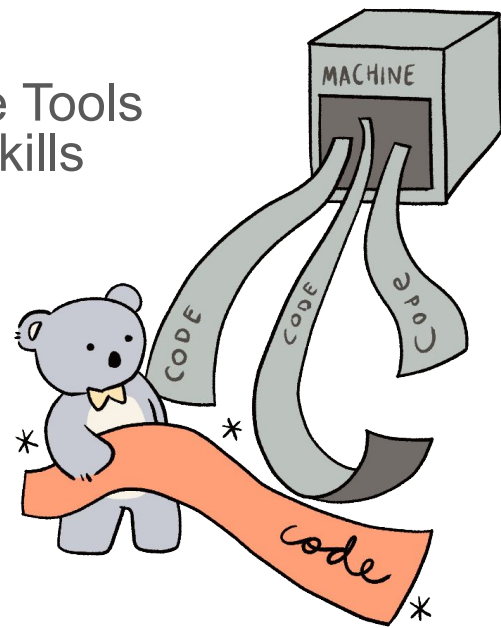
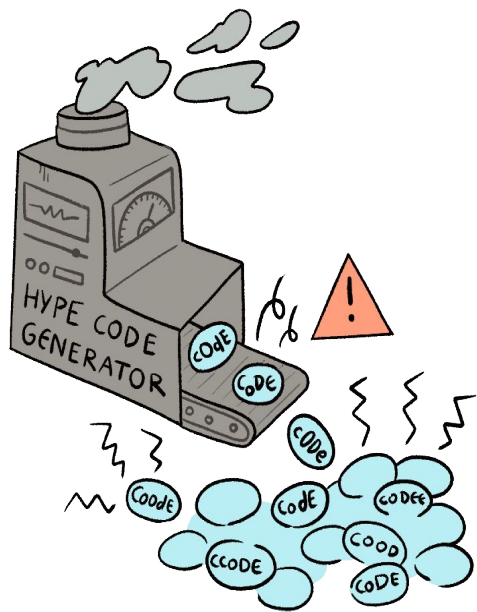


Forgetting Machines

Impacts of AI Code Tools
On Developer Skills



[FFConf](#), Brighton 2025-11-14

CC BY-NC-ND

Eda Eren & Jessica Rose

Slide art by [Kiri](#)

Hi, I'm Eda!

- Passionate about how our understanding of humanities can make us better technologists
- [Code Your Future](#) Volunteer
- Happy to chat if you're hiring developers who love to learn

🕸: rivea0.github.io

🐘: mastodon.social/@rivea0

🦋: @rivea0.bsky.social



Hi, I'm Jess



- Works in developer outreach and education
- Co-founded [Bad Website Club](#), a free online webdev bootcamp
- Loves languages*
- Always excited to hear about what you're excited about or to see pet photos. Find me online and say hello!

🕸: jessica.tech

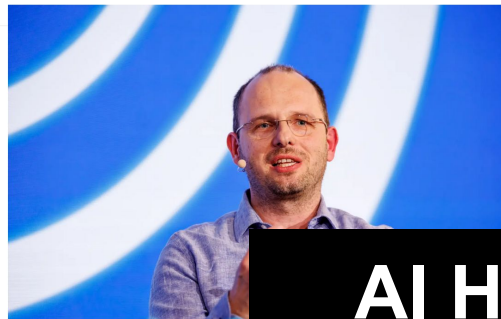
🦋: @jesslynnrose.bsky.social

🐘: mastodon.social/@jessie

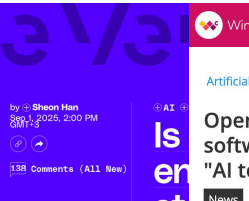
★ Please do not tell me anything interesting about your language, I am interested in everything and have no self control

GitHub CEO delivers stark message to developers: Embrace AI or get out.

By [Alistair Barr](#) Author of the [Tech Memo](#) newsletter [+ Follow](#)



Thomas Dohmke, CEO of GitHub, speaking at a conference.



McKinsey & Company



How an AI-enabled software product development life cycle will fuel innovation

February 10, 2025 | Article

[Tech Industry](#) > [Artificial Intelligence](#)

Microsoft's CEO reveals that AI will replace 30% of its code — some projects will have 50% code written by AI

by Nassim Nasir published 30 April 2025

ist one-third.



urchase through links on our site, we may ear Here's how it works.

CompTIA

[Home](#) > [CompTIA Blog](#) > [Will AI Replace Software Developers?](#)

Will AI Replace Software Developers?

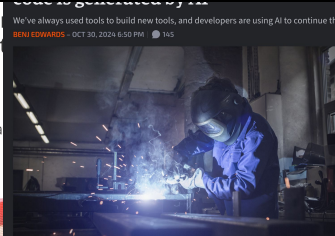
December 18, 2024

AI Hype is everywhere right now, and can feel pretty inescapable

OpenAI's Sam Altman claims AI will "gradually" replace software engineers — Creating an urgent need for "AI tools"

News By [Kevin Okemwa](#) published March 24, 2025

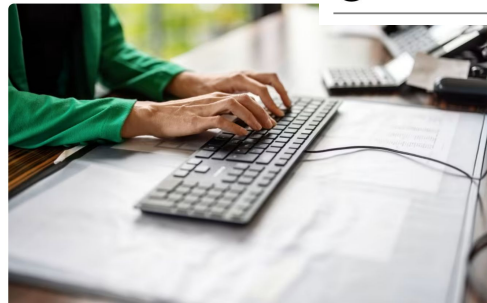
OpenAI CEO Sam Altman says there's a high probability that AI will replace software engineers in a gradual but accelerating manner.



BUSINESS > TECH • 3 MIN READ

Google says 90% of tech work will be done using AI at work

UPDATED SEP 23, 2025
By Lisa Eadicicco



AI assistants are reshaping software development

Published: 11 February 2025 · Last updated: 11 February 2025



 Cybernews Team

code Will Be
time to stop taking

@CRIPTO.COM @3PROCO



[Home](#) / [Innovation](#) / [Artificial Intelligence](#)

AI agents will match 'good mid-level' engineers this year, says Mark Zuckerberg

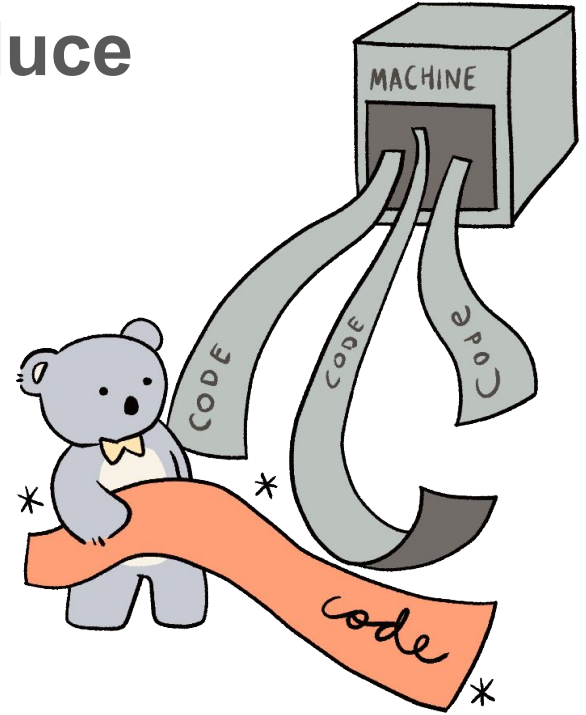
Autonomous software engineering agents will take over significant programming tasks, predicts Meta's CEO. And he's counting on Llama to achieve that goal.



Written by [Tiernan Ray](#), Senior Contributing Writer
Jan. 30, 2025 at 9:41 a.m. PT

First, let's look at how LLMs produce code at a highly simplified level

- Datasets of code are collected
- Models are trained on the data
- Users query the model
- When the model is given a query it produces output based on pattern and frequency matching with what the model has most often “seen” in it's datasets, in contexts it “thinks” are connected to the query





Learning happens when new neural pathways in our mind are built and when existing pathways are strengthened through use and connected to each other.

We can think of this as hedge maze, with the existing connections being the current pathway through the maze. And to create a new connection, we would need to push through the bush to create it and keep using the new path to widen and establish it.

Like pushing through a hedge maze, the first part of establishing a new neural pathway involves a little friction to build it.






Declarative learning:

- You know information that you can recall and describe

Procedural learning:

-  The ability to do stuff like that

(When you develop something through practice, it's intuitive)

Mainichi Shimbun, Public domain, via Wikimedia Commons

Individual Impacts



These tools remove the friction to establish new neural pathways and build the connections needed for procedural learning.

They jump right to the output, the answer.





AI coding tools can only produce output based on what they've seen in their datasets. Which means they can't produce creative or original work.

Or, something weird for weirdness sake, like an esoteric programming language that is created not necessarily for practical use, but as a creative act, maybe as a joke, or to win a competition.



Befunge

The image shows a light gray rectangular box containing the Befunge code for an infinite loop. The code is arranged in a diamond shape: the first character is '>', the second is 'v', the third is '<', and the fourth is '^'. This sequence of characters causes the program to move in a continuous loop: right, down, left, and up, returning to the start of the code.

An infinite loop in Befunge, a 2-dimensional programming language

Or, a hello world program in SPL, the Shakespeare Programming Language where the code is written as a play script, the characters are variables and they edit each other's values using dialogue.

Shakespeare Programming Language

Do Not Adieu, a play in two acts.

Romeo, a young man with a remarkable patience.
Juliet, a likewise young woman of remarkable grace.
Ophelia, a remarkable woman much in dispute with Hamlet.
Hamlet, the flatterer of Andersen Insulting A/S.

Act I: Hamlet's insults and flattery.

Scene I: The insulting of Romeo.

[Enter Hamlet and Romeo]

Hamlet:
You lying stupid fatherless big smelly half-witted coward!
You are as stupid as the difference between a handsome rich brave
hero and thyself! Speak your mind!

You are as brave as the sum of your fat little stuffed misused dusty
old rotten codpiece and a beautiful fair warm peaceful sunny summer's
day. You are as healthy as the difference between the sum of the
sweetest reddest rose and my father and yourself! Speak your mind!

You are as cowardly as the sum of yourself and the difference
between a big mighty proud kingdom and a horse. Speak your mind.

Speak your mind!

[Exit Romeo]

Scene II: The praising of Juliet.

[Enter Juliet]

Hamlet:
Thou art as sweet as the sum of the sum of Romeo and his horse and his
black cat! Speak thy mind!

[Exit Juliet]

Scene III: The praising of Ophelia.

[Enter Ophelia]

Hamlet:

Thou art as beautiful as the difference between Romeo and the square
of a huge green peaceful tree. Speak thy mind!

Thou art as lovely as the product of a large rural town and my amazing
bottomless embroidered purse. Speak thy mind!

Thou art as loving as the product of the bluest clearest sweetest sky
and the sum of a squirrel and a white horse. Thou art as beautiful as
the difference between Juliet and thyself. Speak thy mind!

[Exeunt Ophelia and Hamlet]

Act II: Behind Hamlet's back.

Scene I: Romeo and Juliet's conversation.

[Enter Romeo and Juliet]

Romeo:
Speak your mind. You are as worried as the sum of yourself and the
difference between my small smooth hamster and my nose. Speak your
mind!

Juliet:
Speak YOUR mind! You are as bad as Hamlet! You are as small as the
difference between the square of the difference between my little pony
and your big hairy hound and the cube of your sorry little
codpiece. Speak your mind!

[Exit Romeo]

Scene II: Juliet and Ophelia's conversation.

[Enter Ophelia]

Juliet:
Thou art as good as the quotient between Romeo and the sum of a small
furry animal and a leech. Speak your mind!

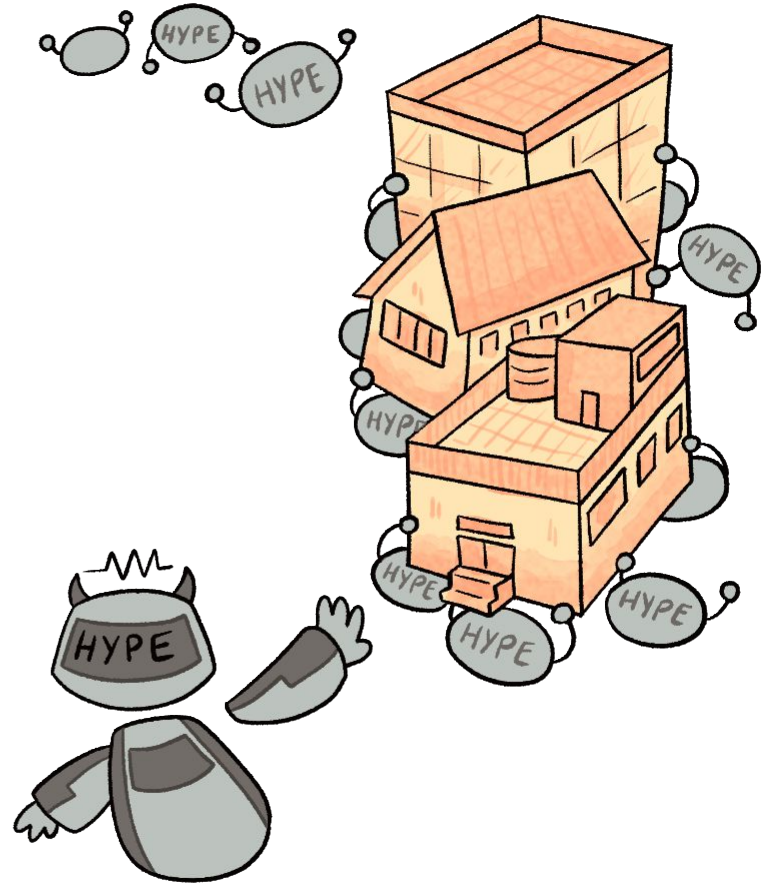
Ophelia:
Thou art as disgusting as the quotient between Romeo and twice the
difference between a mistletoe and an oozing infected blister! Speak
your mind!

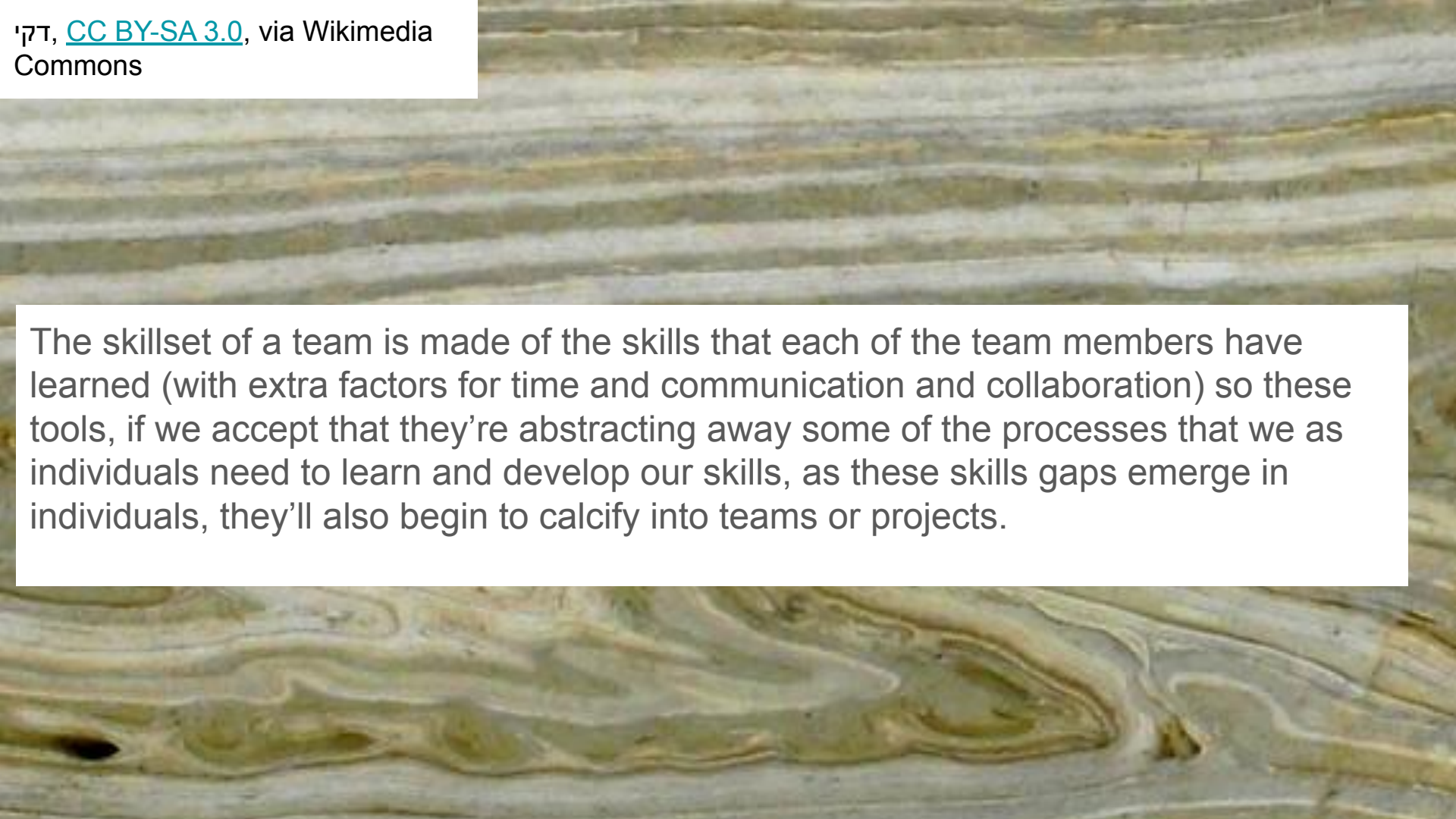
[Exeunt]

“Esoteric languages inherently make little sense and frequently serve little purpose, making them conceptually completely counter to AI-generated code and thus often not even understood by them—almost the code equivalent of wearing clothing to confuse facial recognition software.”

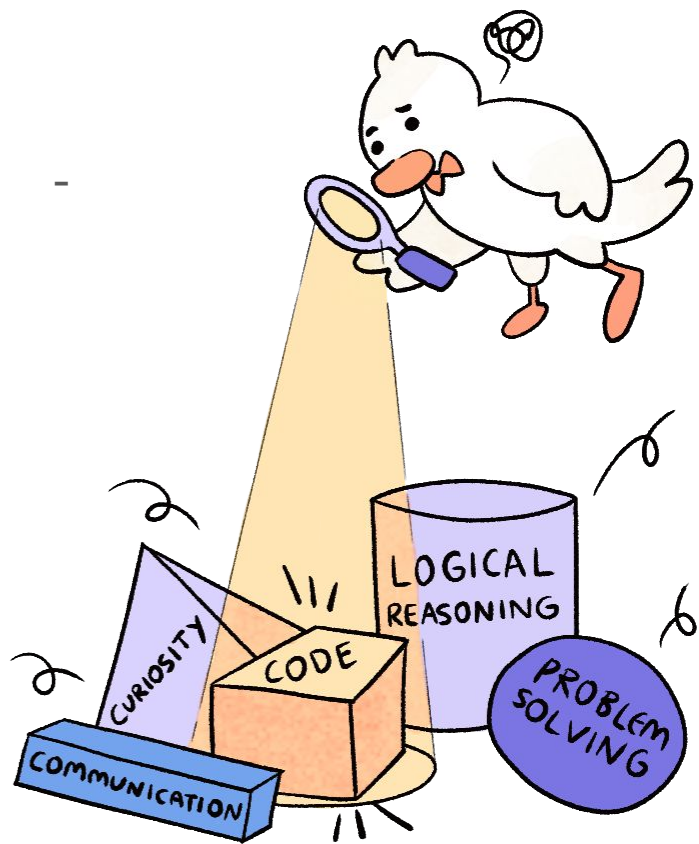
- <https://spectrum.ieee.org/esoteric-programming-languages-daniel-temkin>

Team and Project Impacts





The skillset of a team is made of the skills that each of the team members have learned (with extra factors for time and communication and collaboration) so these tools, if we accept that they're abstracting away some of the processes that we as individuals need to learn and develop our skills, as these skills gaps emerge in individuals, they'll also begin to calcify into teams or projects.



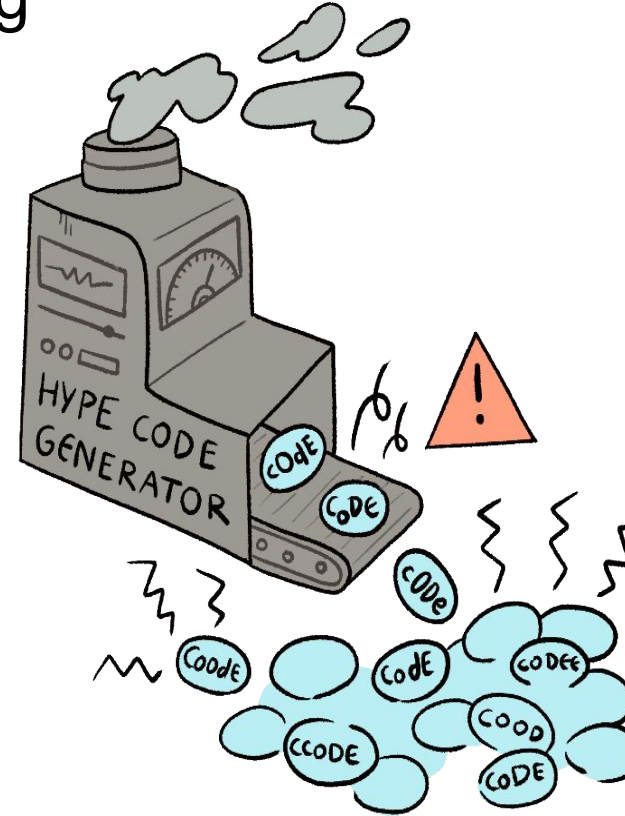
These tools also do learners a disservice by producing code independent of user needs, use cases and user research

Abstracting all this away makes it more likely you'll build the wrong thing, but also prevents you from learning the non-code skills engineers need to perform and thrive.

AI Code Tools are Amazing at Prototyping

AI code tools can be fantastic at build prototypes. But these prototypes are fragile and don't often connect and scale to make real, maintainable codebases for you or your users.

By learning to make quick, disposable prototypes, we're forgetting how to write code that is robust and even more importantly, maintainable.

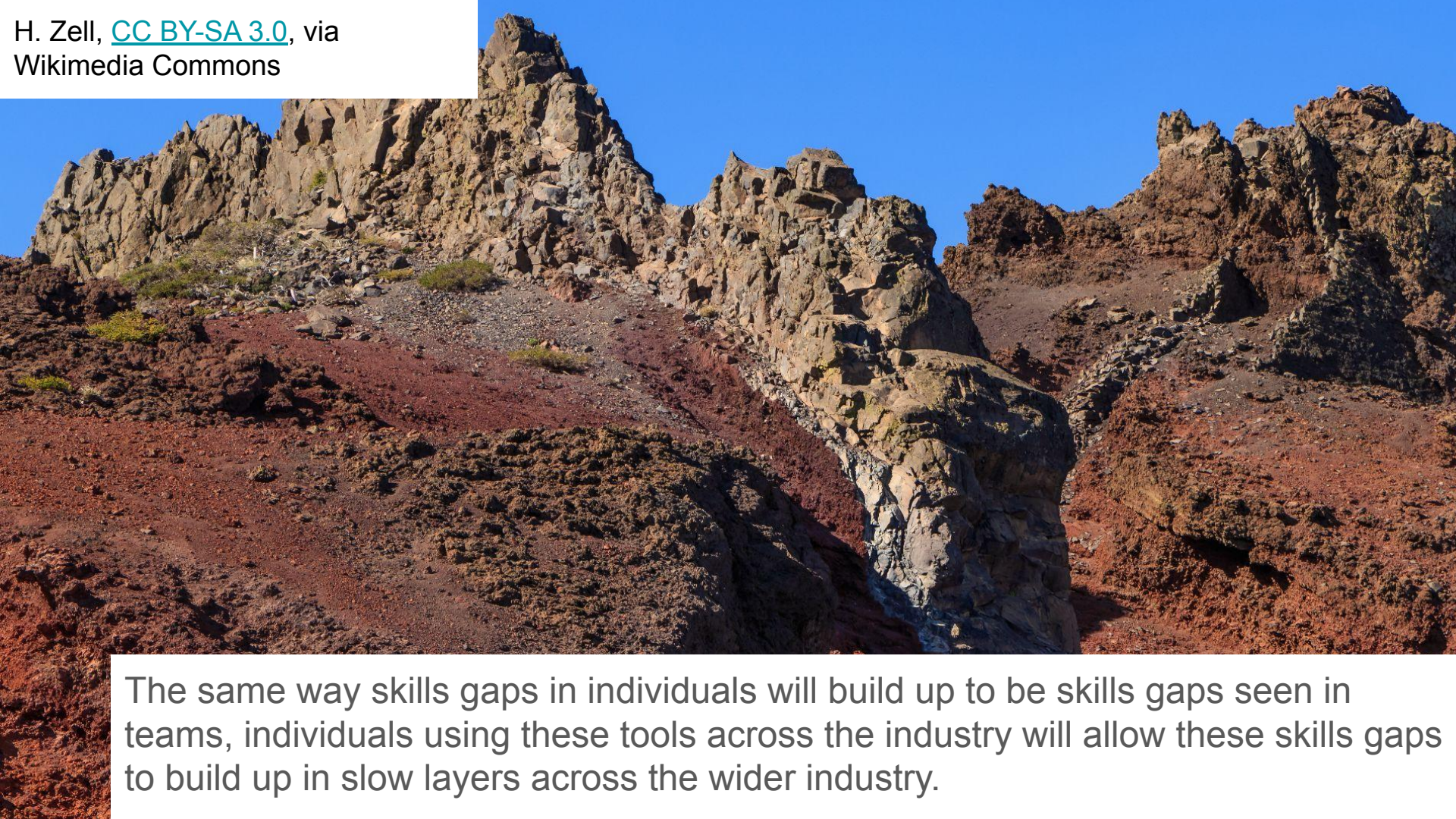


Code you did not write is code
you do not understand.

You cannot maintain code you
do not understand.

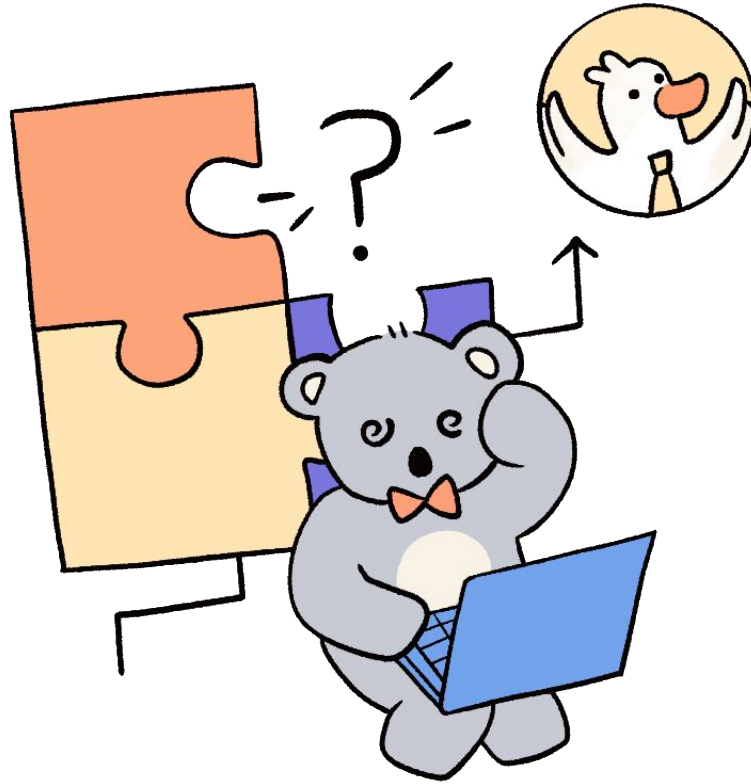
Industry Impacts





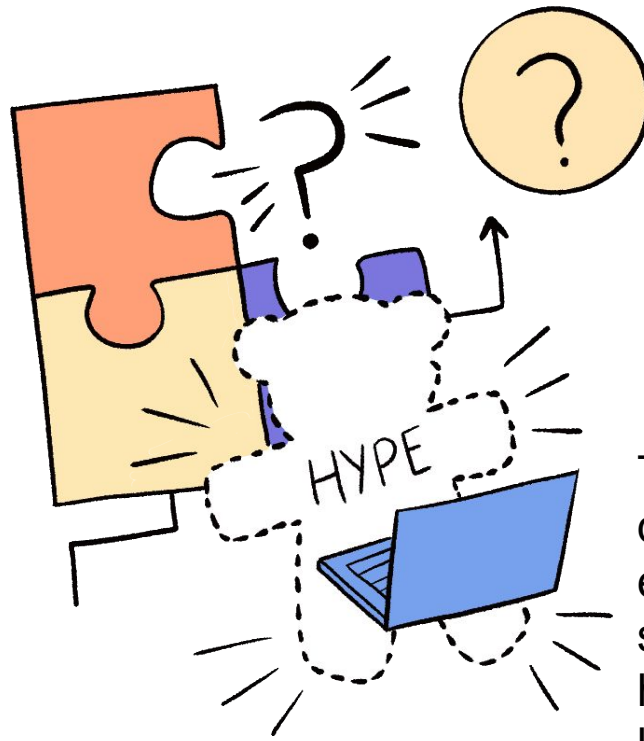
The same way skills gaps in individuals will build up to be skills gaps seen in teams, individuals using these tools across the industry will allow these skills gaps to build up in slow layers across the wider industry.

This industry level gap is likely to be increasingly visible over time for talent emerging now.



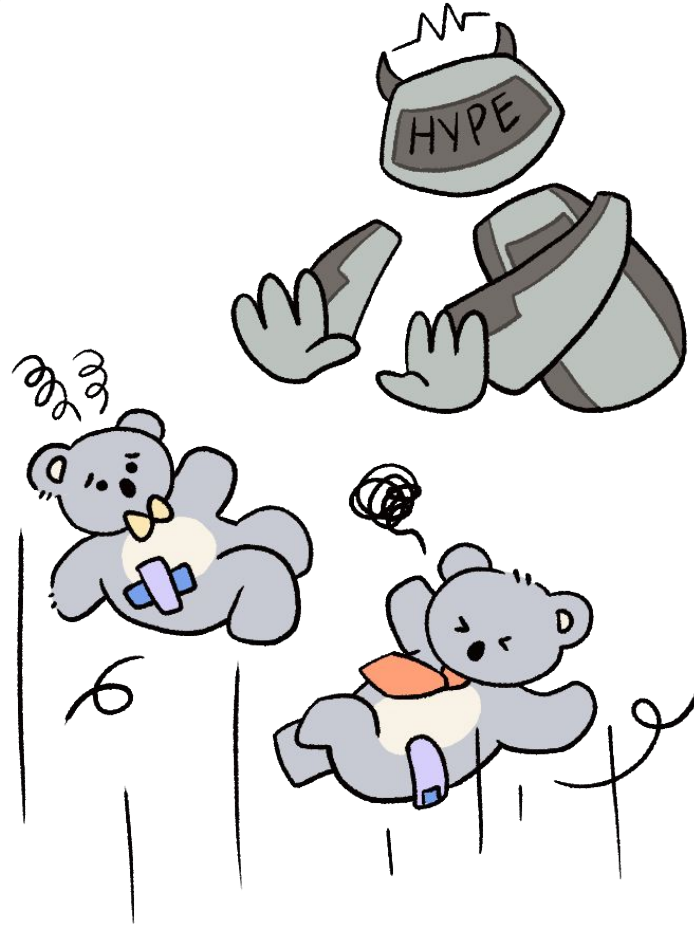
The abstraction these tools provide may prevent the development of procedural learning for these foundational concepts that allow new engineers to become really great seniors.

This gap becomes more literal as we see companies and teams using AI as an excuse to avoid hiring junior and emerging talent.



Taken at face value, this discourse by toolmakers and employers risks dramatically sharpening “senior only” hiring in tech, creating a literal missing generation of new developers.

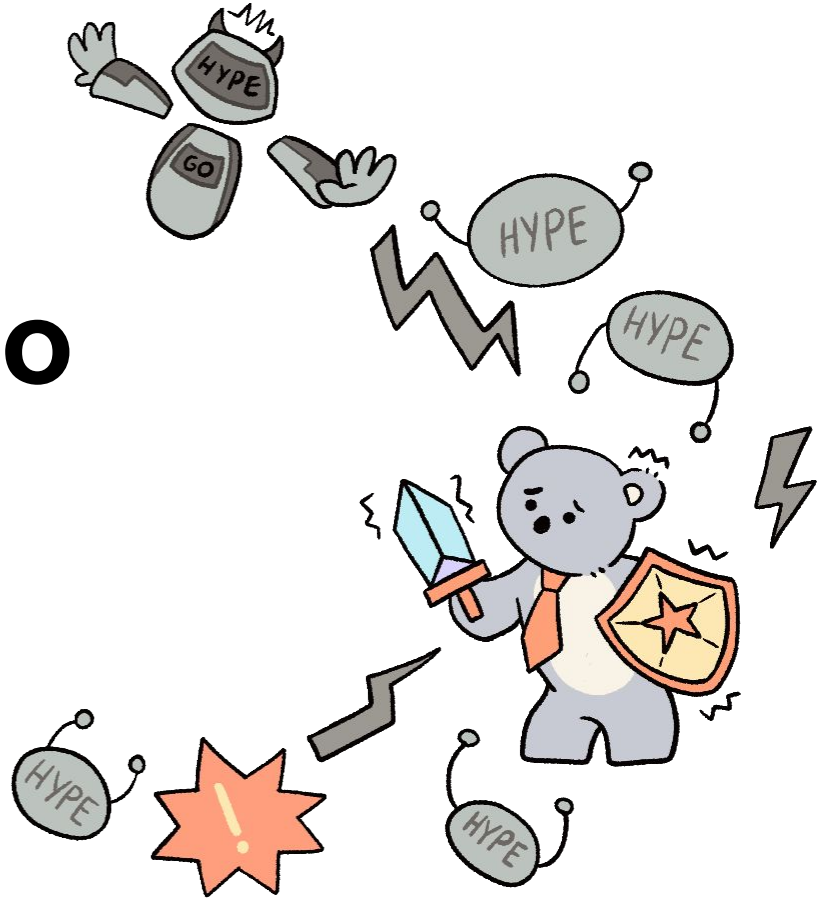
The threat of AI code tools replacing developer isn't just being used against new talent.



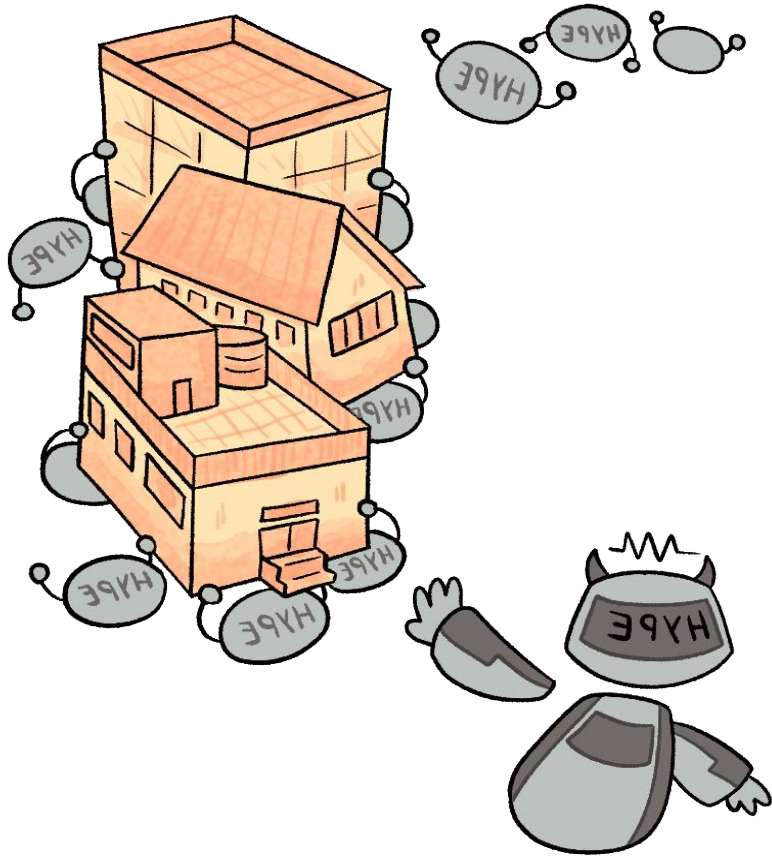
Intentionally or not, hype around these tools are weakening labour power for tech workers and creating a more fragile career security outlook for workers in tech.

...as well as workers from other creative and knowledge work who have had their work scraped into datasets.

**What can we do
for our own
learning**



- Formal courses or learning support programs like Recurse centre
- Silly (or serious) side projects
- Reading
- Keeping learning journal
- Daily challenges / exercises / exercism.org, codewars.com
- Writing blog posts about things you're learning, as notes to your future self
- When using these tools, pause to talk through what your generated code is doing at each step to better understand the code



**What can
we do for
our teams?**

If you're a manager or decisions maker:

- Build learning time into your timelines
- Code reviews that check for understanding
- Skills sharing build into team processes
- Be judicious and thoughtful about where and how these tools are supplied, encouraged and evaluated for your team members
- Training and supports that reinforce skills development

If you're not a manager or decision maker:

- ...there's not a lot you can do. You can try and convince your managers and decision makers that these things are important, but I might softly, kindly, suggest focusing your hope and your energy on developing your skills and supporting your peers in their skills



**What can we
do for the
industry**



It's not your job to fix the tech industry.

It's too big, too messy, too much, to have any individual set out to try and fix it.

But we can change things through individual choices at scale. One of the ways we can do this is by helping each

other.

- Teach and mentor, it's one of the best ways to share knowledge and will help you with your own skills development so it's win/win
- Support emerging talent: Their success as a lever for keeping the industry healthy
- Reinforce healthy working patterns and processes to build time to learn

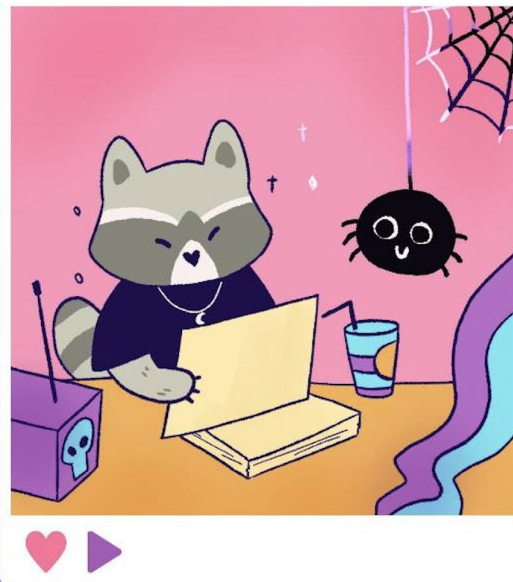


Thank you

Slide art by Kiri at kirionearth.com

Glitch art by [Antonio Roberts](#)

Painting by [Claire Douglass](#)



Neuroscience and programming information based on work by [Barbara Oakley](#) and [Zach Caceres](#)

Endless thanks to FFConf organizers, attendees and all the beautiful, real people making silly, strange real things and helping each other learn