FOUNDATIONS · AIL-FP-2026-04

# How to Get Started With Generative AI

*Skip the training. Try something impossible. Get hooked.*

Gregg Collins · Brandon Dickens · April 2026

An entire industry has sprung up to help organizations get started with generative AI: frameworks, maturity models, readiness assessments, center-of-excellence playbooks. Nearly all of it is unnecessary—and worse, it teaches the wrong lesson, because tame introductory tasks produce expectation confirmations rather than the expectation failures that drive real learning. This paper argues that the fastest path to GenAI fluency is not training but ambition: hand the tool a task so far outside your prior beliefs that, whichever way it goes, your mental model is forcibly updated. We illustrate with three examples from our own practice—a six-week course produced in a day, a twenty-by-twenty-by-twenty domain-sketching drill, and a self-consistent multinational audit simulation—and distill a four-part recipe: an obviously-too-hard task, a domain you know cold, permission to keep pressing, and a willingness to be wrong about what the tool can do. Once the jolt lands, the adoption problem solves itself.

---

> *The best way to predict the future is to invent it.*
>
> — Alan Kay

There is an entire industry springing up around the problem of how to get your organization started with generative AI. Consultants, frameworks, maturity models, readiness assessments, center-of-excellence playbooks, change-management rubrics. Somewhere out there a PowerPoint deck titled “The Seven Pillars of Enterprise AI Adoption” is being presented as we speak.

This is all, we are happy to report, almost entirely unnecessary. Much of the material is going to be analogous to the initial units of a programming class back when I was a computer science student: binary arithmetic, Boolean logic, logic circuits, von Neumann architectures, random access memories, and so on. All massively irrelevant, and also boring, because it had little real purpose as far as the task of

learning to program was concerned. The topics were much more aimed at a sort of domain-specific cultural literacy than trying to teach us to program well.

Here is what we have actually observed, in ourselves and in everyone we have watched become adept at using these tools: you do not learn to use AI by reading about AI. You learn by trying to get it to do something ridiculous. Something you would never dream of attempting with any other tool. You try it, something remarkable happens, and from that moment forward you are hooked. After that, the learning takes care of itself. You start looking for excuses to use it. You start noticing, a dozen times a day, that the thing you are about to spend an hour, a day, or a week on is a thing the AI could do in four minutes. The adoption problem solves itself, because you have become, functionally, an addict.

The trick is getting to that first hit. And the trick to getting to that first hit is to be far, far more ambitious than the tutorials tell you to be.

## Why the Usual On-Ramp Is Backwards

Most introductions to AI teach you the small, tame things first. Summarize this email. Rewrite this paragraph in a friendlier tone. Generate five subject lines. The implicit theory seems to be that you should start with what the tool can do reliably and work your way up to what it can do ambitiously, like a kid learning to swim in the shallow end.

The problem with this approach is that none of those small tasks are actually surprising. You knew, more or less, that a computer could shorten a paragraph. You would have bet money that something called “artificial intelligence” could generate five subject lines. Nothing about these exercises violates your prior expectations, and so nothing about them updates your mental model of what the tool can do. You come away from the tutorial with the correct belief that AI is a useful productivity

enhancer, in roughly the way that spellcheck was a useful productivity enhancer, and you go back to your actual work unchanged.

This is, incidentally, exactly backwards from how people actually learn. We have written at length elsewhere about how learning is driven by expectation failure—the jolt you get when reality does not match your mental model, which triggers you to update the model. Tame tasks do not produce expectation failures. They produce expectation confirmations, which are pedagogically useless.

If you want to actually learn what this technology is, you need to hand it a task so far outside your expectations that, whichever way it goes, you learn something. Either it fails in an interesting way, and you now have a calibrated sense of where the ceiling is. Or—and this is the more common outcome, and the one that changes everything—it succeeds, and your entire mental model of what a computer can do is forcibly upgraded on the spot.

## Exhibit A: A Course in a Day

Days after ChatGPT first became available to the public, before most people had any real idea what it could do, we decided to try an experiment. Our day job involves designing training—the kind of work that, in the conventional world, takes a team of instructional designers, subject matter experts, and reviewers somewhere on the order of six weeks to produce one reasonably polished course.

We asked the AI to build us a complete course. Not an outline. A full course: objectives, scenarios, dialogue, assessment items, coaching feedback, the works.

It did it. In under a day.

And it was better than a lot of courses out there in the world—maybe the majority of them. It was 100% usable as an actual training product.

What mattered, though, was not the course itself. What mattered was what happened to us in the process of watching it come together. Somewhere around hour two, the conversation in our office shifted from “can it do this?” to “oh my gosh, it’s doing it” to “what can’t it do?”—and that shift is the whole game. Once that question takes hold, you are no longer someone learning about AI. You are someone using AI, and the learning becomes a byproduct of use.

The six-weeks-to-one-day compression was not, it turned out, an upper bound. It was the floor.

## Exhibit B: The 20-20-20 Drill

A little later, we designed an exercise we ended up calling 20-20-20. It works like this. Pick a domain—any domain where you have enough expertise to know when you are being bluffed. Then:

1. Ask the AI to sketch twenty training scenarios in that domain.
2. Pick one of those scenarios and ask for twenty variations on it.
3. Pick one of the variations and ask for twenty sub-variations of that.
4. Pick one of those and ask for it to be fleshed out, end to end, as a complete scenario.

The point of the exercise is not the final scenario, though the final scenario is generally astonishingly good. The point is what happens to you in the middle. By the time you are two levels down, you are asking the AI to produce variations on variations of something that did not exist 2 minutes ago—and the variations are coherent, domain-appropriate, and just shockingly detailed.

Try this with something you know cold. Insurance underwriting. Oncology. Oil well drilling. Pearl diving. You will hit the moment, somewhere around the second level

of drill-down, where you realize the AI is not retrieving pre-formed material. It is constructing, in real time, from a model of the domain that is broader and deeper than you can even imagine. The depth comes out when you press on it. The breadth comes out when you step sideways.

You can read about this in a paper. It will not land. Run the 20-20-20 drill for ten minutes on a domain you know and it will hit home permanently.

## Exhibit C: The Multinational Audit Game

The experiment that really rearranged our minds was a simulation we asked the AI to run. The brief was roughly: imagine the learner is a senior auditor dropped into a large multinational corporation and tasked with conducting an audit. We want to train people to spot financial irregularities. Construct the corporation. Populate it. Run the audit. Make a game out of it.

What came back was, frankly, disorienting. The AI invented the company from whole cloth—its industry, its product lines, its organizational structure, its executive team, offices in a dozen countries. It named the CFO. It named the regional controllers. It generated a cast of characters with plausible backstories and motives. It produced financial statements that rolled up from subsidiaries to the parent.

Then we started drilling down. Show us Q3 expense reports from the São Paulo sales office. It produced them, line item by line item. Who approved them? What was the travel policy? We kept drilling, suspicious that somewhere the seams would show—that a number would contradict another number, that a name would flip, that the whole confection would start to shimmer and come apart.

It never happened. The numbers ticked and tied. Personalities, roles, and responsibilities stayed consistent across queries. The stories the characters told were internally coherent across sessions.

And then, the part that genuinely stopped us in our tracks: we asked it to plant problems in the fact pattern—specific irregularities for the learner to find—and bake them in there in a plausible and non-obvious way. Then we told it to guide the learner through the audit without ever giving the game away. It did. It kept the secret. It answered truthfully about the surface facts while preserving the things we had asked it to hide, and it did this across long back-and-forths without losing the thread. The problems it baked in were fully realized in massive depth—the learner could drill down into the data as far as they wanted and it all reinforced the same story.

The AI simulated an entire fictional company in its head, so to speak, observable from any angle, totally self-consistent, and infinitely expandable.

This is the kind of thing that, once you have seen it, you cannot un-see. And it is not the kind of thing you would have discovered by asking the tool to help rewrite your bio or organize your to-dos.

## The Method, Such As It Is

If there is a lesson in any of this, it is dead simple. To get started with generative AI, you do not need a strategy, a roadmap, a pilot committee, or a governance framework. You need, roughly, the following:

- **A task that is obviously way too hard.** Pick something that, if a colleague offered to do it for you by the end of the week, you would be totally skeptical. Building a full course. Running a full simulation. Auditing an imaginary company. The size of the ask is the whole point.
- **A domain you actually know.** This is non-negotiable. The surprise only lands, and the calibration only happens, if you are in a position to judge what you are looking at. The audit simulation would have been wasted on us if we did not know what a real one ought to look like. Run the experiment on your own turf.

- **Permission to keep going.** The first reply is never the experiment. The experiment is the fifth or tenth or twentieth follow-up, when you have stopped being polite and started actually pressing. Demand more. Ask for variations. Ask it to argue against itself. Ask it to plant something and then hide it from you. Just ask it to do better—it will.
- **A willingness—or even eagerness—to be wrong about what it can do.** Most of your prior beliefs about what a computer can and cannot do were formed before this technology existed. They are, with high probability, outdated. Treat every surprise as useful data and every failure as a more interesting surprise.

That is the whole method. The rest is momentum.

## The Knowledge Building Shows Up on Its Own

A curious thing happens a few months into this. You start to get ambitious.

At first, all you want is the chat window in your browser. That is enough to get you hooked. But eventually you hit a wall that the browser cannot get you over. Maybe the AI needs to see a folder full of files instead of a single pasted excerpt. Maybe you want it to keep working while you are asleep. Maybe you have realized, guiltily, that you have been copy-pasting the same 3,000-word context into every conversation for a week, and there has to be a better way.

So you move to a desktop app, because the desktop app can see your local files and act on them. Then you notice that the desktop app still makes you babysit every turn, and you start poking at the API—which sounds intimidating right up until you realize it is maybe thirty lines of code to script the thing you were doing by hand. You write the thirty lines. They work. Something in your head reorders itself.

At some point, probably sooner than you expected, you find yourself downloading a coding agent—Claude Code, or whatever its successor is by the time you read this—

and you give it the kind of software project you would, a year ago, have confidently declared yourself unqualified to attempt. It builds most of it. You correct it. It fixes the corrections. You ship something.

And along the way, almost as a side effect, you start picking up the vocabulary. Agents and agent swarms. RAG. Vector databases. MCP servers. Knowledge graphs. Prompt caching. Matryoshka embeddings. Context engineering. Evals. Guardrails. Coves. If those terms currently read like gobbledygook, don't worry about it—they will still read like gobbledygook tomorrow morning, which is fine. You will learn them when you need them… and mostly you will learn them by having conversations with the AI itself, which is a pretty good explainer of its own innards.

What you will not do is learn them the way you were made to learn the parts of a cell in ninth-grade biology. You will learn them the way someone learns the words "distributor" or "camshaft" when they start working on their own car. This is, for the record, also how humans learn anything in general. The mind is built to assimilate the lessons of experience, not to memorize inert facts. The glossary of AI is no exception.

You will not, it turns out, need to take a course on any of this. You will just need a project that is slightly too ambitious for what you already know how to do, and the curiosity to chase the next piece down when it becomes the thing in the way. The fancy vocabulary is a lagging indicator, not a prerequisite.

Which is, not coincidentally, the same story as the rest of this paper. Ambition first. Everything else follows.

## What Happens Next

Once you have had the experience—the jolt of watching the AI do something you would have bet against it doing—the trajectory from there is 100% predictable. You

start using it. You find a second use case. You find a fifth. You start catching yourself, mid-task, asking whether this is a thing the AI could do. Increasingly, the answer is yes. Your unofficial job title begins to include the phrase “AI wizard.”

None of this requires a training program. None of it requires a center of excellence. It requires, as a starting condition, one afternoon in which you tried to get the AI to do something you secretly suspected was impossible, and watched it nail it first try.

So: our advice, such as it is, comes down to a single instruction. This afternoon, think of the most ambitious thing you can imagine asking a computer to do for you. Not a reasonable thing. An unreasonable thing. The thing you would not dream of asking a new hire to attempt, the thing you would normally consign to a three-month project or a six-figure consulting engagement. Ask for it. Press when the first answer is thin. Drill in when a detail looks suspicious. Stay with it for an hour or three.

Then let us know what happened.