SCOTT WU

LENNY'S PODCAST

BILINGUAL TRANSCRIPT

ORIGINAL BY

Lenny Rachitsky

@lennysan • x.com/lennysan

ANALYSIS BY

@Penny777 • x.com/penny777

Generated: 2026-01-16

Scott Wu - 双语对照

Lenny's Podcast: Scott Wu (Cognition CEO) - Full Bilingual Transcript

[00:00:00] Scott Wu

English:

Our whole team is only like 15 engineers a year. We use a ton of Devin when we're building Devin. Most folks on the team are definitely working with up to five Devins at once, and so Devin merges like several hundred pull requests into production in the Devin code bases every month.

中文翻译:

我们整个团队一年只有大约 15 名工程师。在构建 Devin 的过程中，我们大量使用了 Devin。团队中的大多数人肯定同时在与多达五个 Devin 一起工作，因此 Devin 每月会向 Devin 的生产代码库合并数百个 PR（Pull Request，合并请求）。

[00:00:12] Lenny Rachitsky

English:

What percentage of your PRs are Devin versus humans right now?

中文翻译:

目前你们的 PR 中，Devin 提交的比例和人类提交的比例分别是多少?

[00:00:16] Scott Wu

English:

It's in the neighborhood of a quarter or so.

中文翻译:

大约在四分之一左右。

[00:00:19] Lenny Rachitsky

English:

Where do you think this will be at the end of the year?

中文翻译:

你认为到今年年底这个比例会达到多少?

[00:00:21] Scott Wu

English:

Honestly, we expect it to be a decent bit more than half.

中文翻译:

老实说，我们预计会明显超过一半。

[00:00:24] Lenny Rachitsky

English:

You guys are so ahead of how companies work with AI engineers.

中文翻译:

在公司如何与 AI 工程师协作方面，你们走得太靠前了。

[00:00:28] Scott Wu

English:

AI is going to be the biggest technology shift of our lives, so most of the big tech revolutions that we've had over the last 50 years, like personal computer, and the internet, and the mobile phone, they all had this big hardware component that was a big part of the distribution. Folks who were building for those industries kind of saw their market grow and grow and grow basically steadily year over year as the number of people with mobile phones increased, right, as the number of people connected to the internet increase. One of the things which is already I'd say different in AI, is just how explosive the technology can be. There's no weight on hardware distribution. It means that the space is just growing so exponentially.

中文翻译:

AI 将是我们一生中最大的技术变革。过去 50 年里我们经历的大多数重大技术革命，比如个人电脑、互联网和移动电话，它们都有一个巨大的硬件组件，这是分发的重要组成部分。为这些行业开发产品的人们看到他们的市场随着手机用户或互联网连接人数的增加而逐年稳步增长。而我认为 AI 已经表现出的不同之处在于，这项技术的爆发力有多强。它没有硬件分发的负担，这意味着这个领域正在呈指数级增长。

[00:01:02] Lenny Rachitsky

English:

How is the act of being an engineer and building changing?

中文翻译:

作为一名工程师，其工作行为和构建方式正在发生怎样的变化?

[00:01:05] Scott Wu

English:

I think there's going to be way more programmers and way more engineers a few years from now. Pretty quickly. The form factor of what it means to be a programmer obviously is going to change, but at the end of the day, of course the discipline is all about just being able to tell your computer what's do. And so in that lens, I really think that programming is only going to become more and more important as AI gets more powerful.

中文翻译:

我认为几年后会有更多的程序员和工程师，而且这种变化会很快发生。程序员这一角色的形式显然会发生变化，但归根结底，这门学科的核心就是能够告诉电脑该做什么。从这个角度来看，我真的认为随着 AI 变得越来越强大，编程只会变得越来越重要。

[00:01:20] Lenny Rachitsky

English:

Today my guest is Scott Wu. Scott is the co-founder and CEO of Cognition, which makes a product called Devin, the world's first autonomous AI software engineer. Unlike other AI tools that I've highlighted on this podcast, Devin is designed to act like an actual remote engineer that you chat with like you would with any other human engineer through Slack or through its dedicated website. When Devin launched about a year ago, it was very much a junior engineer. Over the past year, they've made a lot of progress and Devin is now being used by tons of companies in production. We chatted about how their engineering team of 15 uses Devins to build Devin, including how every engineer uses about five Devins each to help them code and move faster. How a quarter of their pull requests today are committed by Devins and that they expect this to be over 50% by the end of the year.

中文翻译:

今天的嘉宾是 Scott Wu。Scott 是 Cognition 的联合创始人兼 CEO，他们开发了一款名为 Devin 的产品，这是世界上第一个自主 AI 软件工程师。与我在本播客中介绍过的其他 AI 工具不同，Devin 的设计初衷是像真正的远程工程师一样工作，你可以像对待其他人类工程师一样，通过 Slack 或其专用网站与它聊天。大约一年前 Devin 发布时，它更像是一个初级工程师。在过去的一年里，他们取得了很大进展，Devin 现在已被大量公司用于生产环境。我们聊到了他们 15 人的工程团队如何使用 Devin 来构建 Devin，包括每位工程师如何使用大约五个 Devin 来帮助他们编写代码并加快进度。目前他们四分之一的合并请求是由 Devin 提交的，他们预计到年底这一比例将超过 50%。

[00:02:05] Lenny Rachitsky

English:

We also talk about how Scott imagines software engineering is going to look in the future and how the role of an engineer changes from a coder to an architect. We also get into the eight pivots that they went through before landing on this path, why Scott believes AI tools like this will lead to more engineer hiring versus less. Also where the name Devin comes from and so much more. This episode is going to blow your mind. I highly recommend you listen to it if you're at all interested about where engineering, product building, and AI is going. A huge thank you to Claire Voue for suggesting a bunch of great questions for this conversation.

中文翻译:

我们还讨论了 Scott 构想的未来软件工程的样子，以及工程师的角色如何从代码编写者转变为架构师。我们还深入探讨了他们在走上这条道路之前经历的八次转型（pivots），以及为什么 Scott 相信像这样的 AI 工具会导致工程师招聘增加而不是减少。此外，我们还聊到了 Devin 这个名字的由来等等。这一集会让你大开眼界。如果你对工程、产品构建和 AI 的未来走向感兴趣，我强烈建议你听一听。非常感谢 Claire Voue 为这次对话提供了许多精彩的问题建议。

[00:02:38] Lenny Rachitsky

English:

If you enjoy this podcast, don't forget to subscribe and follow it in your favorite podcasting app or YouTube. Also, if you become an annual subscriber of my newsletter, you get a year free of Linear, Superhuman, Notion, Perplexity, and Granola. Check it out at lennysnewsletter.com and click bundle. With that, I bring you Scott Wu. This episode is brought to you by Enterpret. Enterpret unifies all your customer interactions from Gong calls, to Zendesk tickets, to Twitter threads, to app store reviews and makes it available for analysis. It's trusted by leading product orgs like Canva, Notion, Loom, Linear, monday.com, and Strava to bring the voice of the customer into the product development process, helping you build best in class products faster.

中文翻译:

如果你喜欢这个播客，别忘了在你不常用的播客应用或 YouTube 上订阅并关注。此外，如果你成为我时事通讯（newsletter）的年度订阅者，你可以免费获得一年的 Linear、Superhuman、Notion、Perplexity 和 Granola。请访问 lennysnewsletter.com 并点击 “bundle” 查看。现在，让我们欢迎 Scott Wu。本集由 Enterpret 赞助。Enterpret 统一了你所有的客户互动——从 Gong 通话到 Zendesk 工单，从 Twitter 线程到应用商店评论，并使其可供分析。它深受 Canva、Notion、Loom、Linear、monday.com 和 Strava 等领先产品组织的信任，将客户的声音带入产品开发过程，帮助你更快地构建一流产品。

[00:03:19] Lenny Rachitsky

English:

What makes Enterpret special is its ability to build and update customer specific AI models that provide the most granular and accurate insights into your business. Connect customer insights to revenue and operational data in your CRM or data warehouse to map the business impact of each customer need and prioritize confidently and empower your entire team to easily take action on use cases like win-loss analysis, critical bug detection, and identifying drivers of churn with Enterpret's AI assistant Wisdom. Looking to automate your feedback loops and prioritize your roadmap with confidence like Notion, Canva, and Linear visit E-N-T-E-R-P-R-E-T.com/lenny to connect with the team and to get two free months when you sign up for an annual plan. This is a limited time offer. That's enterpret.com/lenny. Many of you are building AI products, which is why I'm very excited to chat with Brandon Foo, Founder and CEO of Paragon. Hey Brandon.

中文翻译:

Enterpret 的特别之处在于它能够构建和更新特定于客户的 AI 模型，从而为你的业务提供最细致、最准确的洞察。将客户洞察与 CRM 或数据仓库中的收入和运营数据连接起来，以映射每个客户需求的业务影响，并自信地确定优先级。通过 Enterpret 的 AI 助手 Wisdom，赋能你的整个团队，轻松针对盈亏分析、关键错误检测和识别流失驱动因素等用例采取行动。想要像 Notion、Canva 和 Linear 一样自动化你的反馈循环并自信地规划路线图吗？请访问 enterpret.com/lenny 联系团队，并在注册年度计划时获得两个月的免费试用。这是限时优惠。网址是 enterpret.com/lenny。你们中的许多人正在构建 AI 产品，这就是为什么我很高兴能与 Paragon 的创始人兼 CEO Brandon Foo 聊天。嘿，Brandon。

[00:04:15] Brandon Foo

English:

Hey Lenny. Thanks for having me.

中文翻译:

嘿，Lenny。谢谢邀请我。

[00:04:17] Lenny Rachitsky

English:

So integrations have become a big deal for AI products. Why is that?

中文翻译:

集成（Integrations）对 AI 产品来说已经变得非常重要。这是为什么呢?

[00:04:21] Brandon Foo

English:

Integrations are mission-critical for AI for two reasons. First, AI products need contacts from their customer's business data such as Google Drive files, Slack messages or CRM records. Second, for AI products to automate work on behalf of users, AI agents need to be able to take action across these different third-party tools.

中文翻译:

集成对 AI 来说至关重要，原因有二。首先，AI 产品需要来自客户业务数据的上下文，例如 Google Drive 文件、Slack 消息或 CRM 记录。其次，为了让 AI 产品代表用户自动化工作，AI 智能体（agents）需要能够在这些不同的第三方工具中采取行动。

[00:04:40] Lenny Rachitsky

English:

So where does Paragon fit into all this?

中文翻译:

那么 Paragon 在这其中扮演什么角色呢?

[00:04:42] Brandon Foo

English:

Well, these integrations are a pain to build and that's why Paragon provides an embedded platform that enables engineers to ship these product integrations in just days instead of months across every use case from RAG data ingestion to agentic actions.

中文翻译:

构建这些集成非常痛苦，这就是为什么 Paragon 提供了一个嵌入式平台，使工程师能够在几天内而不是几个月内交付这些产品集成，涵盖从 RAG 数据摄取到智能体行动的每一个用例。

[00:04:57] Lenny Rachitsky

English:

And I know from firsthand experience that maintenance is even harder than just building it for the first time.

中文翻译:

我从第一手经验中知道，维护甚至比第一次构建还要难。

[00:05:01] Brandon Foo

English:

Exactly. We believe product teams should focus engineering efforts and competitive advantages, not integrations. That's why companies like You.com, AI21 and hundreds of others use Paragon to accelerate their integration strategy.

中文翻译:

没错。我们认为产品团队应该将工程精力集中在竞争优势上，而不是集成上。这就是为什么像 You.com、AI21 和数百家其他公司使用 Paragon 来加速其集成战略的原因。

[00:05:15] Lenny Rachitsky

English:

If you want to avoid wasting months of engineering on integrations that your customers need, check out paragon@useparagon.com/lenny. Scott, thank you so much for being here and welcome to the podcast.

中文翻译:

如果你想避免在客户需要的集成上浪费数月的工程时间，请访问 useparagon.com/lenny。Scott，非常感谢你能来，欢迎来到本播客。

[00:05:29] Scott Wu

English:

Thanks so much for having me. Excited to be on.

中文翻译:

非常感谢邀请我。很高兴能参加。

[00:05:31] Lenny Rachitsky

English:

I'm really excited to have you here because you are building and you've been building something that is very different from what a lot of other AI companies have been doing for a long time, although they are starting to converge to where you guys are now. We're going to talk about that and it's also just such a unique point in the history of AI and just the journey of AI. And so it's really cool to be chatting right now. And I feel like we're going to chat again in a few years and be like, wow, we were so right about so much and so wrong about so much.

中文翻译:

我非常兴奋能邀请到你，因为你正在构建的东西与许多其他 AI 公司长期以来所做的非常不同，尽管他们现在开始向你们的方向靠拢。我们将讨论这一点，而且现在也是 AI 历史和 AI 发展历程中一个非常独特的时刻。所以现在聊天真的很酷。我觉得几年后我们还会再聊一次，然后感叹："哇，我们当时在很多事情上是对的，但在很多事情上又是错的。"

[00:05:59] Scott Wu

English:

Yeah.

中文翻译:

是的。

[00:06:00] Lenny Rachitsky

English:

And so I'm excited to have you here. Let's start with talking about Devin, giving people an understanding of just what the heck Devin is, is the main product that you guys build. What is the simplest way to understand what is Devin?

中文翻译:

所以我很高兴你能来。让我们先聊聊 Devin，让大家了解一下 Devin 到底是什么，这是你们构建的核心产品。理解 Devin 最简单的方式是什么?

[00:06:10] Scott Wu

English:

Absolutely. And so Devin is a fully autonomous software engineer that is going to work on tasks end to end, and so there are a lot of great tools for all parts of the stack of the AI code workflow. What Devin does is it is a full asynchronous workflow, and so you can tag Devin on an issue in Slack, you're talking about an issue and you tag Devin, you can tag Devin in Linear, you can have Devin and Devin will make pull requests in your GitHub, and so it's very much built to work with engineering teams as your junior engineer.

中文翻译:

当然。Devin 是一个全自主的软件工程师，能够端到端地处理任务。在 AI 代码工作流的各个环节都有很多优秀的工具，而 Devin 的不同之处在于它是一个完整的异步工作流。你可以在 Slack 的议题中 @Devin，或者在 Linear 中标记它，Devin 就会在你的 GitHub 中创建合并请求。所以它的设计初衷就是作为一名初级工程师与工程团队协作。

[00:06:38] Lenny Rachitsky

English:

Amazing, okay. So I remember when you guys launched this, there was this big pitch of this is your new AI engineer and it was really good at a lot of stuff. It wasn't great at other things. It's been a year now about since you guys launched, is that right?

中文翻译:

太棒了。我记得你们发布时，主打的卖点是“这是你的新 AI 工程师”，它在很多方面表现出色，但在另一些方面还不完美。从发布到现在大约一年了，对吧?

[00:06:50] Scott Wu

English:

Yeah, yeah.

中文翻译:

是的，没错。

[00:06:51] Lenny Rachitsky

English:

What's the best way to think about the level of seniority that engineer had back in the day when you guys launched and then the level of seniority of engineer today if that's, I don't know, measure of how to think about Devin?

中文翻译:

如果用资历来衡量，你们发布时的 Devin 相当于什么级别的工程师? 现在的 Devin 又相当于什么级别?

[00:07:02] Scott Wu

English:

Yeah, and it's crazy to think about by the way, because a year ago when we did the initial launch, I mean people didn't really believe that an agent was possible. Right. And it was, I mean, it was a very different time. So like start of 2024, things with model capabilities were definitely quite a bit earlier on, reasoning especially was quite a bit earlier on. And yeah, I mean, in the time since then it's obviously developed a lot. I think in terms of practical skills, there's some comparisons we make. Sometimes we kind of say, well, when we got started it was kind of like a high school CS student and then as time went on, it became more of a college intern and now it's like a junior engineer. But I would say though that those are more rough guidelines because I really like the phrase jagged intelligence for example, because obviously there are certain things that it is much better at than a human. There are certain things that it's much worse at than a human.

中文翻译:

回想起来确实很疯狂，因为一年前我们最初发布时，人们甚至不相信“智能体（agent)”是可能的。那是一个完全不同的时期。2024 年初，模型能力还处于早期阶段，尤其是推理能力。在那之后，它显然进步了很多。在实际技能方面，我们做过一些对比。有时我们会说，刚开始时它像个高中计算机系学生，后来变成了大学实习

生，现在像个初级工程师。但我会说这些只是粗略的参考，因为我非常喜欢“参差不齐的智能（jagged intelligence）”这个词。显然，有些事情它比人类做得好得多，而有些事情它比人类差得多。

[00:07:52] Scott Wu

English:

And I think over the last year we've learned a lot especially about not just coding agents but agents in general just really building out how all of us should be working and interacting with agents as part of our flow. And so a lot of the things that we built, I mean, there was no Slack, there was no GitHub integration, there was no Linear, there was no interactive planning phase working back and forth. There was no way to touch up Devin's code. And so a lot of the features that we've built on the product side since then have really been about basically yeah, figuring out how to make working with Devin and handing off tasks to Devin as smooth of an experience as possible.

中文翻译:

在过去的一年里，我们学到了很多，不仅是关于编程智能体，还有关于通用智能体的知识，即如何构建我们所有人与智能体协作的工作流。我们最初构建时，还没有 Slack、GitHub 集成，没有 Linear，也没有来回互动的规划阶段，甚至无法修改 Devin 的代码。因此，从那时起我们在产品端构建的许多功能，核心都是为了让与 Devin 协作和向其交付任务的体验尽可能顺畅。

[00:08:27] Lenny Rachitsky

English:

That's so interesting. So a lot of the work has gone not into how do we just make Devin the best possible engineer, but it's how to work with this new type of entity that we haven't ever worked with.

中文翻译:

很有意思。所以很多工作并不是在研究“如何让 Devin 成为最强的工程师”，而是在研究“如何与这种我们从未接触过的新型实体协作”。

[00:08:37] Scott Wu

English:

I think it's a 50/50 of both. I think the capabilities obviously have improved a ton and we've seen these get better and get measurably better. But I think the other side of it is everything to do with yeah, really the product interface and the tools and so on. And I think today folks generally know how to use chatbots and to with chatbots, right, and that's an interface that people are familiar with and obviously with agents it's still like a real curve, I think to learn how to use them and how to get the most out of them. And so it's really exciting to see a lot of others starting to build and do a lot more in the agent space as well. But I think this is the kind of thing that we're all really figuring out together as a space.

中文翻译:

我认为两者各占一半。能力显然提升了巨大，我们能看到明显的进步。但另一方面是关于产品界面、工具等等。今天人们普遍知道如何使用聊天机器人并与之互动，那是大家熟悉的界面。但对于智能体，学习如何使用它们并发挥最大效用仍然有一个陡峭的学习曲线。看到其他人也开始在智能体领域构建更多东西，这非常令人兴奋。我认为这是我们整个行业正在共同探索的事情。

[00:09:15] Lenny Rachitsky

English:

What can you share about just the scale of Devin at this point, whatever you're comfortable sharing, and then just where do you think the level of Devin's coding abilities will be in a year?

中文翻译:

关于 Devin 目前的规模，你能分享些什么（在方便透露的范围内）？另外，你认为一年后 Devin 的编程能力会达到什么水平?

[00:09:24] Scott Wu

English:

So we work with companies of all stages and sizes. On the smallest end, it goes to startups of just one or two people who are using Devin to build out a lot of their kind of initial prototype or initial product all the way up to big public companies, Fortune 100 companies or public banks or things like that who are using Devin across their engineering teams. In general, we've seen a huge range of the use cases there. And obviously the kinds of engineering work that you're doing at a one or two person startup, they're very different from the kind of work that you're doing at a public bank.

中文翻译:

我们与各种阶段和规模的公司合作。小到只有一两个人的初创公司，他们用 Devin 构建最初的原型或产品；大到大型上市公司、财富 100 强企业或公共银行，他们在整个工程团队中使用 Devin。总的来说，我们看到了极其广泛的用例。显然，一两个人的初创公司所做的工程工作，与公共银行的工作是非常不同的。

[00:09:55] Scott Wu

English:

But throughout it's all been basically yeah, being that junior buddy of yours that makes you go faster and really multiplies you, I would say. I think it can multiply you as an engineer obviously by just letting you work with your own team of Devins instead of having to be kind of fully synchronous on a single task. And then it's also kind of multiplying your team and multiplying your team's knowledge base because Devin really accumulates a lot of the knowledge from working with every member of your team and is able to bring that into each new session.

中文翻译:

但贯穿始终的核心是，它就像你的初级伙伴，让你跑得更快，并真正实现“倍增”。作为一名工程师，它能让你通过与自己的 Devin 团队协作来实现倍增，而不是被困在单个任务的同步工作中。同时，它也在倍增你的团队和团队的知识库，因为 Devin 会积累与团队每位成员协作时的知识，并将其带入每一个新的会话中。

[00:10:23] Lenny Rachitsky

English:

Awesome. We're going to show people how it actually works later in the podcast. We're going to do a few live demos, but let's actually go to the beginning of the journey. What's just the origin story of Devin? How did this all begin?

中文翻译:

太棒了。稍后我们会在播客中向大家展示它的实际运作方式，做几个现场演示。但现在让我们回到旅程的起点。Devin 的起源故事是怎样的？这一切是怎么开始的？

[00:10:33] Scott Wu

English:

The founding team, I mean most of us have known each other for years and years and years actually. And for almost everyone, this is our first time working together, but we've known each other a long time. And we all actually had our own kind of journeys in AI for the last decade or so. And so for myself, I ran a company before this called Lunchclub, which was an AI for a professional networking product and I ran that for about five years. And one of my co-founders, Steven was one of the first engineers at a company called Scale AI, which has obviously grown a lot and done very well. My other co-founder Walden, was an early engineer at a company called Cursor, which has also obviously grown a lot and done really well. And our whole team was kind of like that. Many of us knew each other from competitive programming and math competitions, but we had stayed very closely in touch in the decade since then and we've all kind of all had our own journeys.

中文翻译:

创始团队的大多数人其实已经认识很多很多年了。虽然对几乎所有人来说这是我们第一次一起工作，但我们相识已久。在过去的十年左右时间里，我们每个人在 AI 领域都有自己的历程。我之前经营过一家叫 Lunchclub 的公司，那是一个 AI 驱动的职业社交产品，我做了大约五年。我的联合创始人之一 Steven 是 Scale AI 的早期工程师之一，那家公司现在发展得非常好。另一位联合创始人 Walden 是 Cursor 的早期工程师，Cursor 现在也非常火。我们整个团队基本都是这样，许多人是通过竞赛编程和数学竞赛认识的，但在那之后的十年里我们一直保持着密切联系。

[00:11:22] Scott Wu

English:

And so we had one person who was running teams at Neuro, we had one person who was at Waymo, someone who had their own YC tools startup for machine learning, and we were really excited to build something together. And this was around late 2023, so about a year and a half ago at this point. And yeah, when we got started, I mean I think there were a couple of things that we felt really strongly about and one was that reinforcement learning was really working and was going to be the next big paradigm shift in capabilities. Back then it was the initial ChatGPT launch in 2022, and those models were, to first order were what we would call imitation learning in AI, right, which is basically you have the model read all the texts that you can find on the internet and then train it to talk like somebody on the internet would talk. Right. And there are kind of obviously a lot more details on top of that, but that's kind of the first order pass of what was really done.

中文翻译:

我们团队里有人曾在 Neuro 领导团队，有人在 Waymo 工作，还有人拥有自己的 YC 机器学习工具初创公司。我们非常兴奋能一起做点什么。那是 2023 年底，距今大约一年半。刚开始时，有几件事我们深信不疑：一是强化学习（Reinforcement Learning）确实有效，并将成为下一个重大的能力范式转移。当时是 2022 年 ChatGPT 最初发布后，那些模型从本质上讲是我们所说的“模仿学习（imitation learning）”，即让模型阅读互联网上能找到的所有文本，然后训练它像互联网上的人那样说话。当然，这之上还有很多细节，但那是当时的核心逻辑。

[00:12:16] Scott Wu

English:

And it was amazing. Right. I mean, it passed the churn test, it was able to respond and to have encyclopedic knowledge about a lot of things. And I think this new paradigm which we've gotten into over this last year or year and a half is really high compute RL, which is a very different paradigm, right, which is basically the ability to go and do work on task and put something together and then be evaluated on whether that was correct or incorrect and use that knowledge to decide what to do and to learn from that. Right. And so we felt very strongly that that was going to happen. I think for us, code was the natural thing to work on for a couple reasons. One, because we're all programmer nerds ourselves, and so teaching AI to code is about as cool as it gets for us, but also because code has this whole automated feedback loop, right, where you can run the code and that is the kind of automated feedback that really feeds into the RL, which makes these models so great at coding.

中文翻译:

那很神奇，它通过了图灵测试，能够回答问题并拥有百科全书般的知识。但我认为过去一年半我们进入的新范式是“高算力强化学习（high compute RL）”，这是一个非常不同的范式。它基本上是让模型去执行任务、构建东西，然后根据结果的正确与否进行评估，并利用这些知识来决定下一步做什么并从中学习。我们坚信这会发生。对我们来说，选择“代码”作为切入点是很自然的：首先，我们自己都是编程极客，教 AI 写代码对我们来说酷毙了；其次，代码拥有完整的自动化反馈闭环，你可以运行代码，这种自动化反馈能直接喂给强化学习，这也是为什么这些模型在编程方面如此出色。

[00:13:06] Scott Wu

English:

And then the other thing that we felt very strongly about was that the product experience was going to shift from what I'll call text completion to agents basically. Right. And to first order, I would kind of say there's been a lot of great experiences in text completion. It's been used for marketing, it's been used for customer support, it's been used for education and in Coda obviously as well. The GitHub copilot was kind of really the dominant product of that initial wave. Right.

中文翻译:

另一件我们深信不疑的事是，产品体验将从我所说的“文本补全”转向“智能体（agents）”。在文本补全方面已经有很多很棒的体验了，它被用于营销、客户支持、教育，显然还有 Coda。GitHub Copilot 曾是那一波浪潮中的主导产品。

[00:13:34] Scott Wu

English:

But I think that the big shift that we really felt we would see is moving from kind of this text to text model to an actual autonomous system that can make decisions, that can interact with the real world, that can take in feedback, that can iterate and take multiple steps to solve problems. And now we call that agents, but that was what we were really excited about at the time. So it was always coding, it was always agents. And in some ways that kind of feels like it should have been cleared from the start. But even with that, I feel like we've pivoted eight times or something within coding agents over the last year and a half, so.

中文翻译:

但我认为我们预见到的重大转变是，从这种“文本到文本”的模型转向一个真正的自主系统，它能做决策、能与现实世界互动、能接收反馈、能迭代并采取多个步骤来解决问题。现在我们称之为“智能体”，但那是我们当时真正兴奋的点。所以，方向一直是编程，一直是智能体。某种程度上，这似乎从一开始就很明确，但即便如此，在过去的一年半里，我们在编程智能体这个领域内也经历了大概八次转型。

[00:14:08] Lenny Rachitsky

English:

I just noticed recently all the AI, top AI companies sort of, not all, but many of them, the product that is winning is different, has a different name from the company, which is not typical, Cursors, Anysphere, Bolt, StackBlitz, you guys are Cognition Labs, like V0 is Vercel. And it just tells me these all emerge later in the company's journey and they tried a bunch of stuff and like, oh wow, this thing worked and it's so interesting that it's so common amongst these top AI companies.

中文翻译:

我最近注意到，许多顶尖 AI 公司的成功产品名称与公司名称不同，这并不常见。比如 Cursor 的公司是 Anysphere，Bolt 是 StackBlitz，你们是 Cognition Labs，V0 是 Vercel。这告诉我，这些产品都是在公司发展后期出现的，他们尝试了很多东西，然后发现“哇，这个行得通”。这种现象在顶尖 AI 公司中如此普遍，真有意思。

[00:14:34] Scott Wu

English:

Yeah, and there's even, I mean OpenAI, ChatGPT, Anthropic and Claude and Google. Yeah, it's funny. Yeah, yeah, I agree. So when we got started, it wasn't even really a company. I mean, it was more like a project or a hackathon almost. We got a bunch, we booked an Airbnb basically for a couple of weeks. This was around Thanksgiving time and just got a bunch of people together who were just excited to hack on some projects and build something cool. And it's funny, actually the first thing that we were building for actually was more solving these more contest programming problems and using an agentic loop to really do better on that. And so obviously if you run your code on the test cases, you can evaluate, there's a lot of agentic work that you can do there to try and do better, and that we spent some time on that initially. And then we've kind of gone from, I mean the story of the whole company for us in some sense has been going from hacker house to hacker house.

中文翻译:

是的，甚至还有 OpenAI 的 ChatGPT，Anthropic 的 Claude，还有 Google。确实很有趣，我同意。我们刚开始时，甚至还不算一家公司，更像是一个项目或一场黑客松。感恩节前后，我们租了一个 Airbnb 住了几周，召集了一群对开发项目和构建酷炫东西充满热情的人。有趣的是，我们最初做的其实是解决竞赛编程问题，并利用智能体循环（agentic loop）来做得更好。显然，如果你在测试用例上运行代码，你就可以进行评估，通过智能体化的工作来尝试优化结果。我们最初在那上面花了不少时间。然后我们就像是从一个“黑客之家（hacker house）”搬到另一个“黑客之家”，这就是我们公司的成长史。

[00:15:26] Scott Wu

English:

After that we had another hacker house and that's where kind of some of the initial ideas for Devin came and really building a software engineering agent and not just a coding agent and having to interact with a lot of these tools, but even then there were so many iterations. And even the idea of talking to Devin for example was like, it was something that we had to come up with. Right. Initially, it was just like you hand off a task and then it works and then it shows you this whole finished code. Right. And now obviously it's like you can jump in at any time, you can get feedback on the plan, you guys can scope out the task together when you're working with Devin. And a lot of these things we had to develop obviously, and certainly we've learned a lot about the use cases, the form factor. We've made a lot of big improvements and step function improvements on the capabilities and Devin's ability to use tools and to bug and make decisions.

中文翻译:

在那之后，我们又进驻了另一个黑客之家，Devin 的一些最初想法就是在那里诞生的——即构建一个“软件工程智能体”而不仅仅是“编程智能体”，并且需要与大量工具交互。但即便如此，也经历了很多次迭代。甚至连“与 Devin 对话”这个点子也是我们后来想出来的。最初，流程只是你交付一个任务，它去执行，然后展示完成的代码。而现在，你可以随时介入，对计划提供反馈，在与 Devin 协作时共同确定任务范围。这些功能都是我们后来开发的。当然，我们也学到了很多关于用例和产品形式的知识。我们在能力、Devin 使用工具的能力、调试和决策能力方面都取得了巨大的、阶梯式的进步。

[00:16:11] Scott Wu

English:

And so it's yeah, it's been a fun journey. I mean, I think I would say that the grounding question for us really, which is one that we think about all the time, is really just what is the future of software engineering and how should we be working with AI to write code? Because I think at the end of the day, of course that's what underlines all of the product decisions that we make, so.

中文翻译:

所以，这是一段有趣的旅程。我想说，我们一直思考的核心问题其实就是：软件工程的未来是什么？我们应该如何与 AI 协作编写代码？因为归根结底，这才是我们所有产品决策的基石。

[00:16:45] Scott Wu

English:

Yeah, so we started in November of 2023, which was then yeah, just like hackathon mode. We officially made it into a company around the start of 2024, and then our initial launch was in March. And so it was like nonstop, I mean it's been nonstop for the entire last 17 months, but it's getting to the launch and then obviously working with enterprises and developing the product a lot more, building it and getting it to work for a lot of practical use cases and then making it fully available self-serve in December of last year. And now we've rolled out 2.0 obviously just a few weeks ago. And so it's been a very busy time for us.

中文翻译:

是的，我们从 2023 年 11 月开始，当时就是黑客松模式。2024 年初正式成立公司，3 月份进行了首次发布。这 17 个月来我们一直马不停蹄，从发布到与企业合作，再到深入开发产品，使其适用于大量实际用例，然后在去年 12 月全面开放自助服务。几周前我们刚刚推出了 2.0 版本。所以这段时间我们非常忙碌。

[00:17:25] Lenny Rachitsky

English:

Understatement of the century. Let me ask this question because you touched on it a bit, this whole idea of Devin as a person and this idea of creating a personality for Devin. It's unlike any other, I believe, AI app. No one else has a name and you don't think of it as a person. What made you guys decide to go that approach and just how do you design it to work well that way?

中文翻译:

这真是“世纪级”的轻描淡写。我想问个问题，因为你刚才提到了，就是把 Devin 当作一个“人”的想法，以及为 Devin 创造个性的想法。据我所知，这与其他 AI 应用完全不同。没有其他产品有名字，你也不会把它当成一个人。是什么让你们决定采用这种方式？你们又是如何设计让这种方式奏效的？

[00:17:44] Scott Wu

English:

I would say it's a decision we're pretty proud of, I would say. I mean, I think there's a lot of different product experiences out there, and I think the thing that really makes Devin unique in what it does is that you can really hand off and more and more what we've seen honestly is that I think a lot of kind of explaining the Devin experience to folks is really just explain it as, yeah, this is your junior buddy. And that goes for a lot of the parts of the flow where in the onboarding for example, initially I would say we've definitely had a lot of users come in and just kind of see the blank screen and not really know, or they'd ask, "Hey, I'm going to do this whole big re-architecture, the whole code base."

中文翻译:

我会说这是一个让我们感到自豪的决定。市面上有很多不同的产品体验，但我认为 Devin 的独特之处在于你真的可以“交付”任务。老实说，我们发现向人们解释 Devin 体验的最好方式就是：它是你的初级伙伴。这体现在流程的很多环节。例如在新手引导阶段，最初很多用户进来看到空白屏幕会不知所措，或者会问：“嘿，我要对整个代码库进行大规模重构。”

[00:18:21] Scott Wu

English:

And basically what we've learned over time is to basically get folks to think more like whoa, whoa, whoa, let's work on getting the repository set up first. Let's make sure we hand Devin a couple one pointer tasks so it can get familiar with the code base. Let's get it the thing. If Devin needs to be able to test the code or run the linter or CI or things like that. Obviously we want to make sure Devin's got its own virtual machine set up to be able to do that.

中文翻译:

我们逐渐学到的是，要引导用户这样想：喔，等等，我们先来设置代码库。先给 Devin 几个简单的任务（one-pointer tasks），让它熟悉代码库。如果 Devin 需要测试代码、运行 linter 或 CI 等，我们要确保 Devin 已经设置好了自己的虚拟机来执行这些操作。

[00:18:45] Scott Wu

English:

And similarly, I think the usage pattern, I think often it wasn't clear and obviously you can sit and just kind of watch Devin do it action by action and work that way, but we found that the best workflow really as a team building a lot of stuff was to work with multiple Devins and to run them asynchronously and to kick them off and to only jump in basically as you needed to provide feedback or steer the plan or anything like that. And so in many ways, I think Devin as a name really is our attempt to kind of capture the soul of that as a product where it really is treating it like a bit more of an autonomous entity that you can hand off tasks, that you can work with, that you should be teaching and learning with over time.

中文翻译:

同样，关于使用模式，最初可能并不清晰。显然你可以坐着看 Devin 一步步操作，但我们发现，对于一个构建大量东西的团队来说，最好的工作流是同时与多个 Devin 协作，让它们异步运行。你启动它们，只在需要提供反馈或调整计划时才介入。所以从很多方面来说，“Devin”这个名字是我们试图捕捉产品灵魂的尝试——它更像是一个自主实体，你可以交付任务，可以与之协作，并且你应该随着时间的推移与它共同教学相长。

[00:19:29] Lenny Rachitsky

English:

I want to come back to an area you started us down and then I took us away from which is impact on software engineering and how software engineering is going to change. So there's kind of two parts to this. Just like when people are using Devin today, say this year, how is the act of being an engineer and building changing for those companies? What does that look like?

中文翻译:

我想回到你刚才提到但我打断了的话题，即对软件工程的影响以及软件工程将如何改变。这分为两部分：就今年而言，当人们使用 Devin 时，这些公司的工程师工作方式和构建方式发生了怎样的变化？具体是什么样的？

[00:19:49] Scott Wu

English:

By the way, we're all software engineers ourselves. It's like I'm a programmer by training and still a programmer at heart certainly. And I think the way that we've always thought about it is there's layers of abstraction and there's tools, and one way I would say it at a high level is kind of I think of AI in general as, yeah, I mean computers are obviously getting more and more intelligence and are able to do more and more and it's possible there may come a day where computers truly do everything that we do and humans are not responsible for any of it.

中文翻译:

顺便说一下，我们自己都是软件工程师。我受过编程训练，内心深处依然是个程序员。我们一直认为软件工程就是抽象层和工具。从高层次来看，我认为 AI 意味着计算机显然变得越来越聪明，能做的事情越来越多。也许有一天，计算机真的能做我们所做的一切，而人类不再需要负责任何事情。

[00:20:20] Scott Wu

English:

I don't expect that to come particularly soon, but I guess what I would say is until that point, for as long as we're still part of the equation, one of the most important things to do obviously is for us as humans is to instruct our computers on what we want and what we want to build and what we want to do. Right. And software engineering is, we think of it today obviously as Python and C++ and JavaScript and all these things, but at the end of the day, of course the discipline is all about just being to tell your computer what to do. And so in that lens, I really think that programming is, if anything, is only going to become more and more important as AI gets more powerful. And I think the thing that's really exciting for us is yeah, it's really seeing that kind of iterative transformation. And so you ask what things look like today, and I would say, yeah, it really is like having a junior buddy or really a team of junior buddies that you can work with. Right.

中文翻译:

我不认为那一天会很快到来。但在那之前，只要我们还是这个等式的一部分，对我们人类来说，最重要的显然就是指导计算机我们想要什么、想构建什么、想做什么。今天我们认为软件工程是 Python、C++、JavaScript 等等，但归根结底，这门学科的核心就是告诉计算机该做什么。从这个角度看，我真的认为随着 AI 变得越来越强大，编程只会变得越来越重要。让我们兴奋的是看到这种迭代式的转型。你问现在是什么样子，我会说，这真的就像拥有一个初级伙伴，或者说一队你可以协作的初级伙伴。

[00:21:12] Scott Wu

English:

And so every engineer on our team, we use a ton of Devin when we're building Devin, and so Devin merges several hundred pull requests into production in the Devin code bases every month, which is, I mean, our whole team is only 15 engineers, and so it's a pretty sizable fraction of all the code that we write. And the way that we use it is basically, yeah, everyone's got their whole team of Devins. If you're going to be looking through various issues, if you're going through feature requests, if you're going through bugs, if you're going through new paradigms that you want to build, then it is naturally the case that there's a lot of handoff points where you just say, "Hey, at Devin, here's what's going on. Can you please take a pass at this?" Right.

中文翻译:

我们团队的每一位工程师在构建 Devin 时都会大量使用 Devin。Devin 每月会向 Devin 的生产代码库合并数百个 PR。我们整个团队只有 15 名工程师，所以这占了我们编写的所有代码中相当大的比例。我们的使用方式是：每个人都有自己的 Devin 团队。如果你在处理各种 issue、功能请求、bug，或者想构建新的范式，自然会有很多交付点，你只需说："嘿 @Devin，情况是这样的，你能处理一下吗？"

[00:21:47] Scott Wu

English:

Sometimes Devin will be able to do the task 100% autonomously and just makes the PR and then you merge the PR and that's great. Sometimes you want to be able to jump in for the 10 or 20% that really needs your help. Maybe there's a few details with how exactly you want to scope it or how you're architecting this feature, or maybe you want to go and test the front end at the end yourself to make sure it looks exactly the way that you want and give your one or two lines of feedback after that. Right. But a lot of it is really yeah, is kind of like yeah, learning to work with Devin to be able to just do more in parallel and build more.

中文翻译:

有时 Devin 能 100% 自主完成任务并提交 PR，你直接合并就行，这太棒了。有时你需要在剩下的 10% 或 20% 真正需要你帮助的地方介入。也许是关于任务范围的细节，或者是你如何架构这个功能，又或者你想在最后亲自测试前端，确保它看起来完全符合你的要求，然后给出一两行反馈。但很大程度上，这就像是学习如何与 Devin 协作，从而能够进行更多的并行工作并构建更多东西。

[00:22:20] Lenny Rachitsky

English:

What percentage of your PRs are Devin versus humans right now?

中文翻译:

目前你们的 PR 中，Devin 提交的比例和人类提交的比例分别是多少?

[00:22:25] Scott Wu

English:

Yeah, I'd have to look, but it's in the neighborhood of a quarter or so of all of our-

中文翻译:

我得查一下，但大约占我们所有 PR 的四分之一左右。

[00:22:30] Lenny Rachitsky

English:

Wow.

中文翻译:

哇。

[00:22:31] Scott Wu

English:

Yeah. Yeah.

中文翻译:

是的。

[00:22:33] Scott Wu

English:

Oh yeah, it's grown a ton for, I mean, we've seen it grown exponentially internally ourselves as well. And so it's kind of an interesting one where again, it's always both the capabilities and the product interface. And so I think the intelligence has increased a lot. But the other thing of course is that we've spent a lot of time in figuring out how to build and to really kind of build for an interface where you can get Devin's value on tasks where Devin is able to do the 80 or 90%. And so Devin is obviously not perfect and it'll make mistakes and so on. And a lot of the question is basically, yeah, how do you scope out your initial task with Devin and then just kind of set Devin off and have it go and do the things that you want do? How do you come in at the end and review and give feedback? How do you make sure Devin learns over time? How are you able to kind of just check in as needed and course correct if you want to?

中文翻译:

是的，增长了很多。我们内部也看到了指数级的增长。这很有趣，因为这始终是能力和产品界面的结合。我认为智能程度提高了很多，但另一件事是，我们花了很多时间研究如何构建一个界面，让你在 Devin 能完成 80% 或 90% 的任务中获得它的价值。Devin 显然不完美，它会犯错。核心问题在于：你如何与 Devin 确定初始任务范围，然后让它去执行你想要的操作？你如何在最后介入进行审查和反馈？你如何确保 Devin 随时间学习？你如何根据需要随时查看进度并在必要时纠偏？

[00:23:24] Lenny Rachitsky

English:

Okay, so today about quarter of your PRs are Devins. Where do you think this will be at the end of the year? What would you guess?

中文翻译:

好，现在大约四分之一的 PR 是由 Devin 提交的。你猜到年底这个比例会是多少？

[00:23:31] Scott Wu

English:

I think by the end of this year, we expect it to be more than half. And I mean, as time goes on, one of the things that we've seen is just you're able to do more and more and more work asynchronously. Right. And you're able to hand off more and more. I think the soul of programming, the soul of software engineering has really been about through all the areas, not just now, but even when it was assembly, right, and even when it was Pascal and even when it was punch cards or whatever, I think the soul of it has really been basically just about defining the problem that you're facing and really thinking through exactly...

中文翻译:

我想，到今年年底，我们预计会超过一半。随着时间的推移，我们看到的是你可以越来越多地进行异步工作。你可以交付越来越多的任务。我认为编程的灵魂、软件工程的灵魂，贯穿所有时代——不仅是现在，甚至在汇编语言时代、Pascal 时代、打孔卡时代——其灵魂始终在于定义你面临的问题，并仔细思考……

[00:24:00] Scott Wu

English:

... about defining the problem that you're facing and really thinking through exactly what is the solution that you want to build. Thinking through the architecture, thinking through the details, and really mapping out in your mind exactly what you want to build basically and what you want to have your computer do. I think that's what makes software engineering really great and I think that's the funnest part of software engineering.

中文翻译:

……定义你面临的问题，并仔细思考你想要构建的解决方案。思考架构，思考细节，在脑海中勾勒出你想要构建的东西以及你想要计算机执行的操作。我认为这正是软件工程的伟大之处，也是软件工程最有趣的部分。

[00:24:21] Scott Wu

English:

I think, at the same time, that's probably in the neighborhood of 10% of the average software engineer's time, right? Because 90% of the time is you've got this Kubernetes error, then you've got to debug, and you have to see what went wrong and the system crash, or you left some port open and this is messing up, or there's a bug report that you have to take care of, or you've got to migrate your code, or you got it upgraded to a new version or things like that. A lot more implementation.

中文翻译:

但与此同时，这可能只占普通软件工程师 10% 的时间，对吧？因为 90% 的时间你都在处理 Kubernetes 错误、调试、查找系统崩溃的原因、处理没关掉的端口导致的混乱、处理 bug 报告、迁移代码或升级版本等等。大部分时间都在做具体的实现工作。

[00:24:48] Scott Wu

English:

One of the ways that we've kind of thought about Devin in building Devin is really allowing engineers to go from bricklayer to architect, so to speak. A lot of it is just getting to the point where you can do the high-level directing and you can basically specify things exactly how you want. I think it's very much about still having the human in control and having the human able to do the full specification, but just multiplying the magnitude of what you can do and what you can build in one day or one hour or however long.

中文翻译:

我们在构建 Devin 时的一种想法是，让工程师从所谓的“搬砖工”变成“架构师”。很大程度上是为了让你能进行高层次的指导，并按照你的意愿精确地指定要求。我认为这依然是关于让人类保持控制，让人类能够进行完整的规格说明，但却能将你在一天、一小时或任何时间内能做的事情和能构建的东西的量级翻倍。

[00:25:18] Lenny Rachitsky

English:

In the future, say someone is trying to get into software engineering, thinking about becoming an engineer, first of all, do you think people should... Classic question everyone's getting these days. Should you still learn to code?

中文翻译:

在未来，假设有人想进入软件工程领域，想成为一名工程师。首先，你认为人们是否应该……这是现在每个人都会被问到的经典问题：还应该学习写代码吗？

[00:25:29] Scott Wu

English:

Yeah.

中文翻译:

是的。

[00:25:30] Scott Wu

English:

I'd love your perspective there. And then two, for people that are engineers today, what skills do you think will be more and more important and then less important in this discussion of moving from bricklayer to architect?

中文翻译:

我很想听听你的看法。其次，对于现在的工程师，在从“搬砖工”向“架构师”转变的过程中，你认为哪些技能会变得越来越重要，哪些会变得不那么重要？

[00:25:41] Scott Wu

English:

Yeah, for sure. I love this question. First of all, the question of whether you should still learn to code, my answer would be absolutely yes. I think, to a large extent, when you take computer science classes and when you learn these fundamentals, sure you're learning a little bit about how a particular language is, syntax works or something like that. But honestly, most of what you're learning really is about the ability to logically break down problems for number one. And two, I would say is just the model of a computer and a lot of these decisions and a lot of the abstractions that we've built over time.

中文翻译:

当然，我喜欢这个问题。首先，关于是否还应该学习写代码，我的回答是：绝对要学。我认为，在很大程度上，当你上计算机科学课、学习这些基础知识时，你确实学到了一些特定语言的语法之类的东西。但老实说，你学到的核心东西，第一是逻辑化分解问题的能力。第二，是计算机的模型，以及我们随时间建立的许多决策和抽象。

[00:26:11] Scott Wu

English:

What is a database and how should you think about a database? What is a garbage collection system and how do those work and all of these different pieces? The reason I think that's important is because it's the same with a lot of these other... Arguably we've already gone through these phases in programming and I think this next one is going to be somewhat faster and somewhat bigger, but in many ways a similar flavor, which is when you work with Python today, obviously, a lot of things are already abstracted away from you.

中文翻译:

什么是数据库？你应该如何思考数据库？什么是垃圾回收系统？它们是如何工作的？所有这些不同的组成部分。我认为这很重要，因为这和许多其他事物一样……可以说我们在编程中已经经历过这些阶段，我认为下一个阶段会更快、规模更大，但在很多方面风格相似。比如你今天用 Python 工作，显然很多东西已经为你抽象掉了。

[00:26:38] Scott Wu

English:

In some sense, someone from 50 years ago might already call Python. You just get to explain in English what you want and now the computer does it for you. That's great and I think it's really powerful. It's opened it up. I mean, we have far more programmers, obviously, than we ever have before because of that, but I would say certainly as you're building your skills as an engineer, it really helps a lot to understand the abstractions and to be able to peel the layers beneath. Folks will use assembly for example if they're really performance optimizing a piece of code, but also in order to build good systems and to understand these things, you certainly want to understand these abstractions of how does networking work.

中文翻译:

在某种意义上，50 年前的人可能会觉得 Python 就像是：你只需用英语解释你想要什么，计算机就为你完成。这很棒，我认为它非常强大，它降低了门槛。正因为如此，我们现在的程序员显然比以往任何时候都多。但我会说，当你作为工程师提升技能时，理解抽象并能够剥开底层逻辑真的很有帮助。例如，如果人们真的要对一段代码进行性能优化，他们会使用汇编语言。同样，为了构建好的系统并理解这些事物，你肯定想理解这些抽象，比如网络是如何工作的。

[00:27:18] Scott Wu

English:

What is TCP/IP like exactly or what happens with this Python code when it gets interpreted or all of these details. Similarly, I think we will get to a state where, with no experience at all, you're going to be able to build some pretty cool stuff and to do some pretty amazing work just by explaining what it is that you want. But I think that, for quite some time, you really want to be able to think precisely about the details, to peel back the abstractions, to be very precise about what it is that you want to build and how.

中文翻译:

TCP/IP 到底是什么样的？或者这段 Python 代码被解释时发生了什么？所有这些细节。同样，我认为我们将达到这样一个状态：即使完全没有经验，你也能通过解释你想要什么来构建一些非常酷的东西，完成一些了不起的工作。但我认为，在相当长的一段时间内，你仍然需要能够精确地思考细节，剥开抽象层，对你想要构建什么以及如何构建保持极高的精确度。

[00:27:48] Lenny Rachitsky

English:

And then for skills that you think are more and more valuable for engineers, where should engineers today be leaning more and more into versus like, "Forget this. I don't need to think about this anymore"?

中文翻译:

那么对于你认为对工程师越来越有价值的技能，现在的工程师应该更多地向哪些方面倾斜，而不是觉得“算了，我再也不用考虑这个了”？

[00:28:00] Scott Wu

English:

For sure. I think architect... I mean, we already have a term for architect and engineering and I think it is directionally the right term. It's, I think, one thing to just do a routine implementation and write boilerplate code and things like that. I would say that, in many ways, AI coding has already made us much faster at that, but I think a lot of the core questions of understanding very complex systems and working in the context of the whole company and thinking about the product that you're building or the work that you're doing and understanding, "Okay, what are the problems that we want to solve? How do we want to solve those problems? What is exactly the solution that we want to build? What are all of these key decisions and trade-offs that we're going to be making?"

中文翻译:

当然。我认为是“架构师”……工程领域已经有了架构师这个词，我认为这个方向是对的。做常规实现、写样板代码（boilerplate code）是一回事，AI 编程已经在很多方面让我们在这方面快得多了。但我认为核心问题在于理解极其复杂的系统，在整个公司的背景下工作，思考你正在构建的产品或正在做的工作，并理解：“我们要解决什么问题？我们想如何解决这些问题？我们到底想构建什么样的解决方案？我们要做的关键决策和权衡是什么？”

[00:28:40] Scott Wu

English:

Basically, I think folks who are able to do that really, really well are just going to be able to leverage themselves more and more. If anything, I think there's going to be way more programmers and way more engineers a few years from now than there are today. I think pretty quickly the form factor of what it means to be a programmer, obviously, is going to change. And in some sense, it already has, but I think there's just going to be so much more for us to build. Folks talk about Jevons Paradox all the time. I mean, software is truly the shining example of Jevons Paradox, where we have always managed as a society to find more and more things that we want to build software for and build more code for. I really think there's a lot more out there to do.

中文翻译:

基本上，我认为能够把这些事情做得非常好的人，将能够越来越好地杠杆化自己。如果说有什么不同的话，我认为几年后的程序员和工程师会比现在多得多。我认为程序员的形式很快就会发生变化，某种意义上已经变了。但我认为还有太多的东西等着我们去构建。人们总在谈论“杰文斯悖论（Jevons Paradox）”。软件真的是杰文斯悖论的绝佳例子——作为一个社会，我们总能找到越来越多想要为其构建软件、编写代码的东西。我真的认为还有很多事情可以做。

[00:29:23] Lenny Rachitsky

English:

For people that don't know Jevons Paradox, can you briefly explain it?

中文翻译:

对于不知道杰文斯悖论的人，你能简要解释一下吗?

[00:29:26] Scott Wu

English:

Absolutely. Yeah. Jevons Paradox just says that as the price of something goes down, it can still be the case that the total spend on it actually goes up. You can think about this with money, you can think about this with time or resources, but the direct version here is, I think, as it becomes easier and easier to program and as programming becomes more and more effective, I think we're going to have a lot more programmers. I think in a kind of zero-sum view, you might say, "Well, we're going to be 10 times faster at software engineering and it means that we're going need 10 times fewer software engineers." But I think in practice, what really is going to happen is actually we're going build even more than 10 times as much code. And because all of the work that we do is so capped, obviously, on our ability to actually build and execute and iterate, we're going to have so many great ideas out there, we're going to have so many great products out there. People are going to build a lot more personalized experiences, for example, and there's going to be a lot to do.

中文翻译:

当然。杰文斯悖论是指，随着某种东西的价格下降，对其的总支出反而可能上升。你可以从金钱、时间或资源的角度来思考。在这里直接的解释是：随着编程变得越来越容易、越来越高效，我认为我们将拥有更多的程序员。在零和博弈的观点下，你可能会说："既然软件工程效率提高了 10 倍，那就意味着我们需要减少 10 倍的工程师。"但在实践中，真正会发生的是我们编写的代码量会增加超过 10 倍。因为我们所做的工作受限于我们构建、执行和迭代的能力，未来会有这么多伟大的创意和产品。例如，人们会构建更多个性化的体验，会有很多事情要做。

[00:30:22] Lenny Rachitsky

English:

Going back to the way you guys use Devin, you said that every engineer has this fleet of Devins. How many Devins per engineer do you find most people are working with these days at your company?

中文翻译:

回到你们使用 Devin 的方式，你说每个工程师都有一支 Devin 舰队。在你们公司，你发现现在大多数人平均每个工程师会和多少个 Devin 一起工作?

[00:30:33] Scott Wu

English:

Yeah. It's very asynchronous. Obviously, you can kick them up and start them up and shut them down basically as you see fit, but most folks on the team are often working with up to five Devins at once, I would say. It's a nice flow where you think through, "All right, what are the five things that we want to get done today?" You have Devin one do number one, you have Devin two do number two, Devin three... For what it's worth, I think it's taken us some time to adjust to it and get to the point where it's really intuitive for us, but I think it's definitely a different experience where you're handing off most things asynchronously.

中文翻译:

是的，这是非常异步的。显然，你可以根据需要启动或关闭它们。但我会说，团队中的大多数人通常同时与多达五个 Devin 一起工作。这是一个很好的流程，你会想："好，今天我们要完成哪五件事？" 让 Devin 1 做第一件，Devin 2 做第二件，Devin 3……说实话，我们花了一些时间才适应并达到这种直观的状态，但这绝对是一种不同的体验，你大部分时间都在异步地交付任务。

[00:31:14] Scott Wu

English:

The goal for each of your tasks is to be there for the parts that really need your expertise, either you really, really need to define exactly what it is that you're solving for and what you're building or maybe some of the more complex parts where you want to steer Devin towards, particularly what kinds of changes you want to make. "I want the class to be set up this way and we should go and change all the downstream references to this as well or whatever." But basically having Devin do the bulk of the work asynchronously with you.

中文翻译:

你每个任务的目标是把精力放在真正需要你专业知识的部分，要么是你需要精确定义你要解决的问题和要构建的东西，要么是在一些更复杂的部分，你想引导 Devin 朝特定方向修改。比如："我希望这个类这样设置，我们应该把所有下游引用也改掉。"但基本上是让 Devin 与你异步地完成大部分工作。

[00:31:43] Lenny Rachitsky

English:

And then how many engineers do you guys have roughly?

中文翻译:

你们大约有多少名工程师?

[00:31:46] Scott Wu

English:

Yeah, our engineering team today is about 15 people.

中文翻译:

是的，我们现在的工程团队大约有 15 人。

[00:31:48] Lenny Rachitsky

English:

15. 15.

中文翻译:

15 个，15 个。

[00:31:50] Scott Wu

English:

15, yeah.

中文翻译:

15 个，没错。

[00:31:50] Lenny Rachitsky

English:

Holy moly. Okay. And then each one has five-ish Devins.

中文翻译:

天哪。好吧。然后每个人有五个左右的 Devin。

[00:31:54] Scott Wu

English:

Yeah.

中文翻译:

是的。

[00:31:54] Lenny Rachitsky

English:

So there's five times the number of Devins as engineers. What I love about this is just a glimpse into where the future is going. You guys are so ahead of how companies work with AI engineers. Seeing how you operate is going to be, as a sense, essentially how most companies will end up operating.

中文翻译:

所以 Devin 的数量是工程师的五倍。我喜欢这一点，因为它让我们窥见了未来的走向。在公司如何与 AI 工程师协作方面，你们走得太靠前了。看到你们的运作方式，基本上就是大多数公司最终的运作方式。

[00:32:11] Scott Wu

English:

Yeah. For what it's worth, we've already seen this shift, I would say, ourselves where... In terms of the team, obviously, folks don't spend that much of their time just writing out boilerplate or just doing pure implementation of features. People get to spend much more of their time focused on really just thinking about the core questions of, "How do we make Devin better? What is the right interface for Devin? What is the right flow or the right set of features that's really going to make this as great of an experience as possible?" That's how we like things.

中文翻译:

是的。说实话，我们自己已经看到了这种转变……就团队而言，显然大家不再花那么多时间写样板代码或纯粹的功能实现。人们可以将更多时间花在思考核心问题上，比如："我们如何让 Devin 变得更好？Devin 最合适的界面是什么？什么样的流程或功能组合能让体验尽可能完美？" 这就是我们喜欢的方式。

[00:32:46] Lenny Rachitsky

English:

When is the point you reach, where there's takeoff of this being the by... Your Devin starts moving so much further ahead of everyone else. Once you have enough Devins doing all these things, they're just like wherever and you're 10 years, 20 years, 30 years, 100 years ahead.

中文翻译:

什么时候会达到那个临界点，即你们的 Devin 开始远远领先于其他人？一旦你有足够的 Devin 在做所有这些事情，它们无处不在，而你们就领先了 10 年、20 年、30 年甚至 100 年。

[00:33:00] Scott Wu

English:

Honestly, as a community, I think all of us as engineers around the world, I think, are going to have to think about this and build for this and adapt to these new technologies. But what I would say is, I think more and more... Especially as capabilities get better, but certainly even in studies say today, I think more and more things are going to shift towards this asynchronous flow. And one of the reasons I would say for that is in the real world, you're just capped by real world constraints. I think that one way to put it is kind of like... Don't take these numbers exactly, but it's kind of like the first order math of it is, of course, being able to write files or to complete this function or complete this line or things like that.

中文翻译:

老实说，作为一个社区，我认为全世界所有的工程师都必须思考这个问题，为此进行构建并适应这些新技术。但我想说的是，我认为越来越多……尤其是随着能力的提升，即使在今天的研究中，我也认为越来越多的事情会转向这种异步流。其中一个原因是，在现实世界中，你受限于现实世界的约束。一种说法是（不要太纠结于具体数字）：第一层逻辑当然是能够编写文件、完成这个函数或这一行代码。

[00:33:44] Scott Wu

English:

It helps a ton. It's a really great experience. There's a lot of parts of building software that obviously are almost not that at all, right? If you have a bug that you're trying to fix and so you spin up the local server, you click around on your own product on the front end and try to reproduce the bug yourself. Once you have the error, you take a look at Datadog and you see what happened and you try to find other errors in the logs. You look at those files and you see what went wrong, you make some edits, maybe you go and rerun the whole process again now that you make sure your change looks right. That's a lot of what it means to be a software engineer. These are processes that take real time. I think we're going to shift more and more towards this agentic workflow, because that's in some ways the way to really get to that 200%, 500%, 1000% gains that we'll be getting to with software engineering over the next few years.

中文翻译:

这很有帮助，体验也很好。但构建软件的很多部分显然几乎完全不是那样的，对吧？如果你要修复一个 bug，你会启动本地服务器，在前端点击自己的产品，尝试复现 bug。拿到错误信息后，你会查看 Datadog 看看发生了什么，尝试在日志中寻找其他错误。你查看那些文件，找出哪里出了问题，进行修改，然后可能还要重新运行整个流程，确保修改看起来是对的。这才是软件工程师工作的很大一部分。这些过程需要真实的时间。我认为我们将越来越多地转向这种智能体化工作流（agentic workflow），因为在某种程度上，这是在未来几年内让软件工程获得 200%、500% 甚至 1000% 收益的真正途径。

[00:34:39] Lenny Rachitsky

English:

Enough talk. Let's show people what the heck this actually looks like. You've got a couple demos prepped that show a few use cases that you found helpful. You're going to pull up your screen and then we'll kick it off and then we'll talk as it's happening.

中文翻译:

说得够多了。让我们给观众看看这到底是什么样子的。你准备了几个演示，展示了一些你认为有用的用例。你把屏幕投出来，我们开始演示，边看边聊。

[00:34:52] Scott Wu

English:

Yeah. The whole process obviously of working with Devin is working asynchronously. I thought it'd be cool for us to actually just watch Devin a little bit in action and then we can go through some other examples of work that Devin's done or things that Devin does for us, even on our team for example. But then we can check back in asynchronously with our Devin after.

中文翻译:

好的。与 Devin 协作的整个过程显然是异步的。我想我们可以先观察一下 Devin 的实际操作，然后看一些 Devin 完成的其他工作示例，或者它为我们团队做的事情。之后我们可以再异步地回来查看这个 Devin 的进度。

[00:35:12] Lenny Rachitsky

English:

Let's do it.

中文翻译:

开始吧。

[00:35:13] Scott Wu

English:

I'll share this real quick. The key thing that I would just emphasize here is a lot of it obviously is really just about thinking about as a software engineer or as engineers ourselves or engineering teams, PMs, and so on. What are the things that we would want to build that we would want to hand off? We have Devin set up with our own Devin code base, for example, and so I'll go ahead and kick off with Devin for that. I'll just say, "Devin, I'm on with my friend, Lenny."

中文翻译:

我很快分享一下。我想强调的关键点是，这很大程度上是关于作为软件工程师、工程团队或 PM，思考我们想要构建什么、想要交付什么。例如，我们已经为 Devin 设置了它自己的代码库，所以我现在就开始。我会说：“Devin，我正和我的朋友 Lenny 在一起。”

[00:35:45] Lenny Rachitsky

English:

Hi, Devin.

中文翻译:

嗨，Devin。

[00:35:48] Scott Wu

English:

"Can you modify Devin web app two?" Let's feature your newsletter as part of the Devin website.

中文翻译:

“你能修改 Devin web app 2 吗？” 让我们把你的时事通讯（newsletter）作为 Devin 网站的一部分展示出来。

[00:35:59] Lenny Rachitsky

English:

Let's do it. On the real Devin website.

中文翻译:

来吧。就在真正的 Devin 网站上。

[00:36:02] Scott Wu

English:

"Feature Lenny's site."

中文翻译:

“展示 Lenny 的网站。”

[00:36:04] Lenny Rachitsky

English:

Yeah, lose all your features.

中文翻译:

好啊，把你们的功能都删了换成我的。

[00:36:06] Scott Wu

English:

We're going to kick this off. As you can see, Devin gets started instantly and goes ahead and responds. Again, you can work with this asynchronously, you can work with it synchronously as well. For this, we'll just kind of go in a little bit and see exactly what's going on. But as you can see here, Devin's going through files and taking a look through a lot of stuff. We can follow here basically as we need to and see what makes sense. You can see Devin's already called out a few particular pieces where there's the sidebar, which we have implemented on the front end, and there's pieces there. We're going to have a new component and that component's going to link to Lenny's website. That all sounds good. Devin's asking us any questions if there's anything that we have here. Same story here where you can let Devin make its own decisions and hand off, or you can go ahead and give some more thoughts. Should the button open in a new tab or within an application? I'll say let's open it in a new tab.

中文翻译:

我们开始吧。如你所见，Devin 立即启动并做出了回应。同样，你可以异步工作，也可以同步工作。现在我们深入看一下到底发生了什么。你可以看到 Devin 正在浏览文件，查看很多东西。我们可以根据需要在这里跟进。你可以看到 Devin 已经找出了几个特定的部分，比如我们在前端实现的侧边栏。我们将添加一个新组件，该组件将链接到 Lenny 的网站。听起来不错。Devin 正在问我们是否有任何问题。同样，你可以让 Devin 自己做决定然后交付，也可以给出更多想法。比如“按钮应该在新标签页打开还是在应用内打开？”我会说在新标签页打开。

[00:37:03] Lenny Rachitsky

English:

And you could answer these at any point. Is it waiting for the answer?

中文翻译:

你可以在任何时候回答这些问题。它在等答案吗?

[00:37:06] Scott Wu

English:

You answer these at any point, you can hand off, hand back off.

中文翻译:

你随时可以回答，可以交付任务，也可以收回。

[00:37:07] Lenny Rachitsky

English:

It's not going to be like, "Just god damn it. I just wrote it this way. Why didn't you tell me earlier?"

中文翻译:

它不会说：“该死，我都写好了，你为什么不早点告诉我？”

[00:37:11] Scott Wu

English:

That's right. One of the big pieces with Devin is Devin will always be enthusiastic, will always be ready to put in the hours.

中文翻译:

没错。Devin 的一大特点是它总是充满热情，随时准备投入工作。

[00:37:21] Lenny Rachitsky

English:

Thanks for changing scope. Thanks, Scott.

中文翻译:

“谢谢你改需求，Scott。”（模仿 Devin 语气）

[00:37:24] Scott Wu

English:

We'll give Devin a chance to work and it's going to go through these files and it'll make a pull request for us and we'll see and go from there. But I thought it'd be fun to show some other examples of Devin in action as well. One of the examples actually this morning, which I just used Devin for, is I asked Devin to help me brush my own facts up for this podcast. Obviously, a huge fan of the podcast and the newsletter. I asked Devin, "Hey, Devin. I'm to be on the podcast. Could you please research everything you can about him and make a nice website quiz for me so that I can make sure I know my facts?" This was just this morning, I asked Devin to do this and I'll just show what Devin did, it looks like. Went to Wikipedia first. Unfortunately, it's not a page in Wikipedia, which is... Lenny, we will work on that.

中文翻译:

我们给 Devin 一点时间工作，它会浏览这些文件并为我们创建一个 PR。但我想展示一些 Devin 实际操作的其他例子。今天早上我刚用过一个：我让 Devin 帮我温习一下关于这个播客的背景知识。我是这个播客和时事通讯的忠实粉丝。我问 Devin：“嘿 Devin，我要上这个播客了。你能帮我搜集关于他的所有信息，并为我做一个漂亮的网页测验，好让我确保掌握了背景事实吗？”就在今天早上，我让 Devin 做了这件事，我给你们看看它做了什么。它先去了维基百科。不幸的是，维基百科上没有你的页面……Lenny，我们会解决这个问题的。

[00:38:09] Lenny Rachitsky

English:

I'm not a big enough deal yet.

中文翻译:

我还不够出名。

[00:38:12] Scott Wu

English:

They did you dirty. I mean, we need a page list. And so then it went and found it on Spotify.

中文翻译:

他们对你太不公平了。我们需要一个页面列表。然后它在 Spotify 上找到了。

[00:38:19] Lenny Rachitsky

English:

So you're watching what it's researching live?

中文翻译:

所以你是在实时看它研究的内容?

[00:38:21] Scott Wu

English:

Yeah, yeah, yeah. This was this morning, obviously.

中文翻译:

是的，这是今天早上的。

[00:38:24] Lenny Rachitsky

English:

This is a playback of what Devin did. This is part of Devin you could just watch what it did.

中文翻译:

这是 Devin 操作的回放。这是 Devin 的一部分功能，你可以直接看它做了什么。

[00:38:29] Scott Wu

English:

Yeah, yeah. Especially when you're building engineering products or something like that, you can see each of the steps that Devin was doing, or if Devin tested the code locally, obviously you want to be able to go and look and see what Devin was cooking around with and testing or things like that. It found the newsletter. It's going and looking at this and it's going and reading all of this. And then it says, "Okay, let's get started with putting the code together." It says, "Hey, I've researched." It's going through and writing all of this, putting the app together. It plays its own quiz itself. Actually, we should just play this quiz actually. Let's see how much I know.

中文翻译:

是的。特别是当你构建工程产品时，你可以看到 Devin 执行的每一步。如果 Devin 在本地测试了代码，你显然想去看看它在捣鼓什么。它找到了时事通讯，正在阅读。然后它说：“好，让我们开始编写代码。”它说：“嘿，我已经研究过了。”它正在编写代码，构建应用。它甚至自己玩了一下那个测验。实际上，我们也应该玩一下，看看我知道多少。

[00:39:05] Lenny Rachitsky

English:

[inaudible 00:39:05]

中文翻译:

（听不清）

[00:39:06] Scott Wu

English:

What is the name?

中文翻译:

名字是什么?

[00:39:08] Lenny Rachitsky

English:

What is the name of the podcast? Lenny's Podcast. For people not watching, so to say, approximately how many subscribers? A million. Very good.

中文翻译:

播客的名字是什么？Lenny's Podcast。对于没在看视频的人，大约有多少订阅者？一百万。非常好。

[00:39:17] Scott Wu

English:

Yeah. Yeah.

中文翻译:

是的。

[00:39:18] Lenny Rachitsky

English:

What are three main topics Lenny is focused on? Oh, product, growth, and career. Very good. It's a good quiz [inaudible 00:39:25] of people.

中文翻译:

Lenny 关注的三个主要话题是什么？哦，产品、增长和职业。非常好。这是一个很好的测验。

[00:39:25] Scott Wu

English:

What does [inaudible 00:39:26] besides podcasting?

中文翻译:

除了做播客，他还做什么?

[00:39:27] Lenny Rachitsky

English:

What does Lenny do besides podcasting?

中文翻译:

除了做播客，Lenny 还做什么?

[00:39:30] Scott Wu

English:

I'd say writing, investing, and advising. How often does Lenny-

中文翻译:

我会说写作、投资和咨询。Lenny 多久——

[00:39:35] Lenny Rachitsky

English:

[inaudible 00:39:35]

中文翻译:

（听不清）

[00:39:35] Scott Wu

English:

Once a week, right?

中文翻译:

每周一次，对吧?

[00:39:36] Lenny Rachitsky

English:

Once a week.

中文翻译:

每周一次。

[00:39:37] Scott Wu

English:

Yeah. We can go through all these and do all these. I took this quiz, by the way, obviously, to make sure that I was well-prepped, but this is kind of one of the more fun examples, obviously, of [inaudible 00:39:46]

中文翻译:

是的。我们可以把这些都过一遍。顺便说一下，我显然做过这个测验，以确保我准备充分。这是最有趣的例子之一。

[00:39:46] Lenny Rachitsky

English:

Scott, how many subscribers do I have at my newsletter?

中文翻译:

Scott，我的时事通讯有多少订阅者?

[00:39:50] Scott Wu

English:

Over a million, actually.

中文翻译:

实际上超过一百万。

[00:39:52] Lenny Rachitsky

English:

[inaudible 00:39:52]

中文翻译:

（听不清）

[00:39:53] Scott Wu

English:

And then one last one I'll show and then maybe we can come back to our initial run after. Like I was saying, a lot of this is really built to work with all of the existing code workflows out there. For example, we were doing some exploration with the DeepSeek repository on GitHub and we imported it into Devin and we got our own fork of it set up in Devin. A couple of things I just wanted to show here. One is Devin sets up its whole wiki with all of its internal understanding. When Devin indexes the code base, obviously, building a representation of the code base and learning it and improving it over time is one of the big things that Devin does. Funnily enough, we found that naturally humans really are interested to understand this code base representation as well.

中文翻译:

最后我再展示一个，然后我们再回到最初的那个任务。正如我所说，这很大程度上是为了与现有的所有代码工作流协作。例如，我们正在探索 GitHub 上的 DeepSeek 仓库，我们将其导入 Devin，并在 Devin 中设置了我们自己的 fork。我想展示几点：一是 Devin 会根据其内部理解建立完整的 Wiki。当 Devin 索引代码库时，构建代码库的表示、学习并随时间改进它是 Devin 的核心工作之一。有趣的是，我们发现人类自然也对理解这种代码库表示非常感兴趣。

[00:40:38] Scott Wu

English:

Devin Wiki is something that we built here and you can take a look at all these different pieces and see each of these different things. Here are the FP8 operations, here's an SG [inaudible 00:40:46] integration. There's diagrams of how the different layers are built and put together. There's deployment operations. There's a lot of details about the architecture as well. You can ask questions about it as well. For example, you can say, "How does DeepSeek handle multi-token prediction designed for spec deck?" It'll go through and it's able to search through the entire code base and give you an informed answer based off of that.

中文翻译:

Devin Wiki 是我们构建的功能，你可以查看所有这些不同的部分。这里有 FP8 操作，这里有 SG 集成。还有展示不同层如何构建和组合的图表。还有部署操作。关于架构也有很多细节。你也可以提问，例如："DeepSeek 如何处理为 spec deck 设计的多 token 预测？" 它会搜索整个代码库，并据此给出专业的回答。

[00:41:10] Scott Wu

English:

We use this a lot. It helps when you're scoping out a task for Devin and doing an initial prompt. It also helps, obviously just in a vacuum, you often have questions about your code base that are really nice.

中文翻译:

我们经常使用这个功能。当你为 Devin 确定任务范围并编写初始提示词时，它非常有帮助。显然，即使在平时，你经常会有关于代码库的问题，这非常棒。

[00:41:20] Lenny Rachitsky

English:

This episode is brought to you by Attio, the AI-native CRM. Attio is built to scale with your business from day one. Connect your email and calendar and Attio instantly builds a CRM that matches your business model, with all of your company's Contacts and interactions enriched with actionable insights. Sync in your product's usage, billing info, or any other data sources and Attio's flexible data model will handle it all without any rigid templates or workarounds.

中文翻译:

本集由 AI 原生 CRM Attio 赞助。Attio 旨在从第一天起就随你的业务规模而扩展。连接你的电子邮件和日历，Attio 会立即构建一个符合你业务模式的 CRM，并利用可操作的洞察丰富你公司的所有联系人和互动。同步你的产品使用情况、账单信息或任何其他数据源，Attio 灵活的数据模型将处理这一切，无需任何僵化的模板或变通方案。

[00:41:49] Lenny Rachitsky

English:

With Attio, AI isn't just a feature. It's the foundation. You can do things like instantly prospect and route leads with research agents, get real-time insights from AI using customer conversations, and build powerful AI automations for your most complex workflows. Industry leaders like Flatfile, Replicate, and Modal are already experiencing what's next for CRM. Go to A-T-T-I-O.com/lenny to get 15% off your first year. That's A-T-T-I-O.com/lenny.

中文翻译:

在 Attio，AI 不仅仅是一个功能，它是基础。你可以通过研究智能体立即寻找并路由潜在客户，利用 AI 从客户对话中获取实时洞察，并为最复杂的工作流构建强大的 AI 自动化。Flatfile、Replicate 和 Modal 等行业领导者已经在体验 CRM 的未来。访问 attio.com/lenny 即可享受首年 15% 的折扣。网址是 attio.com/lenny。

[00:42:21] Lenny Rachitsky

English:

Something that I've learned as I've been talking to more and more AI building companies and apps is there's a big difference in how large of a code base they could integrate into. That's a big deal for companies that are existing versus startups, people that have large existing code base. How should people think about what kind of code base Devin can plug into?

中文翻译:

在与越来越多的 AI 公司和应用交流时，我学到的一点是，它们能集成的代码库大小有很大差异。对于拥有大型现有代码库的成熟公司（相对于初创公司）来说，这是一个大问题。人们应该如何看待 Devin 可以接入什么样的代码库?

[00:42:41] Scott Wu

English:

Yeah. We go all the way to the biggest code base as possible. One way I'd kind of put it is how... The way that we as engineers would think about a large code base is certainly when you're making changes or when you're thinking about a particular task. You're not bringing in every single line of the code base at once. You have a high level of traction that you're able to think about and look into and then you're obviously able to zoom in and get to higher resolution on each of these different things. Devin works in much the same way, where the first thing it'll do is it's going to figure out the high level architecture of what's going on here and what this is built for and so on. But within each of the components, it's obviously also going to be able to zoom in and give some more detail about each of these. Here's FP8 to be float 16 and how exactly a lot of that is set up. Here's each of the different parts of the code base. Similarly, we built this to be scalable.

中文翻译:

是的。我们支持尽可能大的代码库。我的一种解释方式是：作为工程师，我们在修改代码或思考特定任务时，并不会一次性把代码库的每一行都装进脑子里。你会先从高层次的抽象开始思考，然后显然能够放大并对每一个不同的部分获得更高的分辨率。Devin 的工作方式非常相似：它首先会弄清楚这里发生了什么、这是为了什么而构建的高层架构等等。但在每个组件内部，它显然也能放大并提供更多细节。比如 FP8 如何转为 float 16，以及具体是如何设置的。这是代码库的各个部分。同样，我们将此设计为可扩展的。

[00:43:38] Lenny Rachitsky

English:

It's essentially coming back to the engineer as architect. Now, it's helping you understand the architecture, kind of circling back to that.

中文翻译:

这本质上又回到了“工程师作为架构师”这一点。现在，它在帮助你理解架构，算是绕回来了。

[00:43:46] Scott Wu

English:

Yeah. Yeah, exactly. One of the fun use cases that we've seen actually with folks is they'll often actually get Devin's help to onboard new engineers on the team. When you're new and you're joining, there's obviously a lot of questions that you have about the code base or about how things are set up. It also sometimes can be a little bit awkward to ask your mentor or your manager the questions and if you're worried that they're going to be really dumb questions. It's nice to just be able to ask Devin and to go through Devin's wiki and to understand these internal representations.

中文翻译:

是的，没错。我们看到的一个有趣的用例是，人们经常让 Devin 帮助新工程师入职。当你刚加入团队时，显然会对代码库或设置有很多疑问。有时向导师或经理提问会有点尴尬，尤其是当你担心那是些“愚蠢的问题”时。能直接问 Devin，查看 Devin 的 Wiki 并理解这些内部表示，真的很棒。

[00:44:15] Lenny Rachitsky

English:

I think that's really interesting, because it comes back to your point that Devin is not just a junior engineer. It's what you call a jagged engineer.

中文翻译:

我觉得这很有趣，因为它回到了你之前的观点：Devin 不仅仅是一个初级工程师。它是你所说的“参差不齐的工程师”。

[00:44:21] Scott Wu

English:

A jagged intelligence.

中文翻译:

参差不齐的智能。

[00:44:22] Lenny Rachitsky

English:

A jagged intelligence, where it's almost like a staff engineer at understanding the code base. Usually, you have to ask an engineer that's been there a long time, " What does this do? Where is this thing? How does this work?" It feels like Devin's very good at that.

中文翻译:

参差不齐的智能。在理解代码库方面，它几乎像个主任工程师（staff engineer）。通常你得问老员工：“这个是干什么的？那个在哪？这怎么运行？”感觉 Devin 非常擅长这个。

[00:44:34] Scott Wu

English:

Yeah. Yeah. Obviously, the retrieval and processing a lot of code and a lot of tokens at once is something that language models are really great at. Basically being able to get those gains in the places that you need them is really great. Yeah. Sweet, cool.

中文翻译:

是的。显然，一次性检索和处理大量代码和 token 是语言模型非常擅长的。在需要的地方获得这些收益真的很棒。太酷了。

[00:44:50] Lenny Rachitsky

English:

All right. You got a couple more use?

中文翻译:

好的，你还有几个用例?

[00:44:51] Scott Wu

English:

Yeah. One last thought I'll show is just... We just rolled this out last week actually, but it's a full Devin automation setup with Linear. If you have tasks that you're doing on the DeepSeek repository, for example, and it's all set up, all you have to do is you just add the Devin label and Devin will come through and it'll give you this. It's going to give you its thoughts on what the tasks looks like and you can take a look at each of the particular files that you see, or it'll point out snippets that it thinks are important. From there, if you feel good about what was built or the conclusions that we came to, then you can just start off the Devin session that will go and actually do that work.

中文翻译:

是的。最后展示一个……我们上周刚推出的功能：Devin 与 Linear 的全自动化设置。例如，如果你在 DeepSeek 仓库上有任务，设置好后，你只需添加“Devin”标签，Devin 就会介入。它会给出它对任务的看法，你可以查看它识别出的特定文件，或者它认为重要的代码片段。如果你觉得它的分析或结论没问题，就可以启动 Devin 会话去实际执行工作。

[00:45:30] Lenny Rachitsky

English:

That is insane. That sounds like such a simple idea, but essentially what you're saying is there are tasks in Linear that are fixes and features and now Devin just goes off and can just do them for you.

中文翻译:

太疯狂了。这听起来是个很简单的点子，但本质上你是说，Linear 里的那些修复和功能任务，现在 Devin 可以直接去为你完成了。

[00:45:44] Scott Wu

English:

Yeah. Definitely it's a hands-on process. You certainly want to be involved when Devin is scoping out the task or giving you its thoughts. The nice thing, too, by the way, is Devin will give you its confidence level.

Here's how likely I am to really understand this piece or that piece or whatever, but it helps make things a lot faster. To your point, a lot of product managers, for example, obviously love to be able to use Devin in Linear to understand the code base better or things like that. Claire Vo, for example, from LaunchDarkly is a big Devin user and she loves basically going and scoping out tasks or asking data questions or asking, "Hey, what's going on? Or is this merged into production yet? Or is this a feature flag right now? Or what percent of people are getting this or that feature?" It's a clean way basically to make that intelligence much more accessible.

中文翻译:

是的。这肯定是一个需要参与的过程。当 Devin 确定任务范围或给出想法时，你肯定想参与其中。顺便说一下，Devin 还会给出它的置信度——“我有多大把握理解这一块或那一块”。这有助于加快进度。正如你所说，很多产品经理非常喜欢在 Linear 中使用 Devin 来更好地理解代码库。例如 LaunchDarkly 的 Claire Vo 就是 Devin 的重度用户，她喜欢用它来确定任务范围、询问数据问题，或者问：“嘿，进度如何？合并到生产环境了吗？现在有功能开关（feature flag）吗？百分之几的用户能看到这个功能？”这是一种让智能变得触手可及的简洁方式。

[00:46:40] Lenny Rachitsky

English:

I love, just with the integration with Linear, that you can still keep it really simple. You add a little ticket like, "Hey, this link to this home page, do this," and Devin will be really good at understanding what you mean and then show you, "Here's what I'm thinking." Is this right?

中文翻译:

我喜欢这种与 Linear 的集成，它依然保持了极简。你添加一个小票据（ticket），比如“嘿，把主页的这个链接改一下”，Devin 就能很好地理解你的意思，然后展示：“这是我的想法，对吗？”

[00:46:53] Scott Wu

English:

Yeah. Yeah, cool. Okay. Yeah, Devin did finish working. It seems like there's something going on with the CI and it's debugging that right now, but it went ahead and put up the initial first pass pull request and we can take a look.

中文翻译:

是的，很酷。好，Devin 完成工作了。看起来 CI 出了点问题，它正在调试，但它已经提交了第一版 PR，我们可以看看。

[00:47:04] Lenny Rachitsky

English:

Let's do it.

中文翻译:

看看吧。

[00:47:06] Scott Wu

English:

This is the Devin website, obviously, in this custom deploy and we have Lenny's newsletter right here.

中文翻译:

这是 Devin 的网站，在这个自定义部署中，Lenny 的时事通讯就在这里。

[00:47:11] Lenny Rachitsky

English:

Let's ship this to production. We won't be so confused.

中文翻译:

把它发布到生产环境吧，这样我们就不会困惑了。

[00:47:15] Scott Wu

English:

Yeah.

中文翻译:

好的。

[00:47:16] Lenny Rachitsky

English:

That's amazing. Okay. Show it again real quick. Just added it to the home page of Devin.

中文翻译:

太神奇了。好，再快展示一下。刚把它加到 Devin 的主页上。

[00:47:22] Scott Wu

English:

Yeah. Devin, obviously, has access to our Devin code base. It does a lot here and so it's super familiar with all the pieces here.

中文翻译:

是的。Devin 显然有权访问我们自己的代码库。它在这里做了很多工作，所以对这里的每一块都非常熟悉。

[00:47:28] Lenny Rachitsky

English:

Beautiful.

中文翻译:

漂亮。

[00:47:29] Scott Wu

English:

Yeah, I like how that looks. We've got Devin search, we got Devin [inaudible 00:47:33], and we've got Lenny's newsletter.

中文翻译:

是的，我喜欢这个样子。我们有 Devin 搜索，有 Devin Wiki，还有 Lenny 的时事通讯。

[00:47:34] Lenny Rachitsky

English:

[inaudible 00:47:34]. You link to my site. We'll get some PageRank going.

中文翻译:

你链接到我的网站，我们能增加点 PageRank（网页排名）。

[00:47:37] Scott Wu

English:

Yeah, yeah, yeah.

中文翻译:

哈哈，是的。

[00:47:39] Lenny Rachitsky

English:

Okay. Is that a good example? Oh, there it is. What a beautiful website for my newsletter. Is that just a good example of the kind of thing Devin is very good at like, "Here's a very specific thing to change on the website"? How does your people think about what Devin is very good at and maybe where it starts to fall apart?

中文翻译:

好，这是一个好例子吗？哦，在那。我的时事通讯网站真漂亮。这是否就是 Devin 非常擅长的事情的一个好例子，比如“修改网站上一个非常具体的东西”？人们应该如何看待 Devin 擅长什么，以及它在什么情况下会开始失效？

[00:47:55] Scott Wu

English:

The way that we often describe it is, I think, Devin is best when it is working on tasks that are well-defined. One way to put it, you might...

中文翻译:

我们经常这样描述：我认为当 Devin 处理定义明确的任务时表现最好。一种说法是，你可能……

[00:48:00] Scott Wu

English:

... on tasks that are well-defined. One way to put it is, you want to be giving Devin tasks, not problems. And a lot of these things like what you just saw, which was kind of like a quick front-end feature request or a bug fix or adding testing and documentation or things like that.

中文翻译:

……处理定义明确的任务。换句话说，你应该给 Devin “任务”而不是“问题”。很多事情就像你刚才看到的，比如快速的前端功能请求、bug 修复，或者添加测试和文档之类的。

[00:48:16] Scott Wu

English:

One of the things that makes a loop really nice obviously is a quick way to iterate and test. And so with something like this, obviously super easy for us, for example, to just go pull up the preview and see that the link worked. Obviously it would be easy for Devin to do as well. Devin will often go and log in to Devin and start a Devin session and make sure when it's working on our code base, which is kind of hilarious. But yeah, you generally want something that is kind of easy to verify and easy to test is the main thing. And you can work on bigger projects or bigger asks as well, obviously. But in that case you should certainly expect to need to steer Devin more to make sure it's going the right direction.

中文翻译:

让这个闭环非常高效的一个因素是快速迭代和测试的能力。对于这类任务，我们显然很容易拉出预览并查看链接是否有效。对 Devin 来说显然也很容易。Devin 经常会登录 Devin 并启动一个 Devin 会话，以确保它在我们的代码库上工作正常，这挺搞笑的。但总的来说，最核心的是你希望任务是易于验证和测试的。当然，你也可以让它处理更大的项目或需求，但在那种情况下，你肯定需要更多地引导 Devin，以确保它朝着正确的方向前进。

[00:48:55] Lenny Rachitsky

English:

It's interesting because that's very similar to the way people talk about synthetic data and reinforcement learning, creating data that's very easy. There's a very definitive answer, yes and no.

中文翻译:

这很有趣，因为这与人们谈论合成数据和强化学习的方式非常相似——创建非常容易判断的数据，有一个非常明确的答案，是或否。

[00:49:04] Scott Wu

English:

Yeah.

中文翻译:

是的。

[00:49:04] Lenny Rachitsky

English:

It's very clear.

中文翻译:

非常清晰。

[00:49:06] Scott Wu

English:

Yeah.

中文翻译:

是的。

[00:49:07] Lenny Rachitsky

English:

Okay. Let me ask you this question. What's something that you guys debated a lot as you were designing and building Devin?

中文翻译:

好。让我问你这个问题：在设计和构建 Devin 的过程中，你们争论最多的是什么？

[00:49:15] Scott Wu

English:

I'll give a couple that comes to mind. One I would say is a question of, I'll call it how opinionated we should be. We had the workflows that we used to Devin for, which was very much as you can see for basically integrating to our Slack and GitHub making pull requests for us in our repos, responding to issue reports or things like that. And naturally we've had certainly a lot of other different things that have come up that folks have tried. I mean, we have folks who order their DoorDash with Devin, for example. Even we have folks, certainly a lot of people who are building cool websites from scratch or working on things like that.

中文翻译:

我想到了几个。一个是关于“我们的产品应该有多强的倾向性（opinionated）”。我们使用 Devin 的工作流非常明确，就像你看到的：集成到 Slack 和 GitHub，在仓库中提交 PR，响应 issue 报告等。但自然地，我们也看

到人们尝试了很多其他事情。比如，有人用 Devin 点 DoorDash 外卖。还有很多人用它从零开始构建酷炫的网站。

[00:49:53] Scott Wu

English:

Yeah, I mean it is been an interesting trade off for us where I think the way that I would describe it is in our product, certainly the large bulk of the features that we build are for this kind of making pull requests and engineering teams use case. But I think basically our kind of general stance with all the others is obviously if folks want to use Devin for that, that's great and we want to just make sure that they're fully aware about the limitations and about where things can get caught up.

中文翻译:

这对我们来说是一个有趣的权衡。我会这样描述：在我们的产品中，我们构建的大部分功能确实是针对“提交 PR”和“工程团队协作”这一用例的。但对于其他用例，我们的基本立场是：如果人们想用 Devin 做那些事，那太棒了，我们只想确保他们充分了解局限性以及可能卡壳的地方。

[00:50:20] Scott Wu

English:

It is funny with AI and especially because I would say one of, I would say the most common pieces of advice out there I would say is focus on a really niche cohort. Do things that don't scale, make one use case that's really great and then you grow from there. And I think that's great advice across the board. But yeah, it's kind of interesting because I think with generative AI, you naturally see this where a lot of product experiences can turn out to be more general. And so it's an interesting trade-off for us. This is something that we still always go back and forth on and how much do we want to do more to support all the other kind of use cases out there to handle other things that folks might want to do with Devin.

中文翻译:

AI 很有趣，尤其是因为最常见的建议通常是“专注于一个非常细分的群体”、“做那些无法规模化的事”、“做好一个用例然后再扩展”。我认为这在各行各业都是很好的建议。但有趣的是，在生成式 AI 领域，你自然会看到很多产品体验最终变得非常通用。所以这对我们来说是一个有趣的权衡。我们仍然在反复权衡：我们要在多大程度上支持所有其他类型的用例，去处理人们可能想用 Devin 做的其他事情。

[00:51:03] Scott Wu

English:

I think another one that comes to mind is how much Devin should be, let's say a single comprehensive project experience versus a suite of tools. And as you can see here, we have Devin search, we have Devin Wiki, we have the linear ticket scooping, and certainly these tools interact with each other, but I think as time has gone on, we've seen it more and more as really building this suite of tools. And I think the core agent experience and the core kind of agent that will go off and build each of these build things for you, for example, is always of course going to be that's Devin, and that is the core piece. I think that will always be what's really special about our work. But I think that all of the other features out there, there is a complex suite of work that's required for real-world software sharing and engineering is just messy at the end of the day.

中文翻译:

我想到的另一个争论是：Devin 应该是一个单一的综合项目体验，还是一个工具套件？如你所见，我们有 Devin 搜索、Devin Wiki、Linear 票据分析，这些工具当然会互相交互。但随着时间的推移，我们越来越倾向于将其视为构建一个工具套件。我认为核心的智能体体验——那个会去为你构建东西的智能体——永远是 Devin，那是核心。我认为那永远是我们工作中最特别的部分。但除此之外的所有其他功能，都是为了应对现实世界软件协作所需的复杂工作，毕竟工程工作归根结底是很繁杂的。

[00:51:55] Scott Wu

English:

And so I think there are a lot of different flows and a lot of different use cases that makes sense. And an obvious thing to point out is you could ask the same questions to Devin search as you could to Devin and Devin will go through and it'll do the same thing. It'll go through and look through the files and give you an answer and stuff.

中文翻译:

所以我认为有很多不同的流程和用例是有意义的。一个显而易见的事实是，你可以向 Devin 搜索提问，也可以向 Devin 提问，Devin 都会执行相同的操作：浏览文件并给出答案。

[00:52:10] Scott Wu

English:

But with that said, on the one hand, I think on the capability side, there's certainly a lot you can do to really optimize things for very specifically question answer about this repository and that made sense to really build into a specific kind feature.

中文翻译:

但话虽如此，一方面在能力端，你肯定可以做很多事情来专门针对"关于这个仓库的问答"进行优化，这使得将其构建为一个特定功能变得很有意义。

[00:52:23] Scott Wu

English:

And then on the other side, I would say we found that users actually really, really like having this access of control. Sometimes you have a task that you're thinking about, but you actually don't want Devin to get started on the task just yet. You want to ask Devin and understand what parts of the code base might be relevant. And so you want to be very direct about saying, "This is just an ask and I just want to see the snippets of the code base that relate," or, "I just want to look at the Wiki and understand the existing representation."

中文翻译:

另一方面，我们发现用户其实非常喜欢拥有这种控制权。有时你正在思考一个任务，但你其实还不想让 Devin 立即开始执行。你想先问问 Devin，了解代码库的哪些部分可能相关。所以你想非常直接地说："这只是一个询问，我只想看看相关的代码片段"，或者"我只想看看 Wiki，了解现有的表示方式"。

[00:52:50] Scott Wu

English:

And so it's on both the capabilities and on the UX side, we've found that that's kind of what's naturally made sense over time.

中文翻译:

所以无论是在能力端还是在 UX（用户体验）端，我们发现随着时间的推移，这种方式自然而然地变得合理了。

[00:52:58] Lenny Rachitsky

English:

Well, let's talk about the landscape then of just other companies in the space, which is something a lot of people are always thinking about. There's all these different approaches. You guys are going full on AI engineer, there's obviously ID companies. There's also just models being built that are really good at engineering. Everyone's kind of starting to build agents now. You guys are ahead on this in a lot of ways. OpenAI just recently said they're going to build a software engineering agent. Anthropics got something there. Cursor and Windsurf have their only agents and Replit. Thoughts on just where you guys fit in the landscape and then how do you think you win long term. How do you think about that?

中文翻译:

那我们来聊聊这个领域的竞争格局，这是很多人一直在思考的问题。这里有很多不同的方法：你们是全力投入 AI 工程师，还有 IDE 公司，还有专门为工程设计的模型。现在每个人都开始构建智能体了。你们在很多方面都处于领先地位。OpenAI 最近刚说他们要构建软件工程智能体，Anthropic 也有动作，Cursor 和 Windsurf 也有自己的智能体，还有 Replit。你觉得你们在这个格局中处于什么位置？长期来看你们如何胜出？你是怎么考虑的?

[00:53:31] Scott Wu

English:

Yeah, and for what it's worth, I think all of these are incredible teams. I think really smart and really forward-thinking folks who are building a lot of great products out there. And I think there's a lot to do honestly over the next few years with the advent of AGI or whatever you want to call it. I think one of the quotes that I love is in 2017, if you asked if we had AGI, the answer is no. And in 2025, if you ask if we have AGI, the answer is, "Well, you have to define AGI. And it depends on your subject." And I think it does get to the point of, I mean there is a lot of really amazing stuff happening. I think that it's easy to underrate, I would say just how big of a shift it is that we're seeing where I think there are a lot of great products out there, for example, over the last 10 years, 20 years, 30 years, that have made each of these different niches of the life cycle of building a product a little bit easier, for example, right?

中文翻译:

是的，我认为这些都是非常了不起的团队。他们是非常聪明、非常有前瞻性的人，正在构建很多伟大的产品。老实说，随着 AGI（通用人工智能）或随便你怎么称呼它的到来，未来几年有很多事情要做。我喜欢的一句话是：在 2017 年，如果你问我们是否有 AGI，答案是否定的；而在 2025 年，如果你问我们是否有 AGI，答案是“那得看你如何定义 AGI，以及你的评判标准”。我的意思是，现在有很多令人惊叹的事情正在发生。我认为人们很容易低估我们正在经历的这场变革有多么巨大。在过去的 10 年、20 年、30 年里，有很多优秀的产品让产品构建生命周期的各个环节变得容易了一点，对吧?

[00:54:28] Scott Wu

English:

There's great products out there for instant response, there's great products out there for logging, there's great products out there for billing, there's all of these different tools. And the obvious thing is what we're seeing with AI is all of these spaces are going to be moving multiple times faster and it's going to be like an order of magnitude shift, if anything.

中文翻译:

有优秀的事件响应产品、日志产品、计费产品，有各种各样的工具。显而易见的是，通过 AI，所有这些领域都将以数倍的速度向前推进，如果说有什么变化的话，那将是数量级的转变。

[00:54:47] Scott Wu

English:

And so I think from our perspective, we've obviously had a very specific lens that we've bet on this whole time, and that is autonomous coding agents. And there's a lot of problems to solve there, to be honest, right? There's still a ton to do on the core capabilities, certainly, and we see cases all the time where it's like, wow, why did Devin make that decision? That seems, no human engineer would've ever done that. There's all sorts of spots where, with the product interface, there's obviously a lot to think about.

中文翻译:

所以从我们的角度来看，我们一直押注在一个非常具体的视角上，那就是"自主编程智能体"。老实说，那里有很多问题需要解决。在核心能力方面当然还有很多工作要做，我们经常看到一些案例，心想："哇，Devin 为什么会做那个决定？这看起来没有任何人类工程师会那样做。"在产品界面方面，显然也有很多需要思考的地方。

[00:55:16] Scott Wu

English:

And I think it's by the way, not just a single thing that we're working towards, but something that will change with every edition of capabilities. I kind of think of it as there's 20 generations of agent product, agent coding experiences to come. I think the one that we'll get to over the course of several years is probably something where you don't even look at the code at all, right? And you're actually just looking at your own product and you're just able to look and specify and say, "Hey, this button should be a little bit rounder. Let's do that. And by the way, let's add a new tab here and maybe we should save this information. Let's start up a database table and let's index it on X, Y, and Z columns." And you're just basically working with your products in real time and having your agent build out those things for you. Obviously there's going to be a lot of generations in between here and there, but I think the product experience itself is going to change every single time.

中文翻译:

顺便说一下，我认为我们努力的目标不是单一的，而是会随着能力的每一次提升而改变。我把它看作未来会有 20 代智能体产品和智能体编程体验。我想几年后我们会达到的状态可能是：你根本不需要看代码。你只需看着自己的产品，进行指定并说："嘿，这个按钮应该再圆一点。就这样做。顺便说一下，在这里加一个新标签页，也许我们应该保存这些信息。启动一个数据库表，并在 X、Y、Z 列上建立索引。"你基本上是在实时操作你的产品，让你的智能体为你构建这些东西。显然，从现在到那时会有很多代产品的更迭，但我认为产品体验本身每一次都会发生变化。

[00:56:04] Scott Wu

English:

And then obviously there, there's all of the practicality of just getting it out there in the world. And so folks obviously need to learn how to use the new technology. There's a lot to do to deploy into all of the messiness of real world software. There's a lot of COBOL out there still. There's a lot of FORTRAN out there still. There's lots of kind of abstractions and details that folks have done.

中文翻译:

然后显然还有将其推向世界的实际问题。人们显然需要学习如何使用新技术。要将其部署到现实世界杂乱无章的软件中，还有很多工作要做。现在外面还有很多 COBOL 代码，很多 FORTRAN 代码。还有人们建立的各种抽象和细节。

[00:56:27] Scott Wu

English:

And so I think from our perspective, we have since the beginning have been laser-focused on agentic coding, and that is the one thing that we've really believed in. It's the one thing that we've designed for and that goes all the way to even the revenue model with ACUs and having the usage-based setup. It does into obviously all the product experiences of thinking how, okay, where do you want to talk to Devin? You want to be able to talk to Devin in Slack, you want to be able to spin this up from your issue. You want to be able to all of these things and then of course the capabilities.

中文翻译:

所以从我们的角度来看，我们从一开始就专注于“智能体化编程”，那是我们真正相信的一件事。那是我们设计的核心，甚至延伸到了我们的收入模式（ACU，按使用量计费）。这显然也体现在所有的产品体验中：你想在哪里与 Devin 交流？你想在 Slack 中与它交谈，你想从 issue 中启动它。你想实现所有这些功能，当然还有核心能力。

[00:56:58] Scott Wu

English:

And so I don't think there's any one easy answer. I think it's obviously a combination of things, but this has been the space that we've lived in and spent all of our time in for the last year and a half, and it's going to be that way for the next five or 10 years too.

中文翻译:

所以我认为没有一个简单的答案。这显然是多种因素的结合，但这是我们过去一年半以来一直生活并投入全部时间的领域，未来五到十年也会是这样。

[00:57:14] Lenny Rachitsky

English:

Along these lines, a big question everyone always has in AI's moats and defensibility, it's a question I've been asking every founder that comes on. How do you just think about how to build a moat in this space when it's so much easier to build and so much is built on these models that are themselves advancing so quickly?

中文翻译:

沿着这个思路，每个人对 AI 都有一个大疑问，那就是“护城河”和“防御性”。这是我问每一位来访创始人的问题：当构建变得如此容易，且如此多的东西都建立在这些自身进步神速的模型之上时，你如何思考在这个领域建立护城河?

[00:57:30] Scott Wu

English:

I'd give one slight tweak on that, which is I think it's often less about moats and more about stickiness. And what I mean by that is moats are in some sense, typically what folks mean by moats is something that means that a competitor couldn't even enter the market. I agree that at a high level, a lot of different folks at different layers of the AI spectrum, the foundation labs or the application layer or so on, I don't think there's any kind of hard barrier that would prevent others from entering. I think what does exist is stickiness, which I would kind of define as once you have a product experience that you really like, are you excited to keep using that experience or is there an effect where it is just as easy from now on to just switch on to a new one and learn a new one and so on.

中文翻译:

我想对这个问题做一个微调：我认为这通常与其说是关于“护城河”，不如说是关于“粘性”。我的意思是，护城河在某种意义上通常是指竞争对手甚至无法进入市场。我同意从高层次来看，在 AI 频谱的不同层级（无论是基础模型实验室还是应用层等），我不认为存在任何阻止他人进入的硬性障碍。我认为真正存在的是“粘性”，我将其定义为：一旦你拥有了真正喜欢的产品体验，你是否愿意继续使用它？还是说从现在起切换到一个新产品、学习一套新东西也同样容易?

[00:58:15] Scott Wu

English:

And I think from that perspective, I think there's a few things that are really great about coding agents in particular. One I would say is there is a lot of just inherent stickiness and learning and buildup over time, which is that as you use Devin and as your whole team uses Devin, it's the same thing with an engineer. If you're joining on day one versus you've been at the company for five years, you wrote half the code yourself, you've touched every file you've built every single piece, you know all the engineers. And so similarly, it's like Devin will really learn and build its representation of your code base and of your stack and of your process over time and will be able to do a lot more with that.

中文翻译:

从这个角度来看，我认为编程智能体有几点非常棒。首先，它具有内在的粘性，以及随时间推移的学习和积累。当你和你的整个团队使用 Devin 时，这和人类工程师是一样的。第一天入职和在公司待了五年是不一样的——后者自己写了一半的代码，接触过每一个文件，构建过每一个部分，认识所有的工程师。同样，Devin 也会随着时间的推移真正学习并建立对你的代码库、技术栈和流程的理解，并能据此做更多事情。

[00:58:51] Scott Wu

English:

And the other piece of it, which I think is really exciting I'd say is there really is a lot to do of what I would call a multiplayer aspect of code, which if you think about it, is how a lot of things get done in the real world, certainly. And so it's one thing to have your own experience, which you use yourself as just an engineer, but for example, ourselves, we see this all the time where some engineers are working with Devin and teaching Devin things and as I mentioned, folks will have Devin on board their new engineers and convey that knowledge to them.

中文翻译:

另一部分我认为非常令人兴奋的是代码的“多玩家（multiplayer）”属性。如果你仔细想想，现实世界中很多事情确实是这样完成的。作为一名工程师，拥有自己的个人体验是一回事，但例如在我们自己这里，我们经常看到一些工程师与 Devin 协作并教它东西，正如我提到的，人们会让 Devin 引导新工程师入职并向他们传递知识。

[00:59:24] Scott Wu

English:

Or similarly, it's like I'll start a session with Devin in Slack and I'll say, "Hey, it'd be cool if we could do this thing." And some other engineer will chime in and say, "Oh, by the way, the reason we did it initially was X and Y." And so Devin, just make sure when you do this change that you still support that workflow. And Devin will say, "Okay, sounds great." Or Devin will make a PR, I'll be working with Devin, we'll make pull requests and GitHub and somebody else will be reviewing that PR or give some comments and Devin will work on that too. You'll be in linear.

中文翻译:

或者类似地，我在 Slack 中启动一个 Devin 会话并说：“嘿，如果我们能做这件事就太酷了。”然后另一位工程师会插话说：“哦，顺便说一下，我们最初这么做的原因是 X 和 Y。所以 Devin，确保你在做这个更改时仍然支持那个工作流。”Devin 会说：“好的，没问题。”或者 Devin 提交了一个 PR，我正在和它协作，我们在 GitHub 上提交合并请求，其他人会审查该 PR 或给出评论，Devin 也会据此进行修改。你也会在 Linear 中看到这些。

[00:59:52] Scott Wu

English:

So all these kind of spaces, it really does just kind of set up for an experience basically where Devin can just grow in the value that it can provide for your whole work over time. And so I think from that perspective, if anything, we want there to be a lot of innovation and a lot of new products and so on. I don't think that the goal is to try to lock other people out of building. There's a lot of stuff to build, and I think there's going to be a lot of different experiences. I think from our perspective, what we think about is more like how can we make Devin more and more and more useful as you're using it more?

中文翻译:

所以所有这些场景，实际上都是为了建立一种体验，让 Devin 随着时间的推移，为你的整个工作提供越来越大的价值。从这个角度来看，如果说有什么想法的话，我们希望看到大量的创新和新产品。我不认为目标是试图阻止他人构建。有很多东西需要构建，会有很多不同的体验。从我们的角度来看，我们思考的更多是：随着你使用得越来越多，我们如何让 Devin 变得越来越有用？

[01:00:27] Lenny Rachitsky

English:

It's very similar. We had Michael from Cursor, the CF Cursor on the podcast, and he had a similar point of just he thinks moats are just kind of like consumer, like Google is the way, he thinks it's like Google where people can easily switch. You just have to stay the best, and that's the answer. And it feels like you're adding to that of just like, but also if you can create some stickiness where it is very hard to leave because it's so good at what it's doing and it's built knowledge and integrated to your workflows and builds on that on that stickiness.

中文翻译:

这非常相似。我们之前邀请过 Cursor 的 Michael 参加播客，他也有类似的观点：他认为护城河就像消费级产品，就像 Google 那样——人们可以轻易切换，你只需要保持最强，这就是答案。感觉你在此基础上又增加了一点：如果你能创造一些粘性，让用户很难离开，因为它做得太好了，它积累了知识，集成了你的工作流，并建立在这种粘性之上。

[01:00:58] Scott Wu

English:

Yeah, and I think one of the things that's nice about our space too is, software engineering for better for worse has a very clear tie to value. And what it means is, I guess one way to put it is there is always kind of a clear next level, at least for the next while. I think there could be some point where you're just like, "All right, just build the entirety of YouTube for me." And Devin just does the whole, it's like there's probably been a hundred million hours of human engineering time building YouTube, building the algorithm, building all the infrastructure, all of the, everything, every little detail. And maybe there's some time where Devin just does that out of the box. That's obviously going to be a long time from now.

中文翻译:

是的。我认为我们这个领域还有一个好处，那就是软件工程无论好坏，都与价值有着非常明确的联系。这意味着，至少在未来一段时间内，总会有一个清晰的“下一级”。我想可能会有那么一个时刻，你说：“好吧，帮我构建整个 YouTube。”然后 Devin 就完成了全部工作。要知道，构建 YouTube、算法、所有基础设施、每一个小细节，可能耗费了人类工程师一亿个小时。也许有一天 Devin 拆箱即用就能做到。那显然是很久以后的事了。

[01:01:36] Scott Wu

English:

I think on the interim, on every level in between, obviously it makes a difference the quality of software engineering. And I think one of the cool things with developers obviously is developers are really willing to learn new experiences and to put in effort if it means that they're able to have a higher and higher quality experience.

中文翻译:

我认为在过渡期，在其中的每一个层级，软件工程的质量显然都会产生影响。我认为开发者最酷的一点是，如果意味着能获得越来越高质量的体验，开发者非常愿意学习新体验并投入精力。

[01:01:57] Lenny Rachitsky

English:

Awesome. I'm going to spend a little time on the tech that enables Devin. Without divulging trade secrets, just what allowed you to make Devin so good? Was there an unlock with a certain model? Some folks have shared three points on a 3.5 was a huge unlock for a lot of their products. Just what's kind of the key to the way you've architected or built Devin that makes it work so well?

中文翻译:

太棒了。我想花点时间聊聊支持 Devin 的技术。在不泄露商业机密的情况下，是什么让你们把 Devin 做得这么好？是否有某个模型的突破？有人分享说 GPT-3.5 是他们很多产品的重大突破。你们架构或构建 Devin 的关键是什么，让它运行得如此出色？

[01:02:20] Scott Wu

English:

We obviously, we've been betting on agents for a long time. I think that agents were doable and workable a lot earlier than most folks might've thought. But certainly I think as the community has really rallied around it, I mean you see the impacts of that in the pre-training, you see the impacts of that in a lot of the work that's done with these models. I actually don't think there's been any, from our perspective, I don't think there's been any single step function based model shift or anything that has been kind of like a night and day difference in Devin, but I certainly think that the curve of every point on the chart, I mean there's a new model that comes out every week now has obviously made a big difference in terms of what we've been able to do. And then obviously on top of that, we work with the research teams at all these foundation labs to do a lot of our work on top.

中文翻译:

我们显然在智能体上押注了很久。我认为智能体在比大多数人想象的更早的时候就是可行且有效的。但当然，随着社区真正团结起来，你在预训练中看到了影响，在针对这些模型所做的很多工作中也看到了影响。从我们的角度来看，我不认为 Devin 经历过任何单一的、基于模型的阶梯式转变，或者任何天差地别的变化。但我肯定认为，图表上的每一个点——现在每周都有新模型发布——显然在我们能做的事情上产生了巨大影响。除此之外，我们还与所有这些基础实验室的研究团队合作，在他们的基础上开展大量工作。

[01:03:11] Scott Wu

English:

And so I think that my hot take here at which I would give is, I think in terms of base intelligence, we're honestly basically already there. And I think a lot of what we see actually and what we spend our time on is less so, obviously, we don't our own models or things like that. It's less so increasing the base IQ of a model, for example, and more about teaching it all of the idiosyncrasies of real-world engineering and thinking about here's how you use Datadog and do this, and here's how you might diagnose this error and here are the different things that you could run into and here's how you handle each of those. And when you're ready, here's how you make GitHub PR.

中文翻译:

所以我的一个大胆观点是：在基础智能方面，老实说我们基本上已经达到了。我们实际看到并投入时间的地方，显然不是训练我们自己的模型之类的事。我们较少关注提高模型的基础智商，而更多地关注教它现实世界工程中的所有特质。比如：如何使用 Datadog 做这件事，如何诊断这个错误，你可能会遇到哪些不同的情况，以及如何处理每一种情况。当你准备好时，如何提交 GitHub PR。

[01:03:50] Scott Wu

English:

And this is true in engineering. It's true in every other space as well. I mean, there's so much detail and idiosyncrasy to the work that we all do obviously day to day. And a lot of it is kind of like teaching the model to mirror the complexity of the real world, I would say, rather than getting it to some higher fundamental level of problem solving, which I think the foundation labs are doing a really great job.

中文翻译:

这在工程领域是真的，在其他任何领域也是如此。我的意思是，我们每天所做的工作中都有太多的细节和特质。我会说，很大程度上是教模型去镜像现实世界的复杂性，而不是让它达到某种更高层次的基础问题解决水平——我认为基础实验室在这方面已经做得非常出色了。

[01:04:15] Lenny Rachitsky

English:

There's something you shared when we were chatting before we started recording around the growth of previous transformative technologies were very hardware oriented and there was a limiting factor to their growth and AI is not that. Can you just share that insight?

中文翻译:

在我们开始录音前聊天时，你分享了一些观点：以前的变革性技术的增长非常依赖硬件，这限制了它们的增长速度，而 AI 并非如此。你能分享一下那个见解吗?

[01:04:30] Scott Wu

English:

For a number of reasons? I think AI is going to be the biggest technology shift of our lives, but I think one thing, which is what we were just talking about before this, which is most of the big tech revolutions that we've had over the last 50 years, I mean, I thinking about personal computer and the internet and the mobile phone and stuff, they all had this big hardware component that was a big part of the distribution. And so you had the internet, and initially it was just these universities that were talking with one another, but obviously over time we got the whole world plugged into the internet and it took years and years and years. Same thing was true with mobile phones. Same thing was true with PC.

中文翻译:

出于多种原因，我认为 AI 将是我们一生中最大的技术变革。其中一点就是我们刚才聊到的：过去 50 年里我们经历的大多数重大技术革命——比如个人电脑、互联网、手机等——它们都有一个巨大的硬件组件，这是分发的重要部分。比如互联网，最初只是大学之间在交流，但显然随着时间的推移，全世界都接入了互联网，这耗费了漫长的岁月。手机和电脑也是如此。

[01:05:06] Scott Wu

English:

And the thing that's interesting about that in particular, which is I would say we're already seeing the effects of that, is in these hardware distribution machines, obviously there's a lot that depends on real time. And so folks who were building for those industries saw their market grow and grow and grow basically steadily year over year as the number of people with mobile phones increased, as the number of people connected to the internet increased. Many of those businesses, it's still crazy to think, but many of those businesses got started right in the beginning. I mean, Apple and Microsoft were started right around the same time. And same is true for a lot of the great internet businesses or wherever, but certainly it was something that touched whole world with time or a large fraction of the whole world. And it had a really massive impact, but it took place over several years because of the time that it took. And I think one of the things which is already I'd say different in AI, is just how explosive the technology can be. Once AI could, and I think we're firmly past the inflection point in AI code where it's, as an engineer, if you're not using AI at all to write code, I mean you're falling behind honestly. And it is a technology that everyone should have and should be using, and there's no kind of weight on hardware distribution that is causing that. It means that the space is just growing so exponentially, basically.

中文翻译:

特别有趣的一点是（我认为我们已经看到了这种影响），在这些硬件分发机器中，显然有很多东西取决于实时性。因此，为这些行业开发产品的人看到他们的市场随着手机用户或互联网连接人数的增加而逐年稳步增长。回想起来依然疯狂，很多伟大的企业都是在最初就开始了，比如苹果和微软几乎同时起步。互联网巨头也是如此。这些技术随着时间的推移触及了全世界或大部分地区，产生了巨大的影响，但由于硬件分发的时间，这经历了数年之久。而我认为 AI 已经表现出的不同之处在于，这项技术的爆发力有多强。一旦 AI 能够做到……我认为我们已经牢牢越过了 AI 代码的拐点。作为一名工程师，如果你完全不使用 AI 来写代码，老实说你已经落后了。这是一项每个人都应该拥有并使用的技术，而且没有硬件分发的负担。这意味着这个领域基本上正在呈指数级增长。

[01:06:36] Lenny Rachitsky

English:

Michael Ballin has this interesting point that cliches are cliches because they're so true and that's why they're cliches. I heard that a million times and I think it's like people hear this like, "Yeah, yeah, I know," but it's actually insane what is happening. That's why you're here to help us through this transition.

中文翻译:

Michael Ballin 有个有趣的观点：陈词滥调之所以成为陈词滥调，是因为它们太正确了。我听过无数次这种说法，人们听到后会觉得“行了行了，我知道”，但现在发生的事情实际上非常疯狂。这就是为什么你在这里帮助我们度过这个转型期。

[01:06:52] Scott Wu

English:

Yeah, no, I mean, it is a fun time and I think there will be real investment and real work that it takes. But I think from the perspective of us as engineers, for example, I think it just means it's so important to stay in the loop with everything that's happening. And as we're seeing it's not only because of your learning and your ability to work these technologies, but it's also about basically teaching the AI what there is to know about your code base in order to make it really effective at building with you and doing more of the things that you would want it to do.

中文翻译:

是的，这是一个有趣的时代。我认为这需要真正的投入和努力。但从我们工程师的角度来看，这意味着紧跟时事非常重要。正如我们所见，这不仅是为了你的学习和使用这些技术的能力，也是为了教 AI 了解你的代码库，从而让它能非常高效地与你共同构建，并完成更多你想要它做的事情。

[01:07:27] Lenny Rachitsky

English:

So along those lines, for people listening that they're like, "Hey, we should be using Devin at our company," what are things you've found to be helpful in helping an engineer at a company get adoption and be able to use Devin either culturally or logistically?

中文翻译:

沿着这个思路，对于那些听了之后觉得“嘿，我们公司应该使用 Devin”的听众，你发现有哪些方法能有效帮助公司里的工程师获得采用并开始使用 Devin（无论是从文化上还是从流程上）？

[01:07:42] Scott Wu

English:

So a pattern which we often see with folks is there will be a few folks at the team who are really excited and want to try out the new thing, and they want to put in the investment and are really excited to get it going. And they'll go through all the setup. They'll give Devin the repos, they'll teach Devin how to run the lint and the CI and all of those details. And they'll start off by giving it those initial tasks and help Devin build a foothold basically. And as time goes on, eventually folks will see, "Wow, Devin's writing all these PRs, Devin's doing this [inaudible 01:08:14].

中文翻译:

我们经常看到的一种模式是：团队中会有几个人非常兴奋，想尝试新事物，他们愿意投入精力并热衷于让它运转起来。他们会完成所有设置，把仓库交给 Devin，教 Devin 如何运行 lint 和 CI 等所有细节。他们会从给它一些初始任务开始，帮助 Devin 建立立足点。随着时间的推移，其他人最终会看到：“哇，Devin 正在写所有这些 PR，Devin 正在做这个……”

[01:08:13] Lenny Rachitsky

English:

Who's this Devin person that just joined the company's just knocking out PRs?

中文翻译:

“这个刚加入公司、疯狂提交 PR 的 Devin 是谁啊？”

[01:08:17] Scott Wu

English:

Yeah. And they'll see that, and then naturally they'll get on and they'll get an account. And one of the cool things of course is by the time they join, Devin already knows a good amount of detail about the repositories that they're already working in and they're working with that. And so one of the really cool things which we often see is that the early adopters themselves can really pave the way I think, for everyone else on the team.

中文翻译:

是的。他们看到后，自然就会加入并注册账号。当然，酷的地方在于，当他们加入时，Devin 已经对他们正在处理的仓库有了相当深入的了解。所以我们经常看到的一件很酷的事情是，早期采用者本身真的可以为团队中的其他人铺平道路。

[01:08:37] Scott Wu

English:

But yeah, I think the main thing I would just call out is it really does take, it's a very different product experience, and I think for what it's worth, I think there's still a lot more that we can do to make it as intuitive and as clear as possible to folks like how to use Devin and what the right steps are and how to really maximize value out of Devin. But I think that, yeah, it's the kind of thing where if you put in the investment and understand exactly what it takes to get Devin to be successful, we've found ourselves that as time has gone on, we just use Devin more and more and more with every next update.

中文翻译:

但我认为最重要的一点是，这确实需要时间，因为它是一种非常不同的产品体验。我认为我们还有很多工作可以做，让人们尽可能直观、清晰地了解如何使用 Devin、正确的步骤是什么，以及如何真正最大化 Devin 的价值。但如果你投入精力并准确理解如何让 Devin 获得成功，你会发现随着时间的推移，随着每一次更新，你会越来越频繁地使用 Devin。

[01:09:15] Lenny Rachitsky

English:

So let me follow that thread. There is a question I ask every AI app building founder, which is, if you could sit next to every new user of Devin and whisper something in their ear to help them be successful with Devin, one or two tips, what would those tips be?

中文翻译:

让我顺着这个话题问下去。我问过每一位 AI 应用创始人的一个问题是：如果你能坐在每一个 Devin 新用户身边，在他们耳边低声说一两条建议来帮助他们成功使用 Devin，那会是什么建议？

[01:09:31] Scott Wu

English:

I think the biggest thing I would say is it really is just treat Devin like your new junior engineer. And I think that's the biggest thing. I think folks come in and they see the blank page and they think of all sorts of various things that they want to try out. They think of lots, where I think typically the flow that we see that works best is obviously you can try demos and you can do things, but a lot of it is just like, "Let's figure out what tickets we want to get done today or this week and let's have Devin get started on those and let's start with the easier ones and then work with Devin and understand what things Devin needs to get set up to be able to test its own code and do this well. And then let's scale up over time." And then obviously as you work with your engineer, you understand better how to communicate with them or what are the right tasks or projects to bring them in on. But I think that really is the one-liner for us.

中文翻译:

我想我能说的最重要的一点就是：真的要把 Devin 当作你的新初级工程师。我认为这是核心。人们进来看到空白页，会想到各种各样想尝试的事情。虽然你可以尝试演示，但我们发现最有效的流程通常是："让我们弄清

楚今天或这周想完成哪些票据（tickets），让 Devin 开始处理这些。先从简单的开始，然后与 Devin 协作，了解 Devin 需要设置什么才能测试自己的代码并把工作做好。然后随着时间的推移逐渐增加难度。”显然，当你与工程师协作时，你会更了解如何与他们沟通，或者哪些任务或项目适合让他们参与。我认为这就是我们的核心建议。

[01:10:31] Lenny Rachitsky

English:

Okay. There's a question I'd be meeting task. I just want to get back to this because it's something I think a lot about with Devins. Everyone's going to have five Devins, let's say 10 Devins. Everyone's kind of turning into basically an engine manager with a bunch of junior engineers, which isn't necessarily the best job in the world because it's just a bunch of, at least you don't have to do performance reviews and one-on-ones, but it's sitting around, checking a lot of PRs all day. There's a sense of you become an architect, which is kind of what every engineer wants to become eventually, right? They're all, "I just want to think about the architecture. I don't want to code all these stupid, fix bugs."

中文翻译:

好。我想回到关于 Devin 的一个思考。每个人都会有 5 个甚至 10 个 Devin。每个人基本上都变成了一个带着一群初级工程师的工程经理（EM），这不一定是世界上最好的工作，因为虽然你不用做绩效评估和 1 对 1 面谈，但你得整天坐着检查大量的 PR。有一种感觉是你变成了架构师，这正是每个工程师最终想成为的角色，对吧？他们都想：“我只想思考架构，不想写那些愚蠢的代码和修 bug。”

[01:11:06] Lenny Rachitsky

English:

So I get that there's a good side to that, but just I imagine you're thinking a lot about this, just like how do you make life pleasant and fun and enjoyable as basically an engine manager of say, 500 Devins in the future?

中文翻译:

所以我明白这有积极的一面，但我猜你也在思考这个问题：未来当你作为一个管理着比如 500 个 Devin 的工程经理时，你如何让生活变得愉快、有趣且享受？

[01:11:19] Scott Wu

English:

Yeah, I can just imagine the performance, "Devin, you've done a really great job on your task, but I really would like you to be more proactive in the team meetings." So what I'd say-

中文翻译:

哈哈，我可以想象那个绩效谈话：“Devin，你任务完成得很好，但我真的希望你在团队会议中能更积极主动一点。”我想说的是——

[01:11:29] Lenny Rachitsky

English:

[inaudible 01:11:29].

中文翻译:

（听不清）

[01:11:29] Scott Wu

English:

It's funny actually because something that in terms of the wording that we thought a lot about as well is just, we've used the term manager of Devins in the past, which of course I think is a big part of it. But I think that the bricklayer versus architect is closer to the experience than being a manager. Because I think a lot of the difficulty of management or the reason that people shy away from it is more because of a lot of the various.

中文翻译:

这很有趣，因为在措辞方面我们也思考了很多。我们过去用过“Devin 管理者”这个词，当然我认为这是很大一部分。但我认为“搬砖工 vs 架构师”比“经理”更接近这种体验。因为我认为管理的很多困难，或者人们回避管理的原因，更多是因为各种……

[01:12:00] Scott Wu

English:

... [inaudible 01:12:00] because of a lot of the various, let's say... There's all of the context, and the ownership, and the responsibility and stuff, and then there's also all of the emotional aspects of it. Where, I think working with Devin is a little more like just being, more as having an interface to hand off tasks and build tasks. And so, the parallel that I would draw is when we invented Python, obviously... It's like, in many ways the description and the outlining of tasks, obviously it was a different paradigm, but I think certainly it was nowhere near what folks typically think of as management bureaucracy today.

中文翻译:

……各种因素，比如所有的上下文、所有权、责任等等，还有所有的情感层面。而我认为与 Devin 协作更像是拥有一个交付任务和构建任务的界面。我能想到的类比是，当我们发明 Python 时，显然……在很多方面，任务的描述和勾勒是一个不同的范式，但我认为它肯定远没有人们今天通常认为的管理官僚主义那么复杂。

[01:12:50] Scott Wu

English:

And I think that with Devin, a lot of it is just like, it's more like finding the right levels of abstraction that you could work with Devin on, and just finding the workflows that work really well, and the obvious thing to say here is, it's like you can always have Devin take a first pass at things. And so you have Devin take the first pass, if it's great, you merge it right away, if it needs some touch up, you can obviously give that feedback, for example, but a lot of it is like, it's more about, basically making Devin part of your flow than it losing control, which I think is the main thing that folks are scared of with management.

中文翻译:

我认为对于 Devin，很大程度上是找到与 Devin 协作的正确抽象层级，找到运行良好的工作流。显而易见的是，你总是可以让 Devin 先尝试一下。如果它做得很好，你就直接合并；如果需要修饰，你可以给出反馈。很

大程度上，这更多是关于让 Devin 成为你工作流的一部分，而不是失去控制——我认为失去控制才是人们对管理感到恐惧的核心原因。

[01:13:34] Lenny Rachitsky

English:

Are you thinking about a manager Devin? Like a Devin that manages other Devins?

中文翻译:

你在考虑“经理 Devin”吗？比如一个管理其他 Devin 的 Devin?

[01:13:37] Scott Wu

English:

Yeah, yeah. So, for what it's worth, Devin can start other Devins through the API, right? And so, we've seen this happen quite a bit of times, where naturally if you have some big tasks that you want to do, Devin will do this all the time, it'll chunk up and then [inaudible 01:13:53] into smaller Devins. And so, it's the kind of thing that you need to give Devin the credentials to be able to do that, it's not currently something that is default enabled, but I can certainly imagine as time goes on that there's more and more of that

中文翻译:

是的。事实上，Devin 可以通过 API 启动其他 Devin。我们已经见过很多次这种情况：自然地，如果你有一个大任务要完成，Devin 会经常将其拆解，然后分配给更小的 Devin。当然，你需要给 Devin 相应的凭据才能做到这一点，目前这不是默认开启的功能，但我完全可以想象随着时间的推移，这种情况会越来越多。

[01:14:03] Lenny Rachitsky

English:

Devins all the way down.

中文翻译:

全是 Devin，一直到底。

[01:14:05] Scott Wu

English:

Yeah, yeah. I think the thing that's kind interesting too is, with humans, the way I almost say it, in technical terms, it's like there's this coupling of a context and a thread, and what I mean by that is basically, each human can only operate a single threaded on the work that they do, and they have their set of context, and then there's other humans obviously who can do other stuff at the same time, but they have their own context. With agents, one of the cool things is you can have an agent that's doing multiple lines of exploration at once, but is sharing all of the context of everything that they find, and so I think that this is very early, and I think we'll see this, but folks obviously love to talk about basically systems of agents communicating with one another, and I think that there will be a lot of new paradigms to build for, once we get there.

中文翻译:

是的。我觉得有趣的一点是，对于人类，用技术术语来说，存在上下文和线程的耦合。我的意思是，每个人在工作中基本上只能单线程操作，他们有一套自己的上下文。虽然其他人可以同时做别的事，但他们有自己的上下文。而对于智能体，酷的地方在于你可以让一个智能体同时进行多条线路的探索，但共享它们发现的所有上下文。我认为这还处于非常早期，我们会看到的。人们显然很喜欢谈论智能体相互通信的系统，我认为一旦我们达到那个阶段，将会有很多新的范式需要去构建。

[01:14:55] Lenny Rachitsky

English:

And it's so interesting what you said about the decision between having one Devin and only one Devin do all the things, and you just tell them things and they fire off jobs versus you have five Devins, and they're each doing individual things, it's such an interesting decision to make.

中文翻译:

你提到的那个决策非常有趣：是让一个且仅一个 Devin 处理所有事情，你只管吩咐它，由它去分派任务；还是你直接拥有五个 Devin，每个都在做独立的事情。这是一个非常有趣的决策。

[01:15:10] Scott Wu

English:

Yeah, yeah, yeah, for sure.

中文翻译:

是的，没错。

[01:15:12] Lenny Rachitsky

English:

Okay, two more questions. Maybe the most counterintuitive thing you've learned so far building Devin, that maybe goes against startup wisdom, common startup wisdom?

中文翻译:

好，还有两个问题。在构建 Devin 的过程中，你学到的最反直觉的事情是什么？也许是违背了初创公司常识或智慧的事情？

[01:15:21] Scott Wu

English:

Something I've thought about a lot lately as we've built this is, this is not my first company, actually for a lot of us, it's not our first company, I think of our 26 or 27 people total on the team, I think 18 of us have started our own company before this. And yeah, one of the things I think about is, there's actually your point about cliches I think really spoke to me as well, which is, there's the really common things which you hear all the time in startups, where you're like, you got to move fast, or you got to hire great people, it's like, okay, well, obviously you do, I wasn't planning on not hiring great people, I wasn't planning on going slow. And similarly it's like, yeah, you really got to build something that people want. And there's these three to five things which are always repeated, and they're always the common wisdom in startups.

中文翻译:

最近在构建这个产品时我思考了很多。这不是我的第一家公司，实际上对我们很多人来说都不是。我们团队总共 26 或 27 人，其中大概 18 人之前都创办过自己的公司。我想起你刚才提到的关于“陈词滥调”的观点，这真的引起了我的共鸣。在初创公司中，你总能听到那些非常普通的事情，比如“你必须行动快”、“你必须雇佣优秀的人”。你会觉得：“好吧，这显而易见，我也没打算雇平庸的人，也没打算慢吞吞的。”同样还有“你必须构建人们想要的东西”。总有那么三到五件事被不断重复，成为初创公司的常识。

[01:16:11] Scott Wu

English:

And I definitely had this idea as a founder, when I was starting initially, that, all right, so those are the three to five basic things, but as you get really deep into it, you spend a lot of years into it, you learn all of the thousands of other things that you have to learn to build a company. And I think to some extent that's, of course, true, and there's lots of little details that you'll get into with all these different things, including team building, and product, and strategy, and engineering decision-making, and fundraising, and sales, and every other component. But I also realized that as time has gone on, more and more, I felt like building companies well sometimes just comes down to doing those three to five things just even more than you could possibly expect.

中文翻译:

作为创始人，我最初的想法是：好吧，那是三到五个基本点，但当你深入其中，投入多年时间后，你会学到构建公司所需的成千上万件其他事情。在某种程度上，这当然是对的，你会接触到很多细节，包括团队建设、产品、战略、工程决策、融资、销售等等。但我也意识到，随着时间的推移，我越来越觉得，把公司办好有时仅仅归结为把那三到五件事做得比你预想的还要极致。

[01:16:58] Scott Wu

English:

And so, with us, it's like... And everyone says we go fast, but it's like, yeah, we had a hackathon in November, we had another hackathon in December, we started the company officially in January, we got the prototypes out to initial users in February, we did a launch in March, we got our first customers in April... It's just like, basically truly pushing the pace in every spot where we possibly could has really made a difference for us. And similarly, it's like, yeah, everyone always says you should hire great people, but I think that the truth within that truth is basically you should fight to all ends basically, to get the folks that you really want to bring in. And one of my favorite stories to share is we had a candidate who came and interviewed, he was a junior at MIT, so he was very, very young, and we gave him an interview, and he did way better than almost any of the full-time candidates that we had ever talked to. And so, we said, hey, what do you think about taking some time off of school and working with us and building out Devin?

中文翻译:

对我们来说，每个人都说我们很快，但事实是：我们在 11 月搞了黑客松，12 月又搞了一场，1 月正式成立公司，2 月把原型交给了初始用户，3 月发布，4 月有了第一批客户……基本上在每一个可能的环节都真正地推进速度，这确实让我们与众不同。同样，每个人都说要雇佣优秀的人，但我认为这个真理背后的真相是：你应该不惜一切代价去争取你真正想要的人才。我最喜欢分享的一个故事是：有一个来面试的候选人，他是 MIT 的大三学生，非常年轻。我们面试了他，他的表现比我们谈过的几乎所有全职候选人都要好。于是我们说：“嘿，你觉得休学一段时间来和我们一起构建 Devin 怎么样？”

[01:17:57] Scott Wu

English:

We really think you're just going to be able to come in and just have a ton of impact already from day one. And he thought about it for a while, and he came back and he said, you know what? I'm down, I want to do it, but my parents really want me to graduate from school, and I'm just not sure there's a way to make it work. And so, we talked him more and just understood the situation, and then we flew to North Carolina, went straight from the airport to his parents' house, had dinner with him and his parents, we talked a lot, a really nice Gujarati family, we gave them some gifts and just talked to them about it, and try to understand what would it take, and what would we need to make work. And they just said, it sounds like a great opportunity, but we really want our son to be able to graduate.

中文翻译:

我们真的认为你从第一天起就能产生巨大的影响。他考虑了一段时间，回来后说："我想加入，但我父母非常希望我能大学毕业，我不知道该怎么平衡。"于是我们进一步了解情况，然后飞到了北卡罗来纳州，直接从机场去了他父母家，和他以及他的父母一起吃晚饭。我们聊了很多，那是一个非常友好的古吉拉特人家庭。我们送了礼物，和他们长谈，试图理解需要达成什么条件。他们只是说："这听起来是个很好的机会，但我们真的希望儿子能毕业。"

[01:18:45] Scott Wu

English:

And we talked that through and we figured out a setup basically, where he could work for us essentially full time, but then come in for his required classes, and do what he needed to do to get the diploma basically, but no more than that. And we talked that through, and then we got to a point where everyone was happy with that, and then went straight back to the airport and flew right back, basically. And that was the first and only time that I've ever been in North Carolina, and it was just a great trip. And it is the kind of thing where it's like, hiring great people is one thing, but truly just never giving up, and really giving it everything that you can to make it work for people who really makes sense to be on the team. And he's been with us on the team for over a year now, and he's been an incredible, incredible engineer, and we wouldn't be here without him.

中文翻译:

我们详细讨论了这件事，最终想出了一个方案：他基本上可以全职为我们工作，但要去上必修课，完成拿到学位所需的事情，仅此而已。我们谈妥了，每个人都很满意，然后我们直接回机场飞了回来。那是我第一次也是唯一一次去北卡罗来纳州，那是一次很棒的旅行。这就是我想说的：雇佣优秀的人是一回事，但真正永不言弃，竭尽全力为那些真正适合团队的人解决问题是另一回事。他已经在我们团队待了一年多，是一位非常非常出色的工程师，没有他，我们就不会有今天。

[01:19:31] Scott Wu

English:

And similarly, we had someone else who was again, really, really talented candidate, did amazingly well, very young, and had a lot of great offers at a lot of other companies, and we were talking to them about, he wanted to start his own company someday as well, and we were talking to him about certainly a lot of the obvious things, which are having him meet our investors, or get to do work with customers, or see a lot of these other components so that when the time came that he would have all the experience he needed to start his own company. But one of the other things that was big is he was talking with a lot of great companies already, he didn't want to burn any bridges, and so we actually worked with him and basically hand wrote all of his rejection responses to each of the other companies, and worked with him on it to say, here's how you should say it in a way that's going to come off as that you really did appreciate the time with them, and that obviously you want to remain close with them and stay in touch. And it was the kind of thing obviously where it's like, look, obviously our job is to make sure that he's happy enough that he doesn't want to leave at any time in the near future, but I think it's the kind of thing where the way that you put together a really, really great team is by fighting for what's right for them too.

中文翻译:

同样，我们还有另一位非常有才华的候选人，表现惊人，非常年轻，拿到了很多其他大公司的 offer。他以后也想创办自己的公司，我们和他聊了很多显而易见的事，比如让他见我们的投资者，让他直接接触客户，了解公司的其他组成部分，这样等时机成熟时，他就具备了创业所需的经验。但还有一件大事：他当时正在和很多优秀的公司沟通，他不想断了后路。于是我们实际上帮他一起工作，基本上是亲手帮他写了给其他所有公司的拒绝信，教他如何表达才能显得他真的很感激对方的时间，并且显然希望保持联系。显然，我们的工作是确保他足够开心，在不久的将来不想离开，但我认为组建一支真正伟大团队的方式，也是为他们的利益而战。

[01:20:45] Lenny Rachitsky

English:

Wow, those are incredible stories. And it makes so real these, as you say, cliches, hire the best people, this is what it sounds like to hire the best people. This is what it takes.

中文翻译:

哇，这些故事太精彩了。这让你说的那些陈词滥调变得如此真实——“雇佣最好的人”，这就是雇佣最好的人听起来的样子。这就是代价。

[01:20:56] Scott Wu

English:

Yeah, no, and I was just saying, a lot of things we've fought very hard to just reimagine things from the ground up because a lot of it really is just thinking about where do we think the technology is going over the next five, 10 years, and what is the place that we want to have in that future?

中文翻译:

是的。我刚才想说，我们非常努力地从头开始重新构思很多事情，因为很大程度上这只是在思考：我们认为未来 5 到 10 年技术会走向何方？而我们想在那个未来中占据什么样的位置？

[01:21:16] Lenny Rachitsky

English:

I wonder if people are going to be fighting for the best Devin someday. They're just going to [inaudible 01:21:19] Devins.

中文翻译:

我在想，有一天人们会不会为了争夺“最好的 Devin”而战。

[01:21:18] Scott Wu

English:

Yeah.

中文翻译:

是的。

[01:21:19] Lenny Rachitsky

English:

They're [inaudible 01:21:22] so smart.

中文翻译:

它们太聪明了。

[01:21:22] Scott Wu

English:

I'll give you overtime pay, benefits, free healthcare, and everything, and then the Devin's like...

中文翻译:

“我会给你加班费、福利、免费医疗和一切”，然后 Devin 就像……

[01:21:29] Lenny Rachitsky

English:

Three Devins like Magic: The Gathering cards.

中文翻译:

三个 Devin 就像《万智牌》里的卡片一样。

[01:21:31] Scott Wu

English:

Yeah.

中文翻译:

哈哈，是的。

[01:21:32] Lenny Rachitsky

English:

And then just going back to your three to five things. So, essentially this is incredible advice, essentially, it's like you always hear hire the best people, move fast, build things people want.

中文翻译:

回到你说的三到五件事。这本质上是极好的建议。你总能听到：雇佣最好的人、行动快、构建人们想要的东西。

[01:21:41] Scott Wu

English:

Yeah, build something people want, stay as close as possible to your customers, and then I think the other thing is just always think about where things are going, not where they are today. I feel like those are the five things, which is... Especially in AI, with things moving so fast, and there's so much great talent, I feel like a lot of these are even more true, where it's like it's not just thinking about where things are going to be in 10 years, it's like thinking about what's going to happen next week. And obviously, things are moving very quickly, and it is very hard to predict, but you really have to be very rigorous with yourself, I'd say, about thinking through those things and evaluating all of the decisions that you make in that lens.

中文翻译:

是的，构建人们想要的东西，尽可能贴近你的客户。我认为另一件事是：永远思考事物的走向，而不是现状。我觉得这就是那五件事。尤其是在 AI 领域，事物发展如此之快，人才如此之多，我觉得这些真理甚至更加正确。这不仅仅是思考 10 年后会怎样，而是思考下周会发生什么。显然，事物变化极快，很难预测，但我会说，你必须对自己非常严苛，去深入思考这些事情，并从这个视角评估你所做的每一个决策。

[01:22:25] Lenny Rachitsky

English:

And staying focused is the big takeaway to me here, is it ends up feeling like there's 1000 things you should do, but it's always these five things.

中文翻译:

对我来说，最大的收获是保持专注。最终感觉好像有 1000 件事要做，但核心永远是这五件事。

[01:22:31] Scott Wu

English:

Yeah.

中文翻译:

是的。

[01:22:33] Lenny Rachitsky

English:

Scott, we covered a lot of ground. We went through every question I had, which is great, is there anything else that you want to share? Anything else you want to leave your listeners with, maybe a final nugget or something really you want to double down on that we said before we let you go?

中文翻译:

Scott，我们聊了很多。我所有的问题都问完了，这太棒了。还有什么你想分享的吗？在结束之前，有没有什么想留给听众的，比如最后的金句，或者你想再次强调的观点？

[01:22:49] Scott Wu

English:

The biggest thing that comes to mind for me is there's a lot of different perceptions about AI, right? I think basically every emotion under the sun right now. There's a lot of fear, for example, there's also a lot of skepticism, and we're very skeptical types as well, and we always wanted to try it ourselves to really see it and believe it. And I think the main thing that comes to mind for me, is I'm honestly really optimistic about what we're building here with AI, not just with code and with Devin, but the whole space, and everything that's getting done. And I think one of the cool things that is really actively happening is just the ability for everyone to multiply themselves, and that's how we've always thought about it, it's how we've thought about what we're building. And I think there is a lot more to do out there in the world, I'm not too worried about us running out of things to do, and from that lens, I think that the thing that we've always been most excited about is, how can we all do more?

中文翻译:

我想到最重要的一点是，现在人们对 AI 有很多不同的看法，对吧？基本上涵盖了人类所有的情感。比如有很多恐惧，也有很多怀疑。我们也是非常多疑的人，我们总是想亲自尝试，眼见为实。我想到的核心一点是：老实说，我对我们正在用 AI 构建的东西非常乐观，不仅是代码和 Devin，而是整个领域以及正在完成的一切。我认为正在发生的酷事之一就是每个人都有能力“倍增”自己。这就是我们一直以来的思考方式，也是我们对所构建事物的看法。我认为世界上还有很多事情要做，我不担心我们会无事可做。从这个角度来看，我们一直最兴奋的是：我们如何能做得更多？

[01:23:52] Lenny Rachitsky

English:

I hear you Scott. Well, with that optimism, we've reached our very exciting lightning round. Are you ready?

中文翻译:

我明白你的意思，Scott。带着这份乐观，我们进入了非常令人兴奋的闪电轮环节。准备好了吗？

[01:23:59] Scott Wu

English:

Yeah, let's do it.

中文翻译:

准备好了，开始吧。

[01:24:00] Lenny Rachitsky

English:

Okay, here we go. First question, what are two or three books that you find yourself recommending most to other people?

中文翻译:

好，开始。第一个问题：你最常向别人推荐的两三本书是什么？

[01:24:07] Scott Wu

English:

In terms of nonfiction, I think for folks in startups, I think one of the things that I've really enjoyed is just learning and understanding the history of Silicon Valley. And it's all of these things that we think about, somebody invented them. It's one of the great realizations I feel like, is somebody invented the idea of a seed route, somebody invented the idea of venture capital, somebody invented the idea of product-market fit, and all of these different principles that we talk about. And so, for that, there's a book called The Power Law by Sebastian Mallaby, which I really like. And basically, it's just a tour of many of the great businesses and the great products that have been built over the last 60, 70 years in Silicon Valley, which I really love. I think in terms of fiction, I actually have always really liked The Great Gatsby by F. Scott Fitzgerald, it's one of my personal favorites as a fiction book.

中文翻译:

在非虚构类书籍方面，对于创业者，我非常喜欢学习和了解硅谷的历史。我们思考的所有这些事情，都是有人发明的。我觉得一个伟大的感悟是：有人发明了种子轮的概念，有人发明了风险投资的概念，有人发明了产品市场匹配（PMF）的概念，以及我们谈论的所有这些不同原则。为此，我非常喜欢 Sebastian Mallaby 的《权力法则》（The Power Law）。它基本上是对过去 60、70 年硅谷建立的许多伟大企业和产品的巡礼。在虚构类书籍方面，我一直非常喜欢 F. Scott Fitzgerald 的《了不起的盖茨比》，这是我个人最喜欢的虚构作品之一。

[01:24:57] Lenny Rachitsky

English:

Do you have a favorite recent movie or TV show that you've really enjoyed?

中文翻译:

你最近有没有特别喜欢的电影或电视节目？

[01:25:01] Scott Wu

English:

I have to admit, I have not watched... I can't think of a single movie or TV show that I have watched in the last while, so I'm sure, I'm looking forward to watching a lot of great ones post-AGI.

中文翻译:

我必须承认，我最近没看……我想不出最近看过的任何一部电影或电视节目。我确信，我期待着在 AGI 实现之后看很多精彩的作品。

[01:25:17] Lenny Rachitsky

English:

That's got to be in the trailer, that's great, I like that. And that just shows how hard you're working, just how much shit is going on, and how fast everything's moving. Do you have a favorite product you've recently discovered that you really love? Could be an app, could be something physical, could be a toothbrush.

中文翻译:

这句话一定要放进预告片里，太棒了，我喜欢。这正说明了你工作有多努力，事情有多少，一切发展得有多快。你最近有没有发现什么特别喜欢的产品？可以是应用，可以是实物，甚至可以是牙刷。

[01:25:32] Scott Wu

English:

One I would say is I got an Aura frame recently, it's just like a frame that shows photos, and you can show a new photo every day, or every hour, or every 15 minutes, or whatever you like. I've actually, I've really enjoyed it a lot, I think it's a nice way to basically just have a picture frame memories that come up. And then, the other thing I would say as a general purpose thing, it's not particularly new, but I would say, I think AirPods are extremely well-built and well-designed. And I realize now that it's like, I basically use them for all... I'm taking calls on a walk and I'm using AirPods, obviously I'm doing work at my computer, at my desk, I'm plugged into AirPods, and it works quite well, honestly, for a lot of different situations, and they're very comfortable, they're very consistent.

中文翻译:

一个是我最近买了一个 Aura 相框，它就是一个显示照片的相框，你可以设置每天、每小时或每 15 分钟显示一张新照片。我真的很喜欢它，我觉得这是一种让回忆浮现的好方式。另一个通用的东西，虽然不新，但我认为 AirPods 的制造和设计都非常出色。我现在意识到，我基本上在所有场合都用它们……散步时接电话用 AirPods，在电脑前工作也戴着 AirPods。老实说，它在很多不同情况下都表现得很好，非常舒适且稳定。

[01:26:21] Lenny Rachitsky

English:

I'm going to double down on the Aura frame, I also, I got one of these for my mom and my mother-in-law, and they're so great for just sharing photos of your kids with your family. And people have, they've heard of digital picture frames, but the Aura just does it really well, and it's really easy to add photos, and they're just really nice looking.

中文翻译:

我也要极力推荐 Aura 相框。我也给我的母亲和岳母买了一个，它们非常适合与家人分享孩子的照片。人们听说过数字相框，但 Aura 做得非常好，添加照片非常容易，而且外观也很漂亮。

[01:26:39] Scott Wu

English:

You can imagine not that long from now, we'll have the Aura Frame except Studio Ghiblifies every photo that you have in it, and then it's... Yeah.

中文翻译:

你可以想象，不久之后，我们会有 Aura 相框，只不过它会把你放进去的每一张照片都变成吉卜力工作室的风格，然后……是的。

[01:26:47] Lenny Rachitsky

English:

Or just imagines things you've done that are really cool. Look at my sweet life. Yeah. Cool. And it's Aura, it's A-U-R-A, I believe is how you spell it. Folks who want to check it out, we'll link to it. Not affiliated. Okay, two more questions. Do you have a favorite life motto that you often come back to and find useful in work or in life?

中文翻译:

或者只是想象你做过的一些很酷的事情。“看我甜蜜的生活”。是的，很酷。那是 Aura，拼写是 A-U-R-A。想了解的人我们会放上链接，无利益相关。好，最后两个问题。你有没有最喜欢的座右铭，在工作或生活中经常觉得有用？

[01:27:06] Scott Wu

English:

Yeah, something I've thought about a lot is a lot of the proverbs out there are actually contradictions, right? It's like birds of a feather, and then you also have opposites attract, and you have all... And it's kind of funny, because you feel that both of them are true, and often they both are true, and a lot of it is about understanding why. And one of those that I feel like, especially in the world of startups that I think about all the time is, I think it is very important to be focused and driven, and to really maximize your potential, and then at the same time, it's also very important to not let your own personal emotion get tied up in your success or failure, and I think especially with startups, because there's always ups and downs, honestly, even in the most successful companies ever, it's just like, it's a rocky road, there's a lot that happens, and a lot that goes down.

中文翻译:

是的，我思考过很多的一点是，很多谚语其实是矛盾的，对吧？比如“物以类聚”，但也有“异性相吸”。有趣的是，你觉得两者都是对的，而且通常确实如此，关键在于理解为什么。在初创公司领域，我一直在思考的一个观点是：保持专注、充满动力并真正最大化你的潜力非常重要；但与此同时，不让个人的情感与成功或失败纠缠在一起也同样重要。我认为尤其是在初创公司，因为总会有起伏。老实说，即使是历史上最成功的公司，道路也是坎坷的，会发生很多事，会有很多挫折。

[01:28:03] Scott Wu

English:

And I think one of the things which I've thought a lot about is that somehow you really want to do your best, and put everything you can into it, and do everything you can to... Basically, you want to put it all out on the field. But at the same time, you want to be okay with both wins and losses, and you want to be able to move on, and go into the next one each time. And something, yeah, it's funny, but what I've found personally is that obviously it's really important for your own emotional state and mental state to be able to do that, and we've had lots of mistakes, and I've had a lot.

中文翻译:

我思考很多的一件事是：某种程度上，你真的想尽力而为，投入一切，做你能做的一切……基本上，你想在赛场上倾尽所有。但与此同时，你要能坦然面对输赢，并且能够继续前进，每次都投入到下一个任务中。这很有趣，但我个人的发现是，能够做到这一点对你自己的情绪状态和心理状态显然非常重要。我们犯过很多错误，我也犯过很多。

[01:28:49] Scott Wu

English:

I had my first company, which obviously, which was cool, but there are a lot of tricky spots there, and then over the course of Cognition, it feels like it's been already eight years compressed into one year, and it's still going at that pace. But somehow it also actually makes you more successful, I think, too. It's like, you are just more able to give it your best, and to do the things that will lead to success if you're not tying it up in your own personal worth.

中文翻译:

我的第一家公司虽然很酷，但也有很多棘手的地方。而在 Cognition 的过程中，感觉就像是把八年的经历压缩进了一年，而且节奏依然如此。但不知何故，我认为这实际上也会让你更成功。如果你不把成败与个人价值挂钩，你就更能尽力而为，去做那些通往成功的事情。

[01:29:19] Lenny Rachitsky

English:

That is so interesting, I just had a podcast recording recently where with an executive coach, Jerry Kelowna, that I think will come out before this, might be after this, that's one of his big pieces of advice, and it's a very Buddhist approach of just [inaudible 01:29:33] and attaching to a certain outcome.

中文翻译:

这太有趣了。我最近刚和一位高管教练 Jerry Kelowna 录了一期播客（可能在这一期之前或之后发布），那是他的核心建议之一。这是一种非常佛教的方法，即不执着于特定的结果。

[01:29:35] Scott Wu

English:

Yeah.

中文翻译:

是的。

[01:29:36] Lenny Rachitsky

English:

Okay. Final question, I'm curious if there's a story here, but we could keep it short, is there a story behind Devin as the name, or is there another contender for Devin being the [inaudible 01:29:48]?

中文翻译:

好。最后一个问题，我很好奇这里面是否有故事，我们可以简短点：Devin 这个名字背后有什么故事吗？或者当时还有其他备选名字吗？

[01:29:47] Scott Wu

English:

Devin was the name from pretty early on, we were interested, we were working on coding agents from the beginning, and my co-founders are Steven and Walden, for example, and we had this idea, all right, let's get started, and let's try to expand the box as much as we can. So, have everyone think out of the box and do their own thing, and let's have everyone do their own thing first for a bit, and then we'll consolidate and take everything that we've learned. And so, Walden made a virtual developer version of him, which was called DevWalden, and then Steven made one of him, which is called DevSteven, and we had all these... And then we were kind of combining it all into one thing, and we're like, okay, you know? It's Devin. And that was the thing. And so Devin was, yeah, Devin stuck for us quite early on, I would say.

中文翻译:

Devin 这个名字很早就定下来了。我们从一开始就对编程智能体感兴趣。我的联合创始人是 Steven 和 Walden。我们当时想：“好，让我们开始吧，尽量打破常规。”让每个人都跳出框框去做自己的事情，先让每个人独立尝试一段时间，然后我们再整合并吸收学到的一切。于是，Walden 做了一个他的虚拟开发者版本，叫 DevWalden；Steven 也做了一个，叫 DevSteven。我们有了所有这些……然后我们把它们整合在一起，我们觉得：“好吧，就叫 Devin 吧。”事情就是这样。所以 Devin 这个名字很早就被我们沿用了。

[01:30:31] Scott Wu

English:

One thing which we did have a big decision on though actually is what the image of Devin would be. And so, as folks know, there's the hexagons and then people have seen this more recently, but there's actually also an otter, a little otter with a laptop in its lap, and that is Devin as well. And we had this debate over what to go with, and what not to go with and stuff, and it's been a while now, but somehow we still have both the hexagons and the otter.

中文翻译:

不过，我们确实在 Devin 的形象上做了一个重大决定。大家知道，我们有六边形的标志，但最近人们也看到了，其实还有一个水獭的形象——一只腿上放着笔记本电脑的小水獭，那也是 Devin。我们曾争论该用哪一个，现在已经过去一段时间了，但不知何故，我们仍然同时保留了六边形和水獭。

[01:31:01] Lenny Rachitsky

English:

You skipped over where the Devin, did you have just have, it just came to you?

中文翻译:

你跳过了 Devin 这个词是怎么来的，是突然想到的吗？

[01:31:06] Scott Wu

English:

Oh, so Devin is, it's a dev. Yeah.

中文翻译:

哦，Devin 就是“Dev”（开发者）。是的。

[01:31:10] Lenny Rachitsky

English:

[inaudible 01:31:10].

中文翻译:

（听不清）

[01:31:09] Scott Wu

English:

And so, it was kind of like, when we were consolidating all the names, it just seemed clear then that this would be the universal dev that we all liked to work with.

中文翻译:

所以，当我们整合所有名字时，就很清楚了：这将是我们都喜欢与之协作的通用开发者（universal dev）。

[01:31:17] Lenny Rachitsky

English:

Wow.

中文翻译:

哇。

[01:31:18] Scott Wu

English:

Yeah.

中文翻译:

是的。

[01:31:18] Lenny Rachitsky

English:

Incredible. Scott, this was so much fun. Oh my God, I learned a ton, which is always a really good sign. Two final questions, where can folks find you/Devin/anything else you want to point them to, and how can listeners be useful to you?

中文翻译:

太不可思议了。Scott，这真的很有趣。天哪，我学到了很多，这总是一个好兆头。最后两个问题：大家可以在哪里找到你、Devin 或其他你想推荐的东西？听众可以如何帮到你？

[01:31:29] Scott Wu

English:

Awesome. Yeah, no, we're at App.devin.ai, and you can find us as well on Twitter or a lot of other social media. We'd obviously love to hear any feedback you have about the Devin product, there's so much to figure out, and I think the, like I said, I think we're all still 20 steps away from really the future of software engineering, and so it really means a lot to hear what folks think about the product as they're trying it out. And so, please let us know anytime if there's things that we can do to make it better.

中文翻译:

太棒了。我们的网址是 App.devin.ai，你也可以在 Twitter 或许多其他社交媒体上找到我们。我们显然很想听听你对 Devin 产品的任何反馈。还有很多事情需要摸索，就像我说的，我认为我们距离真正的软件工程未来还有 20 步之遥。所以，听到人们在尝试产品时的想法对我们意义重大。请随时告诉我们，我们可以做些什么来让它变得更好。

[01:32:00] Lenny Rachitsky

English:

Scott, thank you so much for being here.

中文翻译:

Scott，非常感谢你能来。

[01:32:02] Scott Wu

English:

Thank you so much for having me. I had a great time.

中文翻译:

非常感谢邀请我。我聊得很开心。

[01:32:04] Lenny Rachitsky

English:

Me too. Bye everyone.

中文翻译:

我也是。大家再见。

[01:32:07] Lenny Rachitsky

English:

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:

非常感谢您的收听。如果您觉得这期节目有价值，可以在 Apple Podcasts、Spotify 或您喜欢的播客应用中订阅本节目。此外，请考虑给我们评分或留下评论，因为这能真正帮助其他听众发现这个播客。您可以在 LennysPodcast.com 找到所有往期节目或了解更多信息。下期节目见。