

# STARTUP IDEA DEVELOPMENT



PROPERTY OF SHOCKWAVE INNOVATIONS.  
DO NOT DISTRIBUTE WITHOUT PERMISSION.

# STARTUP IDEA DEVELOPMENT

---

## INTRODUCTION

This assignment is designed to help you further develop your initial startup idea to the point of being ready to assemble a full business plan. Even if you already have a starting business plan and are actively developing a minimum viable product (MVP), this assignment might help you realize some important things that were missed.

---

## PREREQUISITE

As an insightful piece of background information, read the excerpt from an article that is copied at the end of this document. It is from the founder of Y Combinator, a world-renowned startup accelerator located in Silicon Valley. The intention of the article excerpt is to give you an idea of what entrepreneurs are often advised regarding initial idea evaluation and formation, before they actually start building a company around it.

Watch the Founders Academy video module titled “Developing Your Idea”. You won’t apply all of the concepts described in the video, but some of them are specifically used for this assignment and the remainder are good “food for thought” and especially recommended for those that are actually working on a real startup project.

---

## ASSIGNMENT

1. What problem have you discovered that needs solving?
  - Even if the world doesn’t know it needs the product you are building, you should try to come up with a problem statement for it.
  - Consider trying to complete one or more of the following sentences to help visualize the problem in the eyes of the customers you will sell to:
    - “Don’t you hate it when \_\_\_\_\_”
    - “Have you ever noticed that \_\_\_\_\_”
    - “Wouldn’t it be great if \_\_\_\_\_”
    - “Did you know that (constituent audience) (problem they encounter)”
  - The problem doesn’t need to be at the severity level of cancer but should be enough of a problem that you could imagine customers paying some amount of money to solve the problem. “Desperately need” is much better than “want”.
  - It is helpful if you have personally experienced this problem or have someone close to you that has. That’s not just because of your personal knowledge on the subject, but also the likely additional personal passion you’ll have for the solution being built.

2. What solution will you create that solves the identified problem?
  - In what form is the product? For example, software, hardware, service offering, online marketplace, etc.
  - What benefit does it provide?
  - How does it work (high-level)?
3. What key assumptions are you making that need to be true in order for your business to eventually become viable?
  - You don't need to worry as much about assumptions related to scalability and long-term sustainability. But list out a dozen or more key assumptions associated with viability and try to make sure they cover at least the problem, the market, and the customer/user.
4. Describe what your first Minimum Viable Product might be?
  - Remember that trying to solve for both "minimum" and "viable" typically means identifying the least quantity of features sufficient to satisfy early adopters (not mainstream buyers). Your MVP will be the first tangible thing your early adopter prospects can experience and then give valuable feedback.
5. Conduct a brief "road test" for #1, 2 and 3 above.
  - First talk to family members or friends to see if they agree the problem you've identified is worth trying to solve and that the solution you've identified has a chance of working and being compelling enough for someone to pay for.
  - Next do some initial customer discovery with people that actually experience the problem you've identified. Use this collection of feedback to gain some initial validation of your assumptions and to refine your initial idea, if necessary.
  - For customer discovery, refer back to the video module titled "Developing Your Idea" for additional guidance and tools to use – especially the topics in the 7 – 12 min range of the video. If you don't have access to that video, see the Appendix at the very end of this document (after the article excerpt).

*Note: Don't worry about excessive levels of precision with this exercise. Throughout the first few modules of Founders Academy, you will have plenty of additional opportunities to further validate and/or refine your idea. You will also want to continue the Customer Discovery work throughout the process of assembling your final business plan.*

*See Article Excerpt on Next Page*

---

## ARTICLE EXCERPT

### Excerpts from "How to Get Startup Ideas"

By Paul Graham, November 2012 (<http://paulgraham.com/startupideas.html>)

The way to get startup ideas is not to try to think of startup ideas. It's to look for problems, preferably problems you have yourself.

The very best startup ideas tend to have three things in common: they're something the founders themselves want, that they themselves can build, and that few others realize are worth doing. Microsoft, Apple, Yahoo, Google, and Facebook all began this way.

#### Problems

Why is it so important to work on a problem you have? Among other things, it ensures the problem really exists. It sounds obvious to say you should only work on problems that exist. And yet by far the most common mistake startups make is to solve problems no one has.

I made it myself. In 1995 I started a company to put art galleries online. But galleries didn't want to be online. It's not how the art business works. So why did I spend 6 months working on this stupid idea? Because I didn't pay attention to users. I invented a model of the world that didn't correspond to reality, and worked from that. I didn't notice my model was wrong until I tried to convince users to pay for what we'd built. Even then I took embarrassingly long to catch on. I was attached to my model of the world, and I'd spent a lot of time on the software. They had to want it!

Why do so many founders build things no one wants? Because they begin by trying to think of startup ideas. That m.o. is doubly dangerous: it doesn't merely yield few good ideas; it yields bad ideas that sound plausible enough to fool you into working on them.

At YC we call these "made-up" or "sitcom" startup ideas. Imagine one of the characters on a TV show was starting a startup. The writers would have to invent something for it to do. But coming up with good startup ideas is hard. It's not something you can do for the asking. So (unless they got amazingly lucky) the writers would come up with an idea that sounded plausible, but was actually bad.

For example, a social network for pet owners. It doesn't sound obviously mistaken. Millions of people have pets. Often they care a lot about their pets and spend a lot of money on them. Surely many of these people would like a site where they could talk to other pet owners. Not all of them perhaps, but if just 2 or 3 percent were regular visitors, you could have millions of users. You could serve them targeted offers, and maybe charge for premium features. [1]

The danger of an idea like this is that when you run it by your friends with pets, they don't say "I would never use this." They say "Yeah, maybe I could see using something like that." Even when the startup launches, it will sound plausible to a lot of people. They don't want to use it themselves, at least not right now, but they could imagine other people wanting it. Sum that reaction across the entire population, and you have zero users. [2]

#### Well

When a startup launches, there have to be at least some users who really need what they're making—not just people who could see themselves using it one day, but who want it urgently. Usually this initial

group of users is small, for the simple reason that if there were something that large numbers of people urgently needed and that could be built with the amount of effort a startup usually puts into a version one, it would probably already exist. Which means you have to compromise on one dimension: you can either build something a large number of people want a small amount, or something a small number of people want a large amount. Choose the latter. Not all ideas of that type are good startup ideas, but nearly all good startup ideas are of that type.

Imagine a graph whose x axis represents all the people who might want what you're making and whose y axis represents how much they want it. If you invert the scale on the y axis, you can envision companies as holes. Google is an immense crater: hundreds of millions of people use it, and they need it a lot. A startup just starting out can't expect to excavate that much volume. So you have two choices about the shape of hole you start with. You can either dig a hole that's broad but shallow, or one that's narrow and deep, like a well.

Made-up startup ideas are usually of the first type. Lots of people are mildly interested in a social network for pet owners.

Nearly all good startup ideas are of the second type. Microsoft was a well when they made Altair Basic. There were only a couple thousand Altair owners, but without this software they were programming in machine language. Thirty years later Facebook had the same shape. Their first site was exclusively for Harvard students, of which there are only a few thousand, but those few thousand users wanted it a lot.

When you have an idea for a startup, ask yourself: who wants this right now? Who wants this so much that they'll use it even when it's a crappy version one made by a two-person startup they've never heard of? If you can't answer that, the idea is probably bad. [3]

You don't need the narrowness of the well per se. It's depth you need; you get narrowness as a byproduct of optimizing for depth (and speed). But you almost always do get it. In practice the link between depth and narrowness is so strong that it's a good sign when you know that an idea will appeal strongly to a specific group or type of user.

But while demand shaped like a well is almost a necessary condition for a good startup idea, it's not a sufficient one. If Mark Zuckerberg had built something that could only ever have appealed to Harvard students, it would not have been a good startup idea. Facebook was a good idea because it started with a small market there was a fast path out of. Colleges are similar enough that if you build a Facebook that works at Harvard, it will work at any college. So you spread rapidly through all the colleges. Once you have all the college students, you get everyone else simply by letting them in.

Similarly for Microsoft: Basic (programming language) for the Altair; Basic for other machines; other languages besides Basic; operating systems; applications; IPO.

## **Self**

How do you tell whether there's a path out of an idea? How do you tell whether something is the germ of a giant company, or just a niche product? Often you can't. The founders of Airbnb didn't realize at first how big a market they were tapping. Initially they had a much narrower idea. They were going to let hosts rent out space on their floors during conventions. They didn't foresee the expansion of this idea; it forced itself upon them gradually. All they knew at first is that they were onto something. That's probably as much as Bill Gates or Mark Zuckerberg knew at first.

Occasionally it's obvious from the beginning when there's a path out of the initial niche. And sometimes I can see a path that's not immediately obvious; that's one of our specialties at YC. But there are limits to how well this can be done, no matter how much experience you have. The most important thing to understand about paths out of the initial idea is the meta-fact that these are hard to see.

So if you can't predict whether there's a path out of an idea, how do you choose between ideas? The truth is disappointing but interesting: if you're the right sort of person, you have the right sort of hunches. If you're at the leading edge of a field that's changing fast, when you have a hunch that something is worth doing, you're more likely to be right.

In *Zen and the Art of Motorcycle Maintenance*, Robert Pirsig says: "You want to know how to paint a perfect painting? It's easy. Make yourself perfect and then just paint naturally."

I've wondered about that passage since I read it in high school. I'm not sure how useful his advice is for painting specifically, but it fits this situation well. Empirically, the way to have good startup ideas is to become the sort of person who has them.

Being at the leading edge of a field doesn't mean you have to be one of the people pushing it forward. You can also be at the leading edge as a user. It was not so much because he was a programmer that Facebook seemed a good idea to Mark Zuckerberg as because he used computers so much. If you'd asked most 40 year olds in 2004 whether they'd like to publish their lives semi-publicly on the Internet, they'd have been horrified at the idea. But Mark already lived online; to him it seemed natural.

Paul Buchheit says that people at the leading edge of a rapidly changing field "live in the future." Combine that with Pirsig and you get: Live in the future, then build what's missing.

That describes the way many if not most of the biggest startups got started. Neither Apple nor Yahoo nor Google nor Facebook were even supposed to be companies at first. They grew out of things their founders built because there seemed a gap in the world.

If you look at the way successful founders have had their ideas, it's generally the result of some external stimulus hitting a prepared mind. Bill Gates and Paul Allen hear about the Altair and think "I bet we could write a Basic interpreter for it." Drew Houston realizes he's forgotten his USB stick and thinks "I really need to make my files live online." Lots of people heard about the Altair. Lots forgot USB sticks. The reason those stimuli caused those founders to start companies was that their experiences had prepared them to notice the opportunities they represented.

The verb you want to be using with respect to startup ideas is not "think up" but "notice." At YC we call ideas that grow naturally out of the founders' own experiences "organic" startup ideas. The most successful startups almost all begin this way.

That may not have been what you wanted to hear. You may have expected recipes for coming up with startup ideas, and instead I'm telling you that the key is to have a mind that's prepared in the right way. But disappointing though it may be, this is the truth. And it is a recipe of a sort, just one that in the worst case takes a year rather than a weekend.

If you're not at the leading edge of some rapidly changing field, you can get to one. For example, anyone reasonably smart can probably get to an edge of programming (e.g. building mobile apps) in a year. Since a successful startup will consume at least 3-5 years of your life, a year's preparation would be a reasonable investment. Especially if you're also looking for a cofounder. [4]

You don't have to learn programming to be at the leading edge of a domain that's changing fast. Other domains change fast. But while learning to hack is not necessary, it is for the foreseeable future sufficient. As Marc Andreessen put it, software is eating the world, and this trend has decades left to run.

Knowing how to hack also means that when you have ideas, you'll be able to implement them. That's not absolutely necessary (Jeff Bezos couldn't) but it's an advantage. It's a big advantage, when you're considering an idea like putting a college facebook online, if instead of merely thinking "That's an interesting idea," you can think instead "That's an interesting idea. I'll try building an initial version tonight." It's even better when you're both a programmer and the target user, because then the cycle of generating new versions and testing them on users can happen inside one head.

## Noticing

Once you're living in the future in some respect, the way to notice startup ideas is to look for things that seem to be missing. If you're really at the leading edge of a rapidly changing field, there will be things that are obviously missing. What won't be obvious is that they're startup ideas. So if you want to find startup ideas, don't merely turn on the filter "What's missing?" Also turn off every other filter, particularly "Could this be a big company?" There's plenty of time to apply that test later. But if you're thinking about that initially, it may not only filter out lots of good ideas, but also cause you to focus on bad ones.

Most things that are missing will take some time to see. You almost have to trick yourself into seeing the ideas around you.

But you know the ideas are out there. This is not one of those problems where there might not be an answer. It's impossibly unlikely that this is the exact moment when technological progress stops. You can be sure people are going to build things in the next few years that will make you think "What did I do before x?"

And when these problems get solved, they will probably seem flamingly obvious in retrospect. What you need to do is turn off the filters that usually prevent you from seeing them. The most powerful is simply taking the current state of the world for granted. Even the most radically open-minded of us mostly do that. You couldn't get from your bed to the front door if you stopped to question everything.

But if you're looking for startup ideas you can sacrifice some of the efficiency of taking the status quo for granted and start to question things. Why is your inbox overflowing? Because you get a lot of email, or because it's hard to get email out of your inbox? Why do you get so much email? What problems are people trying to solve by sending you email? Are there better ways to solve them? And why is it hard to get emails out of your inbox? Why do you keep emails around after you've read them? Is an inbox the optimal tool for that?

Pay particular attention to things that chafe you. The advantage of taking the status quo for granted is not just that it makes life (locally) more efficient, but also that it makes life more tolerable. If you knew about all the things we'll get in the next 50 years but don't have yet, you'd find present day life pretty constraining, just as someone from the present would if they were sent back 50 years in a time machine. When something annoys you, it could be because you're living in the future.

When you find the right sort of problem, you should probably be able to describe it as obvious, at least to you. When we started Viaweb, all the online stores were built by hand, by web designers making individual HTML pages. It was obvious to us as programmers that these sites would have to be generated by software. [5]

Which means, strangely enough, that coming up with startup ideas is a question of seeing the obvious. That suggests how weird this process is: you're trying to see things that are obvious, and yet that you hadn't seen.

Since what you need to do here is loosen up your own mind, it may be best not to make too much of a direct frontal attack on the problem—i.e. to sit down and try to think of ideas. The best plan may be just to keep a background process running, looking for things that seem to be missing. Work on hard problems, driven mainly by curiosity, but have a second self watching over your shoulder, taking note of gaps and anomalies. [6]

Give yourself some time. You have a lot of control over the rate at which you turn yours into a prepared mind, but you have less control over the stimuli that spark ideas when they hit it. If Bill Gates and Paul Allen had constrained themselves to come up with a startup idea in one month, what if they'd chosen a month before the Altair appeared? They probably would have worked on a less promising idea. Drew Houston did work on a less promising idea before Dropbox: an SAT prep startup. But Dropbox was a much better idea, both in the absolute sense and also as a match for his skills. [7]

A good way to trick yourself into noticing ideas is to work on projects that seem like they'd be cool. If you do that, you'll naturally tend to build things that are missing. It wouldn't seem as interesting to build something that already existed.

Just as trying to think up startup ideas tends to produce bad ones, working on things that could be dismissed as "toys" often produces good ones. When something is described as a toy, that means it has everything an idea needs except being important. It's cool; users love it; it just doesn't matter. But if you're living in the future and you build something cool that users love, it may matter more than outsiders think. Microcomputers seemed like toys when Apple and Microsoft started working on them. I'm old enough to remember that era; the usual term for people with their own microcomputers was "hobbyists." BackRub seemed like an inconsequential science project. The Facebook was just a way for undergrads to stalk one another.

At YC we're excited when we meet startups working on things that we could imagine know-it-alls on forums dismissing as toys. To us that's positive evidence an idea is good.

If you can afford to take a long view (and arguably you can't afford not to), you can turn "Live in the future and build what's missing" into something even better:

Live in the future and build what seems interesting.

## **Competition**

Because a good idea should seem obvious, when you have one you'll tend to feel that you're late. Don't let that deter you. Worrying that you're late is one of the signs of a good idea. Ten minutes of searching the web will usually settle the question. Even if you find someone else working on the same thing, you're probably not too late. It's exceptionally rare for startups to be killed by competitors—so rare that you can almost discount the possibility. So unless you discover a competitor with the sort of lock-in that would prevent users from choosing you, don't discard the idea.

If you're uncertain, ask users. The question of whether you're too late is subsumed by the question of whether anyone urgently needs what you plan to make. If you have something that no competitor does and that some subset of users urgently need, you have a beachhead. [11]



The question then is whether that beachhead is big enough. Or more importantly, who's in it: if the beachhead consists of people doing something lots more people will be doing in the future, then it's probably big enough no matter how small it is. For example, if you're building something differentiated from competitors by the fact that it works on phones, but it only works on the newest phones, that's probably a big enough beachhead.

Err on the side of doing things where you'll face competitors. Inexperienced founders usually give competitors more credit than they deserve. Whether you succeed depends far more on you than on your competitors. So better a good idea with competitors than a bad one without.

You don't need to worry about entering a "crowded market" so long as you have a thesis about what everyone else in it is overlooking. In fact that's a very promising starting point. Google was that type of idea. Your thesis has to be more precise than "we're going to make an x that doesn't suck" though. You have to be able to phrase it in terms of something the incumbents are overlooking. Best of all is when you can say that they didn't have the courage of their convictions, and that your plan is what they'd have done if they'd followed through on their own insights. Google was that type of idea too. The search engines that preceded them shied away from the most radical implications of what they were doing—particularly that the better a job they did, the faster users would leave.

A crowded market is actually a good sign, because it means both that there's demand and that none of the existing solutions are good enough. A startup can't hope to enter a market that's obviously big and yet in which they have no competitors. So any startup that succeeds is either going to be entering a market with existing competitors, but armed with some secret weapon that will get them all the users (like Google), or entering a market that looks small but which will turn out to be big (like Microsoft). [12]

## Recipes

While the best way to discover startup ideas is to become the sort of person who has them and then build whatever interests you, sometimes you don't have that luxury. Sometimes you need an idea now. For example, if you're working on a startup and your initial idea turns out to be bad.

For the rest of this essay I'll talk about tricks for coming up with startup ideas on demand. Although empirically you're better off using the organic strategy, you could succeed this way. You just have to be more disciplined. When you use the organic method, you don't even notice an idea unless it's evidence that something is truly missing. But when you make a conscious effort to think of startup ideas, you have to replace this natural constraint with self-discipline. You'll see a lot more ideas, most of them bad, so you need to be able to filter them.

One of the biggest dangers of not using the organic method is the example of the organic method. Organic ideas feel like inspirations. There are a lot of stories about successful startups that began when the founders had what seemed a crazy idea but "just knew" it was promising. When you feel that about an idea you've had while trying to come up with startup ideas, you're probably mistaken.

When searching for ideas, look in areas where you have some expertise. If you're a database expert, don't build a chat app for teenagers (unless you're also a teenager). Maybe it's a good idea, but you can't trust your judgment about that, so ignore it. There have to be other ideas that involve databases, and whose quality you can judge. Do you find it hard to come up with good ideas involving databases? That's because your expertise raises your standards. Your ideas about chat apps are just as bad, but you're giving yourself a Dunning-Kruger pass in that domain.

The place to start looking for ideas is things you need. There must be things you need. [14]

One good trick is to ask yourself whether in your previous job you ever found yourself saying "Why doesn't someone make x? If someone made x we'd buy it in a second." If you can think of any x people said that about, you probably have an idea. You know there's demand, and people don't say that about things that are impossible to build.

More generally, try asking yourself whether there's something unusual about you that makes your needs different from most other people's. You're probably not the only one. It's especially good if you're different in a way people will increasingly be.

If you're changing ideas, one unusual thing about you is the idea you'd previously been working on. Did you discover any needs while working on it? Several well-known startups began this way. Hotmail began as something its founders wrote to talk about their previous startup idea while they were working at their day jobs. [15]

A particularly promising way to be unusual is to be young. Some of the most valuable new ideas take root first among people in their teens and early twenties. And while young founders are at a disadvantage in some respects, they're the only ones who really understand their peers. It would have been very hard for someone who wasn't a college student to start Facebook. So if you're a young founder (under 23 say), are there things you and your friends would like to do that current technology won't let you?

The next best thing to an unmet need of your own is an unmet need of someone else. Try talking to everyone you can about the gaps they find in the world. What's missing? What would they like to do that they can't? What's tedious or annoying, particularly in their work? Let the conversation get general; don't be trying too hard to find startup ideas. You're just looking for something to spark a thought. Maybe you'll notice a problem they didn't consciously realize they had, because you know how to solve it.

When you find an unmet need that isn't your own, it may be somewhat blurry at first. The person who needs something may not know exactly what they need. In that case I often recommend that founders act like consultants—that they do what they'd do if they'd been retained to solve the problems of this one user. People's problems are similar enough that nearly all the code you write this way will be reusable, and whatever isn't will be a small price to start out certain that you've reached the bottom of the well. [16]

One way to ensure you do a good job solving other people's problems is to make them your own. When Rajat Suri of E la Carte decided to write software for restaurants, he got a job as a waiter to learn how restaurants worked. That may seem like taking things to extremes, but startups are extreme. We love it when founders do such things.

In fact, one strategy I recommend to people who need a new idea is not merely to turn off their schlep and unsexy filters, but to seek out ideas that are unsexy or involve schleps. Don't try to start Twitter. Those ideas are so rare that you can't find them by looking for them. Make something unsexy that people will pay you for.

A good trick for bypassing the schlep and to some extent the unsexy filter is to ask what you wish someone else would build, so that you could use it. What would you pay for right now?

Since startups often garbage-collect broken companies and industries, it can be a good trick to look for those that are dying, or deserve to, and try to imagine what kind of company would profit from their demise. For example, journalism is in free fall at the moment. But there may still be money to be made

from something like journalism. What sort of company might cause people in the future to say "this replaced journalism" on some axis?

But imagine asking that in the future, not now. When one company or industry replaces another, it usually comes in from the side. So don't look for a replacement for x; look for something that people will later say turned out to be a replacement for x. And be imaginative about the axis along which the replacement occurs. Traditional journalism, for example, is a way for readers to get information and to kill time, a way for writers to make money and to get attention, and a vehicle for several different types of advertising. It could be replaced on any of these axes (it has already started to be on most).

When startups consume incumbents, they usually start by serving some small but important market that the big players ignore. It's particularly good if there's an admixture of disdain in the big players' attitude, because that often misleads them. For example, after Steve Wozniak built the computer that became the Apple I, he felt obliged to give his then-employer Hewlett-Packard the option to produce it. Fortunately for him, they turned it down, and one of the reasons they did was that it used a TV for a monitor, which seemed intolerably déclassé to a high-end hardware company like HP was at the time. [17]

Are there groups of scruffy but sophisticated users like the early microcomputer "hobbyists" that are currently being ignored by the big players? A startup with its sights set on bigger things can often capture a small market easily by expending an effort that wouldn't be justified by that market alone.

Similarly, since the most successful startups generally ride some wave bigger than themselves, it could be a good trick to look for waves and ask how one could benefit from them. The prices of gene sequencing and 3D printing are both experiencing Moore's Law-like declines. What new things will we be able to do in the new world we'll have in a few years? What are we unconsciously ruling out as impossible that will soon be possible?

## Notes

[1] This form of bad idea has been around as long as the web. It was common in the 1990s, except then people who had it used to say they were going to create a portal for x instead of a social network for x. Structurally the idea is stone soup: you post a sign saying "this is the place for people interested in x," and all those people show up and you make money from them. What lures founders into this sort of idea are statistics about the millions of people who might be interested in each type of x. What they forget is that any given person might have 20 affinities by this standard, and no one is going to visit 20 different communities regularly.

[2] I'm not saying, incidentally, that I know for sure a social network for pet owners is a bad idea. I know it's a bad idea the way I know randomly generated DNA would not produce a viable organism. The set of plausible sounding startup ideas is many times larger than the set of good ones, and many of the good ones don't even sound that plausible. So if all you know about a startup idea is that it sounds plausible, you have to assume it's bad.

[3] More precisely, the users' need has to give them sufficient activation energy to start using whatever you make, which can vary a lot. For example, the activation energy for enterprise software sold through traditional channels is very high, so you'd have to be a lot better to get users to switch. Whereas the activation energy required to switch to a new search engine is low. Which in turn is why search engines are so much better than enterprise software.

[4] This gets harder as you get older. While the space of ideas doesn't have dangerous local maxima, the space of careers does. There are fairly high walls between most of the paths people take through life, and the older you get, the higher the walls become.

[5] It was also obvious to us that the web was going to be a big deal. Few non-programmers grasped that in 1995, but the programmers had seen what GUIs had done for desktop computers.

[6] Maybe it would work to have this second self keep a journal, and each night to make a brief entry listing the gaps and anomalies you'd noticed that day. Not startup ideas, just the raw gaps and anomalies.

[7] Sam Altman points out that taking time to come up with an idea is not merely a better strategy in an absolute sense, but also like an undervalued stock in that so few founders do it.

There's comparatively little competition for the best ideas, because few founders are willing to put in the time required to notice them. Whereas there is a great deal of competition for mediocre ideas, because when people make up startup ideas, they tend to make up the same ones.

[8] For the computer hardware and software companies, summer jobs are the first phase of the recruiting funnel. But if you're good you can skip the first phase. If you're good you'll have no trouble getting hired by these companies when you graduate, regardless of how you spent your summers.

[9] The empirical evidence suggests that if colleges want to help their students start startups, the best thing they can do is leave them alone in the right way.

[10] I'm speaking here of IT startups; in biotech things are different.

[11] This is an instance of a more general rule: focus on users, not competitors. The most important information about competitors is what you learn via users anyway.

[12] In practice most successful startups have elements of both. And you can describe each strategy in terms of the other by adjusting the boundaries of what you call the market. But it's useful to consider these two ideas separately.

[13] I almost hesitate to raise that point though. Startups are businesses; the point of a business is to make money; and with that additional constraint, you can't expect you'll be able to spend all your time working on what interests you most.

[14] The need has to be a strong one. You can retroactively describe any made-up idea as something you need. But do you really need that recipe site or local event aggregator as much as Drew Houston needed Dropbox, or Brian Chesky and Joe Gebbia needed Airbnb?

Quite often at YC I find myself asking founders "Would you use this thing yourself, if you hadn't written it?" and you'd be surprised how often the answer is no.

[15] Paul Buchheit points out that trying to sell something bad can be a source of better ideas:

"The best technique I've found for dealing with YC companies that have bad ideas is to tell them to go sell the product ASAP (before wasting time building it). Not only do they learn that nobody wants what they are building, they very often come back with a real idea that they discovered in the process of trying to sell the bad idea."

[16] Here's a recipe that might produce the next Facebook, if you're college students. If you have a connection to one of the more powerful sororities at your school, approach the queen bees thereof and offer to be their personal IT consultants, building anything they could imagine needing in their social lives that didn't already exist. Anything that got built this way would be very promising, because such users are not just the most demanding but also the perfect point to spread from.

I have no idea whether this would work.

[17] And the reason it used a TV for a monitor is that Steve Wozniak started out by solving his own problems. He, like most of his peers, couldn't afford a monitor.

*Thanks to Sam Altman, Mike Arrington, Paul Buchheit, John Collison, Patrick Collison, Garry Tan, and Harj Taggar for reading drafts of this, and Marc Andreessen, Joe Gebbia, Reid Hoffman, Shel Kaphan, Mike Moritz and Kevin Systrom for answering my questions about startup history.*

## APPENDIX

### Start with a List of Assumptions

**Much of your idea development work will involve validating or refining assumptions**

We believe that (audience) (experiences pain)

We believe they want to eliminate this pain (for benefit)

We believe they would choose us over (alternative) primarily due to (reasons)

We believe that (audience) would spend (amount) to eliminate the pain

...

### How many potential users do you need to talk to?

As many as needed until you start getting  
similar responses from many/most  
*(that possibly means hundreds)*

## Questions should be “open-ended”

(They don't allow simple “yes” or “no” answers)

**X** Do you agree that \_\_\_\_\_?

(vs)

✓ How do you \_\_\_\_\_?

✓ Tell me about a time when \_\_\_\_\_

## There's a Lot You Can Figure Out

- Are you targeting the right customer? (demographic, persona)
- Do they see the problem the same way you do?
- What other problems do they experience in the same general area?
- How do they avoid or deal with the problem currently?
- What solutions have they considered in the past and why didn't they implement any of them?
- If you build it, would they at least consider buying it?