitsky

English:

I see so many books behind you. So I love this first question. I'm excited to see what you pick. What are two or three books that you find yourself recommending most to other people?

中文翻译:

我看到你身后有很多书。我很喜欢第一个问题，很期待你的选择。有哪两三本书是你最常推荐给别人的?

(01:18:12) Dhanji R. Prasanna

English:

Yeah, I very much of the opinion that you shouldn't read books that are about your daily work or your professional life. I read fiction, I read the classics, I read poetry, philosophy, history. These are the books I really enjoy. And I think it expands your mind and gives you creative ideas and helps you question things about the human condition. And that's much more valuable than some self-help book or some get good at being an engineering manager book. So yeah, having said that, the Master in Margarita by Mikhail Bulgakov is one that I really love. It's a masterpiece of Russian literature. And then I've always been drawn to Tennyson's poetry and I find that in the times when I'm most uncertain or grieving, Tennyson's poetry has always resonated with me and helped me find a center.

中文翻译:

是的，我非常坚持一个观点：你不应该读那些关于你日常工作或职业生活的书。我读小说、经典文学、诗歌、哲学和历史。这些是我真正享受的书。我认为它们能开阔你的思维，给你带来创意，并帮你思考关于人类生存状态的问题。这比什么成功学书籍或“如何成为优秀的工程经理”之类的书要有价值得多。话虽如此，我非常喜欢米哈伊尔・布尔加科夫的《大师与玛格丽特》，那是俄罗斯文学的杰作。另外，我一直被丁尼生

（Tennyson）的诗歌所吸引，我发现在我最不确定或悲伤的时候，他的诗总能引起我的共鸣，帮我找到内心的平静。

(01:19:30) Lenny Rachitsky

English:

Wow. Never heard these recommendations before. I'm really excited to check these out. Very cool for a CTO of a big tech company. What is a favorite recent movie or TV show you've really enjoyed?

中文翻译:

哇，以前从未听过这些推荐。我很期待去读读看。对于一家大科技公司的 CTO 来说，这品味很酷。最近有什么你非常喜欢的电影或电视剧吗?

(01:19:30) Dhanji R. Prasanna

English:

Alien Earth I think is pretty awesome. It's by Noah Hawley who did the Fargo TV series. And so it's someone with all of these incredible skills in high art filmmaking who's doing a pulp sci-fi show, and it just looks stunning and it feels stunning and it captures all of that essential alien pulpiness that makes it so interesting and fun. So I really like that, and I'm also watching Slow Horses, which I think is one of the better shows on TV.

中文翻译:

我觉得《异形：地球》（Alien: Earth）非常棒。它是《冰血暴》剧集导演 Noah Hawley 的作品。一个拥有高超艺术电影技巧的人去做一部通俗科幻剧，画面和感觉都非常震撼，它捕捉到了《异形》那种本质的通俗感，非常有趣。我很喜欢。另外我还在看《慢马》（Slow Horses），我觉得那是目前电视上最好的剧集之一。

(01:20:04) Lenny Rachitsky

English:

Love Slow Horses. The new season's Out, think the fifth episode just dropped the day we're recording this. So I love that show. Alien Earth also just watched it, so creepy and just like all these slimy, gooey little creatures just crawling around.

中文翻译:

我也爱《慢马》。新一季出来了，录音当天好像刚更新了第五集。那部剧太棒了。《异形：地球》我也刚看，非常诡异，到处都是黏糊糊的小生物在爬。

(01:20:16) Dhanji R. Prasanna

English:

Yeah, I just love the aesthetic and they captured something essential about the original Alien and yeah, they do it, but every scene in Alien Earth feels like you're watching a painting or something or someone's reading a novel to you. It's really unfolds very thoughtfully.

中文翻译:

是的，我喜欢那种美学，他们捕捉到了初代《异形》的一些本质。剧中的每一帧都像是一幅画，或者像有人在为你读小说，展开得非常细腻。

(01:20:36) Lenny Rachitsky

English:

I've never watched any alien content in my life and I really enjoyed Alien Earth. I will say the ending, I was just like, it felt like it kind of slowed down a bit. I'm just like, "All right, I guess I see where it's going now." But it was really fun to watch. Okay, next question. Do you have a favorite product you really enjoyed? Sorry, a favorite product you've recently discovered that you really enjoy? It could be an app, could be an gadget, could be some kitchen thing.

中文翻译:

我这辈子从没看过任何《异形》系列的内容，但我非常喜欢《异形：地球》。不过我觉得结尾节奏稍微慢了一点，但整体观感很棒。好，下一个问题：你最近发现并非常喜欢的某个产品是什么？可以是 App、小工具或者厨具。

(01:20:58) Dhanji R. Prasanna

English:

Well, I'm a gamer. I love playing games. So for me it's the Steam Deck, the Steam Deck OLED, which is their latest version. It's like this gorgeous piece of hardware that lets you play the best games out there, but it's totally extensible and customizable. And in this era where we're constantly told by big tech companies that we need to lock everything down, we need to lock down the user experience and customizability in order to have things work for people. I think Valve showed that's totally unnecessary and totally wrong, and you can build the Steam Deck. You can install competing app stores, you can install Windows on it. You can treat it like a computer, write programs, which I have done to run on it. So yeah, I think it's an incredible thing and it looks beautiful and it works great. So yeah, big fan.

中文翻译:

我是个游戏玩家，我热爱游戏。对我来说是 Steam Deck，尤其是最新的 OLED 版本。它是一件华丽的硬件，能让你玩到市面上最好的游戏，而且它是完全可扩展、可定制的。在这个大科技公司不断告诉我们必须锁定一切、锁定用户体验和定制化才能让产品好用的时代，Valve 证明了那是完全没必要且错误的。你可以给 Steam Deck 安装竞争对手的应用商店，可以装 Windows，可以把它当电脑用，甚至可以在上面写程序（我也确实这么做了）。它看起来很美，用起来很棒。我是它的死忠粉。

(01:21:50) Lenny Rachitsky

English:

Do you have a favorite life motto that you find yourself coming back to often in work or in life?

中文翻译:

在工作或生活中，你有没有一句经常想起的人生格言？

(01:21:55) Dhanji R. Prasanna

English:

If you're not waking up in the morning feeling energized about what you're going to do that day in your professional life, then change something, quit if that's what it comes down to, or find a new way of doing what you're doing. Just don't accept what's meted out to you. So that's how I've tried to do things, and sometimes it works, sometimes not, but yeah, it's a good thing to ask yourself.

中文翻译:

如果你早上醒来，没有对自己当天要做的职业工作感到充满活力，那就改变点什么。如果到了那一步，就辞职，或者找一种新的方式去做你正在做的事。不要只是被动接受分配给你的一切。这就是我行事的方式，有时奏效，有时不奏效，但这是一个值得问自己的好问题。

(01:22:25) Lenny Rachitsky

English:

I really love this advice. It's really hard to do that for a lot of people. Is there anything that has helped you get over that fear of just like, "Oh man, I'm going to quit this thing. I don't know where I'm going to go next."

中文翻译:

我很喜欢这个建议。但对很多人来说这很难做到。有没有什么东西能帮你克服那种“天哪，我要辞职了，我不知道下一步该去哪儿”的恐惧?

(01:22:36) Dhanji R. Prasanna

English:

The main thing is telling yourself that a year from now, you're going to look back on what looks like a monumental problem, a life-changing thing, and you're going to be like, "Oh, that was so trivial." A lot of times we get into these traps where we're overthinking something or really nervous about making a change, but in hindsight, those don't seem that big. And all the time that's passed since and all the events that have happened teach you that there's more to the world and it's never too late to do something useful or never too late to do something that's for yourself and improving yourself. So yeah, I think just kind of remembering that things are not as big or bleak or decisive as they seem in the moment is always important.

中文翻译:

最重要的一点是告诉自己：一年后，当你回过头来看现在这个看似天大的、改变人生的问题时，你会觉得“噢，那其实微不足道”。很多时候我们陷入过度思考或对改变感到极度紧张的陷阱，但事后看来，那些都没那么严重。随着时间的流逝和经历的增多，你会发现世界很大，做有用的事永远不晚，为自己而活、提升自己也永远不晚。所以，记住事情往往没有当下看起来那么严重、那么暗淡或那么具有决定性，这很重要。

(01:23:30) Lenny Rachitsky

English:

Final question. So you were a mad scientist at Square for many years. Do you have another favorite mad scientist from pop culture or real life?

中文翻译:

最后一个问题。你在 Square 做了多年的“疯狂科学家”。在流行文化或现实生活中，你还有其他喜欢的疯狂科学家吗?

(01:23:41) Dhanji R. Prasanna

English:

That's an interesting one. I think the image that always comes to my mind is Doc Brown from Back to the Future. I feel like he's the canonical mad scientist of my generation anyway, but there've been a lot in video games and stuff too, but he was the one that was like, "I'm just going to do this crazy thing because I almost have this burning desire, need to do it, and whether I want to or not, I must build this time machine." And he spends the entire movie trying to fix the problems that it creates. But yeah, he has always been a really fun character for me.

中文翻译:

很有趣的问题。我脑海中浮现的形象总是《回到未来》里的布朗博士（Doc Brown）。我觉得他是我们那一代人心中最典型的疯狂科学家。虽然游戏里也有很多，但他那种“我就是要干这件疯狂的事，因为我内心有一种燃烧的渴望，无论我想不想，我都必须造出这台时光机”的劲头很吸引我。然后他花了整部电影的时间去修补时光机带来的麻烦。对我来说，他一直是个非常有趣的角色。

(01:24:20) Lenny Rachitsky

English:

You know what? I think about Pinky from Pinky in the Brain.

中文翻译:

你知道吗？我想到了《粉红豹与大头鼠》（Pinky and the Brain）里的 Pinky。

(01:24:24) Dhanji R. Prasanna

English:

Oh yeah, that's a good one too. Yeah.

中文翻译:

噢是的，那个也很经典。

(01:24:27) Lenny Rachitsky

English:

Oh man. Dhanji, this was awesome. You were wonderful. Thank you so much for being here. Two final questions before we actually wrap up. Where can folks find you online if they want to reach out, learn more about say Goose or anything else going on at Block? And how can listeners be useful to you?

中文翻译:

太棒了。Dhanji，这次对话非常精彩。非常感谢你能来。在结束前最后两个问题：如果大家想联系你，或者想了解更多关于 Goose 或 Block 的动态，可以在哪里找到你？以及，听众可以为你做些什么？

(01:24:43) Dhanji R. Prasanna

English:

Check out our GitHub pages for Goose and all of the other open source projects we have at Block. So there's a lot that's useful there. We do a lot on Android open source as well, so check that stuff out. You can always find me on LinkedIn, so feel free to connect. I'm very happy to be contacted. And I would say the way people can be useful is, again, going back to this era we're in of a lot of change and uncertainty, I think people that demand more of their companies, of their employers, of their teams, demand something better. At Block, we always ask, "Can we default to making this open source? Can we build this for people that are not just us or our customers? Can everyone benefit?" And I think that's particularly important in this era of AI where everyone's locking themselves in walled gardens and trying to capture parts of the platform that are emerging. So yeah, just demand more of people. The internet was created as a promise for open sharing of information to the benefit of all, and I think that AI should realize that for us. And so yeah, just demand that of people.

中文翻译:

请查看我们在 GitHub 上关于 Goose 以及 Block 其他开源项目的页面，那里有很多有用的东西。我们在 Android 开源方面也做了很多工作。你可以在 LinkedIn 上找到我，欢迎建立联系，我很乐意交流。至于大家能帮我做什么，我想再次回到我们所处的这个充满变革和不确定的时代：我希望人们能对自己的公司、雇主和团队提出更高的要求，要求更好的东西。在 Block，我们总是问："我们能不能默认将其开源？我们能不能为除了我们和客户之外的人构建这个？每个人都能受益吗？"我认为在 AI 时代这尤为重要，因为现在每个人都在修筑围墙花园，试图占领新兴平台的份额。所以，请对他人提出更高的要求。互联网诞生的初衷是承诺信息的开放共享以造福全人类，我认为 AI 应该为我们实现这一目标。所以，请以此要求身边的人。

(01:26:07) Lenny Rachitsky

English:

A really beautiful way to end it. Dhanji, thank you so much for being here.

中文翻译:

非常完美的结尾。Dhanji，非常感谢你能来。

(01:26:11) Lenny Rachitsky

English:

I appreciate you. Bye everyone. Thank you so much for listening. If you found this valuable, you can subscribe to the show on Apple Podcasts, Spotify, or your favorite podcast app. Also, please consider giving us a rating or leaving a review as that really helps other listeners find the podcast. You can find all past episodes or learn more about the show at lennyspodcast.com. See you in the next episode.

中文翻译:

我很感激你。大家再见，非常感谢收听。如果你觉得本期节目有价值，可以在 Apple Podcasts、Spotify 或你喜欢的播客应用中订阅。此外，请考虑给我们评分或留下评论，这能帮助更多听众发现这个播客。你可以在 lennyspodcast.com 找到往期所有节目或了解更多信息。下期见。