How to Start Google

March 2024

(This is a talk I gave to 14 and 15 year olds about what to do now if they might want to start a startup later. Lots of schools think they should tell students something about startups. This is what I think they should tell them.)

Most of you probably think that when you're released into the so-called real world you'll eventually have to get some kind of job. That's not true, and today I'm going to talk about a trick you can use to avoid ever having to get a job.

The trick is to start your own company. So it's not a trick for avoiding *work*, because if you start your own company you'll work harder than you would if you had an ordinary job. But you will avoid many of the annoying things that come with a job, including a boss telling you what to do.

It's more exciting to work on your own project than someone else's. And you can also get a lot richer. In fact, this is the standard way to get really rich. If you look at the lists of the richest people that occasionally get published in the press, nearly all of them did it by starting their own companies.

Starting your own company can mean anything from starting a barber shop to starting Google. I'm here to talk about one extreme end of that continuum. I'm going to tell you how to start Google.

The companies at the Google end of the continuum are called startups when they're young. The reason I know about them is that my wife Jessica and I started something called Y Combinator that is basically a startup factory. Since 2005, Y Combinator has funded over 4000 startups. So we know exactly what you need to start a startup, because we've helped people do it for the last 19 years.

You might have thought I was joking when I said I was going to tell you how to start Google. You might be thinking "How could *we* start Google?" But that's effectively what the people who did start Google were thinking before they started it. If you'd told Larry Page and Sergey Brin, the founders of Google, that the company they were about to start would one day be worth over a trillion dollars, their heads would have exploded.

All you can know when you start working on a startup is that it seems worth pursuing. You can't know whether it will turn into a company worth billions or one that goes out of business. So when I say I'm going to tell you how to start Google, I mean I'm going to tell you how to get to the point where you can start a company that has as much chance of being Google as Google had of being Google. [1]

How do you get from where you are now to the point where you

can start a successful startup? You need three things. You need to be good at some kind of technology, you need an idea for what you're going to build, and you need cofounders to start the company with.

How do you get good at technology? And how do you choose which technology to get good at? Both of those questions turn out to have the same answer: work on your own projects. Don't try to guess whether gene editing or LLMs or rockets will turn out to be the most valuable technology to know about. No one can predict that. Just work on whatever interests you the most. You'll work much harder on something you're interested in than something you're doing because you think you're supposed to.

If you're not sure what technology to get good at, get good at programming. That has been the source of the median startup for the last 30 years, and this is probably not going to change in the next 10.

Those of you who are taking computer science classes in school may at this point be thinking, ok, we've got this sorted. We're already being taught all about programming. But sorry, this is not enough. You have to be working on your own projects, not just learning stuff in classes. You can do well in computer science classes without ever really learning to program. In fact you can graduate with a degree in computer science from a top university and still not be any good at programming. That's why tech companies all make you take a coding test before they'll hire you, regardless of where you went to university or how well you did there. They know grades and exam results prove nothing.

If you really want to learn to program, you have to work on your own projects. You learn so much faster that way. Imagine you're writing a game and there's something you want to do in it, and you don't know how. You're going to figure out how a lot faster than you'd learn anything in a class.

You don't have to learn programming, though. If you're wondering what counts as technology, it includes practically everything you could describe using the words "make" or "build." So welding would count, or making clothes, or making videos. Whatever you're most interested in. The critical distinction is whether you're producing or just consuming. Are you writing computer games, or just playing them? That's the cutoff.

Steve Jobs, the founder of Apple, spent time when he was a teenager studying calligraphy — the sort of beautiful writing that you see in medieval manuscripts. No one, including him, thought that this would help him in his career. He was just doing it because he was interested in it. But it turned out to help him a lot. The computer that made Apple really big, the Macintosh, came out at just the moment when computers got powerful enough to make letters like the ones in printed books instead of the computery-looking letters you see in 8 bit games. Apple destroyed everyone else at this, and one reason was that Steve was one of the few people in the computer business who really got graphic design.

Don't feel like your projects have to be *serious*. They can be as frivolous as you like, so long as you're building things you're excited about. Probably 90% of programmers start out building

games. They and their friends like to play games. So they build the kind of things they and their friends want. And that's exactly what you should be doing at 15 if you want to start a startup one day.

You don't have to do just one project. In fact it's good to learn about multiple things. Steve Jobs didn't just learn calligraphy. He also learned about electronics, which was even more valuable. Whatever you're interested in. (Do you notice a theme here?)

So that's the first of the three things you need, to get good at some kind or kinds of technology. You do it the same way you get good at the violin or football: practice. If you start a startup at 22, and you start writing your own programs now, then by the time you start the company you'll have spent at least 7 years practicing writing code, and you can get pretty good at anything after practicing it for 7 years.

Let's suppose you're 22 and you've succeeded: You're now really good at some technology. How do you get startup ideas? It might seem like that's the hard part. Even if you are a good programmer, how do you get the idea to start Google?

Actually it's easy to get startup ideas once you're good at technology. Once you're good at some technology, when you look at the world you see dotted outlines around the things that are missing. You start to be able to see both the things that are missing from the technology itself, and all the broken things that could be fixed using it, and each one of these is a potential startup.

In the town near our house there's a shop with a sign warning that the door is hard to close. The sign has been there for several years. To the people in the shop it must seem like this mysterious natural phenomenon that the door sticks, and all they can do is put up a sign warning customers about it. But any carpenter looking at this situation would think "why don't you just plane off the part that sticks?"

Once you're good at programming, all the missing software in the world starts to become as obvious as a sticking door to a carpenter. I'll give you a real world example. Back in the 20th century, American universities used to publish printed directories with all the students' names and contact info. When I tell you what these directories were called, you'll know which startup I'm talking about. They were called facebooks, because they usually had a picture of each student next to their name.

So Mark Zuckerberg shows up at Harvard in 2002, and the university still hasn't gotten the facebook online. Each individual house has an online facebook, but there isn't one for the whole university. The university administration has been diligently having meetings about this, and will probably have solved the problem in another decade or so. Most of the students don't consciously notice that anything is wrong. But Mark is a programmer. He looks at this situation and thinks "Well, this is stupid. I could write a program to fix this in one night. Just let people upload their own photos and then combine the data into a new site for the whole university." So he does. And almost literally overnight he has thousands of users.

Of course Facebook was not a startup yet. It was just a... project. There's that word again. Projects aren't just the best way to learn about technology. They're also the best source of startup ideas.

Facebook was not unusual in this respect. Apple and Google also began as projects. Apple wasn't meant to be a company. Steve Wozniak just wanted to build his own computer. It only turned into a company when Steve Jobs said "Hey, I wonder if we could sell plans for this computer to other people." That's how Apple started. They weren't even selling computers, just plans for computers. Can you imagine how lame this company seemed?

Ditto for Google. Larry and Sergey weren't trying to start a company at first. They were just trying to make search better. Before Google, most search engines didn't try to sort the results they gave you in order of importance. If you searched for "rugby" they just gave you every web page that contained the word "rugby." And the web was so small in 1997 that this actually worked! Kind of. There might only be 20 or 30 pages with the word "rugby," but the web was growing exponentially, which meant this way of doing search was becoming exponentially more broken. Most users just thought, "Wow, I sure have to look through a lot of search results to find what I want." Door sticks. But like Mark, Larry and Sergey were programmers. Like Mark, they looked at this situation and thought "Well, this is stupid. Some pages about rugby matter more than others. Let's figure out which those are and show them first."

It's obvious in retrospect that this was a great idea for a startup. It wasn't obvious at the time. It's never obvious. If it was obviously a good idea to start Apple or Google or Facebook, someone else would have already done it. That's why the best startups grow out of projects that aren't meant to be startups. You're not trying to start a company. You're just following your instincts about what's interesting. And if you're young and good at technology, then your unconscious instincts about what's interesting are better than your conscious ideas about what would be a good company.

So it's critical, if you're a young founder, to build things for yourself and your friends to use. The biggest mistake young founders make is to build something for some mysterious group of other people. But if you can make something that you and your friends truly want to use — something your friends aren't just using out of loyalty to you, but would be really sad to lose if you shut it down — then you almost certainly have the germ of a good startup idea. It may not seem like a startup to you. It may not be obvious how to make money from it. But trust me, there's a way.

What you need in a startup idea, and all you need, is something your friends actually want. And those ideas aren't hard to see once you're good at technology. There are sticking doors everywhere. [2]

Now for the third and final thing you need: a cofounder, or cofounders. The optimal startup has two or three founders, so you need one or two cofounders. How do you find them? Can you predict what I'm going to say next? It's the same thing: projects. You find cofounders by working on projects with them. What you need in a cofounder is someone who's good at what they do and

that you work well with, and the only way to judge this is to work with them on things.

At this point I'm going to tell you something you might not want to hear. It really matters to do well in your classes, even the ones that are just memorization or blathering about literature, because you need to do well in your classes to get into a good university. And if you want to start a startup you should try to get into the best university you can, because that's where the best cofounders are. It's also where the best employees are. When Larry and Sergey started Google, they began by just hiring all the smartest people they knew out of Stanford, and this was a real advantage for them.

The empirical evidence is clear on this. If you look at where the largest numbers of successful startups come from, it's pretty much the same as the list of the most selective universities.

I don't think it's the prestigious names of these universities that cause more good startups to come out of them. Nor do I think it's because the quality of the teaching is better. What's driving this is simply the difficulty of getting in. You have to be pretty smart and determined to get into MIT or Cambridge, so if you do manage to get in, you'll find the other students include a lot of smart and determined people. [3]

You don't have to start a startup with someone you meet at university. The founders of Twitch met when they were seven. The founders of Stripe, Patrick and John Collison, met when John was born. But universities are the main source of cofounders. And because they're where the cofounders are, they're also where the ideas are, because the best ideas grow out of projects you do with the people who become your cofounders.

So the list of what you need to do to get from here to starting a startup is quite short. You need to get good at technology, and the way to do that is to work on your own projects. And you need to do as well in school as you can, so you can get into a good university, because that's where the cofounders and the ideas are.

That's it, just two things, build stuff and do well in school.

Notes

[1] The rhetorical trick in this sentence is that the "Google"s refer to different things. What I mean is: a company that has as much chance of growing as big as Google ultimately did as Larry and Sergey could have reasonably expected Google itself would at the time they started it. But I think the original version is zippier.

[2] Making something for your friends isn't the only source of startup ideas. It's just the best source for young founders, who

have the least knowledge of what other people want, and whose own wants are most predictive of future demand.

[3] Strangely enough this is particularly true in countries like the US where undergraduate admissions are done badly. US admissions departments make applicants jump through a lot of arbitrary hoops that have little to do with their intellectual ability. But the more arbitrary a test, the more it becomes a test of mere determination and resourcefulness. And those are the two most important qualities in startup founders. So US admissions departments are better at selecting founders than they would be if they were better at selecting students.

Thanks to Jared Friedman, Carolynn Levy, Jessica Livingston, Harj Taggar, and Garry Tan for reading drafts of this.
