RYAN J SALVA

LENNY'S PODCAST

BILINGUAL TRANSCRIPT

ORIGINAL BY

Lenny Rachitsky

@lennysan • x.com/lennysan

ANALYSIS BY

@Penny777 • x.com/penny777

Generated: 2026-01-16

Ryan J. Salva - 双语对照

This is the complete bilingual transcript for Lenny’s Podcast featuring Ryan J. Salva, VP of Product at GitHub.

[00:00:00] Ryan J. Salva

English:

We had actually created a snapshot of GitHub's public code for what we call the Arctic Code Vault, right? Essentially, this is up in like way in the Northlands of Finland, there's a seed vault. We were like, you know what? Seed vaults are really there to preserve the diversity of the world's flora in seeds in case of some crazy either natural or manmade disaster. But another really important asset to the world is our code, our open source. This represents actually a lot of the collective, well, certainly software, if not intelligence of kind of the modern world, right?

中文翻译:

我们实际上为 GitHub 的公开代码创建了一个快照，用于我们所谓的“北极代码库”（Arctic Code Vault），对吧？基本上，这就像是在芬兰遥远的北方，那里有一个种子库。我们当时想，你知道吗？种子库的存在是为了在发生疯狂的自然或人为灾难时，通过种子保存世界植物群的多样性。但世界上另一个非常重要的资产是我们的代码，我们的开源项目。这实际上代表了现代世界的许多集体成果，即便不是集体智慧，也肯定是集体软件成果，对吧？

[00:00:44] Ryan J. Salva

English:

We had put this snapshot of public repositories on this silver film that would be preserved for thousands of years in this Arctic Code Vault. Well, we took that same data snapshot and we brought it to our friends over at OpenAI to see like, okay, what can we do with these large language models built on public code? Well, it turns out we can do some pretty cool things.

中文翻译:

我们将这些公开仓库的快照存放在银制胶片上，这些胶片可以在北极代码库中保存数千年。后来，我们带着同样的这份数据快照找到了 OpenAI 的朋友，想看看：好吧，利用这些基于公开代码构建的大语言模型（LLM），我们能做些什么？结果证明，我们可以做一些非常酷的事情。

[00:01:13] Lenny

English:

Ryan Salva is VP of product at GitHub, where, amongst other projects, he incubated and launched GitHub Copilot, which in my opinion is one of the most magical products that you'll come across. If you haven't heard of it, it uses OpenAI's machine learning engine to autocomplete code for engineers in real time as they're coding. I think it's one of the biggest advances in product development and productivity that we've seen in a while. I'm always really curious how a big product like this starts, gets buy in, build momentum, and then launches, especially at a big company like Microsoft and especially a product like Copilot that has surprising ethics challenges, scaling challenges, business model questions.

中文翻译:

Ryan Salva 是 GitHub 的产品副总裁。在众多项目中，他孵化并发布了 GitHub Copilot。在我看来，这是你能遇到的最神奇的产品之一。如果你还没听说过它，它使用的是 OpenAI 的机器学习引擎，在工程师编写代码时实时为他们自动补全代码。我认为这是我们一段时间以来在产品开发和生产力领域看到的重大进步之一。我一直很好奇这样一个大型产品是如何开始的、如何获得支持、如何积攒势头并最终发布的，尤其是在微软这样的大公司，而且像 Copilot 这样的产品还面临着令人惊讶的伦理挑战、扩展挑战和商业模式问题。

[00:01:55] Lenny

English:

Also, this came out of a small R&D team that GitHub has, and it's so interesting to hear what Ryan has learned about incubating big bets within a large company, and then taking them from prototype to Microsoft scale. Ryan is also just super interesting as a human. He's got a very non-traditional background. I am excited for you to hear this conversation. With that, I bring you Ryan Salva. If you're setting up your analytics stack, but you're not using Amplitude, what are you doing? Amplitude is the number one most popular analytics solution in the world used by both big companies like Shopify, Instacart, and Atlassian, and also most tech startups.

中文翻译:

此外，这个项目出自 GitHub 的一个小研发（R&D）团队。听 Ryan 分享在大公司内部孵化“豪赌”型项目，并将其从原型推向微软级别的规模，这非常有趣。Ryan 本人也是个非常有意思的人，他有着非常非传统的背景。我很期待让大家听到这段对话。下面，让我们欢迎 Ryan Salva。如果你正在搭建分析栈（analytics stack）却没用 Amplitude，那你是在干嘛呢？Amplitude 是全球最受欢迎的分析解决方案，无论是 Shopify、Instacart 和 Atlassian 这样的大公司，还是大多数科技初创公司都在使用它。

[00:02:38] Lenny

English:

Amplitude has everything you need, including a powerful and fully self-service analytics product, an experimentation platform, and even an integrated customer data platform to help you understand your users like never before. Give your teams self-service product data to understand your users, drive conversions, and increase engagement, growth, and revenue. Ditch your vanity metrics, trust your data, work smarter, and grow your business. Try Amplitude for free. Just visit Amplitude.com to get started. This episode is brought to you by Athletic Greens. I've been hearing about AG1 on basically every podcast that I listen to, like Tim Ferriss and Lex Fridman.

中文翻译:

Amplitude 拥有你所需的一切，包括强大且完全自助式的分析产品、实验平台，甚至还有一个集成的客户数据平台（CDP），帮助你以前所未有的深度了解用户。为你的团队提供自助式产品数据，以洞察用户、推动转化并增加参与度、增长和收入。抛弃虚荣指标，信任你的数据，更聪明地工作并发展业务。免费试用 Amplitude，访问 Amplitude.com 即可开始。本集节目由 Athletic Greens 赞助。我基本上在听的每个播客里都能听到 AG1，比如 Tim Ferriss 和 Lex Fridman 的节目。

[00:03:20] Lenny

English:

I finally gave it a shot earlier this year, and it has quickly become a core part of my morning routine, especially on days that I need to go deep on writing or record a podcast like this. Here's three things that I love about AG1. One, with a small scoop that dissolves in water, you are absorbing 75 vitamins, minerals, probiotics, and adaptogens. I kind of like to think of it as little safety net for my nutrition in case I've missed something in my diet. Two, they treat AG1 like a software product. Apparently they're on their 52nd iteration and they're constantly evolving it based on the latest science, research studies, and internal testing that they do.

中文翻译:

今年早些时候我终于尝试了一下，它很快就成了我早晨例行公事的核心部分，尤其是在我需要深入写作或录制像这样的播客的日子里。关于 AG1，我有三点非常喜欢。第一，只需一小勺溶于水，你就能吸收 75 种维生素、矿物质、益生菌和适应原（adaptogens）。我喜欢把它看作我营养的小安全网，以防我在饮食中遗漏了什么。第二，他们把 AG1 当作软件产品来对待。显然他们已经进行了第 52 次迭代，并且不断根据最新的科学研究和内部测试进行演进。

[00:03:59] Lenny

English:

And three, it's just one easy thing that I can do every single day to take care of myself. Right now, it's time to reclaim your health and arm your immune system with convenient daily nutrition. It's just one scoop and a cup of water every day. And that's it. There's no need for a million different pills and supplements to look out for your health. Make it easy. Athletic Greens is going to give you a free one year supply of immune supporting vitamin D and five free travel packs for your first purchase. All you have to do is visit AthleticGreens.com/lenny. Again, that's AthleticGreens.com/lenny to take ownership over your health and pick up the ultimate daily nutritional insurance. Ryan, welcome to the podcast.

中文翻译:

第三，这是我每天都能轻松做到的照顾自己的一件事。现在是时候重获健康，用便捷的每日营养武装你的免疫系统了。每天只需一勺粉末和一杯水，就这么简单。不需要为了健康去吃无数种不同的药片和补充剂。让事情变简单。Athletic Greens 将为你首次购买提供免费的一年分免疫支持维生素 D 和五个免费旅行包。你只需访问 AthleticGreens.com/lenny。再次强调，访问 AthleticGreens.com/lenny，掌控你的健康，获取终极每日营养保障。Ryan，欢迎来到本播客。

[00:04:42] Ryan J. Salva

English:

Thank you, my friend. I am genuinely very excited to be here. Lovely to geek out with you for a little while.

中文翻译:

谢谢你，我的朋友。我真的很兴奋能来到这里。很高兴能和你一起钻研技术聊上一会儿。

[00:04:48] Lenny

English:

I'm excited as well. We were chatting briefly before we started recording and you mentioned a little bit about your background, which is really unique for someone that is leading product at GitHub. Could you just share what you studied in school, and then briefly just how that led to your career in product management?

中文翻译:

我也很兴奋。在开始录音前我们简短聊了聊，你提到了你的背景，对于一个在 GitHub 领导产品的人来说，这真的很独特。你能分享一下你在学校学的是什么，以及这又是如何引导你走向产品管理职业生涯的吗？

[00:05:07] Ryan J. Salva

English:

Oh wow! You're going to make me remember all the way back to school. Okay. Back in school, I was not a classic software engineering, CS major. The kind of esoteric answer is philosophy of aesthetics and 20th century critical theory. The easier access answer is philosophy and English. But primarily it was really about how do we, as people, communicate with each other, how do we express ourselves through creativity. As humans since the dawn of time have been painting on cave walls and dancing around the fire and writing stories and novels and singing to each other. I was just really interested in how we convey our experience of the world to others.

中文翻译:

噢哇！你要让我回想很久以前在学校的日子了。好吧。在学校时，我不是典型的软件工程或计算机科学（CS）专业。比较深奥的回答是：美学哲学和 20 世纪批判理论。通俗一点的回答是：哲学和英语。但核心其实是关于我们作为人如何相互交流，如何通过创造力表达自己。人类自古以来就在洞穴墙壁上绘画、围着火堆跳舞、写故事小说、互相歌唱。我只是对我们如何向他人传达我们对世界的体验非常感兴趣。

[00:05:58] Ryan J. Salva

English:

I got started in software development and product management because I wanted to be in the business of creativity. We're at a really, really unique time in human history where we actually get to witness the advent of a brand new medium. Software development and the worlds that it creates wasn't possible, I don't know, maybe 50, 60 years ago now. If I'd been born in the 1700s, I probably would've been the guy making, I don't know, new colors of paint and paint brushes, but I wasn't. I was born kind of at the turn of the 21st century, and so I work in engineering.

中文翻译:

我开始从事软件开发和产品管理，是因为我想投身于“创造力”这一行。我们正处于人类历史上一个非常独特的时期，我们实际上见证了一种全新媒介的诞生。软件开发及其创造的世界在大概 50、60 年前是不可能的。如果我出生在 18 世纪，我可能是一个制造新油漆颜色和画笔的人，但我不是。我出生在 21 世纪之交，所以我从事工程工作。

[00:06:39] Ryan J. Salva

English:

That's what I've been doing for the last about a little bit more than 20 years now, working sometimes in startups, some of them other people, some of them my own, about 10 years at Microsoft and now three years at GitHub.

中文翻译:

这就是我过去 20 多年一直在做的事情，有时在初创公司工作，有些是别人的，有些是我自己的，在微软工作了大约 10 年，现在在 GitHub 工作了 3 年。

[00:06:51] Lenny

English:

Amazing. I didn't know that was a job to make new paint colors for paint brushes. Is there a color you would come up with?

中文翻译:

太神奇了。我以前不知道为画笔制造新油漆颜色也是一份工作。你会想出什么颜色?

[00:06:59] Ryan J. Salva

English:

Oh man! It so happens that yellow... I think I would do a really vibrant gold sunshine yellow if I was in that business.

中文翻译:

噢天哪！碰巧黄色……如果我做那行，我想我会做一种非常充满活力的金色阳光黄。

[00:07:13] Lenny

English:

Very positive, happy. I love it. That could be a new GitHub brand color. Today, you're VP of product at GitHub. Before that, you were a super senior product leader at Microsoft, and I'm always curious how that transition happens when you move from just a longtime senior product leader at a larger company to taking on something like this that was an acquisition. I'm curious what made you decide to take this leap, and then just was there anything interesting about the machination that went into just making that transition and figuring that out?

中文翻译:

非常积极、快乐。我喜欢。那可以成为 GitHub 的新品牌色。今天，你是 GitHub 的产品副总裁。在此之前，你是微软的高级产品领导者。我一直很好奇，当你从一家大公司的资深产品领导者转变为负责像 GitHub 这样被收购的项目时，这种转变是如何发生的。我很想知道是什么让你决定实现这一跨越，以及在完成这种转变和理清思路的过程中，有没有什么有趣的运作过程?

[00:07:45] Ryan J. Salva

English:

Yeah, it's a good question. Like I said, I was working on development tools and developer services when I was there at Microsoft. Specifically, I was leading product for what they call One Engineering System. It's essentially the shared developer infrastructure for all Microsoft products like Windows and Office and

Azure and things like that, as well as Microsoft's DevOps solution called Azure DevOps. When the acquisition happened, it was clear that so much of the energy, so much of the focus and the innovation that was going to be happening around developer tools and services was going to be happening around GitHub. I mean, that's where the community is creating.

中文翻译:

是的，这是个好问题。正如我所说，我在微软时负责开发工具和开发者服务。具体来说，我当时领导着他们所谓的“单一工程系统”（One Engineering System）的产品工作。这基本上是所有微软产品（如 Windows、Office 和 Azure 等）共享的开发者基础设施，以及微软名为 Azure DevOps 的开发运维解决方案。当收购发生时，很明显，开发者工具和服务领域的大部分精力、关注点和创新都将围绕 GitHub 展开。我的意思是，那是社区进行创造的地方。

[00:08:34] Ryan J. Salva

English:

That's where people are learning, that's where so much of the mind share of just the development community is focused. Like I said, I'm motivated. What I care about is helping people create. It was very clear to me that there was no place that I could have a larger impact than working at GitHub. I really took that opportunity to make the transition out of a little bit more enterprise focused internal role at Microsoft to going where I could work on everything from, I don't know, AI technology like Copilot to a cloud hosted development environments like Codespaces, repos, which literally every single developer on the planet is participating in some way GitHub repos in a typical year.

中文翻译:

那是人们学习的地方，也是开发者社区大部分注意力集中的地方。正如我所说，我很有动力。我关心的是帮助人们创造。对我来说非常清楚的是，没有哪个地方能比在 GitHub 工作产生更大的影响力了。我抓住了那个机会，从微软内部一个更侧重于企业的角色转变为可以接触到一切的地方——从像 Copilot 这样的 AI 技术，到像 Codespaces 这样的云托管开发环境，再到仓库（repos），地球上几乎每一个开发者在一年中都会以某种方式参与 GitHub 仓库。

[00:09:28] Ryan J. Salva

English:

That was what I wanted to accomplish, is just like, how do I get more connected to the community, especially the community outside of what Microsoft could reach on its own. The decision to move as well, I think, was really focused not just on what GitHub was and maybe is at the time, but what GitHub also can be. I mean, GitHub has more than a decade, nearly a decade and a half of history of bringing developers together to collaborate on code through repositories. But in the last few years, we've really expanded that portfolio to include so many different parts of the developer life cycle.

中文翻译:

那就是我想实现的：如何与社区建立更紧密的联系，尤其是微软自身无法触及的那些社区。我认为，做出变动的决定不仅关注 GitHub 当时的样子，还关注 GitHub 未来的可能性。我的意思是，GitHub 有超过十年、将近十五年的历史，通过仓库将开发者聚集在一起协作编写代码。但在过去的几年里，我们真正扩展了产品组合，涵盖了开发者生命周期的许多不同部分。

[00:10:13] Ryan J. Salva

English:

Again, I talked there about Codespaces and Copilot, but it's also actions for CI/CD and advanced security. As developers, we are so much more than just where we put our code. There's a whole part of the tool chain there. And to get to an opportunity to work on so many V1 products, like that is creation itself, to be able to build an entirely new product, get it out to market, test it, iterate on it, and really feed on the energy that's coming back from the community.

中文翻译:

我刚才提到了 Codespaces 和 Copilot，但还有用于 CI/CD（持续集成/持续部署）的 Actions 和高级安全功能。作为开发者，我们的工作远不止于存放代码的地方。那里有一整套工具链。能有机会参与这么多“V1”（第一版）产品的开发，这本身就是一种创造——能够构建一个全新的产品，将其推向市场，进行测试、迭代，并真正从社区反馈的能量中汲取养分。

[00:10:46] Lenny

English:

Awesome. There's definitely a lot of energy coming out of GitHub. What I want to spend most of our time chatting about is a product that your team helped launch and incubate, which is GitHub Copilot, which just from my outsider perspective feels like one of the biggest advances in software development in, I don't know, decade, maybe more. It's definitely one of the most magical products out there and your team and you kind of led the incubation and launch of the Copilot.

中文翻译:

太棒了。GitHub 确实释放出了巨大的能量。我想把大部分时间花在聊聊你们团队协助发布和孵化的一个产品上，那就是 GitHub Copilot。从我这个局外人的角度来看，它感觉像是软件开发领域过去十年甚至更久以来最大的进步之一。它绝对是目前最神奇的产品之一，而你和你的团队领导了 Copilot 的孵化和发布。

[00:11:15] Lenny

English:

I'd love to spend most of our time chatting through that. The first question... Okay, cool. My first question just for folks that don't know a lot about Copilot is just like, what is it? Can you just kind of briefly describe what Copilot is?

中文翻译:

我很想花大部分时间聊聊这个。第一个问题……好的。我的第一个问题是针对那些不太了解 Copilot 的人：它到底是什么？你能简要描述一下 Copilot 吗？

[00:11:26] Ryan J. Salva

English:

Yeah, sure. Developers for the last 20 years or more have had essentially simple, intelligent autocomplete. You hit the period and you get the next variable that might come up. It's helpful for moving a little bit faster through your code, helpful sometimes for remembering what the particular syntax might look like for a method or a function. Copilot is essentially that magnified by many lines of code. It is multi-line autocomplete that is fundamentally powered by an AI model called CodeX, which is a derivative of another one that you might be familiar with, GPT-3.

中文翻译:

好的，没问题。在过去的 20 年或更长时间里，开发者基本上使用的是简单的智能自动补全。你输入一个点（.），就会出现下一个可能出现的变量。这有助于稍微加快代码编写速度，有时也有助于记住方法或函数的特定语法。Copilot 基本上就是这种功能的放大版，涵盖了多行代码。它是多行自动补全，其核心动力来自一个名为 CodeX 的 AI 模型，它是你可能熟悉的另一个模型 GPT-3 的衍生品。

[00:12:15] Ryan J. Salva

English:

When you are in the editor, it could be VS Code, it could be IntelliJ, it could be them, essentially, as you are typing, Copilot will provide suggestions usually in kind of this italicized gray text that is really, to your point, kind of magical what it's able to infer. Based upon the variables around it, the class names, the method names around it, your comments, Copilot infers what you intend to create, and then hopefully does a pretty good job at nailing it by providing scaffolding code template that you can then riff on. Now, what we tend to find is that developers love it. They really enjoy it. They kind of find themselves getting a little addicted to it because it helps them stay in the flow.

中文翻译:

当你在编辑器中（可以是 VS Code、IntelliJ 或 Vim），基本上当你打字时，Copilot 会提供建议，通常以这种灰色斜体文本的形式出现。正如你所说，它能推断出的内容简直不可思议。根据周围的变量、类名、方法名以及你的注释，Copilot 会推断你打算创建什么，然后通过提供你可以进一步修改的脚手架（scaffolding）代码模板，希望能精准地完成任务。现在我们发现开发者非常喜欢它。他们真的很享受，甚至发现自己有点上瘾，因为它能帮助他们保持在“心流”（flow）状态。

[00:13:08] Ryan J. Salva

English:

As developers, we love to be in that place. I love to be in that place where I'm creating things, where I'm focusing on some product, some piece of software that I'm going to give to my customers, my users. The labor of remembering what's the order of parameters that need to come into a particular API, or hey, what's the particular syntax of this thing I'm supposed to do, or oh, I've got to create a bunch of dummy data that is days of the week or months in the year. That's just labor. It's not creating. It's just typing.

中文翻译:

作为开发者，我们喜欢处于那种状态。我喜欢处于那种正在创造东西的状态，专注于某个产品、某段我要交给客户或用户的软件。而那些记忆特定 API 参数顺序的劳动，或者“嘿，我该做的这个东西的具体语法是什么”，或者“噢，我得创建一堆关于星期几或一年中月份的模拟数据”，这些只是体力活。这不是创造，这只是打字。

[00:13:47] Ryan J. Salva

English:

Copilot helps developers stay in the flow by bringing all of that information into the editor, preventing them from having to go check out documentation or watch tutorial or go to Stack Overflow and either find an answer or worse, have to ask a question and wait for an answer. It just brings all of that into the editor and gives the developer often multiple suggestions that they can choose from and just pick and choose what is the right solution to solve the problem for the thing they're trying to create.

中文翻译:

Copilot 通过将所有这些信息带入编辑器来帮助开发者保持心流，让他们不必去查阅文档、看教程或上 Stack Overflow 寻找答案，甚至更糟的是——还得提问并等待回答。它直接把所有这些都带进编辑器，通常会给开发者提供多个建议供其选择，让他们只需挑选出最合适的方案来解决他们正在创造的东西所面临的问题。

[00:14:21] Lenny

English:

Awesome. What I'm most curious about, and we're going to spend time on this, is just how a product like this comes to be at a larger company. But before we get into that, what's the craziest story of someone using Copilot to write code? And I'll share one real quick. I was watching some YouTube videos to prepare for this chat and one guy, maybe this is the Turing Test of AI writing code, is he used Copilot to center divs. He's like, "Wow! This did it right." And then another guy, he's an instructor of code.

中文翻译:

太棒了。我最想知道的是（我们也会花时间聊这个），这样一个产品是如何在大公司里诞生的。但在深入探讨之前，关于有人使用 Copilot 编写代码，你听过最疯狂的故事是什么？我先快速分享一个。为了准备这次聊天，我看了一些 YouTube 视频，有一个人——这也许是 AI 写代码的图灵测试——他用 Copilot 来让 div 居中。他惊呼："哇！它居然做对了。" 还有一个人，他是个编程讲师。

[00:14:51] Lenny

English:

He makes YouTube videos teaching people how to code and he's like, "Copilot just gives you the answer immediately, and so I can't make these videos as easily. I have to turn it off so that doesn't just give it away." I'm curious, what have you seen?

中文翻译:

他制作 YouTube 视频教人们如何编程，他说："Copilot 直接就把答案秒出了，所以我没法像以前那样轻松地拍视频了。我必须把它关掉，免得它直接把答案泄露了。" 我很好奇，你见过什么有趣的事?

[00:15:03] Ryan J. Salva

English:

There are so many of those. I'll just kind of give a couple of recent ones that I've heard. I was talking to one developer who was... He's actually an educator and he's teaching kids how to code, usually like kind of high school age, so 16, 15, that kind of thing. His experience matches my own, which is that many of us, we learn to code best not by arbitrary exercises, but by actually building something that's going to be useful solving problems.

中文翻译:

这样的故事太多了。我只分享几个我最近听到的。我曾和一位开发者聊天，他实际上是一名教育工作者，教孩子们编程，通常是高中年龄段，比如 15、16 岁。他的经验和我的一样：我们中的许多人学习编程的最佳方式不是通过随意的练习，而是通过实际构建一些有用的、能解决问题的东。

[00:15:41] Ryan J. Salva

English:

What he does is he matches small businesses and medium size businesses who need to build internal tools with essentially classes of students, like a group of maybe six or eight students, and then gives those students Copilot and says, "Here, small business, medium size business. Group of students, go build this internal tool for this business."

中文翻译:

他所做的是将需要构建内部工具的中小企业与学生班级（比如一组 6 到 8 名学生）进行匹配，然后给这些学生配备 Copilot 并说："来，这是中小企业，这是学生小组，去为这家企业构建这个内部工具吧。"

[00:16:08] Ryan J. Salva

English:

Copilot is essentially kind of whispering in the student's ear, metaphorically speaking, "Hey, here's how you solve this problem. Here's how you do this," and students build not only the tool, the software that the business needs and then get to put that on their resume and their application for college and university, but they also get to learn by using the tools that likely are going to be part of the core DNA of the developer tool chain two, three, four years from now, as AI starts to permeate our entire stack. That was a pretty cool recent one that I talked to.

中文翻译:

Copilot 基本上是在学生耳边低语（打个比方）："嘿，这就是解决这个问题的方法。这就是你该做的。" 学生们不仅构建了工具和企业所需的软件，并能将其写进简历和大学申请中，而且他们还通过使用这些工具进行了学习。这些工具很可能在两三四年后成为开发者工具链核心基因的一部分，因为 AI 开始渗透到我们的整个技术栈中。这是我最近聊到的一个非常酷的案例。

[00:16:48] Lenny

English:

That is very cool. I didn't think about just the education lever here of just making it so much easier to learn to code, not even just building code.

中文翻译:

那真的很酷。我之前没考虑到这里的教育杠杆作用，它不仅能构建代码，还能让学习编程变得容易得多。

[00:16:56] Ryan J. Salva

English:

And that's the thing, Copilot is particularly good not just at taking away some of the effort, but often... There's learning a new language, and then there's also just waiting into a code base that you're not necessarily familiar with, right? I mean, heck, sometimes I don't recognize some of the code that I wrote six months ago or a year ago. It feels like I'm wading into new territory. But maybe you need to fix a bug in an app that you don't often touch, wading into that code base is kind of learning and creating a mental map for that code base.

中文翻译:

这就是关键所在，Copilot 特别擅长的不只是减轻工作量，还有……学习一门新语言是一回事，而进入一个你不熟悉的现有代码库又是另一回事，对吧？我的意思是，见鬼，有时候我连自己六个月或一年前写的代码都认不出来。感觉就像进入了新领地。但也许你需要修复一个你不常碰的应用中的 bug，进入那个代码库的过程其实就是在学习并为该代码库建立心智模型（mental map）。

[00:17:30] Ryan J. Salva

English:

One of the really magical pieces of Copilot here is that, that AI is collecting context of the application that you're going into. It can help you build that mental map and learn the code base, even if it's a language that you're already familiar with.

中文翻译:

Copilot 最神奇的地方之一在于，AI 正在收集你进入的应用程序的上下文。它可以帮助你建立心智模型并学习代码库，即使那是一门你已经熟悉的语言。

[00:17:47] Lenny

English:

Going back to the beginning of Copilot and how it started, I'm always curious how a project that ends up being a huge deal to a larger company begins and especially how it builds momentum, how it gets buy in, and then just gets out the door. Can you talk about just the original seed of this idea like, who did it come from, who had the original vision, how did this idea emerge and build momentum where you put resources into it?

中文翻译:

回到 Copilot 的起点以及它是如何开始的，我一直很好奇一个最终对大公司产生巨大影响的项目是如何开始的，特别是它如何积攒势头、如何获得支持，然后最终发布。你能谈谈这个想法最初的种子吗？比如，它来自谁？谁拥有最初的愿景？这个想法是如何浮现并积攒势头，让你们决定投入资源的？

[00:18:13] Ryan J. Salva

English:

Oh wow, what a long, and I don't know, depending upon your point of view, sorted or exciting story that is. Microsoft and OpenAI have been collaborating for quite a while now on large language models, making its way into all different experiments and different parts of both Microsoft's software portfolio, as well as just helping OpenAI by providing the compute necessary. It takes massive amounts of compute to train these models. They were mostly large language models. Couple years ago now, it kind of dawned on us that, well, language models aren't just English and Spanish and German and Korean and Japanese, but Python and JavaScript and Java and C# and Closure.

中文翻译:

噢哇，这真是一个漫长且（取决于你的视角）曲折或令人兴奋的故事。微软和 OpenAI 在大语言模型上的合作已经有一段时间了，这些模型正进入各种实验以及微软软件产品组合的不同部分，同时也通过提供必要的算力来帮助 OpenAI。训练这些模型需要海量的算力。它们大多是大语言模型。几年前，我们突然意识到：语言模型不只是英语、西班牙语、德语、韩语和日语，还有 Python、JavaScript、Java、C# 和 Clojure。

[00:19:07] Ryan J. Salva

English:

All of these are languages too. In fact, they're kind of nice from an AI perspective because they're relatively constrained in terms of their semantics, right? The number of words, I put that the in scare quotes as it were, that can be expressed in Python, for example, is much smaller than the English language, which has all sorts of different grammar rules and nouns, verbs, adjectives, adverbs. We started to see what it would be like to actually bring code to these large language models. The way that I actually got introduced to it is kind of funny. Microsoft and OpenAI had this idea.

中文翻译:

这些也都是语言。事实上，从 AI 的角度来看，它们挺不错的，因为它们的语义相对受限，对吧？例如，Python 中可以表达的“单词”（我在这里给单词加个引号）数量比英语小得多，英语有各种不同的语法规则、名词、动词、形容词和副词。我们开始观察如果把代码引入这些大语言模型会是什么样子。我最初接触到它的方式其实挺逗的。微软和 OpenAI 有了这个想法。

[00:19:53] Ryan J. Salva

English:

At the time, one of the teams that I was responsible for was GitHub's infrastructure team, the team responsible for our data centers, our reliability, our rep time. We noticed one day that we were getting hammered, I mean absolutely hammered with a tremendous amount of clone requests. We're like, "Oh my gosh! Is this like a denial of service attack? How are we going to respond to this? What's going to happen?" We figured out pretty quickly that it was actually OpenAI. They were cloning all of our repositories to harvest the data out of GItHub.I mean, it's totally legit practice, but it does have a real consequence.

中文翻译:

当时，我负责的团队之一是 GitHub 的基础设施团队，负责我们的数据中心、可靠性和运行时间。有一天我们注意到，我们正遭受猛烈攻击，我是说被海量的克隆（clone）请求狂轰滥炸。我们当时想：“噢我的天！这是拒绝服务（DoS）攻击吗？我们该如何应对？会发生什么？” 我们很快发现，那其实是 OpenAI。他们正在克隆我们所有的仓库，以便从 GitHub 中提取数据。我的意思是，这完全是合法的做法，但确实产生了实际后果。

[00:20:33] Ryan J. Salva

English:

We were able to step in and mitigate it very quickly. There was not a reliability kind of an uptime incident there, but we're like, "Hey, you all, cool. Love this thing. Let's see if we can get that data to you in a more responsible way, in a way that's packaged a little bit more to meet your needs." What we did is just the year before that, We had actually created a snapshot of GitHub's public code for what we call the Arctic Code Vault, right? Essentially, this is up in like way in the Northlands of Finland, there's a seed vault. We were like, you know what? Seed vaults are really there to preserve the diversity of the world's flora in seeds in case of some crazy either natural or manmade disaster.

中文翻译:

我们得以介入并迅速缓解了压力。当时并没有发生可靠性或运行时间事故，但我们说：“嘿，各位，很酷，我们喜欢这玩意。让我们看看能不能以一种更负责任的方式把数据给你们，以一种更符合你们需求的方式进行打

包。” 我们所做的是，就在那一年之前，我们实际上为 GitHub 的公开代码创建了一个快照，用于我们所谓的“北极代码库”，对吧？基本上，这就在芬兰遥远的北方，那里有一个种子库。我们当时想，你知道吗？种子库的存在是为了在发生疯狂的自然或人为灾难时，通过种子保存世界植物群的多样性。

[00:21:25] Ryan J. Salva

English:

But another really important asset to the world is our code, our open source. This represents actually a lot of the collective, well, certainly software, if not intelligence of kind of the modern world, right? This represents actually a lot of the collective, well, certainly software, if not intelligence of kind of the modern world. We had put this snapshot of public repositories on this silver film that would be preserved for thousands of years in this Arctic Code Vault. Well, we took that same data snapshot and we brought it to our friends over at OpenAI to see like, okay, what can we do with these large language models built on public code?

中文翻译:

但世界上另一个非常重要的资产是我们的代码，我们的开源项目。这实际上代表了现代世界的许多集体成果，即便不是集体智慧，也肯定是集体软件成果，对吧？这代表了现代世界的许多集体成果。我们将这些公开仓库的快照存放在银制胶片上，这些胶片可以在北极代码库中保存数千年。后来，我们带着同样的这份数据快照找到了 OpenAI 的朋友，想看看：好吧，利用这些基于公开代码构建的大语言模型，我们能做些什么？

[00:22:03] Ryan J. Salva

English:

Well, it turns out we can do some pretty cool things. Just like a translation tool that goes from English to Spanish, Spanish to German, you can also go from English to Python or Python to C#. We're like, okay, this is cool. We can start to get not only translation, but a little bit of predicted text here as well. We're all I think fairly already familiar with predictive text already in our code editors as IntelliSense. But in, I don't know, you go to your favorite word processor and chances are that you've got some kind of predictive text happening there as well.

中文翻译:

结果证明，我们可以做一些非常酷的事情。就像英语转西班牙语、西班牙语转德语的翻译工具一样，你也可以实现英语转 Python 或 Python 转 C#。我们觉得，好吧，这很酷。我们不仅可以开始进行翻译，还可以获得一些预测文本。我想我们大家都已经对代码编辑器中作为 IntelliSense 的预测文本相当熟悉了。但是，在——我不知道——你常用的文字处理器中，很可能也有某种预测文本功能。

[00:22:43] Ryan J. Salva

English:

We started experimenting with different user experiences, right? Do we want it so that you, I don't know, right click and get a little side panel that comes up with a bunch of different options for things that you might want here. That was nice because it would give you hold functions, but it's out of the cursor, right? You had to really... Even if you weren't switching over to a different window, you still had to switch over to a different panel, which itself was a little bit distracting. We eventually came to this idea of inline autocomplete.

中文翻译:

我们开始尝试不同的用户体验，对吧？我们是想要那种——我不知道——右键点击然后弹出一个侧边栏，里面有一堆你可能想要的不同选项吗？那挺好的，因为它能给你完整的函数，但它不在光标位置，对吧？你必须……即使你没有切换到另一个窗口，你仍然需要切换到另一个面板，这本身就有点让人分心。最终，我们想到了“行内自动补全”（inline autocomplete）这个主意。

[00:23:20] Ryan J. Salva

English:

We were able to with the kind of partnership of some of our friends over on the Microsoft side of things, partner with our friends in Visual Studio Code, they're like, hey, there's not really an extensibility yet in your editor for this multi-line autocomplete, but we've got an idea for how this might work. Played around with the actual presentation of it. What should the key strokes be? What should the presentation layer be? The gray italicized tech seemed to be a good way of indicating that it was ephemeral, as it were. Pretty early on, we landed on this user experience that is Copilot as most developers experience it today. I want to say that was at least 16 months ago, 14, 16 months ago. Since then, we brought it to developers.

中文翻译:

在微软那边一些朋友的合作下，我们与 Visual Studio Code 的朋友们建立了合作。他们说，嘿，你的编辑器里还没有这种多行自动补全的扩展性，但我们有个主意知道该怎么做。我们尝试了它的实际呈现方式。按键应该是怎样的？呈现层应该是怎样的？灰色斜体文本似乎是表明它是“临时性”的好方法。很早的时候，我们就确定了这种用户体验，也就是今天大多数开发者所体验到的 Copilot。我想说那至少是 16 个月前，14 到 16 个月前的事。从那时起，我们把它带给了开发者。

[00:24:15] Lenny

English:

Just to double click on that, you're saying just less than a year and a half ago, this kind of really started as a project and now it's out to the world. Is that right?

中文翻译:

我想再确认一下，你是说就在不到一年半以前，这还只是一个刚开始的项目，而现在它已经推向全世界了。是这样吗？

[00:24:26] Ryan J. Salva

English:

That is exactly right. That's exactly right. It's about a year and a half ago.

中文翻译:

完全正确。一点没错。大约是一年半以前。

[00:24:30] Lenny

English:

That's insane. What was that period between OpenAI almost taking down GitHub to I guess that point?

中文翻译:

太疯狂了。从 OpenAI 差点搞垮 GitHub 到你说的那个时间点，中间那段时间发生了什么？

[00:24:38] Ryan J. Salva

English:

The period in between kind of OpenAI almost taking down GitHub and then us really arriving at the user experience, part of that was, frankly, a lot of really smart researchers at OpenAI experimenting and doing what only world class AI researchers can do. It was a lot of them experimenting, occasionally asking for updates to the data set, tossing back to us a model that we might play with and tinker around with. These models have literally thousands of parameters that you can pass to them. When you're really thinking about GPT-3 and CodeX and then the transition from that to something like Copilot, it was not just like the model...

中文翻译:

在 OpenAI 差点搞垮 GitHub 到我们真正确定用户体验之间的那段时间，坦率地说，很大一部分是 OpenAI 许多非常聪明的研究人员在进行实验，做着只有世界级 AI 研究人员才能做的事情。他们进行了大量的实验，偶尔要求更新数据集，然后扔回给我们一个模型让我们试用和琢磨。这些模型字面上就有成千上万个可以传递的参数。当你真正思考 GPT-3、CodeX 以及从那到像 Copilot 这样的产品的转变时，不只是模型本身……

[00:25:27] Ryan J. Salva

English:

Creating the model is one thing, but then figuring out how to use the model in terms of what parameters do you want to adjust for, what do you want to optimize for in terms of... A great example of this is performance, right? When you're in a code editor, you don't necessarily want to type, type, type and then have to wait one second, two seconds, three seconds to get a suggestion back when your entire goal is to stay in the flow. We would run experiments to see how many milliseconds are the right amount such that a developer doesn't feel like they're being interrupted by Copilot and a suggestion.

中文翻译:

创建模型是一回事，但随后弄清楚如何使用模型——比如你想调整哪些参数，你想在哪些方面进行优化——又是另一回事。一个很好的例子是性能，对吧？当你在代码编辑器中时，你肯定不想打字、打字、打字，然后为了得到一个建议而等上一秒、两秒、三秒，因为你的整个目标是保持心流。我们会进行实验，看看多少毫秒是最合适的，能让开发者觉得没有被 Copilot 和它的建议所打断。

[00:26:06] Lenny

English:

What's the answer to that?

中文翻译:

答案是多少？

[00:26:09] Ryan J. Salva

English:

It seems like right now it's around 200 milliseconds. Depending upon where you're in the world, your latency can go up or down a little bit from there. But it seems like the sweet spot is somewhere around 200 milliseconds.

中文翻译:

目前看来大约是 200 毫秒。取决于你在世界上的位置，延迟可能会略有波动。但看起来“黄金平衡点”就在 200 毫秒左右。

[00:26:20] Lenny

English:

Good to know.

中文翻译:

很有用的信息。

[00:26:22] Ryan J. Salva

English:

We also experimented quite a bit. It's not just about the model, but it's also about what you feed the model. How do you prompt the model to return back a useful response? This kind of began a journey of experimentation for what we call prompt crafting.

中文翻译:

我们也做了相当多的实验。这不仅关乎模型，还关乎你给模型喂什么。你如何提示（prompt）模型返回有用的响应？这开启了一段我们称之为“提示词构建”（prompt crafting）的实验之旅。

[00:26:40] Lenny

English:

Going back to the way this started, it sounds like basically it was kind of this fortunate accident where OpenAI just did something that you didn't expect. And then somebody within this PhD group that you described is like, "Oh wow. Maybe we could do something really good with this." Is that kind of how it began?

中文翻译:

回到这一切开始的方式，听起来基本上像是一个幸运的意外：OpenAI 做了一些你没预料到的事，然后你提到的那个“博士团队”里有人说：“噢哇，也许我们可以利用这个做点非常棒的事情。” 是这样开始的吗?

[00:26:57] Ryan J. Salva

English:

That's fairly accurate. Yeah. I mean, we had a model that really was amazingly good, like a step level change in actual intelligence, right? And then marrying that up against a really good use case that actually changes developers' fundamental experience of the creation process, the creative process.

中文翻译:

相当准确。是的。我的意思是，我们当时拥有一个表现惊人地好的模型，在实际智能上有了阶跃式的提升，对吧？然后将其与一个非常好的用例结合起来，真正改变了开发者在创作过程、创意过程中的根本体验。

[00:27:25] Lenny

English:

Was there kind of a point at which it was clear to you or leadership in general like, we should double down on this thing and go big? Or this smaller team was working on this idea and then you're like, "Oh wow, this is going to work?" Or is it always like, "We will bet on this thing, this is such a big and great idea. We're going to invest resources for sure from the beginning?"

中文翻译:

是否有一个时间点，让你或领导层明确意识到：我们应该在这个项目上加倍投入，做大做强？还是说当时是一个小团队在研究这个想法，然后你觉得“噢哇，这行得通”？或者从一开始就是“我们要赌一把，这是一个非常宏大且伟大的想法，我们肯定要投入资源”？

[00:27:48] Ryan J. Salva

English:

The original team that was working on Copilot at GitHub was the team that we call GitHub Next. Essentially their job is to work on second and third horizon projects. What some folks might call moonshots, right? Things that we never really expect work in the next one or two years, but might three, five years down the line actually turn into something meaningful.

中文翻译:

在 GitHub 负责 Copilot 的最初团队是我们称之为“GitHub Next”的团队。基本上，他们的工作是研究“第二和第三地平线”（second and third horizon）项目。也就是有些人所谓的“登月计划”（moonshots），对吧？这些东西我们从不指望在未来一两年内见效，但可能在三五年后真正变成有意义的东西。

[00:28:17] Lenny

English:

Is there a concrete definition of horizon two and three? Is it like number of years out like Amazon style?

中文翻译:

“地平线二”和“地平线三”有具体的定义吗？是像亚马逊风格那样按年数来划分吗？

[00:28:23] Ryan J. Salva

English:

Not necessarily a concrete definition. For me, I usually ballpark it as first horizon is the next year, second horizon, the next three years, third horizon, the next five years. But we generally think of it more as a measure of ambiguity and confidence level more than calendar dates.

中文翻译:

不一定有具体的定义。对我来说，我通常粗略估算为：第一地平线是明年，第二地平线是未来三年，第三地平线是未来五年。但我们通常更多地将其视为对模糊性和信心水平的衡量，而不是具体的日历日期。

[00:28:47] Lenny

English:

This episode is brought to you by Modern Treasury. Modern Treasury is a next generation operating system for moving and tracking money. They're modernizing the developer tools and financial processes for companies managing complex payment flows. Think digital wallets via crypto on-ramps, right sharing marketplaces, instant lending, and more. They work with high growth companies like Gusto, Pipe, ClassPass, and Marqeta. Modern Treasury's robust APIs allow engineering to build payment flows right into your product, while finance can monitor and approve everything through a sleek and modern web dashboard.

中文翻译:

本集节目由 Modern Treasury 赞助。Modern Treasury 是用于资金流转和追踪的下一代操作系统。他们正在为管理复杂支付流的公司实现开发者工具和财务流程的现代化。想想通过加密货币入口实现的数字钱包、共享出行市场、即时贷款等等。他们与 Gusto、Pipe、ClassPass 和 Marqeta 等高增长公司合作。Modern Treasury 强大的 API 允许工程团队将支付流直接构建到产品中，而财务团队则可以通过时尚现代的 Web 仪表板监控和批准一切。

[00:29:22] Lenny

English:

Enabling realtime payments, automatic reconciliation, continuous accounting and compliance solutions, Modern Treasury's platform is used to reconcile over $3 billion per month. They're one of the hottest young FinTech startups on the market today, having raised funding from top firms like Benchmark, Altimeter, SVB Capital, Salesforce Ventures, and Y Combinator. Check them out at ModernTreasury.com. I'd love to spend a little bit more time on this. It's so interesting. Is this a Microsoft thing, just having these three horizons in a certain percentage of resources or bet on different horizons?

中文翻译:

Modern Treasury 的平台支持实时支付、自动对账、持续会计和合规解决方案，每月对账金额超过 30 亿美元。他们是当今市场上最炙手可热的年轻金融科技初创公司之一，已从 Benchmark、Altimeter、SVB Capital、Salesforce Ventures 和 Y Combinator 等顶级机构筹集了资金。请访问 ModernTreasury.com 了解他们。我想在这上面多花点时间。这太有趣了。这种"三个地平线"并按一定比例分配资源或投注不同地平线的做法，是微软的传统吗？

[00:29:58] Ryan J. Salva

English:

I would say it is not necessarily Microsoft thing, but is definitely at GitHub, how we have really contextualized it. Not to say that there aren't teams at Microsoft who might also use that methodology, but where we've been really maybe explicit or intentional about it is at GitHub where we've actually ring-fenced a team to think about that horizon two and horizon three work and kept them separate from EPD. EPD here being engineering, product, and design, the folks who are working on building productized operational products that we bring to market and we either give away or monetize in some way.

中文翻译:

我会说这不一定是微软的传统，但在 GitHub，这绝对是我们将其具体化的方式。并不是说微软没有团队使用这种方法论，但我们在 GitHub 确实做得非常明确且有意识，我们实际上专门划定了一个团队来思考第二和第三地平线的工作，并将他们与 EPD 隔离开来。这里的 EPD 指的是工程（Engineering）、产品（Product）和设计（Design），即那些负责构建推向市场的、可运营的产品化产品的团队，这些产品我们要么免费提供，要么以某种方式变现。

[00:30:39] Lenny

English:

This is so interesting. There's a lot of companies that have these sorts of R&D groups, new product experience team at Facebook and Google has one. I'm not sure how many successes have come out of these teams. From what I've seen, and I'm curious, what have you... And clearly you had a huge success as far as I can tell so far. Is there anything you've learned about how to do this, where you invest in these big moonshots within a larger company?

中文翻译:

这太有意思了。很多公司都有这类研发小组，比如 Facebook 的新产品体验团队，谷歌也有。我不确定这些团队产出了多少成功案例。据我所见——我也很好奇你的看法——显然到目前为止你取得了一个巨大的成功。关于如何在大公司内部投资这些重大的“登月计划”，你学到了什么经验吗？

[00:31:05] Ryan J. Salva

English:

I mean, I think the first step is to invest in it. The first step is really hire really smart people, attract smart people, and give them the opportunity to be creative. Don't expect anything out of them that is going to turn into a money maker or something that is going to be beholden to fundamentals around security, privacy, uptime, accessibility, all that groovy kind of stuff upfront. They need space to create and experiment.

中文翻译:

我的意思是，第一步是真正去投资它。第一步是雇佣非常聪明的人，吸引聪明的人，并给他们创造的机会。不要指望他们一开始就能做出赚钱的东西，或者受限于安全性、隐私、运行时间、无障碍性等基础要求的束缚。他们需要空间去创造和实验。

[00:31:37] Ryan J. Salva

English:

And also, when you do get to a place where that team has an idea that is clearly connected to a representative set of customers who have a genuine problem and there is signal with at least medium confidence that this solution, whatever it is, solves it in a novel way, that's the time to start thinking about, okay, let's actually put a little bit of... I'm going to call this market testing. It's nothing so formal as market testing. It's really just like, let's start to actually bring prototypes of this in front of more and more customers to kind of test it out and see, hey, is this actually solving a problem for you? Is this something that you would use? This is where the transition between Next and EPD at GitHub really started.

中文翻译:

而且，当你达到这样一个阶段：团队有了一个想法，这个想法明确地与一群有真实问题的代表性客户相关联，并且有至少中等信心的信号表明这个解决方案（无论它是什么）以一种新颖的方式解决了问题，那就是时候开始考虑，好吧，让我们投入一点……我称之为市场测试。它不像正式的市场测试那么严肃。其实就是，让我们开始把原型展示给越来越多的客户，去测试一下，看看：嘿，这真的解决了你的问题吗？这是你会用的东西吗？这就是 GitHub Next 和 EPD 之间转型的真正起点。

[00:32:35] Ryan J. Salva

English:

This is actually where my role in the product cycle kind of really started to increase. I had kind of been in tight connection and been monitoring the work and kind of consulting a little bit with the Next team prior to that. But it was that moment when we identified that, okay, this is actually something real. Customers are saying, developers are saying, "This is magical. This does something extraordinary that I could not do on my own," that we started to think about, okay, how do we transition this over? From there, we're really just like, okay, we think we've got a hit here. We think we've got something that we can actually bring to developers.

中文翻译:

这实际上是我在产品周期中的角色开始加重的地方。在那之前，我一直与 Next 团队保持紧密联系，监控工作并提供一些咨询。但就在那一刻，我们确认了：好吧，这确实是个真家伙。客户在说，开发者在说，“这太神奇了。它做了一些我自己无法完成的非凡之事。” 于是我们开始思考，好吧，我们该如何把它交接过来？从那时起，我们真的觉得：好吧，我们手里有个爆款。我们觉得我们有了一些可以真正带给开发者的东西。

[00:33:21] Ryan J. Salva

English:

We made an intentional decision to take some of the researchers who were in the Next team and for a finite period of time, move them over to create a new EPD squad. We want them to be researchers, but we need to do knowledge transfer and we needed to actually provide the seed for a team that could eventually operationalize and productize. And that kind of began the technical preview where we started to invite tens of thousands, then hundreds of thousands to the technical preview. In that technical preview, we started to see crazy mind-blown emoji tweets and threads on Hacker News about people getting really, really excited about it.

中文翻译:

我们做了一个有意识的决定，从 Next 团队中抽调一些研究人员，在有限的时间内将他们转移到新成立的 EPD 小组中。我们希望他们保持研究员的身份，但我们需要进行知识转移，我们需要为一个最终能够实现运营化和产品化的团队提供种子。这开启了技术预览阶段，我们开始邀请数万名、然后是数十万名用户参与技术预览。在那个阶段，我们开始在推特上看到各种“惊掉下巴”表情包的推文，在 Hacker News 上看到人们对此感到非常非常兴奋的讨论帖。

[00:34:09] Ryan J. Salva

English:

That's how we knew it was time to start scaling and it was time to really start thinking about how do we do hiring so that we can build in some insulation around these researchers so that they can eventually go back to GitHub Next to do what they do best, which is be innovative and creative and think about the next moonshot. That process, that took... Well, we're actually still kind of at the tail end of it now. Here we are, like I said, roughly a year and a half after the initial creation of the product, having gone through technical preview, have achieved general availability. We've now hired in a team around them.

中文翻译:

就这样，我们知道是时候开始扩张了，是时候真正开始考虑如何招聘，以便在这些研究人员周围建立一些“绝缘层”，让他们最终能回到 GitHub Next 去做他们最擅长的事——即保持创新和创意，思考下一个“登月计划”。那个过程花了……嗯，实际上我们现在仍处于这个过程的尾声。就像我说的，在产品最初创建大约一年半后的今天，我们经历了技术预览，实现了正式发布（GA）。我们现在已经围绕他们建立了一个团队。

[00:34:53] Ryan J. Salva

English:

The researchers actually as early as last month have started to gradually move back over to GitHub Next. An EPD squad, multiple EPD squads actually are now taking the product forward and starting to respond to customer feedback to think about, okay, how do we now as a product team, carry this roadmap forward from an idea that originated in GitHub Next?

中文翻译:

研究人员实际上早在上个月就开始逐渐搬回 GitHub Next 了。一个 EPD 小组（实际上是多个 EPD 小组）现在正带着产品前进，开始响应客户反馈，思考：好吧，作为一个产品团队，我们现在如何将这个源自 GitHub Next 的想法继续推进路线图？

[00:35:22] Lenny

English:

I love that insight of bringing the people along and not just kind of like, cool, we'll take it from here. If you were to build a team like this again somewhere to this kind of R&D horizon three or two teams, is there anything else you would do differently, any lessons you take away from this experience for maybe founders or PMs working at larger companies that are like, "Hey, we should have something like this?" Is there anything else that you find is important for making something like this successful?

中文翻译:

我非常喜欢“带人一起走”而不是简单说“谢了，剩下的交给我们”这个见解。如果你要在别处再次组建这样一个针对研发地平线三或二的团队，你会有什么不同的做法吗？对于那些在大公司工作、觉得“嘿，我们也该搞一个这样的团队”的创始人或产品经理（PM），你有什么从这次经历中总结出的教训吗？你觉得还有什么对这类项目的成功至关重要？

[00:35:49] Ryan J. Salva

English:

The criteria for moving researchers back into their R&D team, whatever that happens to be for your organization, that can't be based on a calendar. It needs to be based on a replacement in seat, who's actually doing the job and has picked up all of the skills necessary, and only then can the researcher move back. Make sure that you've got continuity of expertise and sets and domain familiarity before you move over. I feel like we've managed that pretty well today. As well, it's critical that the team who is taking over from the R&D shop feels like they have control over their own future. You can't really delegate roadmap to an R&D team.

中文翻译:

将研究人员移回研发团队的标准（无论你的组织如何称呼它）不能基于日历时间。它必须基于“接替者到位”，即谁在实际接手工作并掌握了所有必要的技能，只有那时研究人员才能撤回。在移交之前，确保专业知识、技能组合和领域熟悉度具有连续性。我觉得我们今天处理得相当不错。此外，至关重要的一点是，从研发部门接手的团队必须觉得他们掌控着自己的未来。你不能真的把产品路线图委托给一个研发团队。

[00:36:44] Ryan J. Salva

English:

The team who's responsible for maintaining the product, for building the product, who has the closest feedback loop with the end customer, they're the ones who really need to own and feel like they control the roadmap. Making sure that you're not outsourcing innovation exclusively to an R&D team, but that is happening within the product team as they take ownership over the idea and over the use case in the customer. Last I would say here is really that engineering fundamentals in a lot of ways are the contracts that differentiate an R&D team from an operational product team.

中文翻译:

负责维护产品、构建产品、与最终客户有最紧密反馈闭环的团队，才是真正需要拥有并感觉自己掌控路线图的人。要确保你不是把创新完全外包给研发团队，而是让创新发生在产品团队内部，因为他们接管了这个想法、用例和客户。最后我想说的是，工程基础（engineering fundamentals）在很多方面是区分研发团队和运营产品团队的契约。

[00:37:30] Ryan J. Salva

English:

Bringing that fundamentals process into it is going to feel candidly a little bit unnatural to the researchers. That takes therefore a little bit of cultural change management for everyone to just adapt their way of working and understand that we're graduating from an experiment and a research project to an operational product, and often because those researchers are... They're the first wave that come over. They're the seed of the project. It's going to feel a little bit unnatural to them and they probably won't have all the right skillsets in order to make that transition.

中文翻译:

坦率地说，引入这些基础流程对研究人员来说会感觉有点不自然。因此，这需要一点文化变革管理，让每个人都适应工作方式的转变，并理解我们正在从一个实验和研究项目“毕业”，转变为一个运营产品。通常因为这些研究人员是第一波过来的人，他们是项目的种子，这种转变对他们来说会有些别扭，而且他们可能并不具备完成这种转变所需的全部正确技能。

[00:38:08] Ryan J. Salva

English:

Making sure that you've got a good mix of engineers who are comfortable maintaining a service, as well as engineers and researchers who are really thinking about, what is the idea that we've created, what is the new thing that we've brought to market, and can bring that vision to it.

中文翻译:

要确保你拥有一个良好的工程师组合：既有擅长维护服务的工程师，也有真正思考“我们创造了什么想法”、“我们推向市场的新事物是什么”并能为此带来愿景的工程师和研究人员。

[00:38:27] Lenny

English:

Yeah, I can totally see the challenge that comes from... This was my thing. I've been working on this. What are you guys doing to this project? Where is this going? I'm not sure I'm feeling... And then there's all these new asks that are coming at you like, oh my God, this was so much fun and now I have to scale this freaking thing.

中文翻译:

是的，我完全能预见到那种挑战…… “这是我的心血，我一直在研究这个。你们在对这个项目做什么？它要走向何方？我不确定我是否喜欢……” 然后所有这些新的需求接踵而至，就像是：“噢我的天，之前那么好玩，现在我居然得去扩展这个该死的东西。”

[00:38:46] Ryan J. Salva

English:

I mean, this is the best problem in the world to have. Talk about kind of customer ask, for Copilot in particular, the amount of chatter, the amount of customer feedback that was coming in especially for us with AI, I mean, the world is still figuring out AI, candidly. I mean, we're getting a lot better at it, especially in the last couple of years with things like Dolly and Copilot. But it brings with it not only engineering challenges, but also, frankly, ethical challenges and legal challenges, like making sense of what our expectations are of AI. If AI produces something that is offensive, who's at fault?

中文翻译:

我的意思是，这是世界上最幸福的烦恼。说到客户需求，特别是对于 Copilot，反馈量非常惊人。尤其是对我们这种 AI 产品，坦率地说，世界仍在摸索 AI。我的意思是，我们正变得越来越擅长，特别是过去几年有了像 DALL-E 和 Copilot 这样的东西。但它不仅带来了工程挑战，坦白说，还有伦理挑战和法律挑战，比如理清我们对 AI 的期望。如果 AI 生成了冒犯性的内容，谁该负责?

[00:39:37] Ryan J. Salva

English:

Our stance on it, what we ended up coming to is actually the framing of Copilot as an AI pair programmer I think is a useful one. Pair programmer, I suspect most of your listeners will know, but pair programmer is usually two developers sitting side by side working on a problem together. One's at the keyboard and the other one's kind of helping them talk through it, talk through the ideas and make corrections, that kind of thing. Well, if Copilot is your AI pair programmer and they're whispering crazy stuff into your ear and they're bringing politics into it or gender identity into it or, I don't know, whatever other...

中文翻译:

我们的立场，以及我们最终达成的共识，实际上是将 Copilot 框架化为一名“AI 结对编程伙伴”（AI pair programmer），我认为这很有用。我猜你的大多数听众都知道“结对编程”，通常是两个开发者并排坐着一起

解决问题。一个在键盘前，另一个帮着理清思路、讨论想法并进行纠错之类的。那么，如果 Copilot 是你的 AI 结对编程伙伴，它在你耳边低语一些疯狂的东西，把政治、性别身份或者我不知道的任何其他东西扯进来……

[00:40:19] Ryan J. Salva

English:

They're spouting off slang and slander and all that kind of stuff. You're probably not going to be able to focus on your work, right? It's going to be really distracting. Really coming down to some principles about what is the use case we're trying to solve, what is appropriate, I put this in scare quotes, behavior of the AI bot sitting side by side with you, helped us create some principles or some guidelines for the developer experience that we wanted to create.

中文翻译:

如果它在喷脏话、诽谤之类的，你可能就没法专注于工作了，对吧？这会非常让人分心。最终归结为一些原则：我们要解决的用例是什么？坐在你身边的 AI 机器人的“适当”（加个引号）行为应该是怎样的？这帮助我们为想要创造的开发者体验制定了一些原则或指南。

[00:40:52] Lenny

English:

Oh, I love that. Just kind of creating a persona of the thing to help you inform how the behavior of the thing should work. How do you work through these challenges? Is it discussions with you and the legal team? I don't know, these ethical things are really tricky, I imagine. How do you approach them like that as a product team?

中文翻译:

噢，我喜欢这个。通过为这个东西创建一个“人格”（persona）来指导它的行为逻辑。你们是如何应对这些挑战的？是和你以及法律团队讨论吗？我不知道，我能想象这些伦理问题非常棘手。作为一个产品团队，你们是如何处理这些问题的?

[00:41:09] Ryan J. Salva

English:

It is conversations with a very, very wide cast of characters. This product in particular, I probably spent more time with legal than any other products that I've ever kind of been responsible for. All wonderful creative people. But it's not just legal. It is also privacy and security champions. It is, frankly, developers, like the people who are using it, listening to them. Hey, what works here? What doesn't work for you here? Why is this offensive? Why is it not offensive? We'll continue on the example of the crazy pair programmer whispering crazy things into our year. When we first started out, we didn't really have any filter on Copilot whatsoever the very, very, very early days.

中文翻译:

这是与非常广泛的角色进行的对话。特别是这个产品，我花在法律事务上的时间可能比我负责过的任何其他产品都多。他们都是很棒、很有创意的人。但不仅是法律部门，还有隐私和安全方面的专家。坦率地说，还有开发者，也就是那些正在使用它的人，听取他们的意见。嘿，这里什么行得通？什么对你没用？为什么这具有冒犯性？为什么不具冒犯性？我们继续用那个“疯狂的结对编程伙伴在耳边低语”的例子。刚开始的时候，在非常非常早期，我们对 Copilot 几乎没有任何过滤。

[00:41:58] Ryan J. Salva

English:

And then eventually we're like, okay, it needs to be slightly more controlled experience. We need to edit out some of the most egregious stuff. We introduced a simple block list of words, and these block lists are always fraught with peril, like which words are okay, which words are not okay. All of a sudden, we become editors of language and that's kind of a scary place to be. I'm not comfortable with it at least. But at a certain level, it has to be done, because otherwise you're going to create a bad developer experience.

中文翻译:

后来我们觉得，好吧，它需要一个更受控一点的体验。我们需要删掉一些最过分的内容。我们引入了一个简单的单词黑名单，而这些黑名单总是充满风险，比如哪些词可以，哪些词不行。突然之间，我们变成了语言的编辑，那是一个挺可怕的处境。至少我觉得不舒服。但在某种程度上，这是必须做的，否则你会创造出糟糕的开发者体验。

[00:42:35] Ryan J. Salva

English:

Often we would get feedback from developers of like, "Hey, this particular word was blocked. That it was blocked either was offensive to me or prevented me from being able to get good value out of the product."

中文翻译:

我们经常收到开发者的反馈，比如：“嘿，这个特定的词被屏蔽了。屏蔽它要么让我觉得被冒犯，要么让我无法从产品中获得应有的价值。”

[00:42:51] Lenny

English:

Oh man.

中文翻译:

噢天哪。

[00:42:52] Ryan J. Salva

English:

Always kind of dancing the dance of editorial content. We're actually at a place now where we're able to partner with the Azure Department of a Responsible AI, and they've created some really extraordinary models that help detect I'll call it sentiment for lack of a better word, but basically when there is something that is patently offensive. Because there are some words that in some contexts may be offensive and in some context may be totally reasonable, especially when you get into software for medical kind of scenarios, right?

中文翻译:

我们总是在编辑内容的边缘试探。实际上，我们现在能够与 Azure 的负责任 AI 部门（Responsible AI）合作，他们创建了一些非常出色的模型，可以帮助检测——由于找不到更好的词，我称之为“情绪”——但基本上是检测何时出现了明显的冒犯性内容。因为有些词在某些语境下可能是冒犯性的，而在某些语境下可能完全合理，特别是当你涉及到医疗领域的软件场景时，对吧?

[00:43:35] Ryan J. Salva

English:

Being able to start to shift a little bit to focus or to rely on AI models that can also do a better job than we could with crude or simple block lists is maybe another proof point both of how AI as a solution for common development problems is getting way better at solving more parts of our stack or filling in for more parts of our stack. At least in our case, we were pretty fortunate to be able to deliver on or depend on a parent company's contributions to solve a real acute problem that GitHub probably could not have solved on our own.

中文翻译:

能够开始转向依赖 AI 模型，而这些模型比我们用粗糙或简单的黑名单做得更好，这也许是另一个证明：AI 作为解决常见开发问题的方案，正变得越来越擅长解决我们技术栈的更多部分，或者填补更多空白。至少在我们的案例中，我们非常幸运能够依靠母公司的贡献来解决一个 GitHub 可能无法独立解决的棘手问题。

[00:44:16] Lenny

English:

I never thought that Copilot would be... That you would have to worry about it saying things that are crazy. That is wild that you guys have to deal with that. Wasn't it Microsoft that had that bot that turned really negative and eventually shut down?

中文翻译:

我从未想过 Copilot 会……你居然得担心它说出疯狂的话。你们得处理这种事真是太疯狂了。以前是不是微软有个机器人变得非常负面，最后不得不关掉?

[00:44:31] Ryan J. Salva

English:

It was.

中文翻译:

是的。

[00:44:31] Lenny

English:

There's experience there.

中文翻译:

那方面有经验了。

[00:44:32] Ryan J. Salva

English:

What was its name? Talia or something like that?

中文翻译:

它叫什么名字来着？Talia 还是类似的？

[00:44:35] Lenny

English:

Something like that.

中文翻译:

差不多。

[00:44:36] Ryan J. Salva

English:

Yeah, something like that. We don't want another one of those incidences.

中文翻译:

对，差不多。我们不希望再发生那样的事件。

[00:44:40] Lenny

English:

Wow. What this makes me think about is your team is at the forefront of AI in this applied way. I'm curious what your thinking is on just where this goes for developers especially. I saw a stat that maybe 40% of people's code is now written by Copilot. I don't know if that's right. But is the vision in the future becomes something like 90? Where do you see this all going?

中文翻译:

哇。这让我想到，你们团队正处于 AI 应用的最前沿。我很想知道你对开发者未来的看法。我看到一个统计数据，说现在人们 40% 的代码是由 Copilot 编写的。我不知道这是否准确。但未来的愿景会变成 90% 吗？你认为这一切将走向何方？

[00:45:02] Ryan J. Salva

English:

Just to put a fine point on that stat, it is 40% is specifically for Python developers. Candidly, it varies depending upon the language. Because as you might imagine, some languages have better representation in the public domain than others. And usually both the volume and the diversity of training data correlates with the quality of suggestions, which is then represented by either the number of lines written or the acceptance rate or any one of a number of other metrics.

中文翻译:

关于那个数据，我得说明一下，40% 特指 Python 开发者。坦率地说，这取决于语言。因为你可以想象，某些语言在公共领域的表现比其他语言更好。通常，训练数据的数量和多样性都与建议的质量相关，而这又体现在编写的行数、接受率或许多其他指标上。

[00:45:35] Lenny

English:

Awesome. Thanks for clarifying.

中文翻译:

太棒了，谢谢澄清。

[00:45:36] Ryan J. Salva

English:

Yeah, totally. We see it range anywhere from the upper twenties to the forties across all the different languages.

中文翻译:

是的，完全正确。我们看到在所有不同语言中，这个比例大约在 20% 多到 40% 之间。

[00:45:43] Lenny

English:

Just to throw this out there, as a not great engineer, I used to be an engineer for about 10 years, I welcome our AI Overlords writing all my code. I'm excited for this to do more and more. And yes, I'm curious where you think this goes.

中文翻译:

顺便说一下，作为一个不太出色的工程师（我曾做了大约 10 年工程师），我欢迎我们的“AI 霸主”代劳写完我所有的代码。我很期待它能做得越来越多。是的，我很想知道你认为这会走向何方。

[00:45:58] Ryan J. Salva

English:

It does. It enables even mediocre developers like myself to be able to do some pretty amazing things. But where's it going? First, I think, I hope it's obvious to most developers that AI is going to infuse pretty much our entire development stack in the not so distant future. Copilot is really just the very tip of the sphere for a lot of innovations and better managing maybe our build queues or helping to... Here's a great one. I don't know about you, but often the comments that I get with commit messages and PRs aren't super great. It puts a lot of effort onto the code reviewer to go figure out what the developer was actually trying to do.

中文翻译:

确实如此。它让像我这样平庸的开发者也能做出一些非常了不起的事情。但它将走向何方？首先，我认为（也希望）大多数开发者都能意识到，在不久的将来，AI 将渗透到我们几乎整个开发栈中。Copilot 真的只是众多创新的冰山一角，比如更好地管理我们的构建队列，或者帮助……举个好例子：我不知道你怎么想，但我收到的 commit 信息和 PR（拉取请求）注释通常写得不怎么样。这让代码审查者（reviewer）得花很大力气去弄清楚开发者到底想干嘛。

[00:46:55] Ryan J. Salva

English:

What if AI could summarize all of your changes with your full request and you just have to, as the contributing developer, just review it to make sure it's accurate, send it on its way, and you don't have to put in extra effort for that. There are lots and lots of different opportunities for AI to essentially be able to take some of the drudgery out of our work so that we can focus on creative acts. What I hear from developers and what I experience myself is that Copilot kind of forces me to think a little bit more about what are the design patterns I'm trying to create?

中文翻译:

如果 AI 能总结你 PR 中的所有更改，而你作为贡献代码的开发者，只需检查一下确保准确，然后发送即可，不需要为此付出额外努力，那会怎样？AI 有非常多的机会将我们从繁琐的工作中解脱出来，让我们专注于创造性活动。我从开发者那里听到以及我自己的体会是，Copilot 某种程度上迫使我更多地思考：我想要创建的设计模式是什么？

[00:47:33] Ryan J. Salva

English:

What is the end user experience or the outcomes that I'm trying to drive with my code, and that I can rely on Copilot to scaffold out a lot of that so that I can focus on more creative work? That is really what I hope for our industry five, 10 years from now, is that not only will we be inviting more developers or more people to become developers by essentially providing a layer of abstraction a little bit, or at least a little bit of a hand in development, but that also the really experienced developers are focusing on much larger problems and focusing on outcomes and creativity rather than really low level difficult rote memorization of things like syntax or ordering of parameters and the like.

中文翻译:

我试图通过代码驱动的最终用户体验或结果是什么？我可以依靠 Copilot 来搭建大部分基础框架，这样我就可以专注于更具创造性的工作。这正是我对 5 到 10 年后我们行业的期望：我们不仅会通过提供一层抽象或至少在开发中提供一些帮助，来邀请更多人成为开发者；而且那些经验丰富的开发者将专注于更宏大的问题，专注于结果和创意，而不是那些低级的、困难的、死记硬背的东西，比如语法或参数顺序之类的。

[00:48:32] Lenny

English:

Great. If nothing else, that'll keep people from just having a tab of Stack Overflow, copy and pasting every function that they're trying to figure out.

中文翻译:

太好了。至少，这能让人们不再总是开着一个 Stack Overflow 标签页，然后复制粘贴每一个他们想搞明白的函数。

[00:48:42] Ryan J. Salva

English:

I want Stack Overflow to stay in business, but I would mind a little bit less contact switching myself.

中文翻译:

我希望 Stack Overflow 能继续经营下去，但我个人确实不介意少一点上下文切换。

[00:48:48] Lenny

English:

In the experience of scaling this thing, what would you say has been the biggest challenge either technologically or even operationally just kind of scaling it to a real product that people are paying for?

中文翻译:

在扩展这个产品的过程中，你认为最大的挑战是什么？无论是技术上的，还是运营上的，比如把它扩展成一个人们愿意付费的真实产品？

[00:49:01] Ryan J. Salva

English:

There's a few dimensions of that. One is a problem that's very much of our time in the world, namely that supply chains have been disrupted dramatically over the course of the last few years. It turns out that Copilot for both training and operating the models requires some very rare and unique GPUs that there's not a lot of global supply of. Part of it is just like, can we get enough hardware in order to run these things? We've actually earmarked quite a bit of capacity, and we are greedy, greedy, greedy for more capacity globally. As soon as we can produce those chips and get them in data centers, we do it.

中文翻译:

这有几个维度。一个是当今世界非常普遍的问题，即过去几年供应链遭到了剧烈破坏。事实证明，Copilot 无论是训练还是运行模型，都需要一些非常稀有且独特的 GPU，而这些 GPU 的全球供应量并不多。部分挑战就在于：我们能获得足够的硬件来运行这些东西吗？我们实际上已经预留了相当多的容量，而且我们对全球更多的容量非常非常渴求。只要我们能生产出那些芯片并把它们送进数据中心，我们就会去做。

[00:49:50] Ryan J. Salva

English:

That's been one kind of unique challenge. I would also say here that operationally, another challenge has been, how do we create a model that the community really feels like ownership over, right? The dialogue that's had to happen as we brought an AI tool to market, especially one that is trained on public code, has required a lot of dialogue between us and our community. Every good product manager should be spending as much of their time as possible with their customers, with their potential customers.

中文翻译:

这是一个独特的挑战。我还想说，在运营方面，另一个挑战是：我们如何创建一个让社区真正有归属感的模型，对吧？当我们把一个 AI 工具推向市场，尤其是基于公开代码训练的工具时，我们需要与社区进行大量的对话。每一个优秀的产品经理都应该花尽可能多的时间与他们的客户和潜在客户在一起。

[00:50:34] Ryan J. Salva

English:

Copilot, in particular, has been a more complicated kind of rollout because we as an industry, as a society are still figuring out how to make sense of it. The amount of give and take between developers and us as a product team has really required us to scale up more of the product team than it has the engineering team.

中文翻译:

特别是 Copilot，它的发布过程更为复杂，因为我们作为一个行业、一个社会，仍在摸索如何理解它。开发者与我们产品团队之间的这种互动，实际上要求我们扩充产品团队的规模，甚至超过了对工程团队的扩充需求。

[00:51:02] Lenny

English:

Interesting. And why is that?

中文翻译:

有意思。那是为什么呢？

[00:51:04] Ryan J. Salva

English:

It's a couple of different reasons. I mean, one, like I said, we are trained on public code. Not all of the community is really sure like, when is it okay to train a model on public code? When is it not okay to train a model on public code? Is Copilot producing secure suggestions? Is Copilot producing bug buggy suggestions? There's a lot of doubt. There's a lot of very healthy skepticism. Actually I mean that genuinely. I want people to be skeptical of Copilot. We owe it to ourselves as a community to be skeptical of any AI.

中文翻译:

有几个不同的原因。首先，正如我所说，我们是基于公开代码训练的。并非所有社区成员都确定：什么时候可以用公开代码训练模型？什么时候不行？Copilot 提供的建议安全吗？Copilot 提供的建议会有 bug 吗？存在很多疑虑，也有很多非常健康的怀疑态度。我是认真的，我希望人们对 Copilot 保持怀疑。作为一个社区，我们有责任对任何 AI 保持怀疑。

[00:51:40] Ryan J. Salva

English:

Because just like there's great potential for benefit, there's also great potential for harm. People keeping us accountable like, how are you preventing things like model poisoning? Is there going to be a new attack vector that we just haven't really thought of yet around AI that might produce negative consequences? We think that we've done a really good and responsible job of that by making sure that first, we are very clear that Copilot is not a replacement for a developer. It will never be.

中文翻译:

因为就像它有巨大的获益潜力一样，它也有巨大的潜在危害。人们在监督我们，比如：你们如何防止模型投毒（model poisoning）？是否会出现我们尚未想到的、围绕 AI 的新攻击向量并产生负面后果？我们认为我们在这方面做得很好且负责任，首先我们非常明确：Copilot 不是开发者的替代品，永远不会是。

[00:52:17] Ryan J. Salva

English:

We do not want Copilot auto generating code where a thinking, reasoning, breathing human being is not on the other side of that keyboard making recent decisions. We do not want Copilot to replace any other part of the stack, whether it is static analysis tools or your unit tests or whatever kind of measures you're putting in today to make sure that your humans produce good quality code. We want you to keep all of those same systems in place to make sure that humans who are leveraging tools like Copilot continue to produce that good quality code.

中文翻译:

我们不希望 Copilot 在键盘另一端没有一个会思考、会推理、有呼吸的人类做决策的情况下自动生成代码。我们不希望 Copilot 取代技术栈的任何其他部分，无论是静态分析工具、单元测试，还是你今天为了确保人类产出高质量代码而采取的任何措施。我们希望你保留所有这些系统，以确保利用 Copilot 等工具的人类能够继续产出高质量的代码。

[00:52:56] Ryan J. Salva

English:

But there's a lot of at the same time anxiety of like, where is AI stack? Is AI eventually going to be... This is back to your question about where will we be five, 10 years from now. Will it be writing 90% of the code? We don't want Copilot to be that... We don't want it to replace anything. We want it to augment. The idea here is really that AI is an enabler for developers to focus on the creative work, to stay in the flow, to be able to move faster. Working through those anxieties, working through that healthy skepticism takes conversation. It takes dialogue. And that takes us on the product side having that guided conversation with the community.

中文翻译:

但与此同时，人们也有很多焦虑，比如：AI 处于什么位置？AI 最终会……回到你关于 5 到 10 年后我们在哪里的问题。它会写 90% 的代码吗？我们不希望 Copilot 变成那样……我们不希望它取代任何东西。我们希望它起到增强作用。这里的核心理念是，AI 是开发者的赋能者，让他们能专注于创造性工作，保持心流，并能跑得更快。化解这些焦虑、应对那些健康的怀疑需要沟通，需要对话。这要求我们产品端与社区进行有引导的对话。

[00:53:50] Lenny

English:

It feels like it connects back to your education back in the day, philosophy and literature. How convenient is that?

中文翻译:

这感觉和你当年的教育背景——哲学和文学——联系起来了。这可真巧，对吧？

[00:53:57] Ryan J. Salva

English:

It often feels very connect... I mean, certainly the education side of things taught me that the importance of dialogue, the importance of skepticism is valuable in so much more than esoteric armchair ponderings. It's actually applicable to the real world.

中文翻译:

确实经常感觉到这种联系……我的意思是，教育背景确实教会了我，对话的重要性、怀疑精神的重要性，其价值远不止于深奥的空谈。它实际上适用于现实世界。

[00:54:17] Lenny

English:

Maybe a final question before we get to our very exciting lightning round.

中文翻译:

在进入我们非常令人兴奋的闪电轮环节之前，最后一个问题。

[00:54:21] Ryan J. Salva

English:

Woo!

中文翻译:

喔！

[00:54:23] Lenny

English:

Just looking back at this whole experience of, one, just building, incubating, launching this big bold bet within a big company, you can go in either direction, either just any lessons on just taking a bold bet versus incremental wins and how you think about investing in these two kind of categories, or just within a large company, a lesson of just how to build something like this, like a massive new product from just a seed of an idea to a large new business line potentially.

中文翻译:

回顾这整个经历：在大公司内部构建、孵化并发布这样一个大胆的赌注。你可以从两个方向来谈：一是关于选择大胆赌注还是渐进式胜利的经验，以及你如何考虑在这两类项目上的投入；二是在大公司内部，如何将一个想法的种子培育成一个巨大的新产品，甚至可能是一个庞大的新业务线。

[00:54:51] Ryan J. Salva

English:

As both a product manager and a portfolio manager of multiple products, I'm responsible for multiple product lines at GitHub, the allocation of time, of focus, energy, and resources becomes a really challenging question. The answer to which isn't always the same, depending upon the time, world circumstances, organizational circumstances, technology circumstances. As a general rule, as a general principle, I certainly try to make sure that we're always reserving some capacity for bold, audacious experimental research projects. You can think of those really uncertain bets as being five to 10% of the team's capacity. About 25, maybe 30% of the team's capacity should generally be on just operations.

中文翻译:

作为一名产品经理，同时也是多个产品的组合管理者（负责 GitHub 的多条产品线），时间、注意力、精力和资源的分配成了一个非常具有挑战性的问题。答案并不总是一成不变的，它取决于时间、世界局势、组织状况和技术环境。作为一个通用规则和原则，我肯定会努力确保我们始终为大胆、无畏的实验性研究项目保留一些空间。你可以将这些极具不确定性的赌注视为团队能力的 5% 到 10%。大约 25% 到 30% 的团队能力通常应该花在纯运营上。

[00:55:54] Ryan J. Salva

English:

How do we keep our in-market products meeting customer expectations? And then the remainder of it, what is that, about 60% or so, is really on incremental progress for our end market products. How do we make iterative improvements and continue to actually realize payoff for the larger bets that we made one, two, three, four years back? And from a rough distribution, that's generally how I run my larger teams. That works when you have larger teams though. At startups, where we were pretty much only a big bet, obviously your percentages get very different and it becomes a matter of you're all in for that one proverbial lottery ticket.

中文翻译:

我们如何保持已上市产品满足客户期望？剩下的部分，大约 60% 左右，主要用于已上市产品的渐进式进展。我们如何进行迭代改进，并继续从我们一、二、三、四年前投下的重大赌注中获得回报？从粗略的分配来看，这通常是我管理大型团队的方式。不过，这只有在你有大型团队时才行得通。在初创公司，我们几乎只有一个"大赌注"，显然你的比例会变得非常不同，那就变成了为了那张众所周知的"彩票"而全力以赴。

[00:56:50] Lenny

English:

Awesome. Thanks for sharing that. I was going to ask you the percentages that you recommend. Thank you for getting to that. With that, we've gotten to our very exciting lightning round. I'm just going to ask you five questions briefly and just whatever comes to mind, whatever answer you have. Let's do it. Sound good? Okay. What are two or three books that you recommend most to other people?

中文翻译:

太棒了，谢谢分享。我正想问你推荐的比例呢，谢谢你提前回答了。那么，我们进入了非常令人兴奋的闪电轮环节。我会简短地问你五个问题，想到什么就说什么。开始吧？好。你最推荐给别人的两三本书是什么？

[00:57:13] Ryan J. Salva

English:

Oh, good question. One of them is a book on user experience called Make It So. It's a reference back to Star Trek, and the idea here is essentially that user experiences that are presented to us in sci-fi often make their way into our everyday products and tools 20, 30 years down the line. It is a great eye-opening, illuminating and just really fun book. That's one. And then completely different take, I'll go outside of tech and I'll just do entertainment value. There's a David Foster Wallace book called Brief Interviews with Hideous Men that I love. It's a collection of short stories.

中文翻译:

噢，好问题。其中一本是关于用户体验的书，叫《Make It So》。这个书名引用了《星际迷航》，核心观点是：科幻小说中呈现给我们的用户体验，往往会在 20、30 年后进入我们的日常产品和工具中。这是一本非常开阔眼界、富有启发性且非常有趣的书。这是第一本。然后是一个完全不同的视角，我跳出科技圈，选一本纯娱乐价值的。有一本大卫・福斯特・华莱士（David Foster Wallace）的书叫《与丑陋人物的简短采访》（Brief Interviews with Hideous Men），我很喜欢。这是一部短篇小说集。

[00:58:04] Ryan J. Salva

English:

And essentially what it is, is it is if you're watching a movie and the villain gets their opportunity to have their big speech, which kind of explains why they are who they are, it makes them maybe a little bit vulnerable in that moment, it's that speech 10 times over for different hideous people, terrible, terrible people. Interesting read. I recommend it.

中文翻译:

基本上，它就像是你在看电影时，反派得到了发表长篇大论的机会，解释他们为什么会变成那样，那一刻让他们显得有点脆弱。这本书就是把这种演讲针对十个不同的丑陋、糟糕透顶的人重复了十遍。读起来很有趣，我推荐它。

[00:58:31] Lenny

English:

I love that. It reminds me of this book that is the interior design of dictators and they show you their homes of Saddam Hussein, Hitler, and all these guys.

中文翻译:

我喜欢这个。这让我想起一本书，讲的是独裁者的室内设计，展示了萨达姆・侯赛因、希特勒等人的家。

[00:58:43] Ryan J. Salva

English:

Dude! Oh my gosh, that's awesome. I got to find that one. You'll have to send it to me.

中文翻译:

伙计！噢我的天，太酷了。我得找找那本。你得发给我。

[00:58:47] Lenny

English:

I found one at an old bookstore, like used bookstore. I don't know if they're around anymore, but I'll find it. Second question. What's a favorite other podcast that you like to listen to or recommend if there's any?

中文翻译:

我在一家旧书店找到的。我不知道那家店还在不在，但我会找找看。第二个问题：你最喜欢的其他播客是什么？或者有没有推荐的？

[00:59:02] Ryan J. Salva

English:

Oh god, there's so many. I consume hundreds of hours of podcasts every month. It is crazy. I can choose many. I'll give you just one. The Memory Palace with Nate DiMeo is an excellent storytelling podcast. He does about 20 minute vignettes, usually selected from kind of American history. He also was the artist in residence at one of the museums in Washington, DC. And if you're ever at I think it's the American History Museum or something like that, if you're ever there, you can go to different rooms in the museum and he'll tell you stories about the objects or the rooms that you see there. It's a magical experience recommended to anyone.

中文翻译:

噢天哪，太多了。我每个月要听几百个小时的播客，简直疯了。我可以选很多，但我只说一个。Nate DiMeo 的《记忆宫殿》（The Memory Palace）是一个非常棒的讲故事播客。他会做大约 20 分钟的小品文，通常选自美国历史。他还是华盛顿特区一家博物馆的驻场艺术家。如果你去——我想是美国历史博物馆之类的——如果你在那儿，你可以去博物馆的不同房间，他会为你讲述你看到的物品或房间的故事。这是一种神奇的体验，推荐给所有人。

[00:59:56] Lenny

English:

Wow! I love those. What's a recent movie or TV show that you've really enjoyed?

中文翻译:

哇！我喜欢这种。最近有没有什么你非常喜欢的电影或电视节目？

[01:00:00] Ryan J. Salva

English:

I don't know if this counts as recent, but it's one that I watched recently, which was Arrival. Yeah, that counts. Arrival. Movie ostensibly about aliens, but is really about language and memory. I found that really, really compelling.

中文翻译:

我不知道这算不算“最近”，但这是我最近看的一部，叫《降临》（Arrival）。是的，这算。表面上是关于外星人的电影，但实际上是关于语言和记忆的。我觉得它非常非常有吸引力。

[01:00:20] Lenny

English:

Have you read Ted Chiang books and short stories?

中文翻译:

你读过特德·姜（Ted Chiang）的书和短篇小说吗?

[01:00:23] Ryan J. Salva

English:

I have not. I have not.

中文翻译:

还没，还没读过。

[01:00:24] Lenny

English:

Oh wow! Oh, you would love it. Arrival is from one of his story, I believe, is one of his stories and there's a whole book of many more short stories by the same guy. They're amazing.

中文翻译:

噢哇！你一定会喜欢的。《降临》就是改编自他的一个故事，而且他还有一整本包含更多短篇小说的书。它们太棒了。

[01:00:34] Ryan J. Salva

English:

Brilliant. I've got my weekend cut out for me then.

中文翻译:

太好了。那我这个周末有安排了。

[01:00:39] Lenny

English:

There you go. Just leave work and get to reading. What's a favorite interview question that you like to ask in interviews?

中文翻译:

这就对了。下班就开始读吧。你在面试中最喜欢问的问题是什么?

[01:00:46] Ryan J. Salva

English:

Let's see here. I'll give you a fun one more than it is a challenging one. This is kind of my icebreaker interview question, particularly for more early to mid career product managers. I ask them to teach me something new in one minute. Usually I'll pull up my phone and I'll start the timer. I'll give them a second to think about it and start the timer. They're graded on three different criteria. One is completeness. Did they actually finish the lesson inside of one minute? Two is complexity. It's one thing if you teach me how to, I don't know, pat my head and rub my stomach at the same time.

中文翻译:

让我想想。我给你一个有趣的，而不是挑战性的。这是我的“破冰”面试题，特别是针对职业生涯早期到中期的产品经理。我要求他们在分钟内教给我一件新事物。通常我会拿出手机开始计时。我会给他们一秒钟思考，然后开始计时。评分标准有三个：一是完整性，他们是否真的在分钟内讲完了？二是复杂度，如果你教我如何——我不知道——同时拍头和揉肚子，那是一回事。

[01:01:28] Ryan J. Salva

English:

It's another thing if you teach me something about 18th century ardent connection to religious trends at the time. And then last is really clarity. Oh yeah, clarity is the last one. Clarity is like, do I actually understand? Did I learn something by the end of the lesson? Did they convey the idea fully and wholly?

中文翻译:

但如果你教我一些关于 18 世纪艺术与当时宗教趋势的紧密联系，那就是另一回事了。最后是清晰度。对，清晰度是最后一个。清晰度是指：我真的听懂了吗？课程结束时我学到东西了吗？他们是否完整、全面地传达了那个想法？

[01:01:52] Lenny

English:

I have to ask, what's the most interesting thing somebody has taught in this question?

中文翻译:

我得问问，有人在这个问题上教过你最有趣的东西是什么？

[01:01:57] Ryan J. Salva

English:

My go-to kind of throwaway answer there about did they teach me something about 18th century art and its connection to religious trends at the time, someone taught me that. It was astounding. It was actually a university candidate, so someone who was still in university, and she was from Vanderbilt University.

中文翻译:

我刚才随口说的那个关于“18 世纪艺术及其与当时宗教趋势联系”的例子，真的有人教过我。太令人震惊了。那实际上是一个大学应聘者，也就是还在上大学的人，她来自范德堡大学。

[01:02:18] Lenny

English:

And was that a strong yes hire?

中文翻译:

那是一个“必须录用”的决定吗?

[01:02:20] Ryan J. Salva

English:

It was an extremely strong yes hire. She was freaking amazing. Such a smart person.

中文翻译:

是一个非常坚定的“录用”。她简直太棒了，非常聪明。

[01:02:28] Lenny

English:

Amazing. Final question, who else in the industry would you say you most respect as a thought leader or just influence person?

中文翻译:

太棒了。最后一个问题：在业界，你最尊敬的意见领袖或有影响力的人是谁?

[01:02:36] Ryan J. Salva

English:

There are many, but I think for today I'd probably beat myself up if I didn't say Uga Damore. Uga is the primary researcher who really kind of is the true innovator for Copilot. He deserves credit for the initial work and is a brilliant technologist and futurists. I really, really respect him a lot.

中文翻译:

有很多，但我想如果今天我不提 Uga Damore，我会自责的。Uga 是首席研究员，他实际上是 Copilot 真正的创新者。最初的工作归功于他，他是一位才华横溢的技术专家和未来主义者。我真的非常非常尊敬他。

[01:03:05] Lenny

English:

Amazing. Cool call out. Ryan, this has been so fascinating. You guys are at the forefront of so much interesting work. I honestly can't wait for Copilot for my newsletter so that I can do less work. Maybe that'll come someday. But in any case, I'm excited to see where this whole thing goes. Thank you for being here. Two last questions. Where can folks find you online if they're curious to learn more, reach out? And then is there a way that listeners can be useful to you?

中文翻译:

太棒了，很有意义的致敬。Ryan，这真的太精彩了。你们处于这么多有趣工作的最前沿。老实说，我迫不及待想让 Copilot 帮我写时事通讯，这样我就能少干点活了。也许那天总会到来。无论如何，我很兴奋能看到这一切的发展。谢谢你能来。最后两个问题：如果大家想了解更多或联系你，可以在哪里找到你？另外，听众有什么可以帮到你的吗?

[01:03:33] Ryan J. Salva

English:

Easy one. How can folks find me? I am Ryan J. Salva everywhere, Twitter, GitHub. Pick your choice. LinkedIn, Ryan J. Salva. And then how can folks be useful to me? Please, there is a 60 day free trial of Copilot that is there for everyone to pick up and use. Go try it out. When you do, post either on Twitter or Hacker News or on discussions, GitHub Discussions, your experience.

中文翻译:

简单。怎么找到我？我在哪儿都叫 Ryan J. Salva，Twitter、GitHub 随你挑。LinkedIn 也是 Ryan J. Salva。至于大家怎么帮我？请去试用 Copilot，它有 60 天的免费试用期，每个人都可以用。去试试吧。试用后，请在 Twitter、Hacker News 或 GitHub Discussions 上分享你的体验。

[01:04:07] Ryan J. Salva

English:

Give us the good feedback. Give us the bad feedback. I am so hungry to see how people are using it in novel ways and where they're running up against the rough edges too. Like I said, there's lots of room for us to grow and improve from here, but I'm pretty confident that developers will be pretty freaking amazed at what it's already capable of.

中文翻译:

给我们好的反馈，也给我们坏的反馈。我非常渴望看到人们如何以新颖的方式使用它，以及他们在哪里遇到了困难。就像我说的，我们还有很大的成长和改进空间，但我非常有信心，开发者会对它已经具备的能力感到非常惊讶。

[01:04:30] Lenny

English:

Awesome. Thanks for being here, Ryan.

中文翻译:

太棒了。谢谢你能来，Ryan。

[01:04:31] Ryan J. Salva

English:

Yeah, dude, thank you so much. It's been a lot, a lot of fun.

中文翻译:

是的，伙计，非常感谢。聊得很开心。

[01:04:35] Lenny

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。如果您觉得本期节目有价值，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，这能真正帮助其他听众找到这个播客。您可以在 lennyspodcast.com 找到所有往期节目或了解更多信息。下期节目再见。