A Conversation with Paul Graham - Moderated by Geoff Ralston

Y Combinator · Y Combinator · 2018-09 · source

Interviewer: Well, thank you for coming this morning. We are trying something a little bit different this startup school year. We are not just having our weekly lectures, but we are having some conversations with notable people, and I couldn't be happier to have one of the most notable startup people here. He'll get embarrassed when I say anything complimentary, so I'll try to avoid it. Paul Graham, my friend and the founder of Y Combinator back in 2005, and we're just going to talk, and hopefully it won't be boring. Paul?

Paul Graham: Yes, Jeff.

Interviewer: So, you and I got to know each other strangely at a—it was kind of a startup at the time—Yahoo back in the late 90s, 98.

Paul Graham: Yeah.

Interviewer: Yeah, guys, he wrote Yahoo Mail, some of it. Yeah, Y Combinator has the people who wrote Yahoo Mail and Gmail. That is weird, right? Paul Buchheit, who actually has more of a claim on actually writing Gmail than I do have a claim on writing Yahoo. I wrote—I wrote some important parts of Yahoo Mail, but Paul was kind of by himself doing Gmail, and we had—we were a little more intentional with—we were a little more intentional with Yahoo Mail and had had a team creating something called Rocketmail. But it is a little weird that both of us ended up at YC years later. Uh, but Paul was at Yahoo, I was at Yahoo because we created something called Rocketmail and we were purchased in 97, but Paul's company was bought in 98. And so, how can you—you had this weird idea for a product, for something you thought people would want in the mid-90s, which had to do with like SaaS before SaaS was a thing. How did you come up with this idea for Viaweb, the company that Yahoo bought?

Paul Graham: Okay, so what we made was something that is now called a web app, I believe. At the time, I was the first one, and that's why the company was called Viaweb, because it worked via the web. And the way I remember very vividly, we thought back then—everybody thought that writing software was identical with writing software that ran on the client, and we're still—for those of you who remember the mid-90s, the client meant Windows, right? And like, we knew how to write software for UNIX, and we liked UNIX. We did not know how to write software for Windows, but we knew enough to know that we did not want to learn. And so we were really,

really—we were really, really highly motivated to figure out how to write software without having to write software to run on Windows. And so we thought to ourselves, well, maybe—well, you know, the way I first thought of it actually was that you would have this—you would have—it was an online store builder, right? And I thought that you could—you could send it updates by email. That was—I thought you could send it updates by email, right? Um, and it would change your website. And then I remember thinking, well, if you can use SMTP for shopping carts or whatever, why not HTTP to update your website? Whoa, you could—the software could run on the server and you could control it by clicking on links in the browser. Would that even work, right? But I remember I was like sleeping on a mattress on the floor in Robert's spare bedroom, and I remember I sat up in bed when I thought of this, like—like the letter L, right? It was actually like one of those scenes in a movie. I'm like, "Oh my god, we could avoid having to write software on Windows at all." We never, at that point, we were never thinking about the benefits of web apps, which are great as it turned out. We were just thinking we could like actually do this without ever having to learn Windows.

Interviewer: You did, yeah. At that—like you could deploy then without having to ship software. Did none of that—avoiding Windows, we could actually write—thank you, Gates—never have to learn Windows. Because like the Netscape people had already gone through the pain of making Netscape work on Windows, and just like that was like a hole onto Windows, you know? And people could get through the hole and click on links in the browser and control the software. And so we immediately—we weren't even sure this would work because it was so weird, but we immediately sat down and tried to write this really, really clunky website builder that worked by clicking on links. It was terrible, but it did actually work. It could make a website, and I'm like, "Holy [unclear], this actually works."

Paul Graham: That's weird that like an innovative idea like that just sort of comes out of—no, I mean, it doesn't seem innovative now because all websites work that way, but the first one—like that's—that's going from 0 to 1. How do you—like it's very common, right? Very common for ideas in retrospect, they seem obvious, and you—and you sort of—historians who weren't there sort of straighten out all the kinks in the development, and it just seems like, "Oh, they had this..." You know, it's like this idea—I always hate it when people represent startup ideas as light bulbs, you know? Because as well as being the most cliched metaphor imaginable, it's also false. You don't just like have this idea and then realize it, right? No, it's more like you have this sort of inkling that something—you could do something that somebody else hasn't done before, and it's probably a bad idea, but you're too lazy to learn Windows, and so you want to do it that way, right? Um, and so—or like, who would—or like Zuck starting Facebook, you know, just like, "Let's see what happens," right? The way—implausible if this would ever be a startup, which is good for you all. Hopefully, some of your ideas seem really implausible, right? I love it when ideas seem implausible in the right way. Over the years, I mean, I already had, just from being a hacker, sort of a good nose for this sort of thing, but it has become refined over all these years of dealing with

startups. And when—when people have—I was talking to some people yesterday, and they were talking about two possible things they could do, and one of them seemed so—it seemed like naughty. Not like it was—it was sort of maltreating anyone; it was just sort of taking advantage of something that seems like it shouldn't even be possible, right? Like, like making this software run on the server and just like never making software that seemed to the users to run on their computer, but actually was just talking to them through the browser. That kind of naughtiness. And then I said, "Like, pick that one," because it seems—it seems so outrageous, it seems so hilarious. It seems like there's all these ways we have of talking about what's a good idea, and when you try to match up ideas with—and when you try to match up ideas with them, it's always sort of a complicated—you know, it's like, is this idea naughty, or is it—is it—is it like not quite intuitive, or not so obvious? And it's really always hard to take any particular idea and say, "Is that the one with potential?"

Interviewer: Well, you can't tell. In fact, that's another thing. Um, it's not just that the outcomes of startups are hard to predict. I think to some degree they're actually indeterminate. There's a huge amount of luck involved, you know? And so I think even if—even if Y Combinator were as good at picking startups as you could possibly get—I mean, in fact, you have like 150 startups per batch, of which maybe 5 will become giant or something like that. Hopefully high—that would be good.

Paul Graham: Yeah, well, depends on your definition of giant, right? It's a power-law distribution where, like, if for some definitions of giant, one of—one of the batches will think if five—any definition of giant, yeah, we'll be happy.

Interviewer: So, talking about—I like, oh yeah, the indeterminacy of—yes, even if you were able to pick perfectly, that doesn't mean you could get the batches down to only five startups. You'd probably have to pick twenty. I think even if you were perfect. So it's just—it's hard. Like, it's hard to know. There's this strange mix of extraordinary hard work, an idea that actually has something to it, but is not obvious. It's hard for founders to know.

Paul Graham: Yeah, you don't know yourselves. I mean, I mean, you don't know—comma—yourselves. Not that you don't know yourself, so that it's also true. So, and that's why it's good to be a hacker, because if you're some business person, you know, and you think, "I am going to work on business opportunities," right? Well, I don't know if this is a business opportunity, so I'm not going to work on it. Um, whereas someone who's a hacker who's just doing things for the fun of it, or because they don't like Windows—yeah, laziness to this is actually in some cases really helpful. At least some kinds of—some kinds of laziness. I'm a little scared about this meme escaping into the wild, right? Can't you just see—well, yeah, you just see—it's—it's truly, they said laziness is good, he said from the sofa as he ate another doughnut. You know, it would be better if people thought laziness wasn't good, because the laziness—the kind of the people to you that's good, they do, and they would do anyway even if they thought it's weird. Yeah, well, using laziness not in the

normal sense of laziness, right? Because if you're lazy—there's no such thing as a lazy founder that builds a big company. But—but a certain sort of—it's sort of a reticence to do something for some reason. I'm calling it lazy, but it's not really. It seemed like sort of—it makes you—you know, if you have a natural hatred for gratuitous slop, like, then that produces elegant solutions.

Interviewer: So, in 1995-ish, I was like trying every single idea I could possibly try to start some internet business, and I was doing it all by myself, which was awful. And what I didn't know how to do was to find people to do it with, and I kind of ran into a bunch of people, and I had no clue whether they would be good people to start companies with. And I was thinking about your experience. You said you were like sleeping on Robert's—well, you started Viaweb with two fascinating people, Robert Morris and Trevor Blackwell. And that was a weird mix of names, anyway. How did you pick them as co-founders? How did that happen? And when did you decide they would be good co-founders? Were they good co-founders?

Paul Graham: Alright, so, yeah, they're extremely—they were—they're extremely—they were extremely truculent co-founders who were very good at programming. But I remember Robert was like not into the whole startup thing the entire way. Like, whenever he was on the board, and whenever we got—and whenever you get an acquisition offer, the board has to at least consider it, right? And so we would get these crap, lowball acquisition offers—"We'll give you like two million dollars in our stock," right? Robert would say, "Yes." No, no, Robert would say—like we would vote on it, and Robert would always say, "Well, you know, I have to be honest here, like this would at least mean we could stop working on this thing." And so, yes, I would.

Interviewer: Take this deal, right? I made a deal with Robert that if he ever made a million dollars out of Viaweb, he would get an earring. Robert is not the kind of guy who would enjoy having an earring, let me tell you. And so as soon as the deal closed, me and Trevor—as soon as the deal closed, me and Trevor frog-marched him to the place in Harvard Square where people get their things pierced. Never heard this before. He got him an earring. There's photographs on the internet of Robert Morris with an earring that he didn't keep it when his—but not for long. We should have specified that. I never thought—I never thought to specify, you know? I thought you'd actually get the million dollars. No, no, I really actually did, but I thought he would wear it for a bit longer. Um, when his fiancee saw it, she fell on the floor.

Interviewer: Why did you choose—why did I try killing Trevor? There's okay. So I chose Robert because he was my co-conspirator in everything, and I don't mean I was always—I was always the lead and he was just the co-conspirator. When he did things, I would be his co-conspirator too. So like the internet worm, if—you guys know Robert Morris is pretty famous all on his own. Yeah, he invented buffer overflow, and I remember when he told me about it and I said, "Wow, what a cool idea, you should totally do that." He was the first famous hacker who really got in trouble, I think. Yeah, really in trouble. He was—the way it's true, he was the first person to be prosecuted under

the Computer Fraud and Abuse Act of 1986. Which, so if you're looking for a co-founder, look at people who have been prosecuted because it works. Yeah, he is actually a convicted felon. Okay, this is another—we don't necessarily want to say out there that convicted felons are what you want to look for as co-founders. They're—to look for as co-founders, they're convicted for that. Maybe it was funny because the FBI, like, law enforcement kind of does things by the book, right? Like, imagination is a big thing in, like, Sherlock Holmes stories, but not in actual everyday law enforcement. And so they have these—the FBI agent told me, like, they have these motives they look for: sex, drugs, money, revenge, right? And so, like, Robert did it out of curiosity, and that's just—it's just not on the list. They really had a hard time figuring out what the hell was going on. I think even within the government, there were lots of people who never understood what was going on.

Interviewer: So back to Robert and Trevor. Who—oh, so why did I pick Robert? Trevor, yeah. Robert, Robert, Robert, I picked because I would—I would do everything with him, right? We had all these schemes, and so you knew you could work with him. I mean, I could work with him when I knew it was really good. He was a really good programmer. He could program as fast as he could type. Of course, this was in C, which is a lot of—very verbose language, but he could program as fast as he could type. He was amazing. He would, like, if he didn't like what the operating system would do, he would edit the source and recompile it, and then it would do what he wanted. So, truculence aside, he was exceptional. Like, when he showed up at Harvard, undergrads were only allowed to have accounts on, like, the official undergrad computing system off in the Science Center, and there were all these real computers in the CS department and the Aiken Lab, and so they wouldn't give him an account there. And so he just walked up to a machine, you know, switched it to single-user with him as the superuser, gave himself an account, and then switched it back. Hey, it was curious. Yeah, well, he also wanted accounts on the right machines. He got kicked—this whole thing will be about Robert. Robert is such an interesting character. I shouldn't talk about it—maybe more interesting than anything else we can talk about. He got—anything else we can talk about. He got kicked out of Harvard as an undergrad for reconnecting Harvard to the internet. Um, Harvard—Harvard had been one of the early internet nodes. That's so funny to—to get in trouble for disconnecting a school from the internet. Connecting it can happen for connecting a school. Where were we? Well, we're still trying to figure out—well, we're still trying to figure out why the heck—oh, I should tell you though about how—tell us about the disconnecting really fast. No, no, it's okay. So we have all day. We don't do it. No, we don't. Okay, how long have we—have we have enough time? We have the same watch, more or less. Robert, okay. It was one of the very early internet nodes, and by the time Robert arrived as an undergrad, the connection had died from bit decay. That's like, no one noticed. That's how unimportant the internet was back then, or rather ARPANET, as it was then called. Verily an internet, right? So RTM spent one whole semester working on getting Harvard reconnected. RTM is Robert's—yes, yes, Robert. And he, like, didn't do any work for any of his classes, and he got such bad grades he got kicked out for a year. So we actually used that later as a recruiting technique because we knew there—because

there might be other people who got kicked out for spending all their time on some project. And so we actually went around Harvard and posted up this poster saying, "Did you get kicked out working on some project of your own? We'd like to hire you." I think there's probably—so we got a really good programmer that way. Um, there's probably something—made tons of money off of options because we got acquired, right? And you, like, enjoyed life out of school so much—I enjoyed life out of school so much, I don't think he went back until he was 25. So, um, all right. So why did—why Trevor? Well, so you picked Robert, which was the guy you did everything with, and he was great. Okay, so after we'd been doing—before Viaweb, when you picked Robert, did you know he was gonna be a pain in the ass in so many ways, as you like, or was that a surprise to you? Well, I knew he was a pain in the ass. I didn't know the company was gonna be so hard and it would bring out so much pain. So you didn't realize how hard—tell you the way we got Trevor was very indicative. Yeah, so a month in, we'd been working on this company for a month, and Robert rebels. He says, "We've been working on this company for a whole month and it's still not done." And I thought, "Hmm, maybe we need more programmers." So I said, "All right, Robert, who's the smartest person you know in grad school?" And he said, "Trevor." And I said, "Trevor." And Trevor is one of those people who seems—a lot smarter than you seems so. But it's—Trevor is actually super smart. And so we went and recruited Trevor, and he said, "Oh, okay, that's—that's a really good Trevor." But almost not a Canadian. Trevor has two modes. Oh, okay. And, "Oh, I'm sorry," which is what he does when he does something that breaks everything, which was very common. Very common. And so we got Trevor. I recruited Trevor to work on the thing. He does nothing for two weeks, at least as far as I can tell. He just disappears. And then two weeks later, he says, "Come into my office," and he's rewritten all our goddamn software in Smalltalk. Typical. Yes. Um, so I said, "All right, Trevor, we're not using that, but do this instead," right? And so Trevor, like, Trevor was, like, this super productive hacking monster, but you didn't know that in advance. You just trusted Robert when Robert told you he was really smart. Well, if Robert said he was the smartest, he's definitely the smartest. But there's kind of a deep lesson in that, and how you—how you pick, whether—whether it's employees or co-founders. It's you find someone you trust, and then find someone they trust. Yeah, although here's an interesting point: that works better for some qualities than others. Like intelligence, the intelligent people can judge other intelligent people, but trustworthy people cannot judge other trustworthy people. In fact, trustworthy people are often fooled by untrustworthy people. So, like, Kate Quarto is, like, our architect who did this, is, like, super trustworthy, but she's always fooled by easily sneaky schemers, you know, who take advantage of her. So I guess that means you take recommendations from people, but then you have to add your own—no, no, you can totally trust. But if Robert says someone's smart and I disagree, I think I—someone's smart and I disagree, I think I must be wrong, right? Yeah, but you need to—it works for some qualities better than others. Let's hold, right? You want them to be trustworthy when they join—them to be trustworthy when they join you as a co-founder, so you have to add that for the trustworthiness. Yes, this whole thing of, like, networking doesn't work. It doesn't. You have to have someone you can judge whether people are trustworthy, which I am—I am not even good at. I am not good at. The person who does that for Y

Combinator is Jessica, the social radar, the secret power behind everything.

Interviewer: So you did hire people at Viaweb, yes? What did you learn about hiring in your, like, your first startup? Like, so you learned that they—if they were getting lousy grades, you start working on a project that seemed good, which is—which is kind of counterintuitive. A lot of things we seem to do at YC end up being counterintuitive. Startups in general are very, very counterintuitive, and so that's actually why YC exists. If it was obvious what to do in starting a startup, we wouldn't—YC couldn't really add much by teaching people how to do it, right? And so I often say—I often, in fact, practically every day if I do office hours, I'm reminded of this—that, like, what Y Combinator does is tell founders things that they ignore, right? So we tell people, "Don't hire too fast," and then they go off and hire too fast, and then they come back later and say, "Oh, I wish we'd listened." But all the things we tell people, we tell people the counterintuitive stuff, not the obvious stuff. And then what—the obvious stuff, and then what—the obvious stuff, and then what—counterintuitive means is it sounds wrong, and so they go with their gut and do the wrong thing instead, and then hopefully catch the mistake in time. Why is it that startups are so counterintuitive? You know, that would be—if that would be very much—writing an essay about that would be worth writing an essay about, why startups are so counterintuitive. I don't know. I don't know. It's—I could come up with some theories, but I have a feeling the answer is so interestingly complicated that would be unlikely to come—and it's really just—we were just talking about one of the things that—talking about one of the things that—talking about one of the things that you—you did write an essay about, which is somewhat counterintuitive. When you start a startup and you're, like, "Okay, now I'm gonna grow," and then you write a whole bunch of stuff, but you've already software that helps you grow, and you're—at a whole—he says, "Don't do that. Do things that don't scale in the beginning." Yeah, do things that don't scale, because growth is—maybe you can start by saying what it means to do things that don't scale, because I'm afraid the best thing you did not already—I'm sure you've all read all as well as I say, but this is a—this is actually very important, because most of you all have early startups, and when you do an early startup, it's become—

Interviewer: Our mantra at Y Combinator, and I've seen it again and again, you should not try to do too much in the beginning. You should do things that don't scale. What "doing things that don't scale" means specifically is doing things in a sort of handmade, artisanal, painstaking way that you feel like, "Yeah, it would be great if you could do things that way forever," but you, in the back of your mind, you think to yourself, "Well, there's no way we can keep doing this and become giant," right? And what "doing things that don't scale" means is do those things early on anyway, because you know if you don't do them, you'll never be big, and you've got nothing to lose. And also, you learn a lot from it. You learn a lot. And so, I discovered a lot of these things that we teach startups at Y Combinator—or you teach startups at Y Combinator, I'm retired, I don't know if you know that—but a lot of things you guys teach startups are things that I hit myself and didn't realize that they were actually common startup lessons. One of them is doing things very manually for your

early customers. Like, it's so important to get early customers that if you have to do a ton of manual stuff, that's okay. You'll learn a lot from it. Did you guys do manual stuff?

Paul Graham: Oh yeah, well, totally. And we thought, "Oh, we're doing it wrong. This is so lame." And in fact, it was exactly the right thing. We made an online store builder. You could build a store on the internet and sell stuff. And we would go to would-be customers and say, "Would you like to use our easy online store builder?" And they would say, "No." And we would say, "But you want an online store, right?" And they would say, "Yeah." And they would say, "Well, what if we..." We would say, "What if we used our software for you to make an online store and then you could have it? Would that be good?" And they'd say, "All right." And so we'd like... we can't get anyone to use our software, but at least they're willing to let us use it for them. And it seemed... it seemed so lame. I learned so much [unclear] about like direct marketing, as it turns out it's called. We were members of the DMA, the Direct Marketing Association. That's what, you know, in every business, card-carrying member of the DMA. We were in every business. They have a name for the business that's not the name outsiders call it. Like fast food, no, no, no, they don't call it that. They call it fast casual within the industry, right? And in the catalog business, they used to call the direct marketing industry, right? Um, so that's what the DMA meant back in those days, the catalog business. We signed up for every catalog, you know, these catalogs you get in the mail. We would just like write away and ask for more.

Interviewer: Well, you had this like bookshelf full of every catalog ever.

Paul Graham: Yeah, it seems full of every catalog ever. Yeah, it seems like almost like if you were building a new search engine, what you ought to do is have a... have your box out there and then just take all the searches manually and like get results and throw them out there and see what... see what happens. I'm trying to think what we did for mail that was manual in the beginning. We... sort of a manual ad server. Like we didn't really have ad-serving technology. I just threw it... like I wrote it, so it wasn't really good. And we would just throw them in there... that there's a deep... throw them in there... that there's a deep lesson there for... for... for people, though. Like, what happens if you think you know the solution and you build lots of stuff for it, you're going to be wrong because you just don't know. And having to use our software myself made it much better because I would be using our software to make someone's website, right? And I was also the guy who wrote the site builder. And so I would be using it and think to myself, "This is inconvenient." It's like, in the middle of building their site, I would change the software. And I mean ship, as in like MVP, and like that's like that's the great thing about server-based software. Ship... the thing about server-based software, ship is the UNIX MVP command, so well, no, CP. I would want to keep a copy, but... but I would like change the software in the middle of using it and then go back to working on their website. And boy, did that make... that made the software much better, that I had to use it.

Interviewer: Yeah. So, um, so picking co-founders, you want to pick the best people who you can somehow believe are trustworthy. It's probably helpful for a lot of people. I think a lot of people taking the class are sole founders now. Yeah, why is it that that sole founders... that being a sole founder is so difficult?

Paul Graham: Wow. Um, there's so many different reasons. Like, I think the hardest part is morale. There's no one... there's like, if you have multiple people, they keep one another going when things are going badly. And there's no one to cheer you up. No one to cheer you up. That's what's hard for me. I remember '95, I'd be like, "I know this internet thing is going to be real, but damn."

Interviewer: Oh, so you were a sole founder?

Paul Graham: Yeah, yeah. That's why his hair is grayer than mine. The... yeah, like being a sole founder is hard. Like, it's hard because there's a million reasons why what you're doing isn't going to work, and mostly you're right. So it's easy to convince yourself. You used to say this, and I always remember, you used to say, "Look, you have two choices as a founder." Do you remember what you would say after that?

Interviewer: Well, there's so many different cases, this versions of this. What would he would say? You could either quit or get rich.

Paul Graham: Mmm, yeah, die or get rich. Like the company... a lot of versions. Well, that's what happens with startups mostly. The company either dies or it... something succeeds, you get rich. And you know, getting rich can mean, you know, you have a nice little business where you're the boss and you have 20 or 50 employees, or it could mean your Airbnb, right? They're both good outcomes. Not as good for your investors necessarily, but for you, it's really good. And doing things that don't scale, that's hard, you know, and internalizing what that really means. I was thinking about, you know, one of the... it's the sort of things that don't scale we talked about at YC all the time. We say you're supposed to, for success, talk to users and write code. You... you talked. And if you think about it, talking to users, especially as sort of the founder, doesn't really scale very well. But that's what it forces you to do. If you... that's what it forces you to do if you have to do everything. It doesn't scale. It forces you to learn, right? In the beginning, you only have 10 customers. You want to grow to 20%. You want to grow 10% next week. 10% a week is an ambitious goal. You only got to get one more customer. You can go out and do that very manually, right? And then next week you have 11 customers. You have to get 1.1 customers, right? So this is basically one, right? You just keep going out there and doing things manually. And as long as your growth rate is good, it doesn't matter how small the number is, because a grow... a constant growth rate means exponential growth, and that means the base number will soon take care of itself. It's kind of thinking about what makes companies good, what makes companies fail. And I

was thinking about maybe how we could talk a little bit about some of those... those aspects of how people can kind of figure out how they're doing here. We have thousands of companies out in Startup School and trying to figure out what to do next and whether they're doing the right thing and whether they're failing and whether they're on the right path. What makes companies fail most of the time is poor execution by the founders, right? Um, we... we often talk to startups who are worried about competitors. And one good event... one advantage of YC having funded so many companies, you have a really large data set. And like, how many companies have been killed by competitors? One. More than one out of 1,900, right? So I tell startups that basically you have the same protection against competitors that light aircraft have against crashing into other light aircraft when flying through clouds. You know what the protection is? Space is large. Hey, these little Cessnas don't have any radar in them, you know? You can't see in clouds. It's... it... it falls under the rubric of counterintuitive advice, because one of the things that happens all the time is people come up and say, "Oh my god, there's another new company. What should I... you know, that's in exactly my space. What should I do?" Right? And it's sort of like if you're running the 100 meters, right, and suddenly there appears another lane with another runner in it, what should you do? You run as fast as you can, just like you presumably were. Or, hey, if they're better than you, they'll win. And if not, then... then you'll win. And that's a good... then... then you'll win. And that's a good reminder. Could everyone please silence your phones if you haven't already?

Interviewer: Yeah. What else? Like... like one of the... one of the things we do a lot is try to judge, right, companies and... and founders, Nick, who's gonna succeed. And... and you guys are doing the same thing, except you're trying to pick yourselves, hopefully. What... what is it that then makes startup succeed? That makes... then make startup succeed? That makes startup succeed?

Paul Graham: Well, you have to start up succeed. Well, you have to start up succeed. Well, you have to start with a good founding team. Ideally not just one person. That's my quote. You can do it with one person. Well, I couldn't. I... I ended up... I ended up... couldn't... I... I ended up... I ended up having to join other people. Like, I actually think... oh, that's majority. That was not rock, you know? I was... no way. Like, I couldn't do it. I'm not built that way. I think most people aren't built that way. That's... that's my questions about Robert and Trevor, that, you know, it's just so difficult. I would not try and start a startup just with... just myself as the founder. Hmm. I don't think I could do it. Yeah, it's lonely enough anyway. Do it. Yeah, it's lonely enough anyway. Think you can, but I couldn't. So, um, you need a good, solid founding team who know one another well and who can work together. Just like stuff... pause there. Like, knowing one another well, why does that even matter? Because what happens in a startup will shake your relationship. If there's any sort of flakiness or uncertainty or disloyalty or whatever, like lingering in there, it will emerge under the stresses of a startup. Yeah, it's like a marriage, only harder. You spend more time with... at purpose... other differences. There's that purpose... other differences. There's potentially, yeah. Okay, so you get a good founding team. You get someone who you can stand stress with. So there's two categories of stuff. There's two categories of answers we could give here.

Like, there's the answer you give to investors about how to pick startups, and there's the... and then there's like only a subset of that is useful to founders, right? So, like, if... like, if you, you know, you could tell investors you want to pick smart people, right? But founders are as smart as they are. If they aren't, what can we tell them? Get smarter, right? And actually, it turns out it's not that important to be smart. It's much, much more important to be determined. If... if you imagine... if you imagine this hypothetical person who's like 100 out of 100 for smart and 100 out of 100 for determination, right? And then you start taking away determination, you know, it doesn't take very long before you have this...

Interviewer: Sort of ineffectual but brilliant person, whereas if you take someone who's like super, super determined and you start taking away smartness bit by bit—I mean, taking away smartness bit by bit, I mean eventually you get to some guy who owns a lot of taxi medallions, but he's still rich, right? Or like a trash hauling business or something like that. But like you can take away a lot of smart—yeah, you take a lot of smart and take away a bunch of ethics at the same time, and then you get that guy, right? He could be President.

Paul Graham: Yeah, I've been thinking about what like, who thought that Trump would actually teach an important lesson. Yeah, for startups, it's all about determination. I was thinking about these axes we think about, and those are two, like, the intelligence axis and the determination axis. And obviously, if you were really intelligent, really determined, that's awesome. There's, by the way, only one person who has to be super determined, right? So like, you know, you can have one person who really wants to start the company and one person—you and Robert Morris, exactly—who after a month is like, "What, we're still working?" But that depends on this tight bond, where you can drag that person along with you; otherwise, they'll go away. But I was thinking there's actually another axis, which is you have to have the ability somewhere on your team to project—or think of—but because your idea is gonna suck, and then it'll suck less. And so you have to have the ability to think, "Where should we navigate?" Like, you have to have some sort of creativity that allows you to make the right choices. And you know, actually, you might not have to be that creative if you care enough about users. You can just follow what will make users happy, the way a scientist follows the truth, and eventually, without much thinking on your part, the need to grow will give you this product idea that's actually the product. That's the result of evolution. So maybe you're either sort of Steve Jobs and can intuit what customers need, or you just have to have the skill of actually talking to customers and understanding what they're saying, which isn't always obvious. You know, Steve Jobs, his trick was he satisfied users to accept he was the user. It's like, "I don't want to have any phone jack anymore, so no more phone jacks for anybody," right? Maybe he was just lucky that his desires were generic enough that it was closer to everybody.

Interviewer: Yeah, yeah, he was living in the future.

Paul Graham: Yeah, like he didn't really like the internet except for reading the New York Times,

so he never really got that right.

Interviewer: Hmm, interesting. Didn't you work at Apple at one point after you got bought?

Paul Graham: Did for, it was months.

Interviewer: What was it like? What was Steve like? Could you tell from where you were in Apple what Steve was like?

Paul Graham: Yeah, I got to talk to him a few times. He, um, he's different.

Interviewer: Tell us, tell us how. Are there useful lessons you can learn from him, or is he a one-off?

Paul Graham: I wouldn't presume. I only met with him a handful of times, and I wouldn't presume to know too much about him.

Interviewer: Yeah, you had a lot of indirect evidence and stories and stuff like that. Yeah, like, do you ever see founders at Y Combinator you think to yourself, "Oh, this is like Steve Jobs"?

Paul Graham: I don't think we'd—I don't think we'd fund—I don't think we'd fund a Steve Jobs at Y Combinator.

Interviewer: Well, that's not good.

Paul Graham: Yeah, and the reason is, especially when Steve Jobs was starting off, he was an [unclear]. Oh, I think actually you would find him, because Steve Jobs would make you find him, maybe, right? Maybe we try—like, we try to avoid people who seem like they're [unclear]. And I think like, and like, you know, you wrote a whole essay on being nice.

Interviewer: Yeah, the Ronco principle, right? There's a couple—yeah, be nice, yeah, and called "Be Good."

Paul Graham: Be good, right. And I don't think he would have passed the test, especially when he was younger. Which is—I worry about that, because it's true that it's usually good to be nice and friendly and benevolent. He wasn't. Maybe if Y Combinator had caught him early, we could have helped. Maybe, maybe that would be—maybe that's part of what made him successful in the beginning. I don't know. I don't know. It worked for Donald Trump.

Interviewer: Or did it? That's a different kind of example, right? Yeah, we should know what—like,

in business, determination matters more. But like, business is not identical with startups. I want to leave these guys some time to ask questions, but I want—like, what are they gonna get wrong? What mistakes will these people make that maybe hearing them might help? It probably won't, because you won't listen.

Paul Graham: But what you will get wrong is that you will not pay enough attention to users. You will have—you will make up some idea in your own head that you will call your vision, and then you will spend a lot of time thinking about your vision in a cafe by yourself and build some elaborate thing without going and talking to users, because that's doing sales, which is a pain in the ass. And they might say no. And you'd be way better off finding someone, anyone, who has a problem that they will pay you to fix, and fixing it, and then seeing if you can find more people like that. Best case is if you yourself have the problem, right? But like, you will not ship fast enough because you don't want to face the—you're embarrassed to ship something unfinished, and you don't want to face the likely feedback. It's humbling and humiliating when they tell you how shitty what you built actually is, right? But that's the only way you get it better. Yeah, so you will shrink from contact with the real world or contact with your users. That's the mistake you will make.

Interviewer: Yeah, there's a Paul Graham-ism in there, which is all around launching—it's launching before you're ready. That's something—tell people the story of—it's Reid Hoffman actually who harped on that one early, not me.

Paul Graham: He—there's, I think he said, "If you're not embarrassed by what you launched, then you launched too late."

Interviewer: I remember when I was joining Y Combinator and at the same time this—not necessarily good idea of launching Imagine K12, this EdTech version of Y Combinator, and I came in to talk to Paul and he's like, "So how's it going with Imagine K12?" Because he helped out a lot. And I said, "Well," it was probably called something else—well, no, no, still, it was by then you'd already—he actually gave us the name Imagine K12—and so it was called Imagine K12, but it's like, "When you gonna launch?" And I was like, "Well, we're doing our PR plan and we're finishing the software." So this was like, you know, we're gonna have a summer batch, and this was probably in February or something. And I said, "So we have like, you know, probably in like six weeks." He said, "Why don't you just launch now?" And I was like, "Damn, he just did it to me." And we did, we launched—we launched like within a week. We launched.

Paul Graham: And so that is a mistake you'll make. You'll wait too long. You'll be embarrassed. You'll—we launched Y Combinator probably in less than a week of having the idea. Of course, launching it was merely saying—was merely posting a website. And we didn't have—we didn't have any software in the beginning. People—we just had this ASCII form that people would fill out and reply to us by email. The first couple batches, the applications were by email. We would print

them out and pass them around amongst the partners, and like, we would all write grades on the top like we were grading papers, and then we would sort this heap of papers, and that was the application process. Talking about doing things that don't scale—try to do that with thousands of applications now. It would be harder. So we have lots of software that does that, because eventually you scale, you know? Eventually you build software when you learn what software you ought to be building. It'd be fun to spend a lot of time—more time talking about how YC got started, but I think rather than that, let's—how do you have left? We can probably go to around 12:00, and or all videos that get too long, like, no one wants to watch anyway. So we've already been going for about 40 minutes or so, so let's open it up to any questions you guys might have.

Interviewer: Yeah. Are you saying that if you recruit your friends as co-founders, you might lose them as friends?

Paul Graham: I don't think that's a big danger. I don't think you have—so the question was, "Should I have my friends be co-founders and then maybe risk losing them as friends, or should I just take someone off the street where I don't have the risk of losing them as friends, but they might break my company?" Is that—was that your question? Do I get that right?

Interviewer: I already answered it, dude. Come on. I just wanted to make sure they heard the question too, just because I repeated the question. How do you deal with different commitment levels of co-founders? And if you do have to fire a co-founder, what should you do?

Paul Graham: Okay, so the way you deal with different commitment levels is you ask yourself, "Would I rather have 30% of this person or 100% of some other person?" And in my case, I would rather have like 10% of Robert Morris's brain than 100% of almost anybody else's. So it's an easy decision. How do you fire co-founders in Y Combinator? The answer is you ask Carolyn Levy. Y Combinator has so much experience with dealing with crap like firing co-founders, and like, you just go and talk to them and they say, "Here, you do this and this and this," right? And it's probably online too, so you could Google it. Really, is it? YC published how to fire?

Interviewer: We've—we have published. We've actually, if you look at some of the talks Carolyn has given and Kirsty, I'm pretty sure they've covered that particular issue.

Paul Graham: One way you deal with it, by the way, is making sure that someone has a little bit more stock, perhaps. Like, it's usually better to have somewhat equal shares, not—but not completely equal. And if one person has 51% and the other person has 49%, then they can, you know, deadlock.

Interviewer: Yes. So when you talk about launch, is launching to like a private beta to a couple of people good enough? Or what do you—you just launch? So what does launch really mean? It

means you have actually—

Paul Graham: Okay, so the question is, "Is launching in a private beta to a few people good enough for you to have to launch the general public?" I will take launching in just a private beta to a couple people, because that's closer to the—that's closer to full launch than like just sitting thinking of your idea in a cafe.

Interviewer: Next. Yes. What do you think about the currently raising in the project? Maybe a little bit late. So basically, you're asking about ICOs, yeah? What do you think about ICOs, Paul?

Paul Graham: I haven't—I know nothing about this world. I've heard there's a huge amount of crypto money floating around. So like, if there's money floating around, it's often a good idea for startups.

Interviewer: To try and get it beyond that, I don't know. But I will say, like, one of the things we've seen a lot at Y Combinator is it's not always right to raise too much money. And even if you don't—it's like the IPO thing, by definition, it's always wrong. Yeah, the ways a lot of money, and the thing about—there's a couple reasons for that. One, you dilute more than you necessarily need to. But the other reason is that money just sitting there has sort of a gravitational—it makes you suspend it, and it makes you do stupid things. So it's like these companies that have raised a billion dollars on ICOs, and I don't know if they'll ever have a real company where you get, like, you get a bunch of LPs and raise a billion dollars, and then you just hire so many more people than you actually need, right? I know that's the difference when I see—oh, I get it. But just having a billion dollars when you only barely have an idea is not necessarily—you know, Magic Leap raised 2.3 billion dollars, and they seem to have almost nothing. By the way, that is the trick for getting your question answered. I learned that a long time ago: you raise your hand as people are finishing the preceding question. And I see that guy's hand over there; he knows the trick. Go ahead.

Paul Graham: What is it?

Interviewer: My best tips for a great user interview?

Paul Graham: Um, well, you want to figure out not just what they think is wrong, but what's actually wrong—like what's missing in their life, right? So just start talking to them, like say, "What would you like to do? What would you like to be able to do that you can't?" And they might just, you know, they'll tend to give you an answer that's a subset, right? Um, like, you know, if you were asking someone about email, they'd like to say, "I like, you know, it's really important to me to be able to mark emails as unread," right? And like, what does it mean "unread"? That whole unread thing is a sign that the whole inbox is being misused. What's really going on is it's a to-do

list, you know? And that's why you have to mark items as unread so they stay on your to-do list. Um, and so you could—so you start asking them what's wrong, and then you try and figure out what they really mean, what they're really getting at, which they themselves might not know, you know? And then you start asking them, "So what if you could do such-and-such? What if you could do such-and-such?" Right? So start by asking them questions about their life, and then get hypothetical on them. Okay, we should ask someone on this side. You've taken over, so you just keep calling them. You're doing great. But you—you pick. Go all right there in the way corner.

Interviewer: How do you justify the financial risk of launching a shitty product?

Paul Graham: Okay, so the risk of launching early is not as great as the risk of launching late. It's not like there's only risk on one side, right? There's a risk of launching late, too. So you got to have a rule of thumb about when to launch. And my rule of thumb—the way I like—I have this phrase for it that didn't become as sticky as Minimum Viable Product because it describes what the Minimum Viable Product has, but it's a "quantum of utility," right? Um, launch as soon as you have a quantum of utility, which means as soon as there's one person in the world who is glad that you launched because now they can do something that they couldn't do. When there's—if you don't—if you have something that if you launched it no one would be happy, then you're not ready to launch, right? As soon as there's at least somebody out there who would say, "Oh, here's this thing I can use because I can do X," and before that I couldn't do it, then you should launch. As soon as you have a quantum of utility. It sounds like there's some [unclear] utility function, like if it makes too many people happy and, like, one person happy and, like, a hundred really unhappy, maybe it's not ready. But if you, you know, Paul Buchheit, he says if you have ten people who really love your product, that's a good place to be. Yeah, all you need is—if there's ten people who are super excited, totally, totally launch. And nobody else cares, that is perfectly fine. That's great.

Interviewer: Okay, well, the danger of building—okay, so the question is how do you decide between building what people need and building what people want, right?

Paul Graham: Well, the problem is what you really want to be doing is build something people will choose to use or buy, more becoming your user or something like that. And so here's a counterexample to the whole "need versus want" thing: what people need is healthy food; what people buy is unhealthy food. So if you're a business, which one do you want to do? Okay, I mean, yeah, ideally you want to make healthy food that also tastes good, right? But really, um, you can't —you can't push this too far. I mean, I myself would not start some company making junk food or something like that. But if you are making food—or something where need and want are very different, be real careful about picking "need," you know, or you'll go out, and your goal will be to cleanse the world of sin or something like that. It's easy to delude yourself as a founder to think you're—yeah, the other danger is what they need is what you think they—what you're implicitly talking about is what you think they need. Cleanse the world of your definition of sin, right? Yes,

yeah. Normal as a founder to see multiple different solutions. Yeah, sure, absolutely. You just have to pick one. Pick the one that you think could get usage quickest.

Interviewer: What's the founder DNA? Founder DNA bothers you? You mean what makes someone a founder?

Paul Graham: Someone a founder, yes. It choose to be one. What makes someone a good founder? A combination of positive and negative things. I think positive things are things like determination and, you know, willingness to try new things. But then there may also be negative things, like if you are—I think if you have worked for a large company for twenty years, you might not be a founder unless you were forced to for visa reasons. Because if you were the kind of person that would make a good founder, you wouldn't be able to stand working for a large company for twenty years, right? And so actually, we've noticed certain companies where the alumni of these companies tend to make bad founders, and it's because no one who would make a good founder could have stood working for these companies for very long. Okay, maybe we should pick one on this side. Okay, what do you—what are you pricing during launch?

Paul Graham: Well, you can probably guess if you know your business. You know roughly what you should charge, so you should probably just go with whatever you'd guess. If you want to be more scientific about it, you could try talking to some potential users. Most of the time you have some sort of tame users who are actual users but are also friends of yours or something like that, or family, and you can ask them, "Tell me honestly, like, what would you actually pay for this?" Right? Um, you can always change your prices later, though. If you want to lower your prices, no one's gonna complain. And if you want to raise your prices, you just grandfather your existing users, which, if you have exponential growth, will always be a tiny subset of your total users, and then no one will complain about that either. So don't sweat about it too much. One way I think about that is it just seems like in the beginning what you need more than anything is customers, and you want them to pay. So you will need to pick a price that gets you customers so you'll learn, because you learned so much from those customers. And you're right, don't leave your customer price—yeah, 'cause in the beginning you just want growth, because the customers teach you. The first customers don't just give you money; they teach you. All right, two more questions. You choose. You pick two. Okay, right up front here.

Interviewer: Angel investing. So my name is Karen, and I'm with Chat Goose, which is a chat manager. So three tips for dealing with and finding angel investors if you don't have, you know, the original goals, etc.?

Paul Graham: Well, we never talked about fundraising at all. Well, maybe I'll have to do this whole thing again. That's good. Fundraising, as it is, is the thing to occupy your thoughts as a founder is overrated. It's better not to be spending all your time thinking about fundraising. We come to it

much later in the class; we don't need to. All right, so the question was how to find angel investors. Well, that's the part—that's the thing I'm particularly ill-suited to answering, that question, because Y Combinator is like this machine for funneling angel investors into all the startups that it accepts. So, like, how do you find angel investors if you don't? It's like asking a pilot how to, you know, walk along and track on the ground. I know the least about that. Here's the thing: angel investors are looking for you. So get out there. Go to—I don't know—well, there's lots of startup events. The angels are ancient, like, they're not necessarily good angels, but angels are out there looking. And the best way to actually meet angels is to get one to invest in you and then to introduce you to people. Actually, I know the answer to this. I know the answer: find people who work at startups and ask them to introduce you to their investors. That's the way to do it. That's good. That's a good answer. Okay, last question. Paul, you get to choose. Maybe from this side right here?

Interviewer: Why is he thinking of going after high school students to fund?

Paul Graham: Well, I don't know, because I don't think on behalf of YC anymore. But you better not be, because that would be an evil thing to do. There are plenty of high school students who could start successful startups, but they shouldn't. Just because you could start a successful startup doesn't mean you should, because if you start a successful startup, like, that means the, you know, the footloose and fancy-free days of your life are over. Hey, that—like, you're going for that company. But we have funded kids in high school, but only because they're already going—like, it's that we're not encouraging that. I agree. You think to go to high school, encourage kids to, like, do this incredibly hard thing instead of having fun and, like, being kids? That's awful. I actually think that it's [unclear] optimization to—what you should be doing when you're in high school and even college is you should be figuring out what the options are, not picking one option and running with it, right? And, you know, it just turns out that usually people who are in school and then stop to start companies—it's like, you know, there are dropouts from Harvard, we know.

Interviewer: The classic examples, but if you have like if you drop out and just go do it and it's you're on a path, that's one thing. But most people, like, they're dabbling and they have a fallback, and that's not a good way to be a startup founder. You have to be a hundred percent committed, and so we'll fund people in college, maybe even high school, if they're already clearly a 100% committed. Otherwise, no. Whatever trend I think that trend you're talking about is that is that you graduate from high school and why bother getting a college degree because you're already a fantastic hacker, so I'm going to hire you. And if that person wants to enter the professional workforce, well, that that's great. I don't know. I was I don't think I would it would have been good for me.

Paul Graham: It wouldn't have been good for me either. But you know, like, go off and start just some startup. I just think between 18 and 22 you grow a lot as a human being who's human and

humane, and it's not clear what society will turn into if we if all those folks enter the workforce. It's good to like mess around and do a whole bunch of different things in your early 20s if you could have work. Yeah, whether it's whether this messing around doing a whole bunch of things takes the form of college or something else, just like don't be careful because like starting a startup is like catching a dragon by the tail. If it works, it like be careful at what point in your life you do that.

Interviewer: Paul, thank you very much for coming in.

Paul Graham: Thank you.
