MY 11 STARTUP RULES

The Winning Startup Perspective, distilled into eleven rules

Hard-earned truths that save you time, money, and unnecessary pain.

TOM COLLOPY

UNBIASED INSIGHTS

MY 11 STARTUP RULES

The Winning Startup Perspective, distilled into eleven rules

By Tom Collopy

Unbiased Insights

Hard-earned truths that save you time, money, and unnecessary pain.

Eleven rules experienced founders mostly know and rarely say out loud to a first-timer. Each one comes with the core idea it rests on and a short, true story of the rule in the wild.

Among the rules:

- Customers buy outcomes, not solutions
- A problem is not a wanting
- Warm traction and cold traction are different businesses
- Run the audit before you start over

The pocket companion to *The Winning Startup Perspective: Outcomes, Not Solutions*. Read it in one sitting, or take one rule at a time.

Contents

What this is 4
The one idea under all of them 5
Rule #1: Customers buy outcomes, not solutions 5
Rule #2: The advice wasn't wrong; the question got swapped 7
Rule #3: A problem is not a wanting 8
Rule #4: Behavior is the only answer that counts 9
Rule #5: Most waste is correct building aimed wrong 10
Rule #6: Ask the question, don't check the box 11
Rule #7: You don't need a small product, you need a small test 12
Rule #8: Warm traction and cold traction are different businesses 13
Rule #9: Scaling multiplies what already works without you 14
Rule #10: The discomfort is the point 15
Rule #11: Run the audit before you start over 16
A last word: you were never behind 17

What this is

Eleven rules I wish someone had said out loud to me when I started, the things experienced founders mostly know and rarely explain to a first-timer. Each one is short enough to argue with and, I hope, true enough to keep.

For each rule you'll find three things: the rule itself, an inset block that explains the core idea it rests on, and a short, true story of the rule in the wild.

This is a companion to the longer book, *The Winning Startup Perspective: Outcomes, Not Solutions*. That book makes the full case, two ways to build a startup, why the frameworks confuse founders, and what they were actually built for. This one is the pocket version.

The one idea under all of them

Before the rules, the single idea they all rest on. It takes about a minute, and the rest of the book is just consequences of it.

Picture someone who wants to sit on a Caribbean beach. Between them and it: booking a flight, the security line, a cramped seat, baggage claim, a shuttle to the hotel. Nobody wants any of that. They tolerate the plane because they want the beach.

That's the whole thing. **The outcome is the beach, the state your customer already wants. The product is the plane, what they tolerate to get there.** Nobody has ever wanted a flight. The test for whether you've found a real outcome: would they still want it if your product never existed? People wanted Caribbean beaches long before airlines, and they'll want them after.

From that one distinction come two ways to build a startup, what these rules call the two **spines**.

The **product-first spine** starts with your solution: I see a problem, I build a product, I demo it, I get people to try it, I refine it, I chase leads, I sell. The product is the center, and the customer's wanting is assumed rather than examined.

The **outcome-first spine** starts with your customer's outcome: what do they already want badly enough to act? Who wants it? What's blocking them? What are they already doing instead? Only then, what do I build, and how do I make that outcome clear and reachable?

Same pieces. Different center. Every rule that follows is a consequence of moving from the first spine to the second.

Rule #1: Customers buy outcomes, not solutions

Customers do not buy solutions. They buy outcomes, and tolerate solutions to reach them. Find the beach before you describe the flight.

> Picture someone who wants to sit on a Caribbean beach. Between them and it: booking a flight, the security line, a cramped seat, baggage claim, a shuttle. Nobody wants any of that, they tolerate the plane because they want the beach. Your product is the plane. The outcome is the beach. The test for whether you've found the real outcome: would they still want it if your product never existed?

A founder I know spent the first twenty minutes of every pitch on his product. The architecture. The integrations. The clever thing it did that competitors couldn't.

Prospects nodded politely and didn't buy.

One day a prospect cut him off. "Okay, but what do I actually *get*?"

He stopped. Then, almost by accident, he said: “You stop spending your Sundays reconciling spreadsheets and you get your weekend back.”

She leaned in. “That. Why didn’t you lead with that?”

He’d been selling the plane. Booking, seats, baggage claim. She didn’t want a plane. She wanted to be on the beach by Saturday morning.

Nobody has ever wanted a flight. They tolerate the flight because they want the beach.

Rule #2: The advice wasn't wrong; the question got swapped

Frameworks are built to create judgment. Founders translate them into tasks. The advice wasn't wrong: the question underneath it got swapped.

> Startup frameworks were built to create judgment, to help you find out whether you're wrong before it gets expensive. Founders hear them as tasks to complete instead. "Do customer discovery" was never "confirm your idea." It was "try to kill it." The advice is usually fine. The question underneath it quietly gets swapped for an easier one.

"I finished customer discovery," a founder told me, proud. "Thirty interviews."

"Great," I said. "What did you find out that could kill the idea?"

Silence.

He'd asked thirty people if they had the problem. Thirty people said yes. He'd written it all down and checked the box.

But discovery was never built to confirm your idea. It was built to attack it, to find out, while it's still cheap, whether anyone wants the outcome badly enough to act.

He'd taken a tool meant to build judgment and turned it into a task to complete. So it confirmed what he already believed and taught him nothing.

The advice, "go do discovery", was good. He just heard a different question underneath it than the one it was asking.

Rule #3: A problem is not a wanting

The product-first spine assumes the wanting is already there. A real problem is not the same as a customer who wants the outcome badly enough to act, and a great solution to an unwanted outcome is still an empty plane.

> People hit problems all day and act on almost none of them. You have problems right now you'll never do anything about. A real problem doesn't create demand, wanting an outcome badly enough does. The problem only matters when it's the roadblock between someone and the outcome they already want, their beach.

She had proof. Real, documented proof that her customers had the problem. She'd watched them struggle with it. She'd timed how long it cost them. She'd built something that solved it cleanly.

Nobody bought.

"But the problem is real," she kept saying. And it was. That was never in doubt.

Here's what she'd missed: everyone has that problem, and almost everyone ignores it. It's annoying, not unbearable. It sits on a list of fifty other annoying things people have decided to live with.

A real problem doesn't create demand. Wanting an outcome badly enough creates demand. Her customers had the problem and didn't want the fix badly enough to change anything.

A great solution to an unwanted outcome is still a beautiful plane with no passengers.

Rule #4: Behavior is the only answer that counts

The experienced founder starts at the outcome, not the solution. The question shifts from "do they have my problem?" to "do they already want this badly enough to act?" and that question only accepts behavior as an answer.

> There are two ways to build: start with your solution, or start with your customer's outcome. The experienced founder starts at the outcome and asks one question: do they want this badly enough to act? "I'd use that" is an opinion, and opinions are free. Only behavior, switching, paying, committing, sticking their neck out, counts as an answer.

He had fifty-two "I would definitely use this." He showed me the spreadsheet. Names, dates, quotes. "Definitely." "Absolutely." "Where do I sign up?"

He launched. Zero of the fifty-two signed up.

"They lied," he said.

They didn't lie. They were being kind. "I'd use this" costs nothing to say. It's an opinion, and opinions are free.

The only question that matters is whether anyone did anything. Switched. Paid. Committed. Stuck their neck out.

A founder who collects opinions is collecting applause. A founder who collects behavior is collecting evidence. They feel similar and they are not the same. One of them pays rent.

Rule #5: Most waste is correct building aimed wrong

Start at the outcome; the product still matters, but as the path to the beach, not the reason anyone wants to go. Most founder waste is correct building aimed at the wrong target.

> Starting at the outcome doesn't mean the product stops mattering. The product is how people reach the outcome they want, the plane that carries them to the beach, essential, but never the reason anyone wants to go. And most founder waste isn't sloppy building. It's excellent building aimed at the wrong target. The fastest possible work in the wrong direction is still wasted.

He shipped a gorgeous feature in four days. The team was proud. The code was clean. It worked perfectly.

It moved nothing. No new signups, no new retention, no new revenue.

"But we shipped it so fast," he said, and that was exactly the trap.

The speed wasn't the win. The speed was the problem in disguise. He'd built the wrong thing, beautifully, in record time, and the faster he built wrong things, the faster he burned the runway.

Most founder waste isn't sloppy building. It's excellent building, aimed at the wrong target. The fastest possible work in the wrong direction is still wasted work; it just feels productive while it's happening.

Rule #6: Ask the question, don't check the box

Frameworks aren't steps to complete. Each one was built to answer a specific question, usually about the outcome, the customer, or whether anyone will act. Ask the question, don't check the box.

> A wall of checked boxes, discovery done, MVP done, deck done, and still stuck. That's because a framework isn't a box; it's a question. Discovery asks "am I wrong about what people want?" The MVP asks "what's the cheapest test of my riskiest belief?" You don't *finish* a question. You answer it, or you skip it.

A founder showed me her plan. Every box was checked. Discovery: done. MVP: done. Landing page: done. Pitch deck: done.

A wall of green checkmarks. And she was as stuck as someone who'd done none of it.

"I did everything," she said. She had. That was the strange, painful part.

She'd treated each framework as a box. But a framework isn't a box; it's a question. Discovery asks "am I wrong about what people want?" The MVP asks "what's the cheapest test of my riskiest belief?" You don't finish a question. You answer it, or you don't.

She'd performed the rituals and skipped the questions. So she had a wall of green and no judgment.

She re-titled her plan. Not "things to do." "Questions I haven't answered yet." Nothing else changed, and everything did.

Rule #7: You don't need a small product, you need a small test

Discovery, Lean, and the MVP were built to help you learn whether you're wrong, cheaply. Run as product tasks, they make you efficient at building the wrong thing.

> "Minimum viable product" was never "the smallest real product." The V is the viability of your riskiest belief, not the usability of a build. Discovery, Lean, and the MVP all exist to help you learn whether you're wrong, cheaply. Run as product tasks, they just make you efficient at building the wrong thing.

"I ship every week," he told me, like it was the answer to everything.

And he did. Tight build loop. Clean releases. He'd A/B tested his onboarding four times. By every measure of engineering discipline, he was a machine.

He'd also never once tested whether anyone wanted the outcome badly enough to switch off the tool they already had.

So he'd built the wrong thing twelve times, very efficiently.

"Lean" was supposed to help him avoid wasted building. He heard "build fast" and deleted the word "wasted." Speed became the goal instead of learning, and speed without aim just gets you to the wrong place sooner.

A faster plane to a destination nobody wanted is still empty. It just lands sooner.

Rule #8: Warm traction and cold traction are different businesses

Fit and traction are evidence labels: what matters is what they're labeling. Warm traction proves people will buy from you; cold traction proves the market will buy from your content. Only one repeats, and more leads never fix weak commitment.

> Sort your customers into two columns: people who knew you, and strangers. Warm buyers trust you and forgive the product's gaps. Strangers buy only from your content, with zero trust to start. Both are real money, but only cold traction repeats, because you run out of friends fast. And more leads never fix weak commitment; they just pile up more "loved it" and silence.

Twenty customers. She put it in every investor update. "Strong early traction."

Then I asked her to make two columns: people who knew her, and strangers.

Nineteen and one.

Nineteen had bought because they trusted her, friends, classmates, friends of friends. They'd have given almost any version of the product the benefit of the doubt. The one stranger had bought from the words on her page alone, with no reason to be kind.

That one was worth more than the nineteen, as evidence. The nineteen proved people would buy from *her*. The one proved the market might buy from her *content*, and only that second thing repeats, because you run out of friends fast.

Her instinct was to get more leads. But more leads onto a page that doesn't convert strangers just makes a bigger pile of "loved it!" and silence.

Rule #9: Scaling multiplies what already works without you

The value prop, business model, go-to-market, selling, and scaling are all questions about the outcome, the customer, and what repeats. Selling is learning why people commit; scaling is multiplying only what already works without you.

> Selling isn't a pushy thing you bolt on once the product's done; it's a live experiment in why people commit and why they don't. And scaling isn't how you *find* what works; it's how you multiply what already works without you in the room. Multiply a motion that only works because you know the customers, and you don't grow the business. You grow the leak.

He raised money and hired two salespeople to "scale what was working."

Three months later, the new reps had closed almost nothing. The founder was baffled. "It worked when I did it."

Of course it did. It worked because *he* knew the customers. Every deal he'd closed traced back to a relationship, a warm intro, a reputation he'd spent years building. That wasn't a sales motion. It was him.

He'd tried to multiply something that only worked because he was personally holding it up. He didn't scale the business. He scaled the leak. Now three people were burning money on a motion that was never repeatable.

Scaling isn't how you find what works. It's how you multiply what already works without you in the room. And selling, before that, is how you learn why people commit, not a thing you bolt on once the product is done.

Rule #10: The discomfort is the point

The outcome-first questions are uncomfortable because they can reveal your story is weaker than you hoped. That discomfort is the point. The goal isn't to stop building. It's to stop building around a story customers don't share.

> The product-first path feels great. Every step gives you something to build and show. The outcome-first path feels worse, because it asks questions you might not want answered: what if fewer people want this than I hoped? What if they won't switch? That discomfort isn't a sign you're doing it wrong. It's the sign you're finally asking the question that can save you.

For four months she didn't ask her customers the one question that scared her: *would you actually pay for this, or do you just like me?*

She knew, somewhere, that the answer might hurt. So she stayed busy instead. New features. A redesign. A blog. Anything but the question.

When she finally asked it, out loud, to five real customers, three of them hesitated in a way that told her everything. The outcome she'd built around wasn't one they wanted badly enough to pay for.

It stung. It also saved her a year.

Because the alternative wasn't avoiding the bad news. The alternative was finding it out in twelve months, with the runway gone, instead of in one afternoon.

The hard questions feel like a threat because they can reveal your story is weaker than you hoped. That's not a reason to avoid them. That's the whole reason to ask.

Rule #11: Run the audit before you start over

Don't start over and don't add a system. Run one outcome audit on the idea, content, and customers you already have. It tells you which spine you've been on and where the real work is, and it comes before any tool.

> You almost never need to start over, and you don't need another system. Run one honest audit of the idea, content, and customers you already have, start by reading your own page as a stranger with zero trust. It tells you which of the two paths you've been building from, solution-first or outcome-first, and where the real work is. And it comes before any tool.

He was ready to rebuild the whole product from scratch. New stack, new design, new everything. He was sure the product was the problem.

I asked him to do one thing first. Read his own homepage as if he were a stranger who'd never met him and had zero trust.

He read it out loud and stopped halfway. "This is all about the product. It never says what you actually *get*."

He didn't rebuild anything. He rewrote the words. The page now led with the outcome instead of the mechanism, the beach, not the plane.

His conversion moved that month. The product he'd been about to throw away was fine. The story wrapped around it wasn't.

A last word: you were never behind

If you've done all the right startup things and you're still stuck, here's the rule under all the other rules.

You're probably not behind. You're probably not bad at this. Your idea might be fine.

You were running a sensible process from the only perspective anyone ever handed you, solution-first, the product at the center, the customer's wanting assumed instead of examined. You heard *problem* and *solution* so many times that no one showed you the other place to stand.

Change the angle. Start at the outcome. Watch the whole map reorganize.

You don't need to work harder. You need to aim before you build.

The full case, the two spines, why the frameworks confuse founders, and what they were built for, is in the companion book, The Winning Startup Perspective: Outcomes, Not Solutions.

The founders in these stories are composites drawn from more than a hundred early-stage founders. None is a single real person. All of them are true.