Continuous learning framework

Jarek Orzel · Follow
Published in Level Up Coding
9 min read · Jan 23, 2024

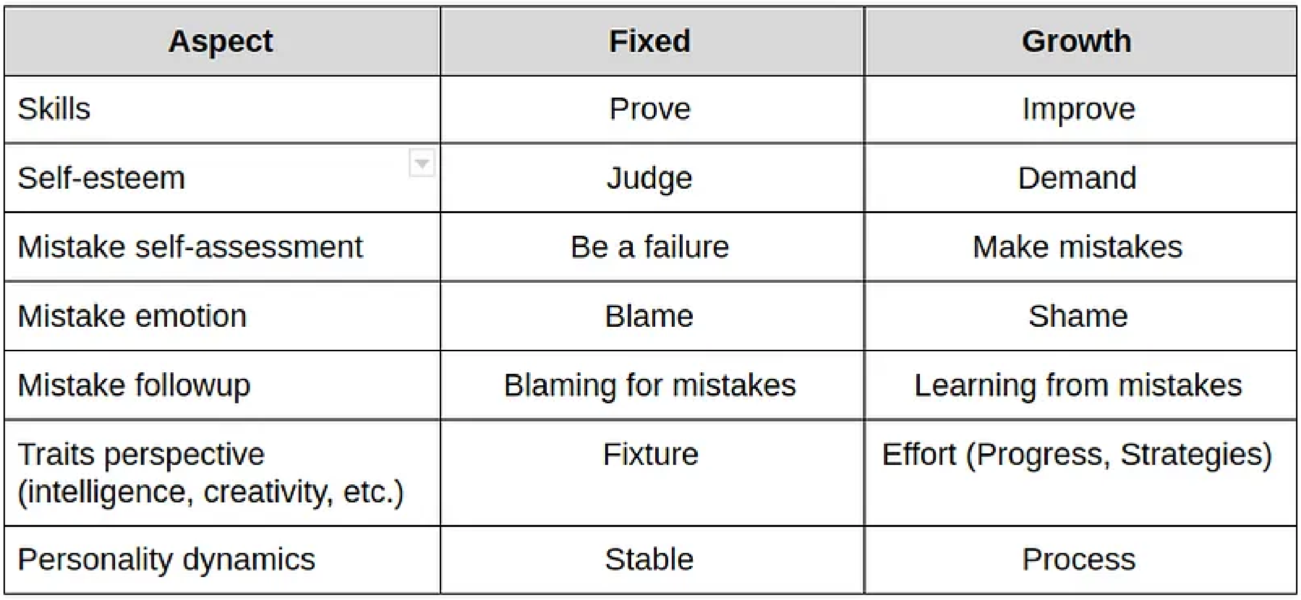
Listen Share

Photo by Tim Mossholder on Unsplash

Software development is a field that demands continuous skill improvement. Technology advances rapidly and to be successful you must find a balance between a destructive attempt to be up-to-date at all costs and clinging to your comfort zone by taking the same repetitive tasks all the time. There is a huge difference between 5 years of experience doing the same and 5 years of experience practicing new things and taking on challenges. Here I have some tips on how to build your continuous improvement framework.

Cultivate growth mindset

Mindset is often underestimated as a part of the soft skills family that is not the core of developer activity. However, a proper attitude can be crucial for implementing continuous learning ability. Carol Dweck in her seminal book *Mindset: The New Psychology of Success* claims that a growth mindset is a natural way for the successful knowledge acquisition process. Here we have a short comparison between a growth mindset that encourages improvement and a fixed mindset that rather inhibits gathering new skills.



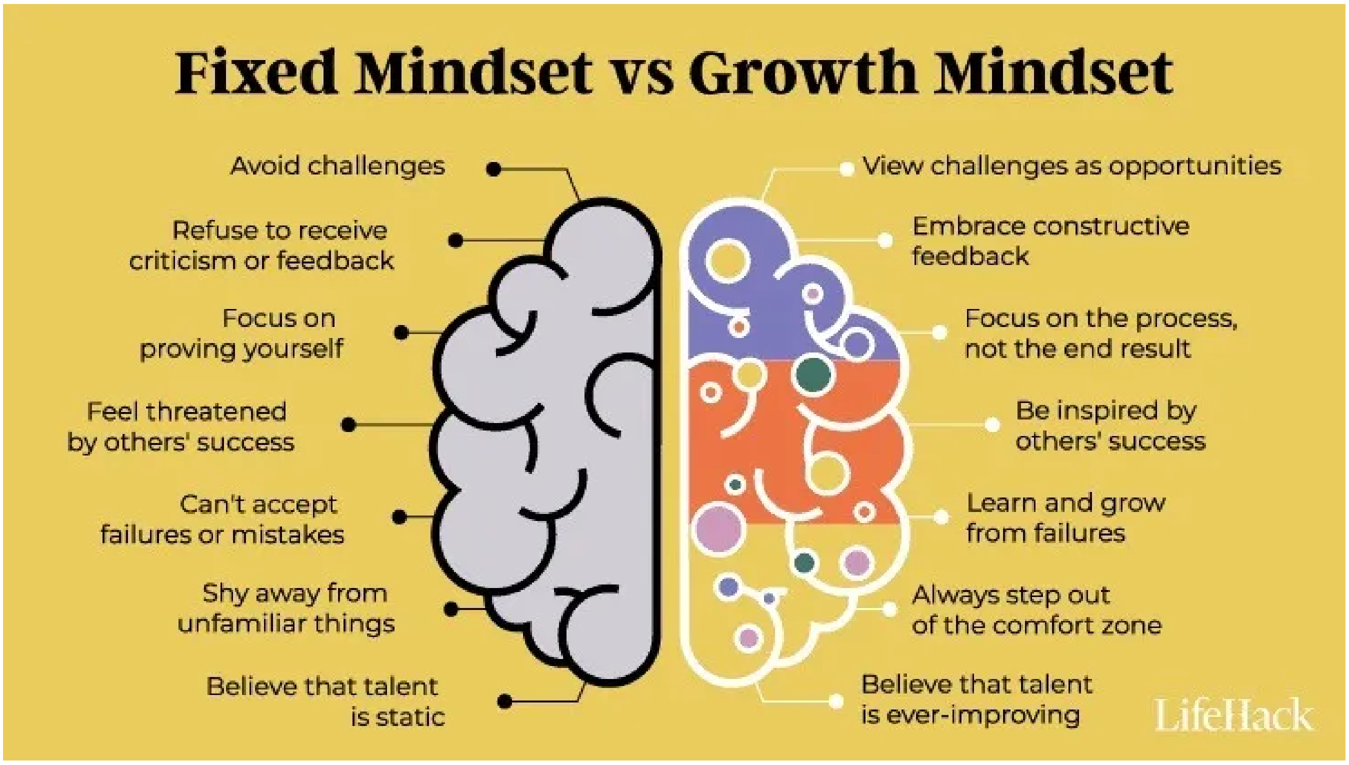
Aspect	Fixed	Growth
Skills	Prove	Improve
Self-esteem	Judge	Demand
Mistake self-assessment	Be a failure	Make mistakes
Mistake emotion	Blame	Shame
Mistake followup	Blaming for mistakes	Learning from mistakes
Traits perspective (intelligence, creativity, etc.)	Fixture	Effort (Progress, Strategies)
Personality dynamics	Stable	Process

Fixed vs Growth Mindset

A growth mindset can unlock your potential and make your mistakes a part of an improvement process rather than an indicator of failure.

Believe you are “talented” enough

People with a fixed mindset often limit themselves by thinking they can only succeed in areas where they believe to have innate talent or where they’ve demonstrated prior competence. This can create a fear of failure in new areas or a reluctance to take on challenges that might stretch their abilities.


Fixed Mindset vs Growth Mindset
Avoid challenges
Refuse to receive criticism or feedback
Focus on proving yourself
Feel threatened by others' success
Can't accept failures or mistakes
Shy away from unfamiliar things
Believe that talent is static
View challenges as opportunities
Embrace constructive feedback
Focus on the process, not the end result
Be inspired by others' success
Learn and grow from failures
Always step out of the comfort zone
Believe that talent is ever-improving
LifeHack

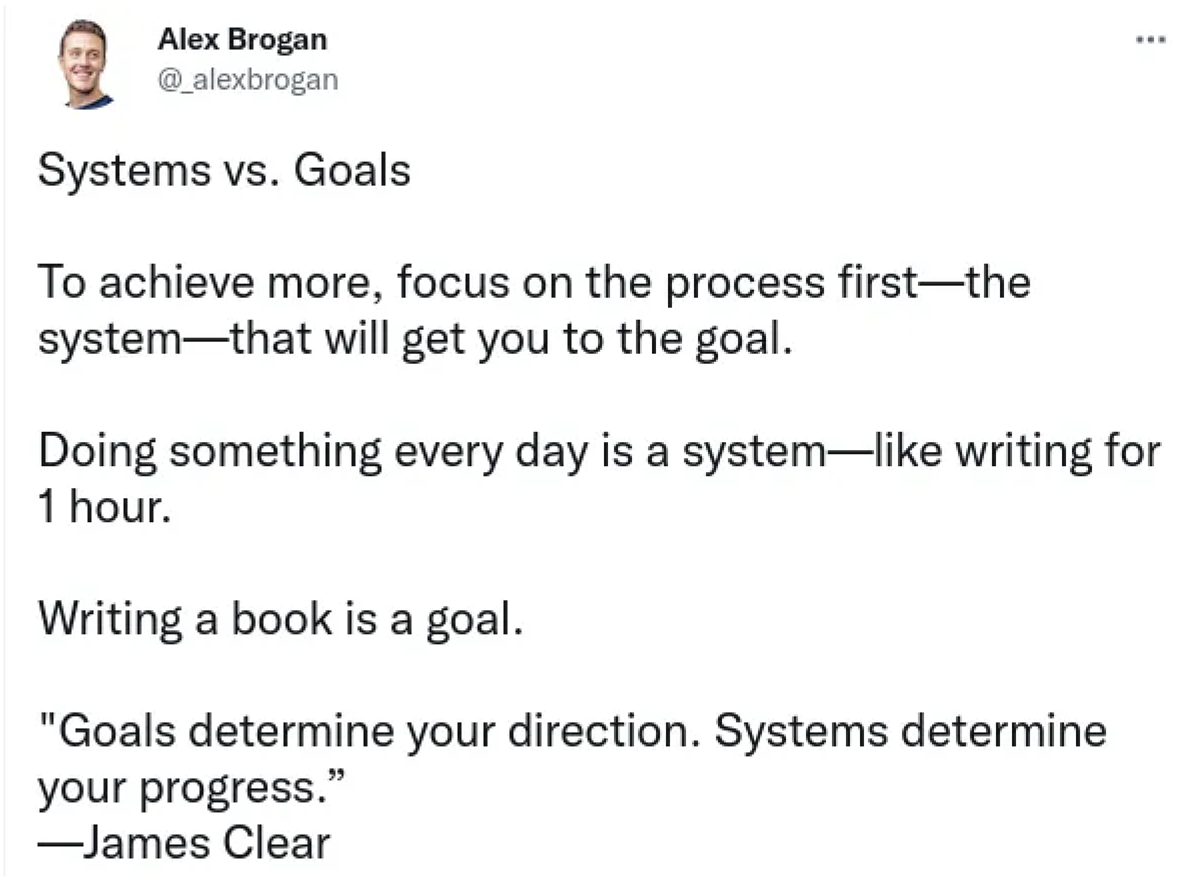
Fixed vs Growth mindset behavior, source

I'm firmly convinced that nothing is truly innate or inherent. Of course, it does not mean that everybody has the same chance to succeed. Every skill and capability is a product of prior experiences. While it's true that some individuals appear to acquire skills more rapidly than others, there must always have been preceding practice or experience that contributed to their current level of ability.

From the argument that expanding our horizons and attempting to excel in areas where we might not consider ourselves "talented" (do not have enough expertise or skill-improvement rate now) can lead to success, it's crucial to understand that trying doesn't guarantee surefire success. Achieving success in virtually any field is challenging. It requires determination and grit, and the support of an enabling environment. This includes people who can guide and mentor us, financial resources to fund our practice, and other essential resources like good health that are necessary for competitiveness and eventual success. Nonetheless, the growth mindset illuminates the possibility of success. It underscores our capacity to seek out resources, nurture our environment, methodically design our practice, and continually invest in training—akin to buying a ticket for the lottery of our aspirations.

Build a system with goals

It is important to have goals that motivate us to improve ourselves. However, to be successful in achieving your goals you should place them within some repeatable process. On the one hand, writing a blog post can be a goal. On the other hand, writing every day something for 30 min can be a system that makes our goals sizable.



Alex Brogan
@_alexbrogan

Systems vs. Goals

To achieve more, focus on the process first—the system—that will get you to the goal.

Doing something every day is a system—like writing for 1 hour.

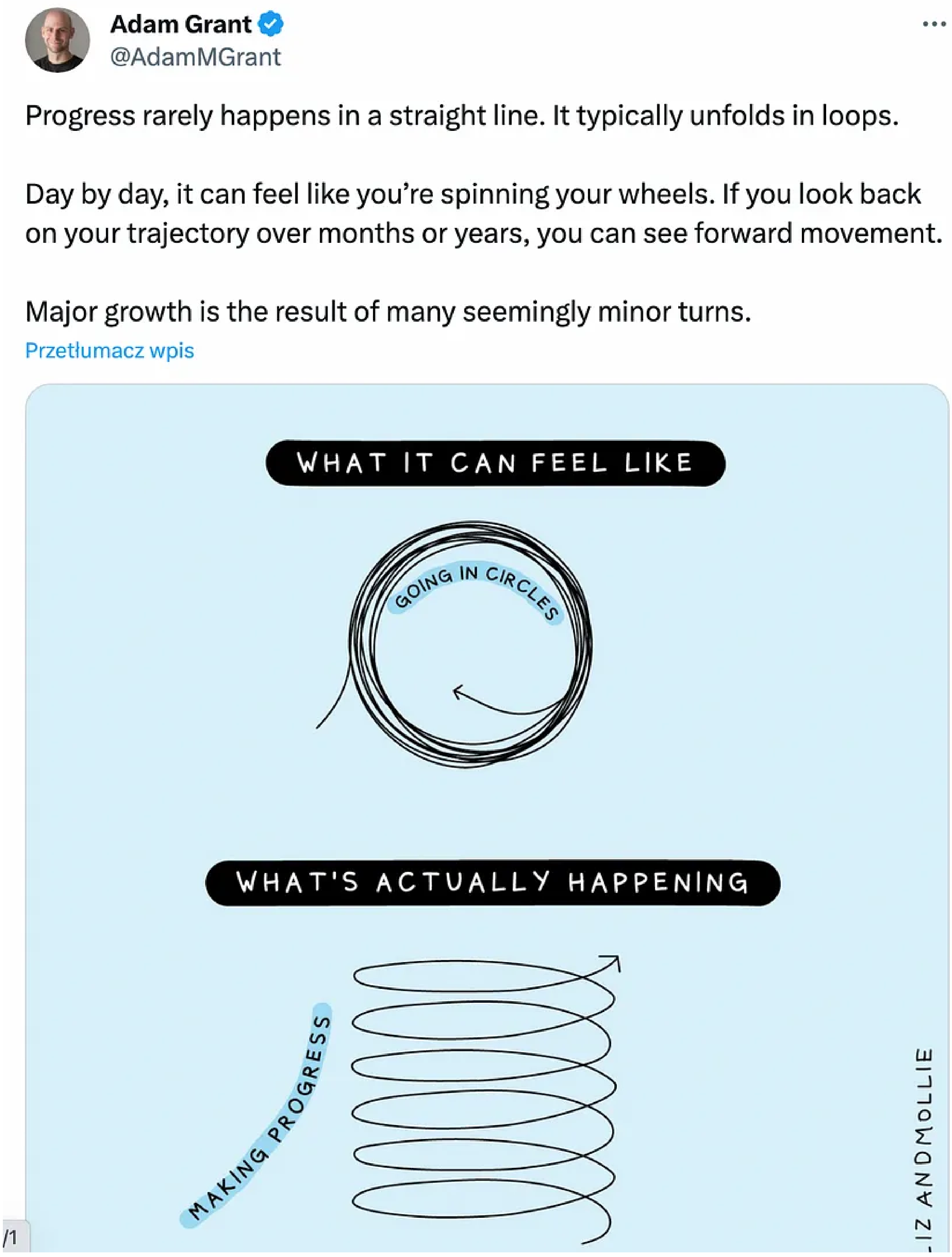
Writing a book is a goal.

"Goals determine your direction. Systems determine your progress."
—James Clear

Alex Brogan's tweet source

Embrace radical incrementalism

Many people tend to focus on the end result, which can seem daunting. We often fail to see the detailed journey they took to get there.



Adam Grant
@AdamMGrant

Progress rarely happens in a straight line. It typically unfolds in loops.

Day by day, it can feel like you're spinning your wheels. If you look back on your trajectory over months or years, you can see forward movement.

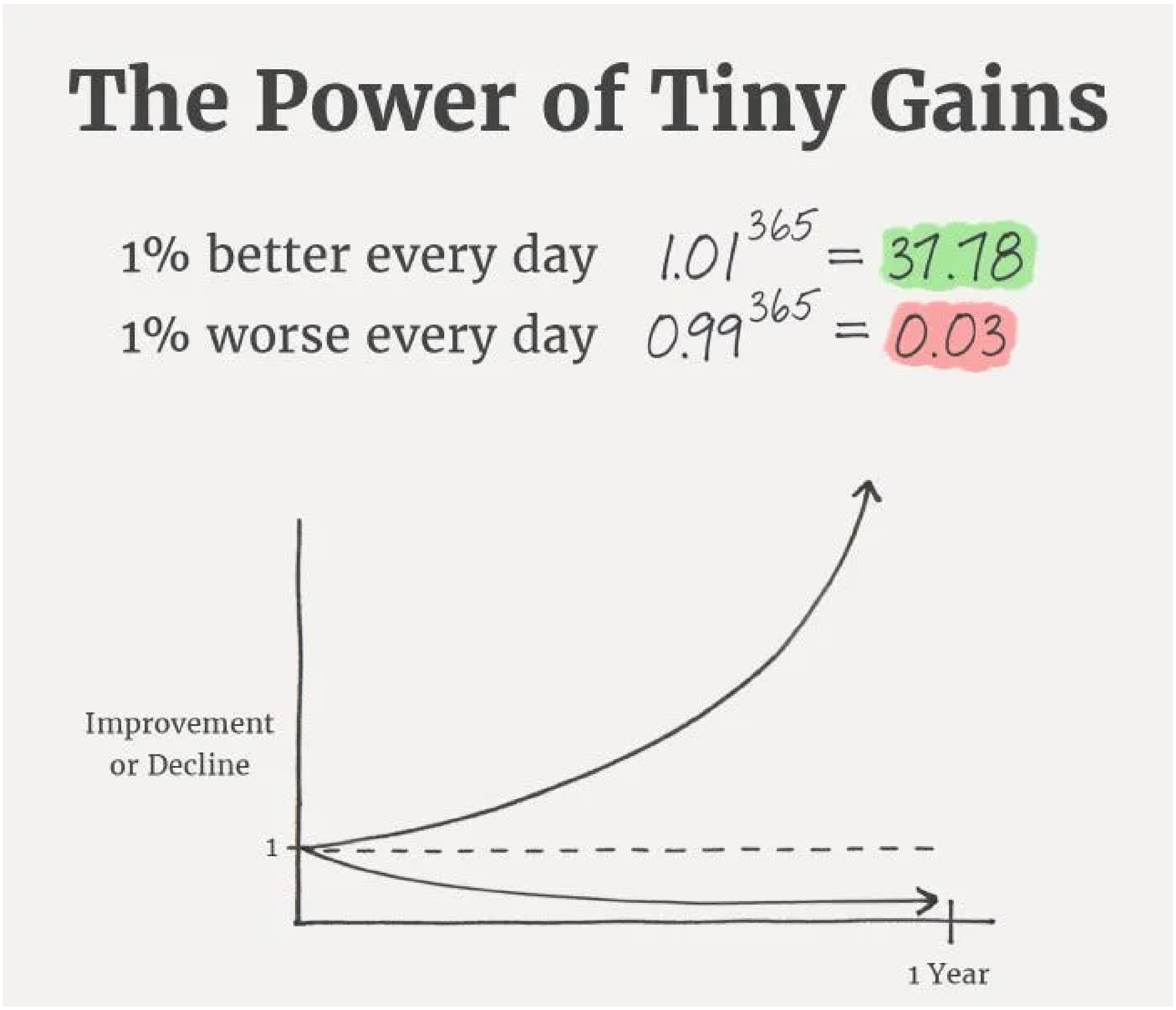
Major growth is the result of many seemingly minor turns.

Przetłumacz wpis

Adam Grant's tweet source

Take writing a great book, for instance. Many people might believe it's beyond their reach, deeming themselves insufficiently skilled. However, the creation of a book is far from an instantaneous feat; it demands a significant, enduring effort. The sheer scale of the task can feel overwhelming. Therefore, a more constructive approach is to acknowledge that substantial progress in any field doesn't happen overnight. Patience is the key. Perhaps we cannot craft an entire book in one go, but we can start with a blog post. Even writing a blog post is a process that requires time and commitment.

More examples how small positive actions repeated with consistency over time can lead to great things you can find in Darren Hardy's inspirational book—The Compound Effect.

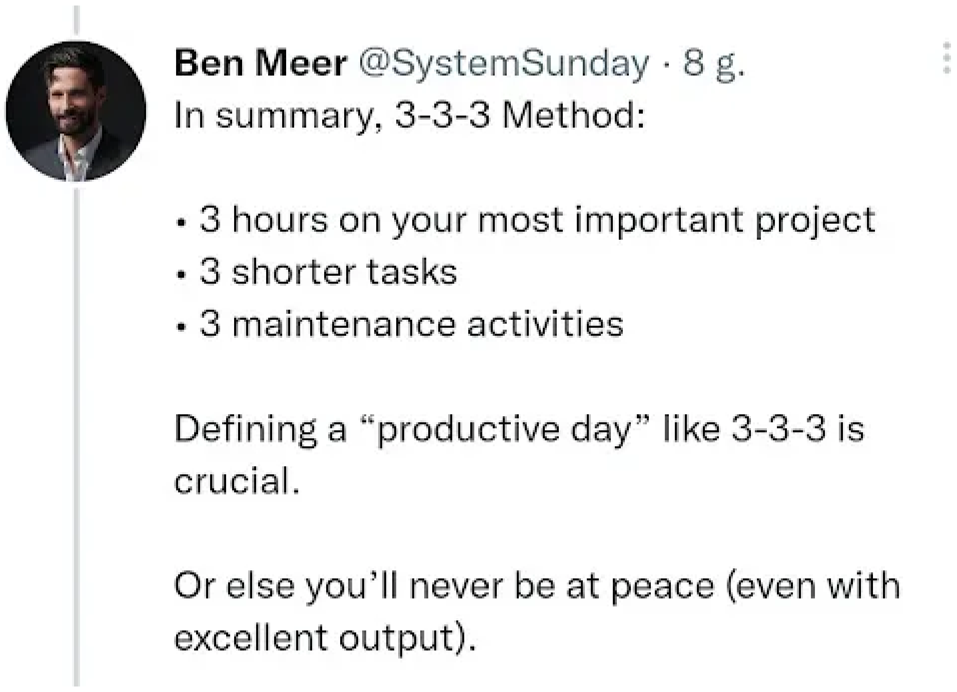


https://thuong.substack.com/p/compound-effect

Put effectiveness over efficiency

Doing a good job for your employer does not always mean maximizing output. The best way to speed up and improve your work results is to build a range of skills. Therefore, it should be no surprise that you devote some of your work time to developing your competencies. Another important factor for productive work is the quality of the work process itself (efficiency). It can be improved by implementing strategies and conventions. Some examples:

- Pomodoro technique
- find out your peak state of energy (morning for most), and allocate it for deep work without interruptions (Maker's Time)
- 3–3–3 method



Ben Meer @SystemSunday · 8 g.
In summary, 3-3-3 Method:

• 3 hours on your most important project
• 3 shorter tasks
• 3 maintenance activities

Defining a "productive day" like 3-3-3 is crucial.

Or else you'll never be at peace (even with excellent output).

Ben Meer's tweet source

However, we must remember the most important thing: efficiency alone is not sufficient if you're not working on tasks that truly matter or align with your goals. It is the most stupid thing to do efficiently something that should not be done at all.

Seek knowledge of the business domain

The goal of software development is not to write code, but to solve business problems (usually by coding). However, you cannot find an optimal solution if you do not have a good understanding of your business domain. Again, it is significant to be efficient and productive in your work (doing things right), but more important is to be effective (doing the right things). The focus should be on outcome maximization (user impact, user experience) while minimizing output (features, tasks). It is the main point of Jeff Patton's great talk about user stories. Be a partner for business experts, stay curious, and question requirements. It is possible only if you understand the business domain.

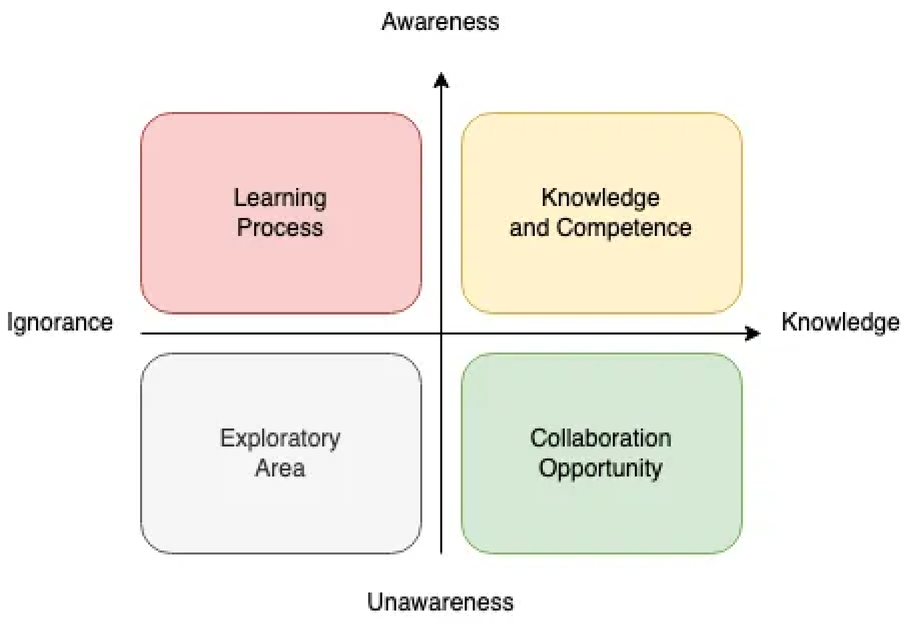
Find your inspirations

The worst scenario for you is not being aware of your knowledge gaps and deficiencies. To avoid the Dunning-Kruger effect you have to be proactive, look into fields that you are not familiar with, and take activities that broaden your horizons, e.g.:

- take part in conferences,
- skim hackernews regularly,
- listen to tech podcasts, watch tech videos, and read blog posts.

Explore unknown unknowns

The concept of "known unknowns" refers to the things you are aware you don't know. For example, you know you don't know how to play a musical instrument. "Unknown unknowns" are the things you don't even realize you don't know, like a new field of science or a skill you've never heard of. These can be game-changers in personal development. Broadening your horizons means exposing yourself to diverse knowledge and experiences.



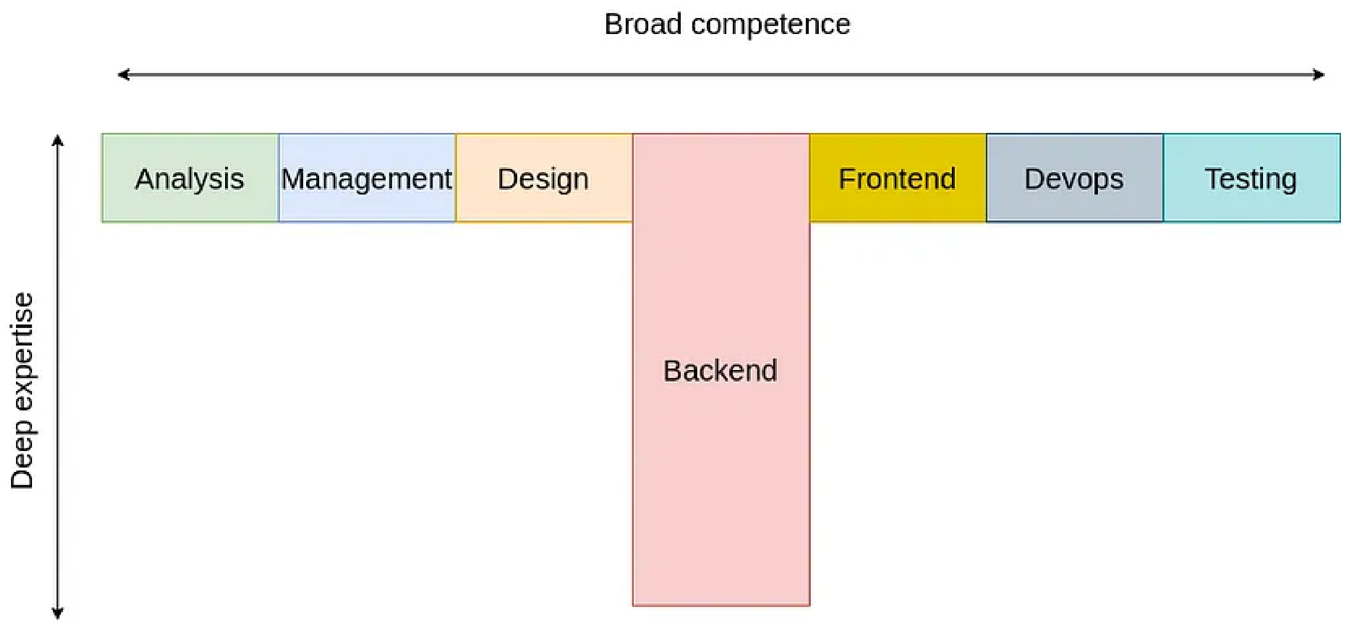
Knowledge and awareness matrix

A conservative approach to learning often involves sticking to what you already know or improving existing skills. While this is valuable, it can limit your growth potential. Curiosity, on the other hand, drives you to explore the unknown and learn new things. It opens doors to unexpected opportunities and is a powerful driver of success because it pushes you to explore beyond your comfort zone.

Simple examples of venturing into the realm of 'unknown unknowns' include reading a blog post or book, watching a conference video, or listening to a podcast in a field that falls outside your usual interests or expertise.

Determine a career path

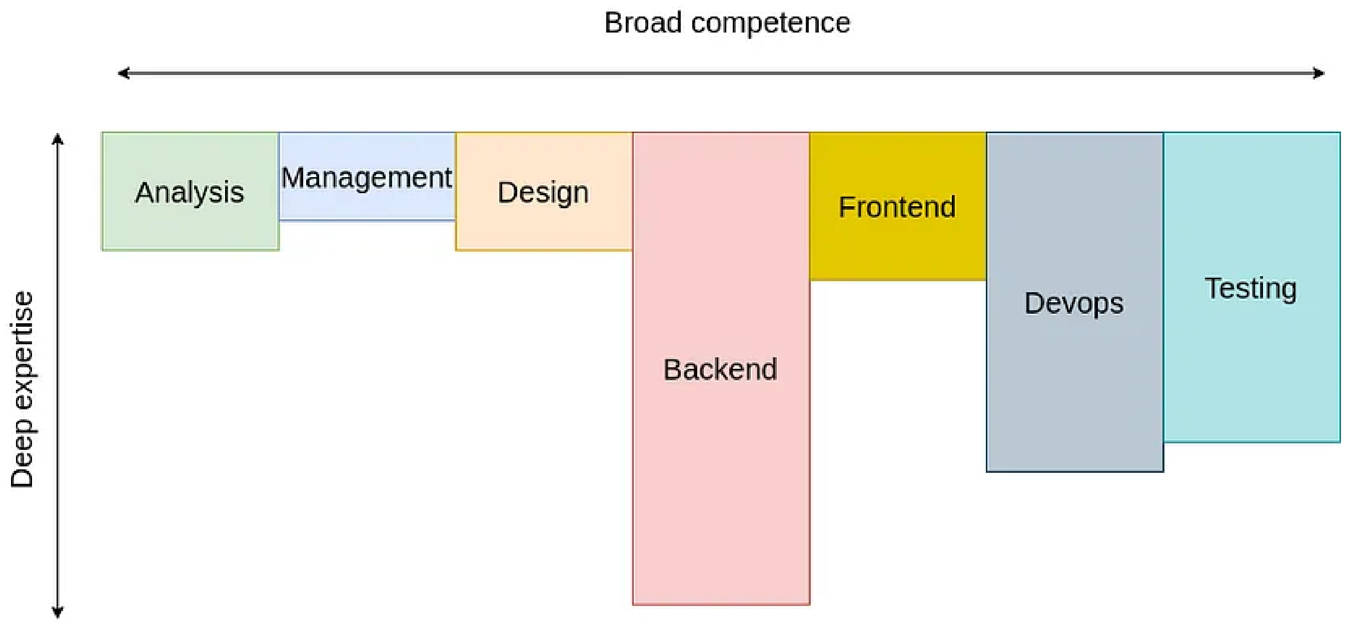
The range of inspirations for improvement cannot be too broad, because for sure, you do not have enough time to master all possible domains. You should choose a career path and define areas to grow. Proper direction is as important as fast progress. One of the useful concepts here is T-shaped engineers who are deep experts in 1–2 domains and are also competent in several other fields to be a valuable member of a team.



T-shaped engineer example

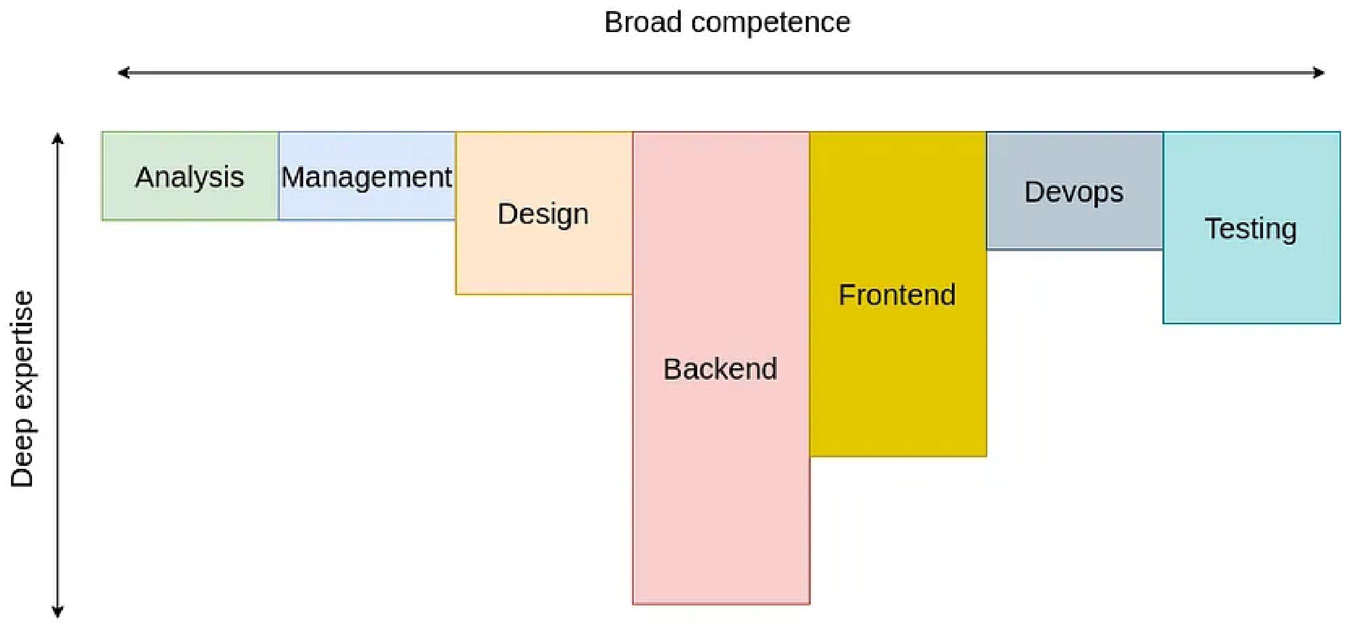
The above example is probably a little bit oversimplified for a Backend Developer. However, we can design some more suitable charts for the following positions:

- Senior Backend Developer



T-shaped Backend Developer

- Full Stack Developer



T-shaped Full Stack Developer

You can do a similar exercise for other career paths, like Software Architect, Engineering Manager, or QA Engineer. While the approach is qualitative (it is not easy to assess the current level of my competence in a given domain), it can be a good starting point for defining personal goals and learning paths.

Remember about fundamentals

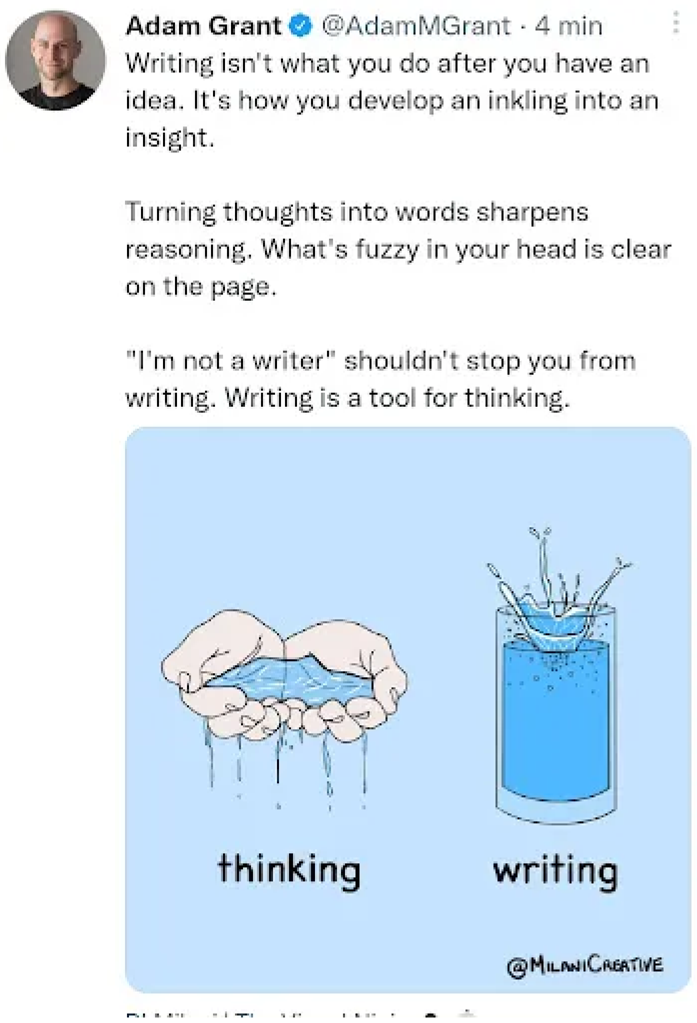
Fundamentals give you means for solving technical challenges and can be divided into two groups:

- technology agnostic, like algorithms, system design, TDD approach, etc.
- technology bounded, like a programming language, web framework, database, etc.

The former can frame your solution, while the latter provides implementation tools. Given their significance, both should receive adequate attention.

Document your work

The best approach to boost your understanding of a concept is to write about it or try to explain it to other people.


Adam Grant @AdamMGrant · 4 min
Writing isn't what you do after you have an idea. It's how you develop an inkling into an insight.

Turning thoughts into words sharpens reasoning. What's fuzzy in your head is clear on the page.

"I'm not a writer" shouldn't stop you from writing. Writing is a tool for thinking.

Adam Grant's tweet source

You can use the medium platform (or similar) for blogging to document your work. Have you been doing something interesting? Write about it. You do not have to share it. Do it, to organize your reasoning and thought process. If you are unsure of your writing skills use Grammarly and Quilbot to make the process less erroneous.

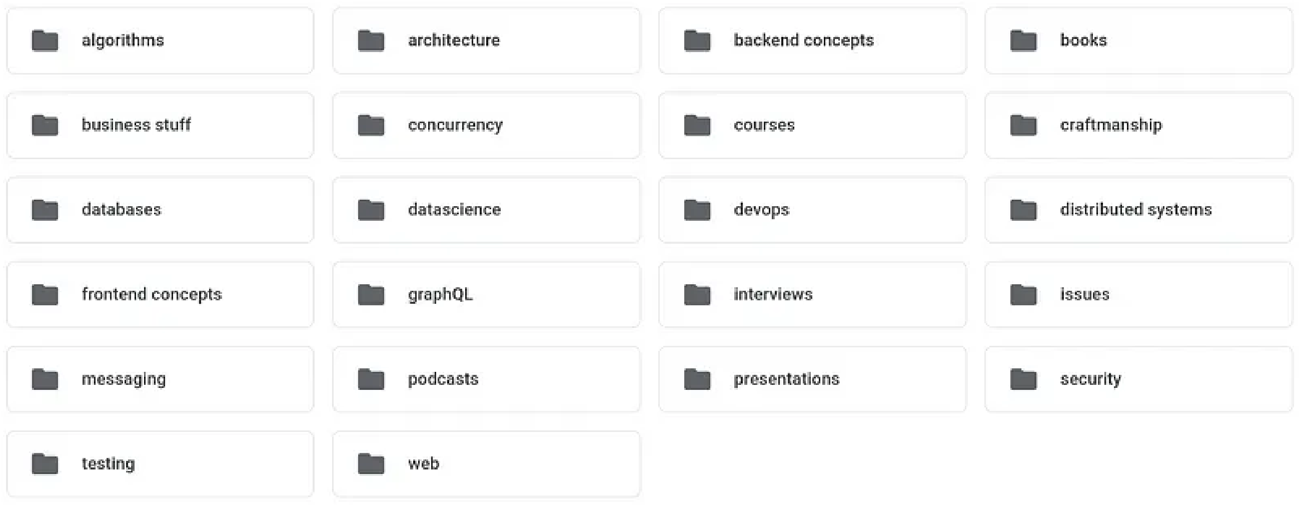
Sharpen your tools

If something makes you inefficient, tackle it. Sometimes, it would be worth stopping to sharpen your tools. It can be many things:

- your slow computer/lack of additional monitors
- lack of familiarity with basic keyboard shortcuts (run given test, switch file in the IDE, one-click rename variable/class/method)
- clicking a lot of pages to test/check something, instead of writing one SQL query
- etc.

You have to recognize what slows you down and address it. The learning process itself also demands care and maintenance. I will give two examples of how to upgrade it:

- organize sources you have gathered. Take a convention to keep it ordered using for example Google Docs and a proper folder tree. Sometimes, I come across a possibly interesting blog post or presentation speech, but I do not have enough time or focus to acquire it at that time. So the best way to not let it slip away is to save it in a retrievable place.



Folder ordering to gather notes

- take notes when you read a book or an article. I found it difficult when it is a paper book, but I discovered a great tool: https://raindrop.io/ which makes it easy to take a text excerpt photo and tag it.

Conclusion